MAIRON FIGUEIREDO MARQUES

CONTROLE POR ESTRUTURA VARIÁVEL E MODOS DESLIZANTES DE DISPOSITIVO ROBÓTICO COM MODELO DINÂMICO INCERTO:

IMPLEMENTAÇÃO NO ROBÔ INDUSTRIAL DENSO VP6242

Orientador: Prof. Dr.Márcio Roberto Covacic

Co-Orientador: Prof. Dr. Ruberlei Gaino

LONDRINA

2018

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Marques, Mairon.
CONTROLE POR ESTRUTURA VARIÁVEL E MODOS DESLIZANTES DE DISPOSITIVO ROBÓTICO COM MODELO DINÂMICO INCERTO: IMPLEMENTAÇÃO NO ROBÔ INDUSTRIAL DENSO VP6242 / Mairon Marques. - Londrina, 2018.
108 f.

Orientador: Márcio Covacic.
Coorientador: Ruberlei Gaino.
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Estadual de Londrina, Centro de Tecnologia e Urbanismo, Programa de Pós-Graduação em Engenharia Elétrica, 2018.
Inclui bibliografia.

1. Controle de Sistemas Robóticos Incertos - Tese. 2. Controle por Estrutura Variável - Tese. 3. Redes Neurais - Tese. 4. Controle por Torque Computado - Tese. I. Covacic, Márcio . II. Gaino, Ruberlei. III. Universidade Estadual de Londrina. Centro de Tecnologia e Urbanismo. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

MAIRON FIGUEIREDO MARQUES

CONTROLE POR ESTRUTURA VARIÁVEL E MODOS DELIZANTES DE DISPOSITIVO ROBÓTICO COM MODELO DINÂMICO INCERTO:

IMPLEMENTAÇÃO NO ROBÔ INDUSTRIAL DENSO VP6242

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Estadual de Londrina como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Prof. Dr. Márcio Roberto Covacic
Orientador
Universidade Estadual de Londrina

Prof. Dr. Emerson Ravazzi Pires da Silva
Universidade Tecnologica Federal do Paraná – Campus Cornélio Procópio

Prof. Dr. Rodrigo Cardim
Universidade Estadual Paulista "Júlio de Mesquita Filho" – Campus de Ilha Solteira

Londrina, Agosto de 2018

DEDICATÓRIA

Aos meus familiares e todos que me ajudaram na minha caminhada.

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me concedido forças, conhecimento e entendimento para superar as adversidades e ter chegado até o presente momento.

Sou imensamente grato aos meus pais, Paulo e Andria, que sempre me apoiaram e nunca mediram esforços para sempre me proporcionar as melhores condições possíveis para me dedicar puramente aos estudos. Sou grato pelo suporte financeiro, mas principalmente por serem exemplos de pessoas na minha vida e me proporcionarem o melhor amor que um filho poderia desejar. Sou grato a minha noiva Débora Machado por todas as vezes que tem me ajudado e me aconselhado durante meu período de mestrado, além de ter sido motivo de inspiração para mim. Também sou grato aos meus familiares e amigos por terem me apoiado e incentivado a seguir a carreira acadêmica.

Agradeço ao meu orientador, Dr. Márcio Roberto Covacic, pelas orientação no trabalho e em especial nos tópicos relacionados ao controle por estrutura variável. Agradeço ao meu coorientador, Dr. Ruberlei Gaino, pelas orientações no trabalho e em especial nos tópicos relacionados a manipuladores robóticos industriais. Sou grato a ambos por terem me acolhido no programa de mestrado da UEL, pelas orientações, amizades e por todas as conversas que me proporcionaram crescimento técnico e pessoal.

Sou grato ao Murillo Magan e ao Igor Oliveira pelo apoio, companherismo e motivação nos estudos. Também sou grato aos companheiros de laboratório, em especial ao Paulo Bronieira por ter me ajudado a trabalhar com redes neurais artificiais e aplicar esse conceito no mundo de robótica.

Fica aqui expressa a minha gratidão pelos representantes da Techsim (Glauco Mangolin e Joel Mangolin) bem como pelos colaboradores da Quanser (Roberto Chan e Michel Levis) que tem nos ajudado a solucionar problemas de software e hardware para que o presente projeto fosse concluído.

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código do financiamento 001.

“O temor do Senhor é o Princípio da Sabedoria”
Provérbios 9 - 10

“És Minha Força e Segurança,
És Meu Apoio, Minha Esperança,
És Minha Vida e Confiança;
Já Me Fizeste, Cidadão dos Ceus.”

MARQUES, Mairon Figueiredo. **Controle por Estrutura Variável e Modos Deslizantes de Dispositivo Robótico com Modelo Dinâmico Incerto:** Implementação no Robô Industrial Denso VP6242. 2018. 108 páginas. Dissertação de Mestrado em Engenharia Elétrica – Universidade Estadual de Londrina, Londrina, 2018.

RESUMO

O presente trabalho aborda o projeto e a implementação de controle dinâmico do manipulador robótico da marca Denso modelo VP6242, que possui seis graus de liberdade e modelo dinâmico incerto, utilizando condições baseadas em desigualdades matriciais lineares (do inglês *linear matrix inequalities* - LMIs) para efetuar o controle por estrutura variável e modos deslizantes. O método estudado foi aplicado em dois sistemas incertos do robô, sendo que o primeiro método fez o uso da modelagem dinâmica do sistema incerto e o segundo método fez o uso de redes neurais artificiais. Ambos métodos foram comparados com um controlador por torque computado com pensador proporcional, integral e derivativo (PID) padrão. Para obter o controlador padrão, inicialmente foi abordado um método clássico para modelagem de manipuladores robóticos dinâmicos: o método recursivo de Newton-Euler. Na sequência, com o auxílio do Robotic Toolbox do Matlab, foi obtido o modelo dinâmico do manipulador robótico Denso VP6242. Com o modelo dinâmico do robô foi desenvolvido o controle por torque computado com compensador PID. Uma vez que o controle por torque computado PID padrão foi desenvolvido, foi abordado condições baseadas em LMIs que garantem a positividade real estrita do sistema e consequentemente a estabilidade assintótica da planta, mesmo na presença de incertezas paramétricas. Em seguida foi projetado um controlador por estrutura variável e modos deslizante para ser implementado no manipulador robótico. Uma vez que o controle por estrutura variável e modos deslizantes para sistema incerto foi abordado, aplicou-se o controlador em duas abordagens diferentes: utilizando o modelo dinâmico incerto baseado no método recursivo de Newton Euler e utilizando redes neurais. Resultados práticos mostraram que o método utilizado para controlar sistemas incertos obteve estabilidade com baixo erro de trajetória. A aplicação do método utilizando o modelo resultante do método de Newton Euler obteve ótimo resultado, enquanto que a aplicação do método utilizando redes neurais obteve resultado muito bom, tendo como vantagem o fato de não necessitar do modelo dinâmico do sistema.

Palavras Chaves: Controle de Sistemas Robóticos Incertos. Controle por Estrutura Variável. Redes Neurais. Controle por Torque Computado.

MARQUES, Mairon Figueiredo. **Variable Structured Control Applied in Robotic Arm with Uncertain Dynamic Model:** Implementation in the Denso Industrial Robot VP6242. 2018. 108 pages. Dissertation of the Master's Degree Program in Electrical Engineering – State University of Londrina, Londrina, 2018.

ABSTRACT

The present work deals with the design and implementation of dynamic control of the Denso robotic manipulator model VP6242, which has six degrees of freedom and uncertain dynamic model, using conditions based on linear matrix inequalities (LMIs) to perform variable structure and sliding modes control. The studied method was applied in two uncertain systems of the robot, the first application made use of dynamic modeling of the uncertain system and the second application made use of artificial neural networks. Both methods were compared with a standard proportional, integral and derivative (PID) computed torque controller. To obtain the standard controller, it was first developed a classical method for dynamic modeling of robotic manipulators: Newton-Euler's recursive method. Then, with the help of the Robotic Toolbox for Matlab, the dynamic model of the robotic manipulator Denso VP6242 was obtained. Once the dynamic model of the robot was obtained, it was developed the computed torque control with PID compensator. Once that the standard computed torque controller with PID compensator was developed, it was developed LMI based conditions that guarantee that the is strictly positive real and consequently the asymptotic stability of the plant, even in the presence of parametric uncertainties. Next, a variable-structure and sliding modes controller were designed to be implemented in the robotic manipulator. Once the variable structured and sliding mode controller was developed, it was applied in two different approaches: using the uncertain dynamic model based on Newton Euler's recursive method and using neural networks. Practical results showed that the method used to control uncertain systems obtained stability with low trajectory error. The application of the method using the Newton Euler method resulted in an excellent result, while the application of the method using neural networks obtained a very good result, having the advantage of not needing the dynamic model of the system.

Key-words: Uncertain Robotic System Control. Variable Structure Control. Neural Networks. Computed Torque Control.

Sumário

Lista de Figuras

Lista de Tabelas

1. INTRODUÇÃO

Nos últimos anos, uma idéia que tem sido amplamente expressa é a de que os robôs são ferramentas capazes de substituir grande parte dos trabalhos manuais. De fato, o crescimento de robôs trabalhando nas indústrias já é consideravelmente elevado, e a perspectiva é de um crescimento ainda maior. De acordo com a Federação Internacional de Robótica (International Federation of Robotics – IFR) em 2016 foi constatado um volume de venda de manipuladores 16% maior do que em 2015. O mercado Asiático ganha grande destaque na quantidade de manipuladores, e o Brasil é apontado como o quarto país com maior crescimento de aquisição de robôs industriais nas Américas durante o ano de 2016 (International Federation of Robotics, 2017).

Com a crescente demanda por robótica, se faz necessário o desenvolvimento de tecnologias que possibilitam a construção, montagem, controle e aplicação de robôs industriais. O controle de manipuladores é um campo que demanda grande pesquisa, pois os manipuladores robóticos são sistemas não lineares, multivariáveis, com propriedades dinâmicas acopladas e que normalmente possuem imprecisões no modelo decorrentes de dinâmicas não modeladas e imprecisão paramétrica. Portanto, os robôs industriais necessitam de bons controladores para que possam movimentar-se com alta velocidade e com baixo erro Spong e Vidyasagar (1989), Siciliano e Khatib (2008) e Craig (2005).

Grande parte dos controladores de manipuladores robóticos necessita do modelo dinâmico do dispositivo. Algumas técnicas são encontradas na literatura para obter o conjunto de equações dinâmicas que regem o comportamento do sistema, as duas principais técnicas são: método recursivo de Newton Euler e técnica de Lagrange como mostrado em Spomg e Vidysagar (1989) e Craig (2005). Nessa dissertação faz-se uso do método recursivo de Newton Euler para obter o modelo dinâmico do manipulador robótico Denso VP6242.

A técnica recursiva de Newton Euler tem a vantagem de ser recursiva, isto é, o cálculo das forças e reações é feito junta por junta e elo por elo, desse modo é mais fácil transformar esse método em um algoritmo. Na formulação de Newton Euler cada elo é tratado individualmente (considerando a influência que outros elos causam nele), são encontradas as equações que descrevem os movimentos lineares e angulares de cada elo, e posteriormente as forças e reações causados por cada elo (Spong e Vidyasagar, 1989). Em Spong e Vidyasagar (1989), Siciliano e Kabit (2008) e Craig (2005) são encontrados pseudo-algoritmos para a obtenção do modelo dinâmico de um manipulador robótico utilizando o método recursivo de

Newton-Euler. Em Djuric (2010) é desenvolvida uma técnica que automaticamente separa os elementos das matrizes de inércia, de Coriolis e centrífuga, e gravitacional utilizando o método recursivo de Newton Euler, essa técnica recebe o nome de método de separação automática.

Embora o método de Newton Euler já se encontre amplamente difundido na literatura, esse método possui elevada complexidade para ser resolvidas manualmente para o caso de robôs com 6 graus de liberdade. Devido à complexidade que as equações de movimento de um manipulador podem assumir, algoritmos foram desenvolvidos para facilitar o processo de obtenção do modelo de um manipulador. Em Dean-Leon, Nair e Knool (2012) é apresentado um toolbox para Matlab/Simulink com a vantagem de ser mais didático e de fácil visualização, esse toolbox utiliza a metodologia de Lagrange para obter o modelo dinâmico dos robôs e tem a vantagem de obter a Matriz regressora do robô. Em Khalil, Vijayalingam, Khomutenk, Mukhanov et.al (2014) é apresentado o *OpenSYMORO*, trata-se de um software open-source que se destaca pela capacidade de obter o modelo simbólico de robôs com juntas flexíveis, robôs em cadeia aberta ou fechada, além de robôs com base móveis. Em Corke (1996) e Corke (2011) é apresentado o *Robotic Toolbox*, um toolbox que permite usuários do Matlab/Simulink criarem, analisarem e simularem dispositivos robóticos. Dentre as suas funcionalidades, essa ferramenta é capaz de gerar o modelo cinemático direto e inverso, matriz jacobiana, e modelo dinâmico direto e inverso de um manipulador (tanto momentâneo como simbólico) entre outras. Para o presente trabalho se fez uso do toolbox desenvolvido por Peter Corke devido a dois motivos: a confiabilidade e aceitação que esse toolbox tem no meio acadêmico, e a grande quantidade de material didático desenvolvido e disponível para uso.

Manipuladores robóticos são sistemas não lineares, multivariáveis e com propriedades dinâmicas acopladas. De acordo com Siciliano, Sciavicco, Villani e Oriolo (2008), a estratégia mais simples para controlar um dispositivo robótico que pode ser pensada é tratando o manipulador como um conjunto de N sistemas independentes (onde N é a quantidade juntas), e, então, projetar N controladores para os N sistemas SISO (single-input/ single-output), para essa estratégia de controle é dado o nome de controle descentralizado. Contudo, nesse caso os efeitos de acoplamentos entre as juntas devem ser tratados como distúrbio, e quanto maior a velocidade de operação do sistema, maior será o efeito causado pelo acoplamento e maiores serão as imprecisões e o erro de rastreamento do sistema. Portanto, essa arquitetura deixa de ser aplicável na prática quando se necessita de alto desempenho do sistema.

Objetivando arquiteturas de controle com maior precisão e rapidez, foi então proposta a utilização de uma arquitetura de controle que em vez de minimizar o efeito do distúrbio, elimina a causa do distúrbio: O controle por torque computado. Essa arquitetura é capaz de utilizar informações do modelo dinâmico do manipulador para cancelar todo efeito causado pelo acoplamento e não linearidade do sistema.

O controle por torque computado utiliza uma lei de controle capaz de cancelar as dinâmicas acopladas do sistema, restando apenas a parte desacoplada para ser controlada, e então pode ser projetado um compensador PID (Proporcional, Integral e Derivativo) para controlar a parte linear do sistema (Murray, Li, Sastry 1994 e Lewis, Dawson e Abdallah 2004, Siciliano, Sciavicco, Villani e Oriolo 2008, Craig 2005, Sciviacco e Siciliano 2000). Desse modo, é possível controlar um manipulador robótico com maior rapidez e eficiência. O presente trabalho faz uso da técnica de controle por torque computado para o robô industrial Denso VP6242 e posteriormente é aplicada uma técnica que proporciona maior robustez ao sistema.

Embora o controle por torque computado seja muito mais eficiente do que o controle descentralizado, uma desvantagem dessa arquitetura é a ausência de robustez. Para que o controle por torque computado possua bom rendimento é necessário ter uma aproximação do modelo dinâmico do sistema muito próximo do real, e muitas vezes isso é difícil devido às incertezas paramétricas do sistema. Uma maneira de adicionar robustez ao sistema é utilizar o controle por estrutura variável. O funcionamento básico dessa topologia baseia-se em guiar o sinal de erro até uma superfície de chaveamento, após isso o sistema entra em modo deslizante e não será afetado por nenhum erro de modelagem ou distúrbio.

O avanço de técnicas de controle por estrutura variável aplicado em robótica teve contribuições positivas em Slotine (1985), onde é apresentada uma arquitetura de controle por estrutura variável que também faz uso da estimativa do modelo dinâmico do manipulador. Já em Chen, Mita e Wakui (1990) o sistema possui incertezas paramétricas e o ganho do controlador projetado é robusto a essas incertezas. Em Medjebouri e Mehennaoui (2016) é proposto um controlador por modos deslizantes adaptativo, onde através de redes neurais é estimado o sinal de controle para descrever melhor o comportamento do sistema próximo da superfície de chaveamento, assim sendo, uma vez que é conhecido o comportamento dinâmico próximo da região que o manipulador encontra-se, então é possível descrever uma lei de controle que se adeque à posição em questão, resultados simulados foram obtidos utilizando essa técnica. Em Coung e Nan (2016) se faz uso do controle por estrutura variável juntamente com redes neurais adaptativas para controlar duas juntas de um manipulador

robótico, embora tenham sido apresentados bons resultados práticos, essa metodologia só foi aplicada para dois graus de liberdade. Os controladores aplicados a manipuladores acima citado possuem desempenho satisfatório, contudo é difícil agregar restrições de projetos tais como: taxa de decaimento, incertezas politópicas ou restrição de distúrbio para os projetos acima. Para o presente projeto faz-se uso das condições formuladas por LMIs (do Inglês: *Linear Matrix Inequality*, em Portugues, Desigualdade Matricial Linear*)* para projetar uma arquitetura de controle por estrutura variável para controle de trajetória de um braço robótico, posteriormente é abordado uma estratégia que dispensa a necessidade de conhecimento do modelo e parâmetros do manipulador.

Paralelamente aos avanços de controle por estrutura variável aplicado aos manipuladores, também houve avanço nas formulações dos controladores por estrutura variável. Em Covacic (2001) são encontradas formulações baseadas em LMI's para projeto de controle por estrutura variável, e em Covacic (2006) as formulações baseadas em LMI's são expandidas para garantir a estabilidade, robustez, taxa de decaimento e restrição de entrada e saída para sistemas com diferentes números de entradas e saídas.

Redes neurais é um tema que tem sido bastante utilizado no controle de sistemas dinâmicos, principalmente por sua propriedade de aproximação universal e capacidade de aprendizado. A propriedade de aproximação universal garante que sempre existirá um conjunto de pesos sinápticos e função de ativação na qual uma rede neural multicamada aproxime o comportamento de uma função não linear com um erro de aproximação $\in_n$ (erro de aproximação funcional da rede neural), para qualquer valor de $\in_n$. Uma das aplicações de redes neurais no campo de controle de dispositivos robóticos é na criação de redes neurais que tenham o mesmo comportamento dinâmico dos dispositivos robóticos (Lewis, Dawson e Abdallah 2004, Lewis, Jagannathan e Yesildirek 1999, Medjebouri e Mehennaoui 2016, Cuong e Nan 2016).

De acordo com Lewis, Dawson e Abdallah (2004), antes dos anos 90 o projeto e análise de controle utilizando redes neurais para sistemas incertos era bastante empírico, e muitas das vezes não era possível garantir a estabilidade dos sistemas. Em Lewis, Dawson e Abdallah (2004) e Lewis, Jagannathan e Yesildirek (1999) são dadas condições de projetos para controladores utilizando redes neurais, tanto de camada simples como de múltiplas camadas, contudo não são apresentados resultados dessa teoria. Em Medjebouri e Mehennaoui (2016) são apresentadas redes neurais adaptativas, e de maneira interativa é estimada a dinâmica do classico manipulador robótico PUMA 560 ao redor do ponto de atuação, contudo os resultados são simulados e a trajetória do dispositivo é relativamente

lenta. Em Cuong e Nan (2016) são utilizadas redes neurais com pesos sinápticos adaptativas para controlar 2 graus de liberdade de um robô, mas essa topologia foi abordada apenas para robô com 2 graus de liberdade e sabemos que, o acréscimo de mais juntas acarretará em forças de acoplamentos não modeladas. O presente projeto faz uso de redes neurais juntamente com controle por estrutura variável e modos deslizantes para efetuar o controle de junta do robô industrial Denso VP6242.

Diversas são as pesquisas encontradas na literatura que abordam arquiteturas de controle robusto para o controle de manipuladores robóticos. Em Vikas (2016) é apresentado resultados acerca do FTSMC (Fast Terminal Sliding Mode Control) utilizando redes neurais para controle de trajetória de um manipulador robótico com modelo dinâmico incerto, essa metodologia garante a estabilidade assintótica do sistema e a convergência de erro para zero em um tempo finito. O trabalho acima difere do trabalho proposto por não fazer uso de formulações baseadas em LMIs, e o fato de usarmos LMIs para garantir a estabilidade trás benefícios devido a facilidade de incorporar uma grande variedade de especificações e restrições de projetos, alem de existir diversos algoritmos eficiente para solucionar as LMIs. Em Jabili e Kazemi (2017) é difundido a idéia de um controlador com condições formuladas por LMIs que controla a trajetória de um manipulador com incertezas politópicas, contudo além da dificuldade de representar o sistema com incertezas politópicas essa metodologia não foi implementada em um dispositivo real. Em Tseng e Chen (2001) são expostos resultados a cerca de um controlador que utiliza redes neurais e condições baseadas em LMIs para efetuar o controle de trajetória de um dispositivo robótico, contudo, o trabalho acima diferencia do trabalho proposto por não utilizar sistemas ERP e também por não ter sido extraído resultados de aplicações reais utilizando essa metodologia.

A contribuição desse trabalho está fundamentada no fato de agregar à comunidade acadêmica resultados acerca da implementação de controle dinâmico para um robô com 6 graus de liberdade e modelo dinâmico incerto fazendo uso de controle por estrutura variável e modos deslizantes. O sistema incerto é decorrente do erro no cancelamento das não linearidades do sistema. Duas metodologias diferentes foram utilizadas para efetuar o cancelamento das não linearidades: utilizando uma estimativa do modelo dinâmico do robô e utilizando redes neurais artificiais. Para ambas as metodologias foram utilizadas condições baseadas em LMIs para projetar controle por estrutura variável que garante a estabilidade do sistema mesmo mediante a presença de imprecisões no cancelamento das não linearidades.

A presente pesquisa tem como objetivo o controle de trajetória em nível de juntas do manipulador robótico Denso VP6242, que possui modelo dinâmico incerto, em todo o ponto

de operação utilizando condições formuladas por LMIs para obter controle por estrutura variável e modos deslizantes. Para isso, os objetivos específicos são: desenvolvimento de trajetória suave; desenvolvimento, análise e implementação de controle puramente baseado no modelo do sistema (torque computado) para ser utilizado como comparação; garantia de que o sistema seja estritamente real positivo; desenvolvimento, análise e implementação de técnica de controle capaz de aumentar a robustez do sistema estritamente real positvo quanto a incertezas paramétricas (estrutura variável e modos deslizantes); obtenção de redes neurais que se comportam como o manipulador; utilização de redes neurais juntamente com controle por estrutura variável para controle de trajetória do manipulador.

O presente trabalho está organizado da seguinte maneira: o Capítulo 2 faz uma abordagem sobre as diferentes maneiras de obter o modelo de manipuladores robóticos. Neste Capítulo foi discutidos inicialmente conceitos teóricos sobre o modelo dinâmico direto e inverso de um manipulador robótico. Posteriormente, é detalhada uma abordagem que confia nas propriedades de energia do sistema para obter o modelo do mesmo. Em seguida é apresentado uma abordagem recursiva que utiliza as leis formulados por Newton e Euler para obter o modelo final do sistema. Em seguida, é abordado brevemente uma maneira computacional e eficiente de obter o modelo dinâmico de um robô. Por fim, é abordado o algoritmo usado para geração de uma trajetória suave para o manipulador.

O Capítulo 3 explora uma arquitetura de controle que faz uso do modelo dinâmico do sistema para controlar o manipulador com alta performance. Inicialmente é abordada uma técnica muito conhecida em robótica e controle de sistemas não lineares, o cancelamento das não linearidades. Posteriormente é agregado um compensador PID junto ao cancelamento das não linearidades, também são feitas análises e são obtidas condições de estabilidade para o projeto do controlador por torque computado com compensador PID.

O Capítulo 4 trata de solucionar um problema que comumente é encontrado nos controladores desenvolvidos no Capítulo 3: a incerteza paramétrica. Para alcançar maior robustez no controle de plantas robóticas é discutida a idéia de controle utilizando estrutura variável. É abordada uma técnica baseada em LMI's que assegura que o sistema seja estritamente real positivo, e também é fundamentada a teoria da utilização de controle por estrutura variável utilizando sistemas estritamente reais positivos. Por fim são expostas condições utilizando LMI's para projeto de controle por estrutura variável.

No Capítulo 5 é exposta uma técnica de controle que visa à substituição do modelo dinâmico dos manipuladores para efetuar o controle dos mesmos: as redes neurais. A princípio é explanado a respeito das redes neurais e suas propriedades que possibilitam a

utilização dessa técnica para o controle de manipuladores robóticos, posteriormente é abordado e discutido uma arquitetura de controle que faz o uso de redes neurais e estrutura variável para controle de trajetória de dispositivos robóticos.

No Capítulo 6 são apresentados os materiais utilizados para efetuar as implementações dos controladores. Inicialmente, é mostrado e discutido o princípio de funcionamento da bancada, posteriormente são citados e explanados sobre os principais blocos da bancada: o robô, o controlador, e a interface Simulink/controlador.

No Capítulo 7 são explorados os métodos e resultados obtidos para três arquiteturas de controle. A primeira arquitetura de controle faz uso puramente das equações dinâmicas que regem o comportamento do sistema. O segundo controlador utiliza o controle por estrutura variável, projetado por intermédio de LMIs, para inserir maior robustez ao controlador por torque computado com compensador PID. O terceiro controlador explora as redes neurais e controle por estrutura variável e controla o robô mesmo sem o conhecimento das equações dinâmicas do dispositivo. Todos os casos são implementados no robô real e são extraídos dados práticos para análise dos resultados.

Por fim, o Capítulo 8 apresenta a conclusão do trabalho. As referências utilizadas e alguns apêndices importantes para o desenvolvimento dessa pesquisa encontram-se após a conclusão do mesmo.

2. MODELAGEM DE MANIPULADORES ROBÓTICOS

A derivação do modelo dinâmico de um manipulador tem uma função muito importante para análise, projeto e simulação de malhas de controle para dispositivos robóticos. Esse Capítulo abordará o método recursivo de Newton-Euler para obtenção do modelo dinâmico de um manipulador. Na sequências será abordada a modelagem de dispositivos robóticos utilizando o "Robotic Toolbox for Matlab" desenvolvido por Peter Corke (Corke 2011), e por fim esse Capítulo aborda a problemática de geração de trajetória para controle de juntas de um manipulador.

2.1 MODELO DINÂMICO DIRETO E INVERSO

O modelo dinâmico de um manipulador pode ser utilizado para realizar simulações do movimento dos manipuladores, permitindo testar o desempenho de diferentes arquiteturas de controles de maneira segura e sem a necessidade de utilizar um protótipo real (Sciviacco e Siciliano, 2000). De acordo com Corke (1996; 2011) e Khalil e Dombre (2004), o modelo dinâmico de um manipulador é a relação entre torque aplicado em cada uma das juntas e posição, velocidade e aceleração angular ou linear (dependendo do tipo de junta) das juntas. O modelo dinâmico pode ser dividido em modelo dinâmico direto e modelo dinâmico inverso.

2.1.1 MODELO DINÂMICO INVERSO

O modelo dinâmico inverso computa o torque requerido para obter posição, velocidade e aceleração das juntas e normalmente é simplesmente chamado de modelo dinâmico. Embora suas equações diferenciais sejam complexas e, muitas das vezes, altamente acopladas, o modelo dinâmico direto pode ser representado como mostrado na Equação (2.1):

$$\zeta_i = M(\theta_i)\ddot{\theta}_i + V(\theta_i, \dot{\theta}_i)\dot{\theta}_i + F(\dot{\theta}_i) + G(\theta_i) = M(\theta_i)\ddot{\theta}_i + N(\theta_i, \dot{\theta}_i), \qquad (2.1)$$

onde:

θ_i é a i-nésima posição angular (caso de junta rotacional) ou linear (caso de junta translacional);

$\dot{\theta}_i$ é a i-nésima velocidade angular (caso de junta rotacional) ou linear (caso de junta tranlacional);

$\ddot{\theta}_i$ é a i-nésima aceleração angular (caso de junta rotacional) ou linear (caso de junta translacional);

ζ_i é o i-nésimo torque (para junta rotacional) ou força (para junta translacional) e tem dimensão $i \times 1$;

$M(\theta)$ é a matriz inercial derivada em espaço de junta de dimensão $i \times i$;

$V(\theta, \dot{\theta})$ é a matriz Coriolis e centrífuga de dimensão $i \times i$;

$F(\dot{\theta})$ são os torques, ou forças, resultantes de atrito de dimensão $i \times 1$;

$G(\theta)$ são os torques, ou forças, resultantes da força gravitacional;

$N(\theta, \dot{\theta}) = V(\theta, \dot{\theta})\dot{\theta} + F(\dot{\theta}) + G(\theta)$ são as forças resultantes do efeito Coriolis, centrifuga, atrito e de força gravitacional e tem dimensão $i \times 6$;

i é a quantidade de graus de liberdade do sistema.

O objetivo de projetos de controle de juntas de um manipulador robótico é que o dispositivo siga uma trajetória pré-estabelecida. Se tratando de manipuladores robóticos, a trajetória é definida como sendo o conjunto de posição, velocidade e aceleração de cada junta no decorrer do tempo. Portanto, o modelo dinâmico inverso tem uma função muito importante para o controle de junta, pois esse modelo tem a função de computar o torque resultante quando cada uma das juntas segue uma determinada trajetória. Ou seja, o modelo dinâmico informa qual o torque necessário que cada junta deve exercer para que a trajetória definida seja percorrida. A Figura 2.1 exemplifica um bloco responsável por informar o modelo dinâmico de um manipulador.

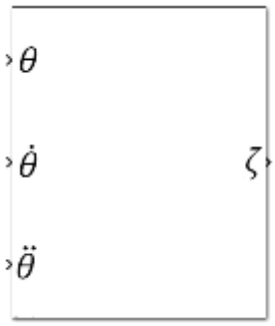

Figura 2.1 - Bloco de Modelagem Dinâmica Inversa (Adaptado de Corke 2011).

Na Figura 2.1 é notório a presença de três sinais de entradas $(\theta, \dot{\theta}, \ddot{\theta})$, e de um sinal de saída (ζ). O conjunto de entradas do bloco representa a trajetória pré-definida e a saída do bloco é o torque resultante para que o manipulador siga a trajetória. Mais especificamente: θ é um vetor de dimensão i (onde i é a quantidade de juntas do manipulador) que contém a posição angular (ou linear no caso de uma junta prismática) de todas as juntas; $\dot{\theta}$ é um vetor de dimensão i que contém a velocidade angular (ou linear no caso de uma junta prismática) de

cada todas as juntas; $\ddot{\theta}$ é um vetor de dimensão i que contém a acelerações angular (ou linear no caso de uma junta prismática) de cada todas as juntas. Por fim, ζ é um vetor de dimensão i que contém os torques necessários para que o manipulador siga a trajetória requerida.

2.1.2 MODELO DINÂMICO DIRETO

Em contrapartida, o modelo dinâmico direto é o conjunto de equações diferenciais que computa a aceleração de cada junta e tem como entrada os estados de posição, velocidade e torque aplicado a cada junta. Esse modelo é muito utilizado para simulação de plantas robóticas e possui fórmula genérica dada pela Equação (2.2):

$$\ddot{\theta} = M^{-1}(\theta)\left(\zeta - V\left(\theta,\dot{\theta}\right)\dot{\theta} - F\left(\dot{\theta}\right) - G(\theta)\right). \tag{2.2}$$

De acordo com Craig (2005), dado a posição, velocidade e torque aplicado em um instante t, é possível aplicar técnicas de integração numéricas com passo de integração Δt para integrar a aceleração resultante de (2.2) e obter a velocidade e posição futura. Uma maneira simples de computar as respectivas acelerações e posições no tempo $t + \Delta t$ é utilizando a integração de Euler. Iniciando no tempo $t = 0$, dado $\theta(0)$, $\dot{\theta}(0)$ e utilizando $\ddot{\theta}(t)$ resultante de (2.2), as respectivas velocidades e posições no instante $t + \Delta t$ são dados por (2.3):

$$\begin{aligned} \dot{\theta}(t + \Delta t) &= \dot{\theta}(t) + \ddot{\theta}(\mathrm{t})\Delta t, \\ \theta(t + \Delta t) &= \theta(t) + \dot{\theta}(t)\Delta t + 0{,}5\ddot{\theta}(\mathrm{t})\Delta t^2. \end{aligned} \tag{2.3}$$

O modelo dinâmico direto representa o funcionamento real de um dispositivo robótico. Em dispositivos reais é aplicado um sinal de torque em cada uma das juntas, e resultante do torque injetado é obtido a posição, velocidade e aceleração (angular ou linear dependendo do tipo de junta) de cada junta. A Figura 2.2 representa um bloco de simulação responsável por informar o modelo dinâmico inverso do dispositivo.

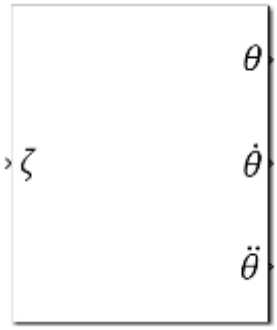

Figura 2.2 - Bloco de Modelagem Dinâmica Direta (Adaptado de Corke 2011).

Na Figura 2.2 é notório a presença de um sinal de entrada (ζ), e de três sinais de saídas $(\theta, \dot{\theta}, \ddot{\theta})$. A entrada é o torque aplicado, e a saída é a trajetória obtida (conforme explicação detalhada de cada elemento descrita anteriormente).

A seguir será explanado sobre como obter o modelo dinâmico de um manipulador e como gerar uma trajetória suave para um dispositivo robótico.

2.2 MÉTODO RECURSIVO DE NEWTON-EULER

O método recursivo de Newton-Euler é uma maneira de obter o modelo dinâmico indireto de um manipulador. Esse método conduz a uma formulação onde cada elo é tratado individualmente (considerando a influência que outros elos causam nele), e são encontradas as equações que descrevem os movimentos lineares e angulares de cada elo, e posteriormente as forças e reações causados por cada elo Spong e Vidyasagar (1989).

De acordo com Siciliano e Katib (2008), inicialmente são computadas as velocidades (linear e angular) e acelerações (linear e angular) do primeiro elo e recursivamente se computa as velocidades e acelerações dos elos subseqüentes, até computar as velocidades e acelerações da ponteira do manipulador, e a essa etapa é dado o nome de computação dinâmica direta.

Uma vez que as velocidades e acelerações para cada elo foram computadas, as forças e torque de interações e torques aplicado pelos atuadores são calculados e relacionados com as posições, velocidades e acelerações através das equações de Newton e Euler, e a essa etapa é dado o nome de computação dinâmica indireta. Segundo Spong e Vidyasagar (1989) e Creig (2005), a Equação de Newton diz que a taxa de variação do momento linear é igual à força aplicada em um corpo. Matematicamente essa relação pode ser descrita como: $f = \frac{d(m \cdot v)}{dt}$, sendo f força, m massa e v velocidade. Como a massa não varia em relação ao tempo a expressão pode ser reescrita como $f = m \cdot a$, sendo a aceleração. De acordo com Spong e

Vidyasagar (1989) a teoria de Euler afirma que a taxa de variação do momento angular é igual ao torque aplicado ao corpo. Matematicamente essa relação pode ser descrita como: $\tau = \frac{d(I_0 \cdot \omega_0)}{dt}$, onde I_0 é o momento de inércia de um corpo em relação ao centro de coordenadas com origem no centro de massa do corpo, ω_0 é a velocidade angular do corpo representado no sistema de coordenadas com origem no centro de massa e τ a soma de torque aplicado no corpo. Diferentemente do caso em que a massa é constante independente da força aplicada, o momento de inércia é descrito em relação a um eixo de coordenadas com origem acoplada no centro do corpo, e desse modo, qualquer rotação do corpo pode causar um desalinhamento entre o sistema de coordenadas do momento de inércia e o sistema de coordenadas da junta. De acordo com Spong e Vidyasagar (1989), uma maneira de superar esse obstáculo é descrevendo o movimento angular em relação ao sistema de coordenadas que está fixo ao corpo (sistema sobre a junta), desse modo a variação de momento angular aplicado ao corpo é dado por: $\tau = I\dot{\omega} + \omega \times (I\omega)$, onde I é constante e representa o momento de inércia do corpo em relação ao sistema de coordenadas fixo do corpo (sobre a junta), ω é a velocidade angular representada no sistema de coordenadas fixo.

2.2.1 ALGORITMO PARA MÉTODO DE NEWTON-EULER

O método de Newton-Euler é chamado de método recursivo por que é necessário computar as velocidades e acelerações linear e angular do centro de massa do elo i para depois computar as mesmas grandezas do elo $i + 1$, da mesma forma é necessário computar as forças de reação em i para depois computar as forças e reações de $i - 1$. Devido a essa recursividade, o método de Newton-Euler tem maior eficiência computacional, sendo assim, é possível encontrar na literatura instruções gradativas para obtenção do modelo dinâmico indireto através do método recursivo de Newton-Euler, como em Siciliano e Kabit (2008), Craig (2005), Djuric (2010).

Em Craig (2005) é encontrada uma formulação recursiva para obter o modelo dinâmico indireto de um manipulador robótico. A formulação se inicia com o levantamento da dinâmica direta, isso é, levantamento de velocidades linear e velocidade e aceleração angular do centro de massa de cada elo (iniciando no elo 0 até o elo N) e posteriormente é feito o levantamento da dinâmica indireta, isto é, através das equações de Newton ($f = m \cdot a$) e Euler($\tau = I\dot{\omega} + \omega \times (I\omega)$) são obtidos os momentos de reação de cada eixo para um

manipulador com 6 graus de liberdade. A dedução detalhada de cada Equação é encontrada com detalhe em Creig (2005) e é brevemente mostrado abaixo.

Inicialmente é necessário determinar as velocidades e acelerações dos elos. Para i iniciando em 1 e incrementando unitariamente até *N*, calcule as velocidades, acelerações, força e momentos dado pelas equações (2.4) – (2.8):

$$\omega_i^i = R_{i-1}^i \cdot \omega_{i-1}^{i-1} + \dot{\theta}_i \cdot Z_i^i, \tag{2.4}$$

$$\dot{\omega}_i^i = R_{i-1}^i \cdot \dot{\omega}_{i-1}^{i-1} + R_{i-1}^i \cdot \left(\omega_{i-1}^{i-1} \times \left(\dot{\theta}_i \cdot Z_i^i\right)\right) + \ddot{\theta}_i \cdot Z_i^i, \tag{2.5}$$

$$\dot{v}_i^i = R_{i-1}^i \left[\left(\dot{\omega}_{i-1}^{i-1} \times P_i^{i-1}\right) + \omega_{i-1}^{i-1} \times \left(\omega_{i-1}^{i-1} \times P_i^{i-1}\right) + \right] \dot{v}_{i-1}^{i-1}, \tag{2.6}$$

$$\dot{vc}_i^i = \dot{\omega}_i^i \times Pc_i^i + \omega_i^i \times \left(\omega_i^i \times Pc_i^i\right) + \dot{v}_i^i, \tag{2.7}$$

$$F_i^i = m_i \cdot \dot{vc}_{i-1}^{i-1}, \tag{2.8}$$

$$N_i^i = I_i^{Ci} \dot{\omega}_i^i + \omega_i^i \times \left(I_i^{Ci} \omega_i^i\right). \tag{2.9}$$

Posteriormente é necessário determinar as forças e momentos de inércia resultantes. Para i iniciando em *N* e decrementando unitariamente até 1, calcule os torques e forças dado pelas equações (2.10) – (2.12):

$$f_{i-1}^{i-1} = R_i^{i-1} f_i^i + F_i^i, \tag{2.10}$$

$$n_{i-1}^{i-1} = N_{i-1}^{i-1} + R_i^{i-1} \cdot n_i^i + \left(Pc_{i-1}^{i-1} \times F_{i-1}^{i-1}\right) + \left(P_i^{i-1} \times \left(R_i^{i-1} \cdot f_i^i\right)\right), \tag{2.11}$$

$$\tau_{i-1} = {n_{i-1}^{i-1}}^T \cdot Z_i^i. \tag{2.12}$$

onde: ω_i^i representa a velocidade angular do elo i nas coordenadas i;

R_{i-1}^i representa a matriz rotacional que o sistema de coordenadas i com $i-1$;

$\dot{\theta}_i$ representa a velocidade angular procedente do atuador i;

Z_i^i representa o versor com a orientação do atuador i no sistema de coordenadas cartesiano i, e normalmente $Z_i^i = [0\ 0\ 1]'$;

$\dot{\omega}_i^i$ representa a velocidade angular do elo i;

$\ddot{\theta}_i$representa a aceleração angular do procedente do atuador i;

$\dot{v}_i^i$ representa a aceleração linear do elo i no sistema de coordenadas cartesiana i;

P_i^{i-1} representa o vetor translação entre os sistemas $i-1$ e i;

$\dot{v}c_i^i$ representa a velocidade linear do centro de massa i em coordenadas cartesiana i;

Pc_i^i representa o vetor que localiza o centro de massa de cada elo i em coordenadas cartesianas i;

F_i^i representa as forças atuando em i em coordenadas cartesianas i;

N_i^i representa os momentos angulares atuando em i em coordenadas cartesianas i;

I_i^{Ci} representa o momento de inércia do link i em relação ao sistema de coordenadas com origem no centro de massa de i e dado em coordenadas cartesianas i;

f_{i-1}^{i-1} representa as forças exercidas na junta $i-1$ pela junta i em coordenadas cartesianas i;

n_{i-1}^{i-1} representa os torques exercidas na junta $i-1$ pela junta i em coordenadas cartesianas i;

τ_{i-1} representa o torque (para junta rotacional) ou força (para junta translacional) da junta $i-1$;

De acordo com Craig (2005), o efeito da gravidade é facilmente adicionado nas equações acima, apenas substituindo $\dot{v}_i^i = G$, onde G possui a magnitude do vetor gravitacional, mas com direção apontando para baixo.

2.3 SOFTWARE APLICADOS EM SIMULAÇÃO DE MANIPULADORES

O Robotics Toolbox desenvolvido por Peter Corke (Corke 1996, Corke 2011) é um software que utiliza o método recursivo de Newton-Euler para permitir aos usuários do Matlab criarem, analisarem e manipularem dispositivos robóticos. Dentre as suas funcionalidades, essa ferramenta é capaz de gerar o modelo cinemático direto e inverso, matriz jacobiana, e modelo dinâmico direto e inverso de um manipulador (tanto momentâneo como simbólico). Para isso, é necessário que o usuário forneça os parâmetros de Denavit-Hartenberg e parâmetro inerciais (massa, centro de gravidade, momento de inércia e tensor de

inércia) do manipulador. Além disso, a ferramenta possui a capacidade de modelar outros fenômenos envolvidos em manipuladores, tais como inércia do rotor dos motores, relação de transmissão, atrito viscoso e atrito de Coulomb.

A função “SerialLink” permite que o usuário descreva um robô como sendo um conjunto de elos que são definidos pela função “Link”. O toolbox também é capaz de retornar ao usuário a modelagem simbólica, bem como blocos de simulação do Simulink com as informações de cinemática direta, cinemática inversa, matriz jacobiana e jacobiana transposta, matriz inercial, centrípeta e gravitacional, modelo dinâmico direto e inverso do dispositivo através da função *“CodeGenerator”*. O toolbox também conta com blocos do Simulink feitos em S-Function que fazem interface com informações contidas no workspace do Matlab (Corke 2011).

Segundo Corke (1996), o método recursivo de Newton-Euler (abordado na Seção anterior) é uma abordagem eficiente para gerar algoritmos para calcular o modelo dinâmico inverso, e, portanto foi implementado no toolbox através da função *“rne( )”*. Essa função utiliza o método recursivo de Newton Euler para calcular o torque necessário para que uma trajetória pré-definida seja seguida. A seguir é extraída a modelagem de um manipulador robótico com 2 graus de liberdade por meio do Robotic Toolbox, e são abordadas algumas funções desse toolbox. Considere o manipulador robótico com 2 graus de liberdade mostrado na Figura 2.3, cujos parâmetros de Denavit-Hartenberg são dados na Tabela 2.1, e cujos parâmetros dinâmico do dispositivo são dados na Tabela 2.2. Considere também que na Figura 2.3 xy, x_0y_0, x_1y_1 e x_2y_2 são os respectivamente eixos da abcissa e ordenadas do sistemas cartesianos representado.

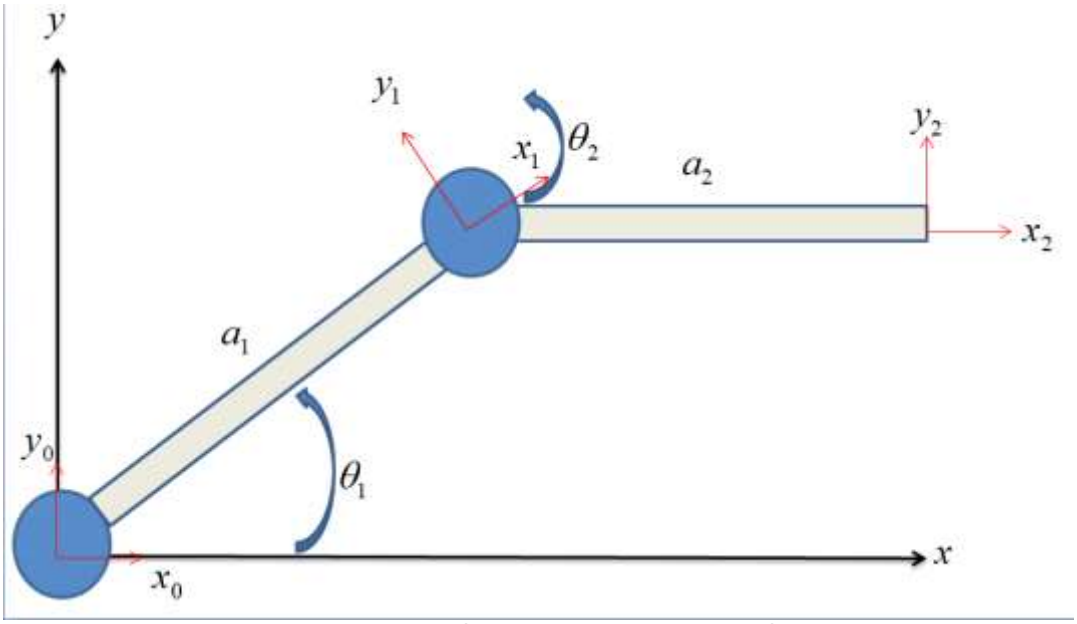

Figura 2.3 - Manipulador Robótico com 2 Graus de Liberdade (Próprio Autor)

Junta	Ângulo da Junta	Offset do Elo	Comprimento do Elo	Angulo de Torção
1	θ_1	0	$a_1 = 0{,}3m$	0
2	θ_2	0	$a_2 = 0{,}21m$	0

Tabela 2.1 - Parâmetros de Denavit Heartenberg de Robô com 2 Graus de Liberdade (Próprio Autor)

Parâmetro	Junta 1	Junta 2
Massa (kg)	1	1
Centro de Gravidade em relação as coordenadas cartesianas do próximo referencial (metros)	$[0{,}15 \quad 0 \quad 0]$	$[0{,}105 \quad 0 \quad 0]$
Tensor Inercial (kgm^2)	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0{,}0075 & 0 \\ 0 & 0 & 0{,}0075 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0{,}003675 & 0 \\ 0 & 0 & 0{,}003675 \end{bmatrix}$
Relação de Transmissão	1:1	1:1

Tabela 2.2 - Parâmetros Dinâmicos de Robô com 2 Graus de Liberdade (Próprio Autor)

Então, é possível utilizar a função *"Link"* do Robotic Toolbox para criar elos de um dispositivo robótico. Uma vez que os links são criados, então é possível acoplar um elo ao outro através da função *"SerialLink"*. O código abaixo foi desenvolvido em Matlab e utiliza o Robotic Toolbox para criar o modelo de um robô com as propriedades mecânicas do dispositivo apresentado nas Tabelas 2.1 e 2.2. É notável que alguns dos parâmetros são considerados como nulos, portanto esse modelo desconsidera os efeitos que esses parâmetros causariam no manipulador.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%    This code loads the parameters to derive Denso Dynamic Model        %%
%%     into a Serial Link variable. It's required the use of Robotics     %%
%%     Tolbox for Matlab (at least release 9.9) available at             %%
%%     http://www.petercorke.com.                                        %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Dynamic Link Viscous Frictions (N.m.s/rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
B_1 = 0;
B_2 = 0;

%% Actuator Gear Ratio %%
%%%%%%%%%%%%%%%%%%%%%%%%%%
G_1 = 1;
G_2 = 1;
```

```
%% Actuator Motor Inertia (Kg.m^2) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J_1 = 0;
J_2 = 0;

%% Link Mass (Kg) %%
%%%%%%%%%%%%%%%%
m_1 = 1;
m_2 = 1;

%% Link Center Of Gravity (m)%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Pc_1 = [-0.15 0 0];
Pc_2 = [-0.105 0 0];

%% Inertia Matrix of Link about link COF (Kg.m^2) %% *Chekar se referencial é o mesmo
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
I_1 = [0 0.0075 0.0075];
I_2 = [0 0.00367 0.00367];

%% Joint Variable Limits [min max] (rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J_1_lim = [-160 160]*pi/180;
J_2_lim = [-160 160]*pi/180;

%% Joint Variable offset (rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J_1_offset = 0;
J_2_offset = 0;

%% Link Offset (m) %%
%%%%%%%%%%%%%%%%%%%%%
d_1 = 0;
d_2 = 0;


%% Link Lenght (m) %%
%%%%%%%%%%%%%%%%%%%%%
a_1 = 0.300;
a_2 = 0.210;

%% Link twist Angle (rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
alpha_1 = 0;
alpha_2 = 0;

%% Definition of Links %%
%%%%%%%%%%%%%%%%%%%%%%%
```

```
L(1) = Link('d', d_1, 'a', a_1, 'alpha', alpha_1, 'standard', 'offset', J_1_offset, 'qlim', J_1_lim, 'I', I_1, 'r', Pc_1, 'm', m_1, 'G', G_1, 'B', B_1, 'Jm', J_1);
L(2) = Link('d', d_2, 'a', a_2, 'alpha', alpha_2, 'standard', 'offset', J_2_offset, 'qlim', J_2_lim, 'I', I_2, 'r', Pc_2, 'm', m_2, 'G', G_2, 'B', B_2, 'Jm', J_2);

%% Definition of Robot %%
%%%%%%%%%%%%%%%%%%%%%%%%%%
D2 = SerialLink(L, 'name', 'D2Robot', 'comment', '2 DOF manipulator');
```

Uma vez que uma variavel do tipo SerialLink contendo as caractristicas dinâmica do manipulador da Figura 2.3 foi criada através do Robotic Toolbox é possivel extrair dados e informações uteis no projeto de controle, implementação e simulação do mesmo. Através da função *"rne"* o toolbox resolve o modelo dinâmico de um dispositivo definido como *"SerialLink"*, isto é, o toolbox resolve a Equação (2.1) para qualquer dispositivos definidos como *"SerialLink"*. O código *"D2.rne([pi/2, 0],[1 2],[3 4])"* resulta no vetor [0.779 0.1974], ou seja o toolbox calcula que é necessário aplicar um torque de intensidade 0.779 Nm na primeira junta para que esta fique na posição $\frac{pi}{2} rad$, com velocidade instantanea de 1 rad/s e aceleração instantânea de 3 rad/s^2, e que é necessario aplicar um torque de intensidade 0.1974 Nm na segunda junta para que esta fique na posição 0 rad, com velocidade instantânea de 2 rad/s e aceleração instantânea de 4 rad/s^2.

Além de computar o modelo dinâmico indireto instantâneo, o toolbox tambêm trabalha com a modelagem simbólica, como é mostrado no código abaixo.

```
D2_sym=D2.sym(); %It will generate symbolic D2 Robot

syms q1 q2; % Creates symbolic variables q1 and q2
syms qd1 qd2; % Creates symbolic variables qd1 and qd2
syms qdd1 qdd2; % Creates symbolic variables qdd1 and qdd2
D2_InvDyn=D2_sym.rne([q1 q2],[qd1 qd2],[qdd1 qdd2])
```

O código acima tem como resultado expressão (2.13):

$$D2_{InvDyn} = [\frac{26939qdd1}{200000} + \frac{2939qdd2}{200000} - \frac{63qd2^2sin(q2)}{2000} + \frac{63qdd1cos(q2)}{1000} + \frac{63qdd2cos(q2)}{2000} - \frac{63qd1qd2sin(q2)}{1000} + \frac{63sin(q2)qd1\text{^}2}{2000} + \frac{2939qdd1}{200000} + \frac{2939qdd2}{200000} + \frac{63qdd1cos(q2)}{2000}]. \quad (2.13)$$

A expressão acima pode ser representado pelo em seu formato clássico pela Equação (2.14):

$$\begin{bmatrix}\tau_1\\ \tau_2\end{bmatrix} = \begin{bmatrix}\frac{26939}{200000}+\frac{63}{1000}\cos(\theta_2) & \frac{2939}{200000}+\frac{63}{2000}\cos(\theta_2)\\ \frac{2939}{200000}+\frac{63}{2000}\cos(\theta_2) & \frac{2939}{200000}\end{bmatrix}\begin{bmatrix}\ddot{\theta}_1\\ \ddot{\theta}_2\end{bmatrix} + \begin{bmatrix}\frac{63}{1000}\dot{\theta}_2\sin(\theta_2) & -\frac{63}{2000}\dot{\theta}_2\sin(\theta_2)\\ \frac{63}{1000}\dot{\theta}_1\sin(\theta_2) & 0\end{bmatrix}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix} + \begin{bmatrix}0\\ 0\end{bmatrix}. \quad (2.14)$$

Uma vez que o robô D2 foi criado, podemos plotar o dispositivo em qualquer combinação de ângulos através da função *plot.* Caso a entrada da função *plot* seja uma sequência de ângulos, então essa resultará em uma movimentação do dispositivo. A Figura 2.4 é resultado da utilização do código *"D2.plot([0 0])"* e representa o dispositivo na sua posição zero.

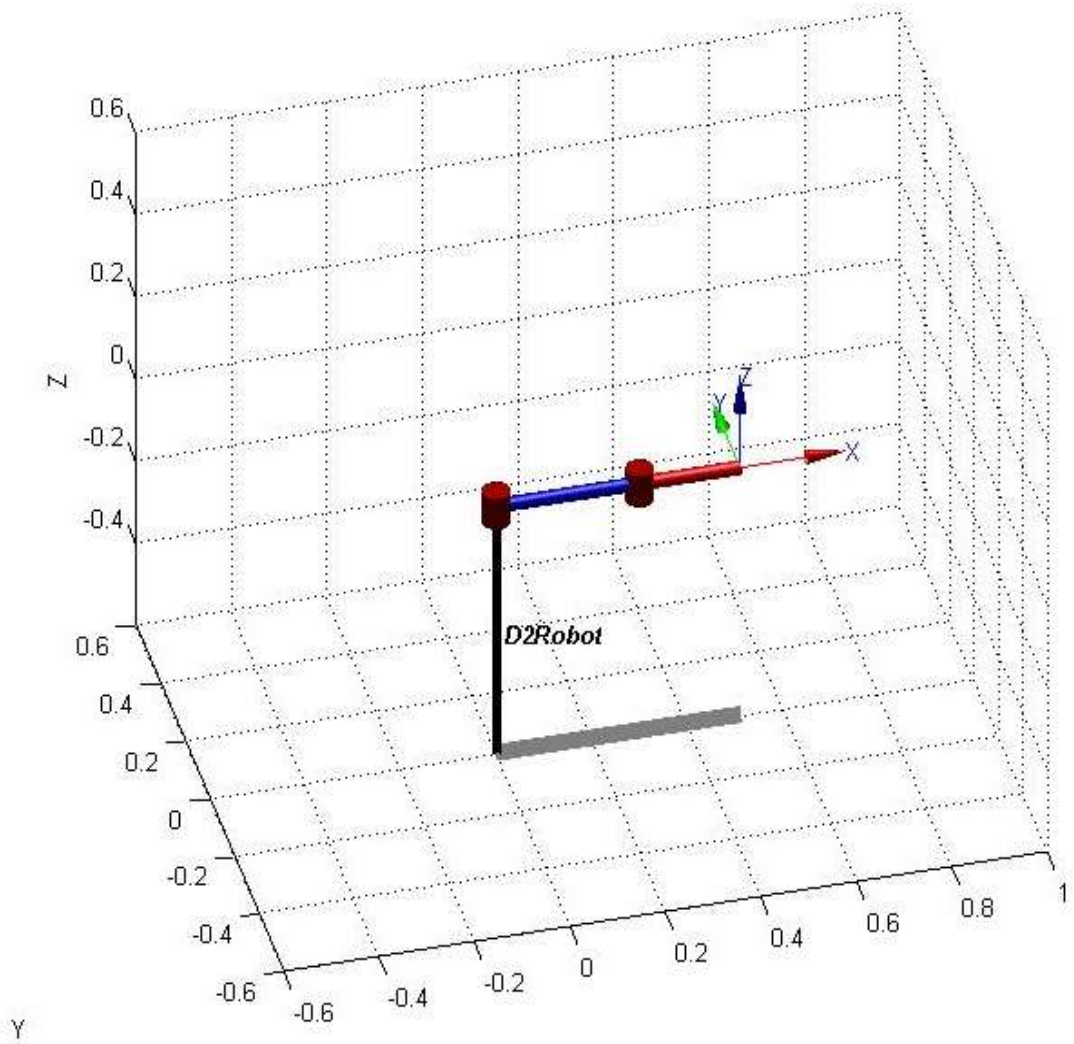

Figura 2.4 - Representação Gráfica de Manipulador Usando Robotic Toolbox (Próprio Autor)

Como visto na Seção 2.1.2, o modelo dinâmico direto tem a característica de simular o comportamento de um manipulador. A Figura 2.5 exemplifica o método desenvolvido pelo Robotic Toolbox para simular o modelo dinâmico direto do robô D2 definido anteriormente.

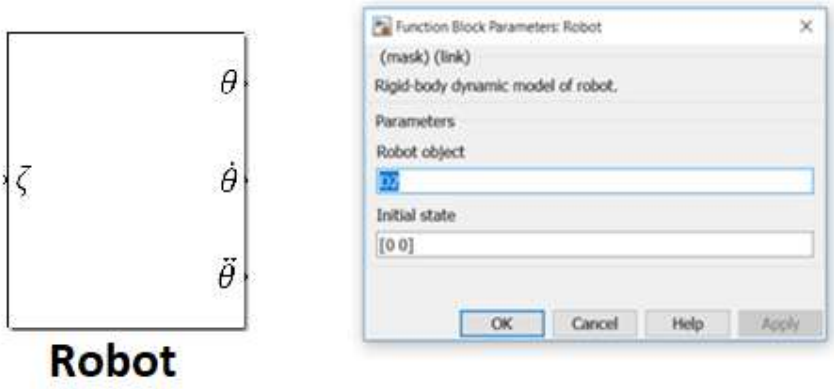

Figura 2.5 - Modelo Dinâmico Direto do Robô D2 (Próprio Autor)

A Figura 2.5 mostra o modelo dinâmico direto do robô D2 pronto para ser utilizado no ambiente de simulação Simulink. O sinal ζ representa um vetor de dimensão 2 que contém os torques a serem aplicados em cada uma das duas juntas do dispositivo. Os sinais $\theta, \dot{\theta}, \ddot{\theta}$ também possuem dimensão 2 e representam as posições, velocidades e acelerações angulares de cada junta quando o robô recebe o sinal de torque ζ. A caixa de mensagem à direita aparece quando se clica duas vezes sobre o bloco *"Robot"* e foi carregada com as informações dinâmicas do robô D2 alem da posição inicial do mesmo.

Outro recurso implementado no Robotic Toolbox é a criação do modelo dinâmico inverso do dispositivo. Como será visto no Capítulo seguinte, o modelo dinâmico inverso é muito utilizado para aplicação de uma técnica de controle muito popular em robótica. A Figura 2.6 mostra como utilizar o modelo dinâmico inverso do robô R2 definido anteriormente.

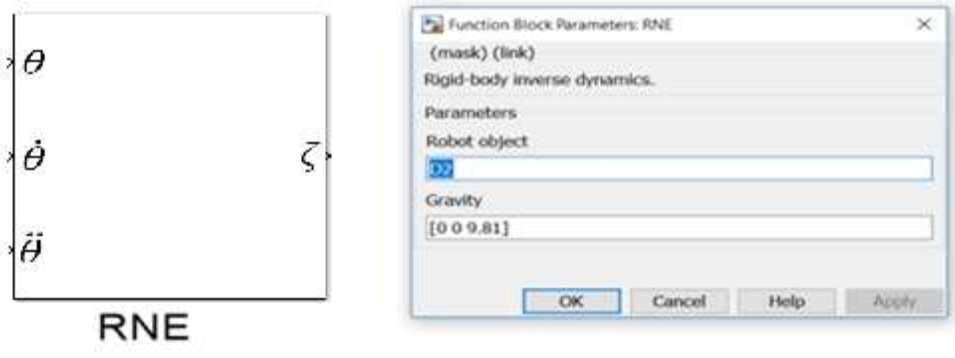

Figura 2.6 - Modelo Dinâmico Inverso do Robô R2 (Próprio Autor)

O bloco *"RNE"* da Figura 2.6 computa o torque requerido para que o robô R2 siga uma trajetória. Os sinais de $\theta, \dot{\theta}, \ddot{\theta}$ são vetores de dimensão 2 e representa a posição, velocidade e aceleração desejada que cada junta do robô siga, e ζ também possui dimensão 2

representa o torque que cada junta deve executar para que a trajetória desejada por $\theta, \dot{\theta}, \ddot{\theta}$ aconteça.

De maneira análoga ao que foi feito com o Robô R2, o apêndice A mostra o código desenvolvido e aplicado no Robotic Toolbox para obter o modelo dinâmico do manipulador industrial Denso. As topologias de controle apresentadas nos Capítulo 3 e 4 utilizam o modelo extraído do apêndice A, mas devido ao tamanho das informações, não será colocado nesse material o modelo dinâmico do robô Denso.

2.4 GERAÇÃO DE TRAJETÓRIA

O objetivo da geração de trajetória é gerar as entradas de referências para o manipulador de modo que certifique que o manipulador obteve a posição desejada. Para isso o planejamento de trajetória gera uma sequência temporal de posições segundo um interpolador.

De acordo com Sciviacco e Siciliano (2000), o requerimento mínimo de um manipulador é que esse possa se mover de uma posição inicial para uma posição final. Essa transação deve ser caracterizada por movimentos que não requerem um sinal de torque maior do que o atuador pode fornecer, e também não deve excitar dinâmicas não modeladas. Portanto, é necessário que a trajetória seja planejada e executada de modo que ocorram transações suaves entre a posição inicial e posição final.

2.4.1 INTERPOLADOR DE TERCEIRA ORDEM

A maneira mais simples de planejar uma trajetória é utilizando um interpolador polinomial de terceira ordem, onde a posição é genericamente expressa por: $q(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$. Nesse caso, a velocidade do manipulador se comportaria como uma parábola dado por $\dot{q}(t) = 3a_3 t^2 + 2a_2 t + a_1$, e aceleração possui um perfil afim dado por $\ddot{q}(t) = 6a_3 t + 2a_2$. Como foi deduzido um interpolador de terceira ordem genérico, é possível projetar as constantes de modo que satisfaça as condições de posição inicial e final, além de velocidade inicial e final.

Das deduções acima, é possível constatar que para $t = 0$ a posição encontrada é $q(0) = a_0$, e a velocidade encontrada é $\dot{q}(0) = a_1$. Desse modo, para o interpolador de terceira ordem é possível determinar as constantes do polinômio a partir do conjunto de equações (2.15).

$$\begin{gathered} a_0 = q_i, \\ a_1 = \dot{q}_i, \\ a_3 {t_f}^3 + a_2 {t_f}^2 + a_1 t_f + a_0 = q_{f,} \\ 3a_3 {t_f}^2 + 2a_2 t_f + a_1 = \dot{q}_f. \end{gathered} \tag{2.15}$$

Na Equação 2.15 t_f, q_i e $q_{f,}$ são respectivamente tempo de trajetória, posição inicial e posição final da trajetóra. Um problema recorrente do uso de um interpolador de terceira ordem é que não é possível explicitar a aceleração inicial e aceleração final, e isso pode causar a necessidade de conjugado de partida elevado. A Figura 2.7 é retirada de Sciviacco e Siciliano (2000) e mostram a utilização de um interpolador polinomial de terceira ordem para a execução de trajetória. As condições de projeto utilizadas são $q_i = 0$, $q_f = \pi$, t_f=1, e $\dot{q}_i = \dot{q}_f = 0$. É notório que a aceleração inicial e final para essa trajetória são muito elevadas, fato esse que pode causar impactos direto no manipulador, portanto será utilizado uma trajetória que tem como característica de projeto explicitar a aceleração inicial e final.

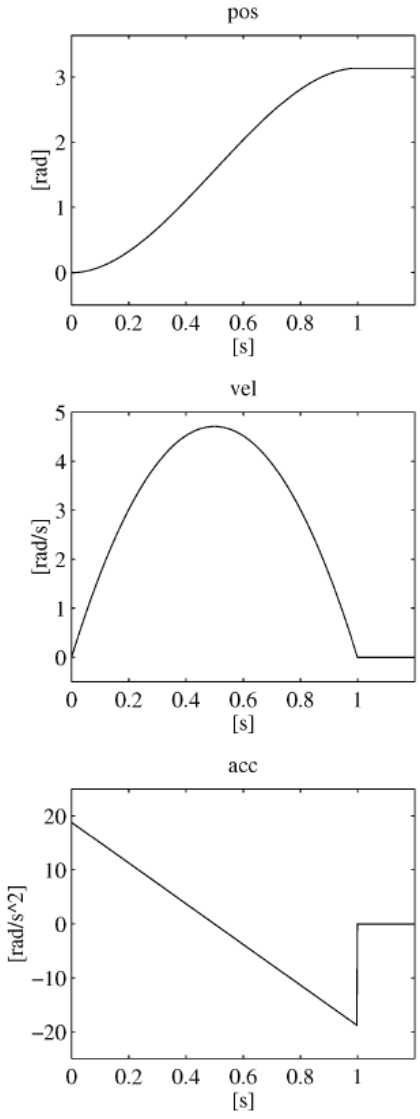

Figura 2.7 - Posição, Velocidade e Aceleração de um Interpolador de Terceira Ordem (Sciviacco e Siciliano 2000).

2.4.2 INTERPOLADOR DE QUINTA ORDEM

Uma maneira mais sofisticada de planejar uma trajetória é utilizando um interpolador polinomial de quinta ordem, onde a posição é genericamente expressa por: $q(t) = a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0$. Nesse caso, a velocidade do manipulador se comportaria de acordo com a seguinte relação $\dot{q}(t) = 5a_5t^4 + 4a_4t^3 + 3a_3t^2 + 2a_2t + a_1$, e aceleração possui um comportamento dado por $\ddot{q}(t) = 20a_5t^3 + 12a_4t^2 + 6a_3t + 2a_2$. De maneira análoga ao desenvolvimento feito para o interpolador de terceira ordem, é possível projetar as constantes de modo que satisfaça as condições de posição inicial e final, velocidade inicial e final, e aceleração inicial e final.

$$\begin{gathered} a_0 = q_i, \\ a_1 = \dot{q}_i, \\ 2a_2 = \ddot{q}_i, \\ a_5{t_f}^5 + a_4{t_f}^4 + a_3{t_f}^3 + a_2{t_f}^2 + a_1t_f + a_0 = q_f, \\ 5a_5{t_f}^4 + 4a_3{t_f}^3 + 3a_3{t_f}^2 + 2a_2t_f + a_1 = \dot{q}_{f,} \\ 20a_5{t_f}^3 + 12a_4{t_f}^2 + 6a_3t_f + 2a_2 = \ddot{q}_f. \end{gathered} \tag{2.16}$$

Para o presente projeto foi requerido que a geração de trajetória satisfaça as seguintes condições: $q_i = \dot{q}_i = \ddot{q}_i = \dot{q}_f = \ddot{q}_f = 0$, $q_f = 1$ e $t_f = 1$. Substituindo os requisitos de projeto nas equações acima, é chegado no seguinte conjunto de condições de projeto:

$$\begin{gathered} a_0 = 0, \\ a_1 = 0, \\ 2a_2 = 0, \\ a_5 + a_4 + a_3 + a_2 + a_1 + a_0 = 1, \\ 5a_5 + 4a_3 + 3a_3 + 2a_2 + a_1 = 0, \\ 20a_5 + 12a_4 + 6a_3 + 2a_2 = 0. \end{gathered} \tag{2.17}$$

que pode ser reescrito no formato matricial dado em (2.18):

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 \\ 0 & 0 & 2 & 6 & 12 & 20 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}. \tag{2.18}$$

A solução para a Equação (2.18) é: $a_0 = 0;\ a_1 = 0; a_2 = 0; a_3 = 10; a_4 = -15; a_5 = 6$. Portanto, dado que uma junta do robô saia da posição 0 e deseja atingir a posição 1 em um tempo de 1 segundo, então essa junta passará por pontos intermediários dado pela seguinte equação: $q(t) = 6t^5 - 15t^4 + 10t^3$. De forma análoga a velocidade, a aceleração de cada junta são dadas pelos seguintes polinômios respectivamente: $\dot{q}(t) = 30t^4 - 60t^3 + 30t^2$, e

$\ddot{q}(t) = 120t^3 - 180t^2 + 60t$ (Corke 2011). A posição, velocidade e aceleração angular resultante desse planejamento de trajetória são apresentados na Figura 2.8.

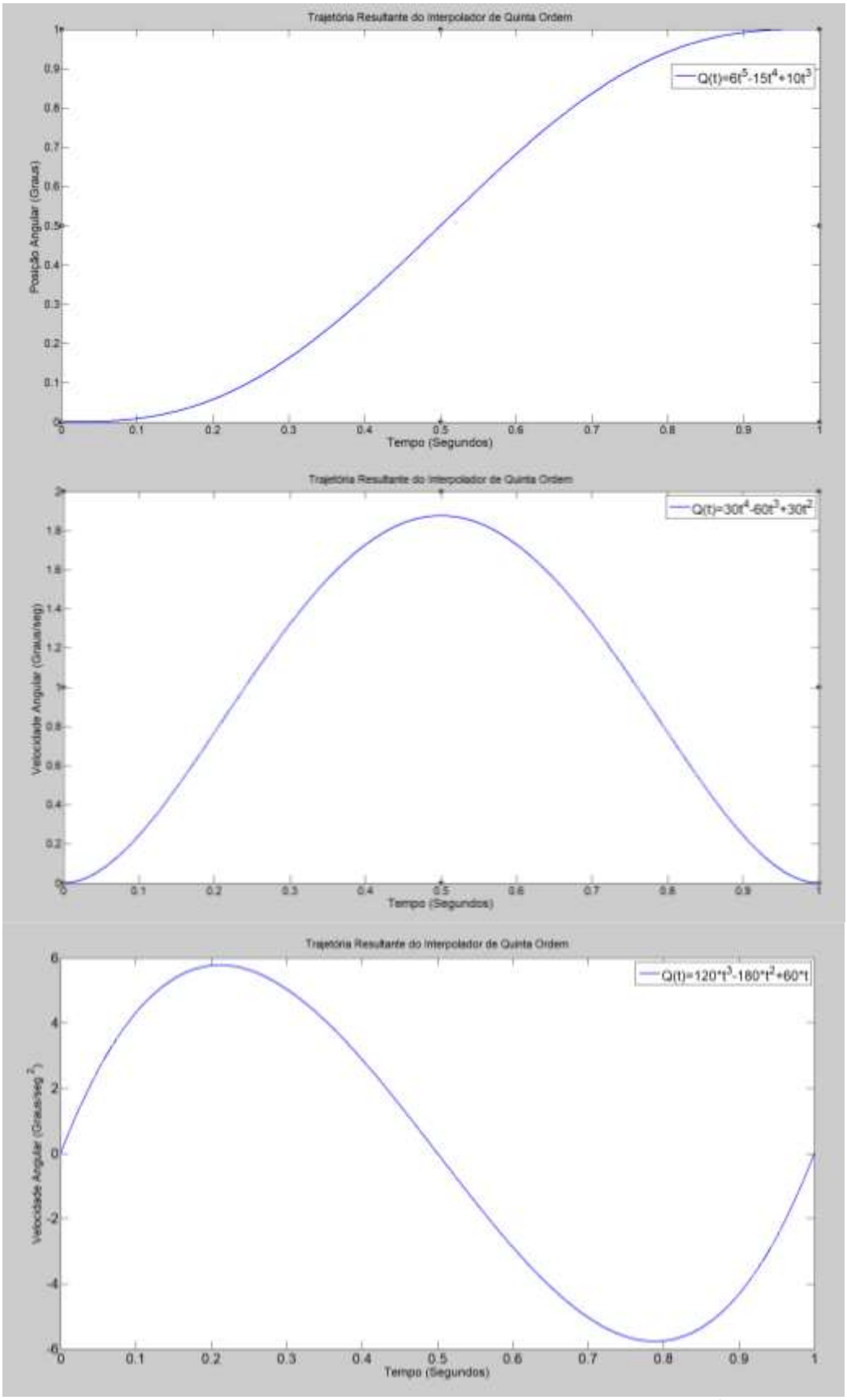

Figura 2.8 - Posição, Velocidade e Aceleração de um Interpolador Unitario de Quinta Ordem (Próprio Autor)

Embora a Figura 2.8 satisfaça todos os requisitos de projeto, é necessário que ajustes sejam feitos para solucionar dois problemas: a trajetória deve sair da posição atual e atingir a posição final desejada (que nem sempre é igual a 1 grau); a trajetória deve ser efetuada em um período de tempo desejado (que nem sempre é igual a 1 segundo).

Para solucionar os dois problemas acima criados, sem perder as características das curvas apresentadas acima, foram propostas duas modificações nos interpoladores de trajetória propostos acima. A primeira é a multiplicação de todo o polinômio por ΔQ, onde ΔQ é a diferença entre a posição real e a posição desejada e não necessariamente precisa ser 1, isto é, $\Delta Q = Q - Q_d$. A segunda é a substuição de t por Δt, onde Δt é a diferença ente o tempo total para que a trajetória seja executada e o tempo atual, isto é, $\Delta t = t_{total} - t$.

Desse modo, o interpolador polinomial de quinta ordem capaz de proporcionar uma variação ΔQ na posição de uma junta em um tempo t_{total} de maneira suave e que garanta que as velocidades e acelerações iniciais e finais sejam nulas é dada pela Equação (2.19).

$$\begin{aligned} Q(t) &= 6\Delta Q\Delta t^5 - 15\Delta Q\Delta t^4 + 10\Delta Q\Delta t^3, \\ \dot{Q}(t) &= 30\Delta Q\Delta t^4 - 60\Delta Q\Delta t^3 + 30\Delta Q\Delta t^2, \\ \ddot{Q}(t) &= 120\Delta Q\Delta t^3 - 180\Delta Q\Delta t^2 + 60\Delta Q\Delta t^3. \end{aligned} \tag{2.19}$$

A Figura 2.9 mostra a posição, velocidade e aceleração angular resultante do interpolador acima deduzido. Para mostrar a eficiência do método, a Figura 2.9 mostra uma trajetória saindo de 0 º até 60º em um tempo total de 15 segundos.

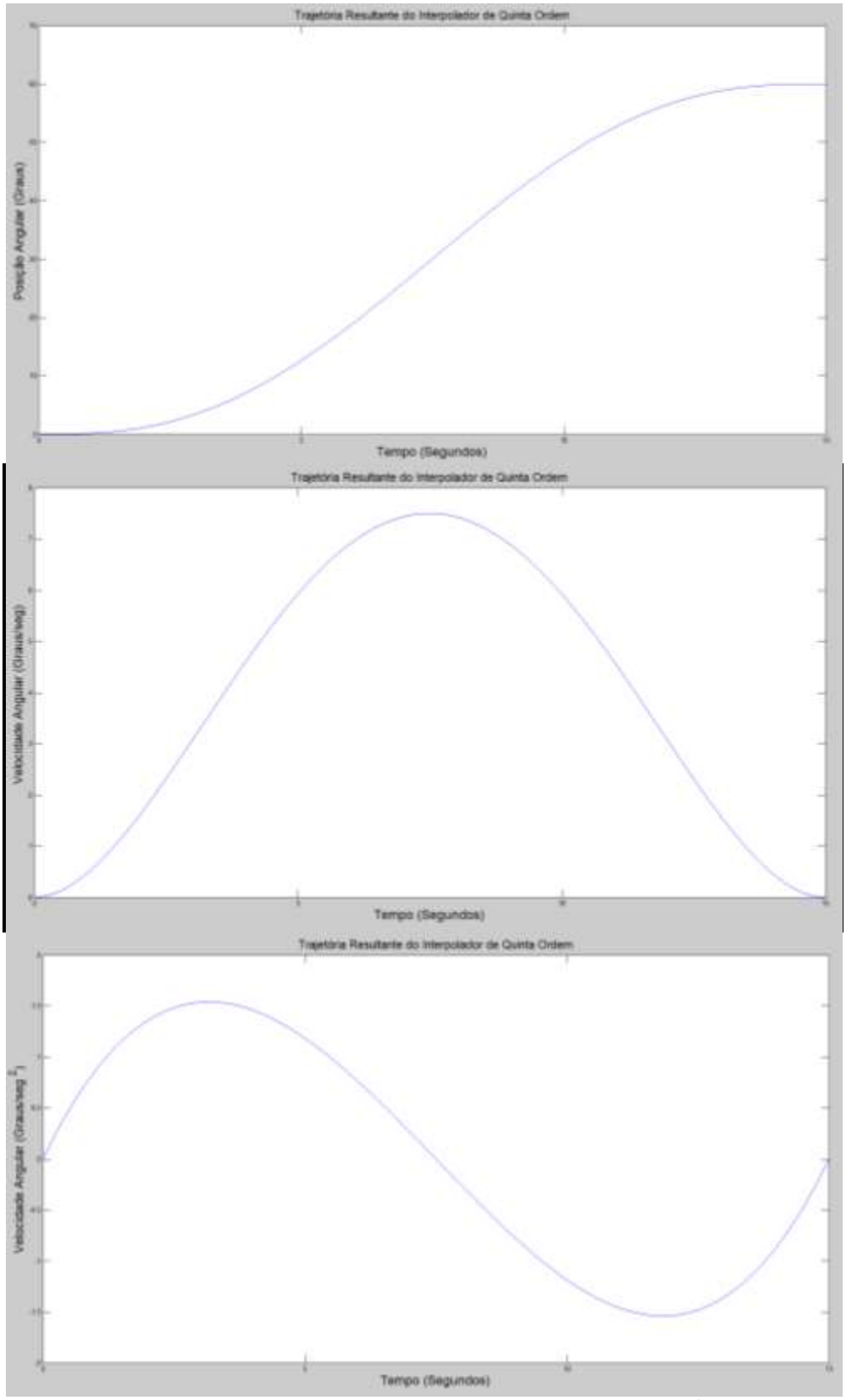

Figura 2.9 - Posição, Velocidade e Aceleração de um Interpolador de Quinta Ordem (Próprio Autor)

A Figura 2.10 mostra como a geração de trajetória foi implementada em Simulink no manipulador industrial Denso VP6242.

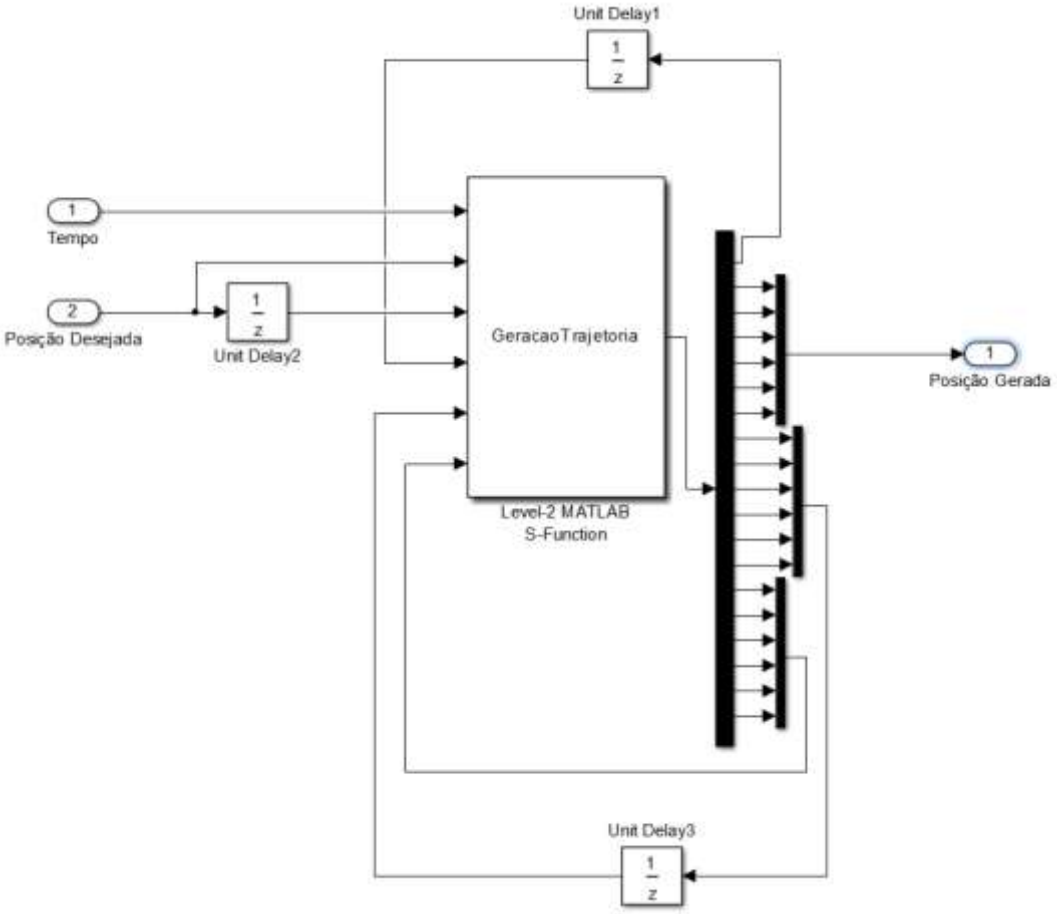

Figura 2.10 - Implementação de Gerador de Trajetória em Ambiente Simulink (Próprio Autor)

O diagrama de bloco da implementação de geração de trajetória possui dois sinais de entrada e um sinal de saída. A variável “Tempo” é um escalar e informa ao código o tempo (em segundos) requerido para que o manipulador exerça a movimentação. A variável “Posição Desejada” é um vetor com seis informações informando os seis ângulos desejados por cada junta (em graus). A variável “Posição Gerada” é um vetor com seis informações que informa a posição gerada pelo interpolador de quinta ordem. O bloco “GeracaoTrajetoria” possui a lógica do interpolador, e o arquivo .m que é executado dentro desse bloco é informado no apêndice B.

As entradas do bloco “GeracaoTrajetoria” (de cima para baixo) representam: tempo, posição desejada, posição desejada com atraso, flag, delta Q delay e Q zero. As saídas são respectivamente: flag, posição gerada, variação de posição a ser gerada, posição antes do início da trajetória.

Quando há a variação da posição desejada, o algoritmo salva a posição inicial, a variação de posição desejada e faz uso do interpolador de quinta ordem para informar qual o próximo ponto para que a trajetória ocorra no tempo desejado. Após o tempo requisitado para

a execução da trajetória o algoritmo mantém a posição atual e só volta a gerar pontos intermediários caso haja variação na posição desejada. O algoritmo utilizado encontra-se no apêndice B do presente trabalho.

A Figura 2.11 mostra o sinal de saída do gerador de trajetória, isto é, o sinal “Posição Gerada” para uma trajetória em que a junta sai da posição 0° e atinge a posição 60° em 15 segundos.

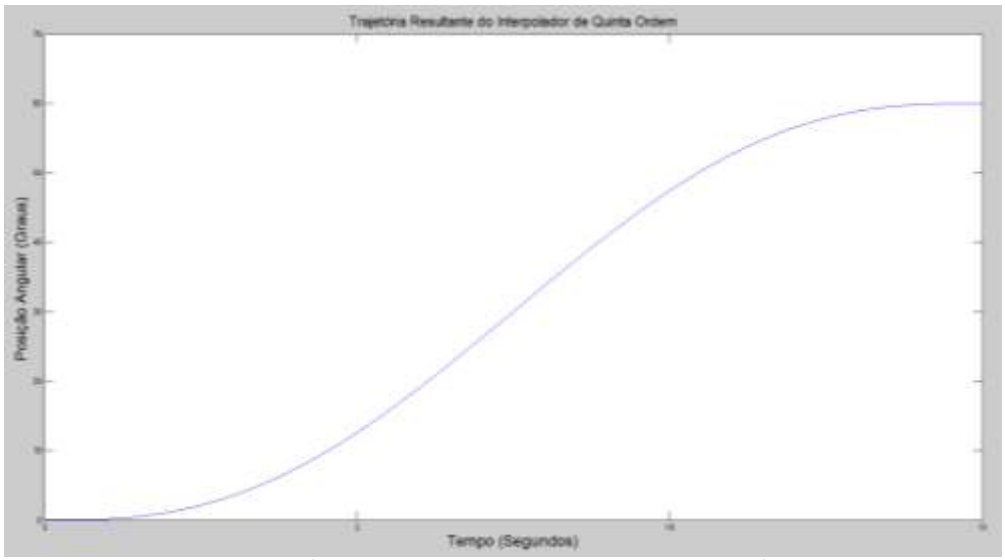

Figure 2.11 - Trajetória Resultante do Interpolador de Quinta Ordem (Próprio Autor)

2.5 CONSIDERAÇÕES FINAIS SOBRE MODELAGEM DE MANIPULADORES ROBÓTICOS E GERAÇÃO DE TRAJETÓRIA

O modelo dinâmico de um manipulador robótico possui suma importância no projeto de controle para o sistema dinâmico de um robô. Esse Capítulo discutiu duas maneiras de obter o modelo dinâmico de um manipulador robótico alem de abordar maneiras de gerar trajetórias suaves para um manipulador robótico.

Inicialmente foi abordado o método clássico de modelagem dinâmica de um manipulador utilizando o método recursivo de Newton Euler. Apesar de existir outras metodologias que proporcionam a modelagem dinâmica de um manipulador robótico, o método de Newton Euler é muito difundido na literatura pelo fato de ser recursivo e consequentemente ser muito utilizado para elaboração de algoritmos. A segunda abordagem utilizada para modelgem de manipulador robótico é utilizando o Robotic Toolbox para Matlab desenvolvido por Corke (1996). Esse método trata-se de um Toolbox que faz uso do método recursivo de Newton Euler para fornecer ao usuário o modelo dinâmico de um manipulador

robótico. Nesse Capítulo foi abordado um exemplo de utilização do Robotic Toolbox para obtenção do modelo dinâmico de um robô com 2 graus de liberdade e posteriormente foi desenvolvido o modelo dinâmico do manipulador robótico Denso VP6242 utilizando o Toolbox.

Esse Capítulo também abordou a geração de trajetória suaves para juntas robóticas. Foi constatado que a utilização de interpolador de terceira ordem não é muito eficiente para a elaboração de trajetórias suáveis pois esse método não permite definir aceleração inicial nem final das juntas. Por fim, foi utilizado um interpolador de quinta ordem para definição da trajetória de cada junta considerando restrições de posição inicial e final desejada, velocidade inicial e final desejada e aceleração inicial e final desejada.

3. CONTROLE DE JUNTAS POR TORQUE COMPUTADO

No Capítulo 2 foi abordado como obter o modelo dinâmico de um manipulador. O objetivo desse capítulo é introduzir o controle por torque computado, sendo essa uma arquitetura de controle que utiliza o modelo dinâmico dos manipuladores para fazer o controle de posição do mesmo. Essa classe de controle é baseada no modelo dinâmico indireto do manipulador, portanto essa arquitetura tem alto desempenho quando o modelo dinâmico é conhecido com precisão.

3.1 SISTEMA NÃO LINEAR COM ACOPLAMENTO

De acordo com a expressão deduzida (2.1) um manipulador possui acoplamento entre as entradas e as saídas desde que as características não sejam puramente diagonais. Portanto um manipulador com acoplamento e n graus de liberdade não pode ser visto como o conjunto de n equações independentes, mas deve ser interpretado como o conjunto de n equações não lineares e com acoplamento entre si. Desse modo, para controlar a posição de um manipulador, duas possíveis alternativas podem ser utilizadas: fazer a linearização das equações em diferentes pontos de operação, ou utilizar algum controlador não linear para compensar as não linearidades existentes.

Embora trabalhar com modelos linearizados localmente seja uma técnica bastante difundida na literatura, como encontrada em Ogata (2010), Nise (2010) e Dorf e Bishop (2011), quando se trata de manipuladores essa técnica não é bem vista pelos seguintes motivos:

- A combinação de modelos lineares locais necessários para aproximar o modelo não linear global em modelos lineares é muito grande, pois cada junta aumenta exponencialmente a quantidade de modelos lineares locais necessários para cobrir toda a área de atuação do sistema não linear;
- O modelo dinâmico inverso algébrico é difícil de ser encontrado para manipuladores com vários graus de liberdade;
- O modelo dinâmico inverso não linear (2.1) tem uma complexidade elevada para ser linearizado.

Em contrapartida, uma arquitetura de controle não linear baseado no modelo dinâmico inverso do manipulador pode ser utilizada para cancelar as não linearidades e efeito de

acoplamento de manipuladores. A essa topologia damos o nome de controle por torque computado, ou ainda controle Feedforward ou Linearização por Realimentação (para fins de unificação, nesse trabalho chamaremos de torque computado). De acordo com Murray, Li, Sastry (1994), ao eliminar as não linearidades e acoplamentos, controladores lineares tendem a ter uma resposta muita satisfatória para manipuladores robóticos.

3.2 TORQUE COMPUTADO

De acordo com Murray, Li, Sastry (1994) o problema de controle de posição ou trajetória se resume em escolher quais devem ser os torques aplicados em cada junta para que o robô siga a trajetória desejada. A Equação (2.1) relaciona o torque aplicado em cada junta com as posições, velocidades e aceleração angular, portanto se fosse possível obter o modelo perfeito de um manipulador e conseguir assegurar que $\theta(0) = \theta_d(0)$, e $\dot{\theta}(0) = \dot{\theta}_d(0)$, então o torque requerido para que o manipulador seguisse a trajetória desejada seria dado por:

$$\zeta = M(\theta_d)\ddot{\theta}_d + V(\theta_d, \dot{\theta}_d)\dot{\theta}_d + F(\dot{\theta}_d) + G(\theta_d), \qquad (3.1)$$

onde, θ_d, $\dot{\theta}_d$ e $\ddot{\theta}_d$ são respectivamente a posição angular desejada, velocidade angular desejada e aceleração angular desejada(Murray, Li, Sastry 1994).

Com o auxílio do modelo dinâmico direto e inverso é possível exemplificar essa topologia de controle com blocos de Simulink obtidos a partir do apêndice A. Considere a Figura 3.1:

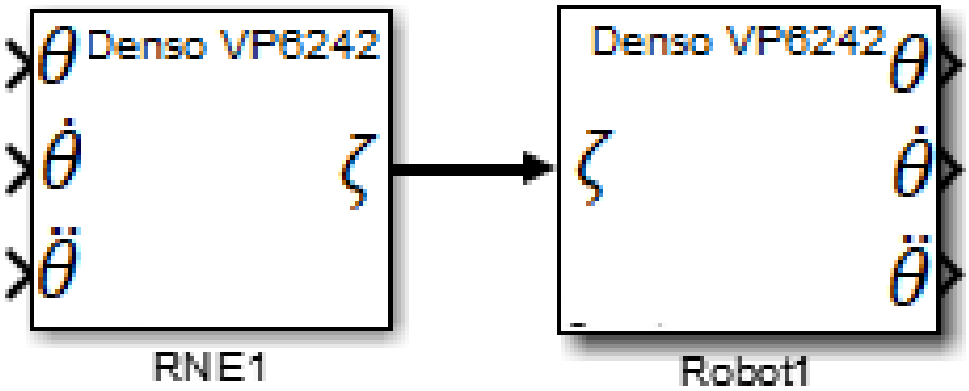

Figura 3.1 - Arquitetura de Torque Computado (Próprio Autor)

O bloco “Robot” representa um manipulador robótico, e o bloco “RNE” possui o modelo dinâmico inverso do mesmo manipulador. Quando uma determinada trajetória (posição, velocidade e aceleração angular) for injetada no bloco “RNE”, então da saída “Q” sairá o troque computado para que o robô siga a trajetória desejada.

Essa estratégia é chamada de controle em malha aberta e não é muito robusta. Caso $\theta(0) \neq \theta_d(0)$ então essa arquitetura de controle não corrigirá o erro, e não é garantido que o erro de trajetória sempre permanecerá sendo o erro entre a posição inicial real e a posição inicial desejada (Murray, Li, Sastry 1994). Essa arquitetura de controle também só garante a estabilidade do sistema para o caso em que o modelo dinâmico inverso descreve perfeitamente o robô; caso haja qualquer imperfeição no modelo, essa arquitetura não garante a estabilidade nem o erro de trajetória nula.

A Figura 3.2 mostra uma simulação da arquitetura de torque computado considerando uma imprecisão nas condições iniciais de 1°, isto é, a posição inicial real é de $[1° \; 1° \; 1° \; -91° \; 1° \; -91°]$, enquanto a posição estimada é de $[0° \; 0° \; 0° \; -90° \; 0° \; -90°]$.

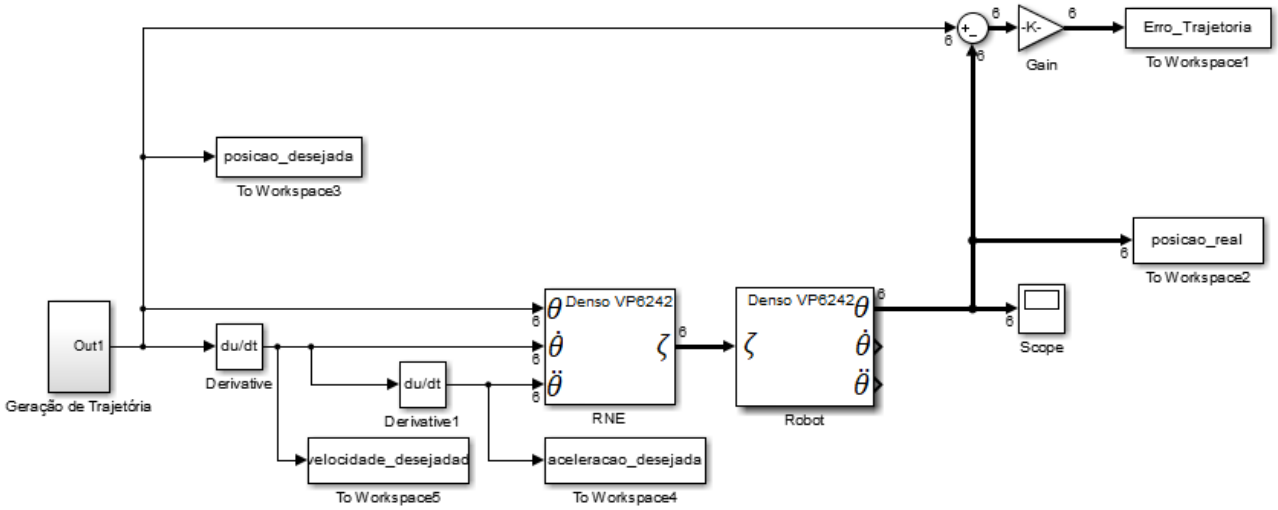

Figura 3.2 - Aplicação de Torque Computado (Próprio Autor)

O bloco de geração de trajetória é descrito na Seção 2.2.4 e no apêndice B desse material. A Figura 3.3 mostra as posições, velocidades e acelerações desejadas, e posteriormente as Figuras 3.4 e 3.5 mostram a posição real e o erro de trajetória do sistema.

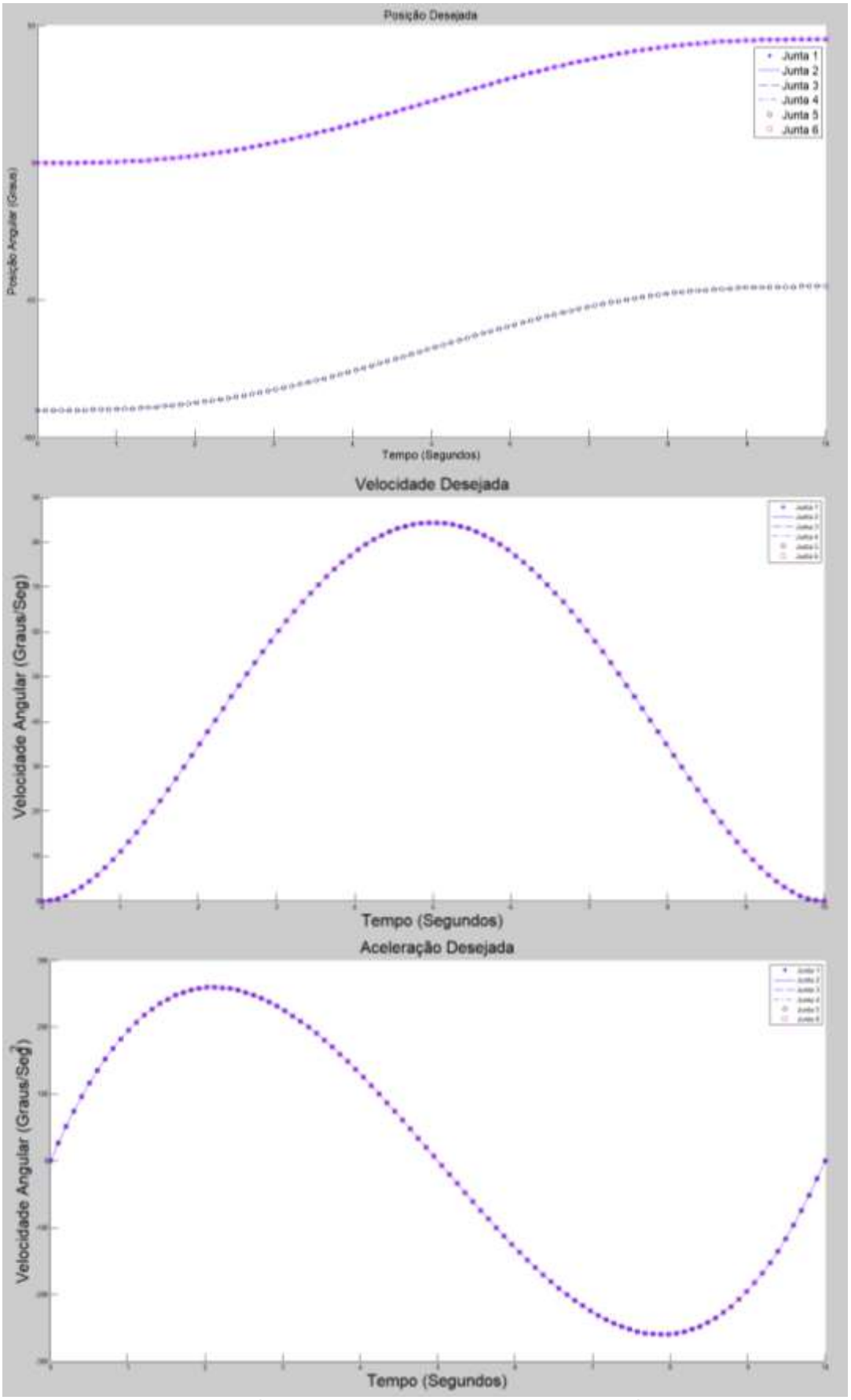

Figura 3.3 - Trajetória Desejada para Controle por Torque Computado (Próprio Autor)

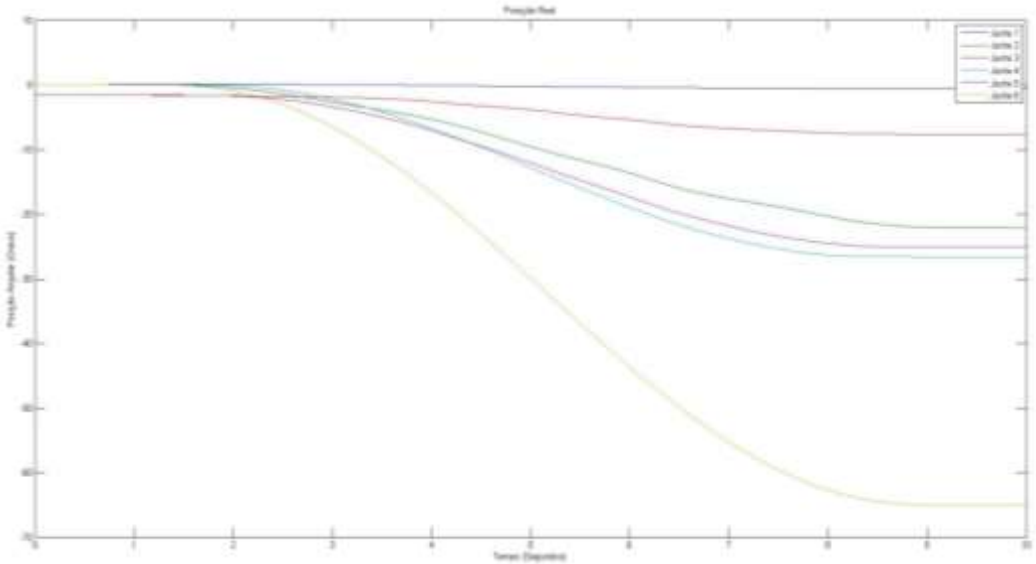

Figura 3.4 - Posição Real do Controle por Torque Computado (Próprio Autor)

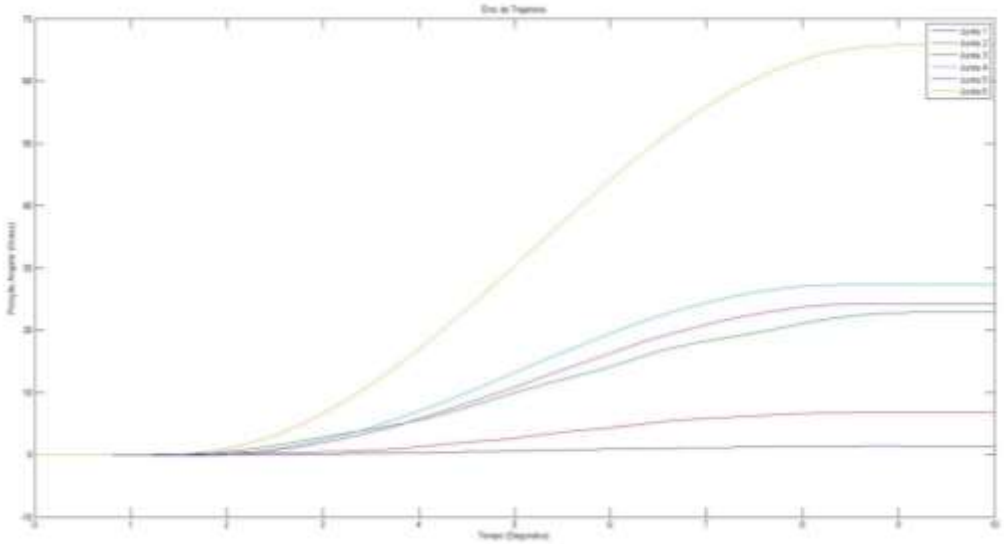

Figura 3.5 - Erro de Trajetória do Controle por Torque Computado (Próprio Autor)

Como visto nas Figuras 3.4 e 3.5, o controle por torque computado é muito sensível à variação de posição inicial e não possui resultados satisfatórios. Embora cada junta tenha iniciado com diferença de apenas 1° da posição desejada, esse erro se ampliou e atingiu o erro de trajetória maior que 60° na junta 6 ao final de uma trajetória de 10 segundos.

Embora o controle em malha aberta não seja aplicável em sistemas reais (devido ao seu baixo desempenho mediante a pequenas variações de posição inicial), essa topologia pode ser utilizada para computar o torque necessário para cancelar as não linearidades e superar a inércia dos elos.

Considere o sinal de controle (3.2):

$$\zeta = \widehat{M}(\theta)\ddot{\theta}_d + \hat{V}(\theta,\dot{\theta})\dot{\theta} + \hat{F}(\dot{\theta}) + \hat{G}(\theta). \tag{3.2}$$

onde $\widehat{M}$, $\hat{V}$, $\hat{F}$ e $\hat{G}$ são respectivamente as matrizes inerciais, de Coriolis e centrifugal, de atrito e gravitacional estimadas, e são dados por: $\widehat{M}(\theta) = M(\theta) - \widetilde{M}(\theta)$, $\hat{V}(\theta,\dot{\theta}) = V(\theta,\dot{\theta}) - \tilde{V}(\theta,\dot{\theta})$, $\hat{F}(\dot{\theta}) = F(\dot{\theta}) - \tilde{F}(\dot{\theta})$ e $\hat{G}(\theta) = G(\theta) - \tilde{G}(\theta)$. Se (3.2) e (2.1) for igualado, ou seja, se o torque computado pela Equação (3.2) for injetado no manipulador descrito por (2.1), se obtém (3.3):

$$M(\theta)\ddot{\theta}_d + V(\theta,\dot{\theta})\dot{\theta} + F(\dot{\theta}) + G(\theta) = \zeta = \widehat{M}(\theta)\ddot{\theta} + \hat{V}(\theta,\dot{\theta})\dot{\theta} + \hat{F}(\dot{\theta}) + \hat{G}(\theta). \tag{3.3}$$

Considerando que $\widetilde{M}(\theta) = \tilde{V}(\theta,\dot{\theta}) = \tilde{F}(\dot{\theta}) = \tilde{G}(\theta) = 0$, isto é, considerando que a lei de controle (3.2) consiga realizar a linearização por realimentação perfeitamente, então o sistema (3.3) seria representado por: $M(\theta)\ddot{\theta}_d = \widehat{M}(\theta)\ddot{\theta}$. Devido ao fato da matriz $M(\theta)$ ser uniformemente positiva definida, é obtido que $\ddot{\theta}_d = \ddot{\theta}$. Contudo, ainda assim o manipulador não corrigiria erros de condições iniciais (Murray, Li, Sastry 1994).

O modelo dinâmico em espaço de estados para o sistema (2.1) mediante a lei de controle $\zeta = \widehat{M}(\theta)\ddot{\theta}_d + \hat{V}(\theta,\dot{\theta})\dot{\theta} + \hat{F}(\dot{\theta}) + \hat{G}(\theta)$, isso é, o modelo dinâmico do sistema (2.1) mediante a linearização por realimentação é dado por (3.4):

$$\begin{bmatrix} \dot{\varepsilon} \\ \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \varepsilon \\ e \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u, \tag{3.4}$$

$$y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \varepsilon \\ e \\ \dot{e} \end{bmatrix}.$$

A representação (3.4) é chamada de modelo dinâmico do erro do sistema, e $\varepsilon = \int e dt$. A seguir será apresentada uma abordagem que utilize um compensador PID junto ao controle por torque computado para compensar as pequenas imperfeições de modelagem e de posição inicial.

3.2.1 TORQUE COMPUTADO COM COMPENSADOR DE REALIMENTAÇÃO PID

De acordo com Murray, Li e Sastry (1994) e Lewis, Dawson e Abdallah (2004), a lei de controle (3.3) pode ser melhorada para a lei de controle (3.5), onde $e = \theta - \theta_d$ é o erro de

trajetória e $\dot{e} = \dot{\theta} - \dot{\theta}_d$ é a sua derivada temporal e $\dot{\varepsilon} = e$, isso é, ε é a integral do erro, então a lei de controle (3.5) é chamada de torque computado com compensador PID:

$$\zeta = \hat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) + \hat{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \hat{F}\left(\dot{\theta}\right) + \hat{G}(\theta), \tag{3.5}$$

sendo que:

$$K_v = diag\left(k_{v_i}\right), \tag{3.6}$$

$$K_p = diag\left(k_{p_i}\right), \tag{3.7}$$

$$K_i = diag\left(k_{i_i}\right). \tag{3.8}$$

Segundo Lewis, Dawson e Abdallah (2004) o sistema (2.1) mediante a arquitetura de controle (3.5) possui o polinômio característico do sistema dinâmico do erro de trajetória conforme definido em (3.9) - (3.13):

$$M(\theta)\ddot{\theta} + V\left(\theta,\dot{\theta}\right)\dot{\theta} + F\left(\dot{\theta}\right) + G(\theta) = \hat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) + \hat{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \hat{F}\left(\dot{\theta}\right) + \hat{G}(\theta), \tag{3.9}$$

$$V\left(\theta,\dot{\theta}\right)\dot{\theta} - \hat{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + F\left(\dot{\theta}\right) - \hat{F}\left(\dot{\theta}\right) + G(\theta) - \hat{G}(\theta) = \hat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) - M(\theta)\ddot{\theta}, \tag{3.10}$$

$$\tilde{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \tilde{F}\left(\dot{\theta}\right) + \tilde{G}(\theta) = \hat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) - \hat{M}(\theta)\ddot{\theta} - \tilde{M}(\theta)\ddot{\theta}, \tag{3.11}$$

$$\tilde{M}(\theta)\ddot{\theta} + \tilde{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \tilde{F}\left(\dot{\theta}\right) + \tilde{G}(\theta) = \hat{M}(\theta)\left(\ddot{e} - K_v\dot{e} - K_p e - K_i\varepsilon\right), \tag{3.12}$$

$$\hat{M}^{-1}(\theta)\left[\tilde{M}(\theta)\ddot{\theta} + \tilde{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \tilde{F}\left(\dot{\theta}\right) + \tilde{G}(\theta)\right] = -\ddot{e} - K_v\dot{e} - K_p e - K_i\varepsilon. \tag{3.13}$$

Considerando que $\tilde{M}(\theta) = \tilde{V}\left(\theta,\dot{\theta}\right) = \tilde{F}\left(\dot{\theta}\right) = \tilde{M}(\theta) = 0$, então o sistema fica sendo representado por (3.14):

$$\ddot{e} + K_v\dot{e} + K_p e + K_i\varepsilon = 0, \tag{3.14}$$

e em espaço de estados é representado por (3.15):

$$\begin{bmatrix} e \\ \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -K_i & -K_p & -K_v \end{bmatrix} \begin{bmatrix} \varepsilon \\ e \\ \dot{e} \end{bmatrix} \tag{3.15}$$

Caso $\tilde{M}(\theta), \tilde{V}(\theta, \dot{\theta}), \tilde{F}(\dot{\theta}), \tilde{G}(\theta) \neq 0$, então o torque gerado pelo erro paramétrico é interpretado como um distúrbio externo ao sistema. De acordo com Lewis, Dawson e Abdallah (2004), o sistema dinâmico do erro de trajetória de um robô com distúrbio é dado por:

$$\begin{bmatrix} e \\ \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -K_i & -K_p & -K_v \end{bmatrix} \begin{bmatrix} \varepsilon \\ e \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w, \tag{3.16}$$

$$y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \varepsilon \\ e \\ \dot{e} \end{bmatrix}.$$

O sinal w é um distúrbio que representa o torque exercido pelas dinâmicas não modeladas. Para que uma arquitetura de controle por torque computado tenha uma boa resposta, é necessário que a influência de w seja reduzida no sistema, ou que w seja diminuído. O escopo da pesquisa desse trabalho gira em torno de reduzir a influência que w causa no sistema. Nessa Seção, a tentativa utilizada para reduzir o efeito do sinal w é optando por valores elevados nos ganhos K_p, K_v e K_i. O sinal w pode ser facilmente medido em implementações práticas, para isso é necessário medir o sinal de torque real no robô (ζ) e o sinal de torque estimado pelo modelo ($\hat{\zeta}$) para uma mesma trajetória. Desse modo, o distúrbio é dado por: $w = \tilde{\zeta} = \zeta - \hat{\zeta}$.

Se considerarmos que $w = 0$, então é possível extrair do sistema descrito por (3.16) que:

$$u = \ddot{e} + K_v \dot{e} + K_p e + K_i \varepsilon. \tag{3.17}$$

É possível então, substituir $e = \dot{\varepsilon}$. Desse modo (3.17) pode ser escrito como:

$$u = \dddot{\varepsilon} + K_v \ddot{\varepsilon} + K_p \dot{\varepsilon} + K_i \varepsilon. \tag{3.18}$$

Consequentemente a função de transferência do sistema (3.18) fica sendo descrita por:

$$\frac{\varepsilon(s)}{U(s)} = \frac{1}{s^3 + K_v s^2 + K_p s + K_i}. \tag{3.19}$$

Portanto, o polinômio característico do sistema dinâmico do erro de trajetória com compensador PID é definido por:

$$\Delta c = s^3 + K_v s^2 + K_p s + K_i. \tag{3.20}$$

Como K_v, K_p e K_i são matrizes diagonais, então utilizando o critério de Routh descrito em Nise(2011), Dorf e Bishop (2011) e Ogata (2010), é possível concluir que considerando $w = 0$, o polinômio característico dado em (3.20) será estável se as condições (3.21) forem satisfeitas:

$$\begin{aligned} k_{i_i} &> 0, \\ k_{p_i} k_{v_i} &> k_{i_i}. \end{aligned} \tag{3.21}$$

A topologia dessa arquitetura de controle é representada pela Figura 3.6.

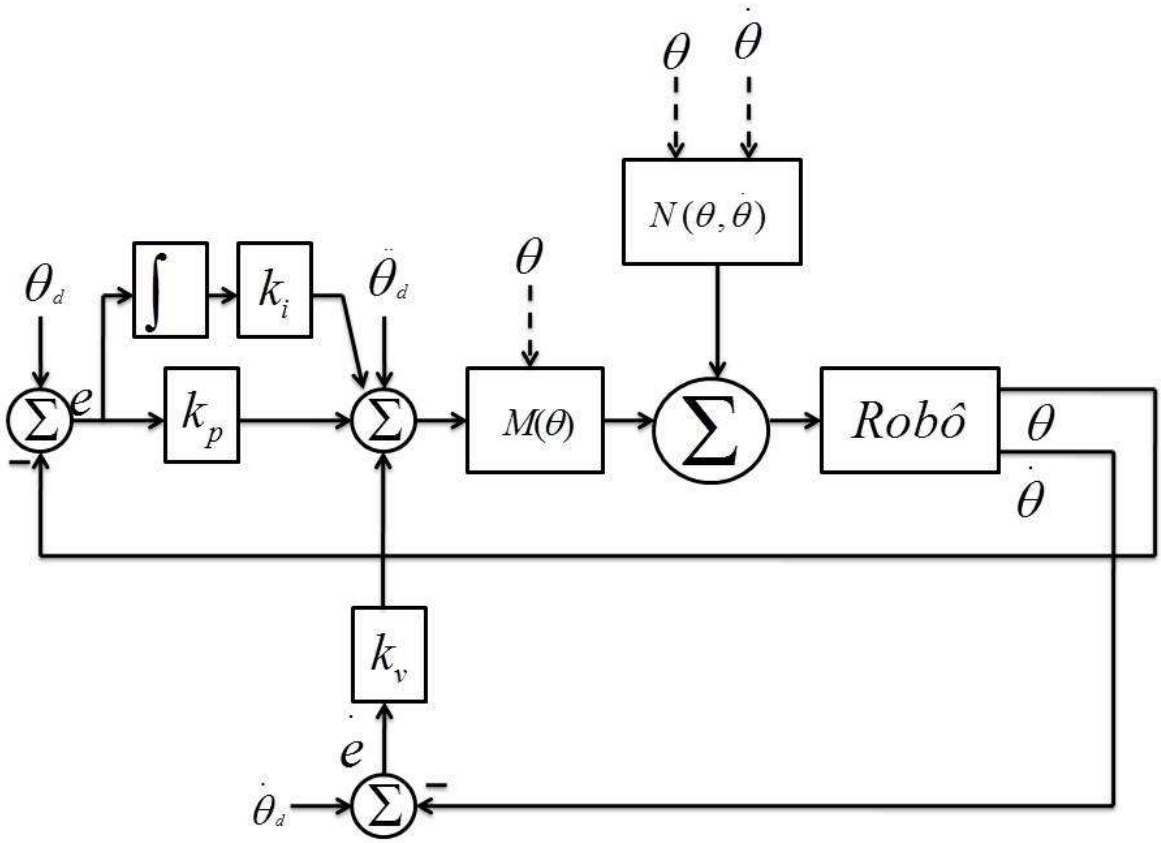

Figura 3.6 - Topologia de Torque Computado com Compensador PID (Adaptado de Lewis, Dawson e Abdallah 2004)

Na Figura 3.6, o bloco Robô representa o manipulador robótico, que tem como entrada um sinal de torque e como saída as posições e velocidades angulares de cada junta. As matrizes $M(\theta)$ e $N(\theta, \dot{\theta})$ são matrizes que descrevem o modelo dinâmico indireto do robô. Por fim, as matrizes K_p, K_v e K_i são matrizes diagonais com os ganhos proporcionais, integrais e derivativos. As condições 3.21 são suficientes para garantir a estabilidade do sistema apenas quando não existe variação paramétrica ou presença de dinâmicas não modeladas, isso é, quando $w = 0$.

A topologia de torque computado com compensador PID pode ser facilmente aplicada no ambiente Simulink com o auxílio do modelo dinâmico inverso do Robotic Toolbox. A Figura 3.7 exemplifica o diagrama de acionamento.

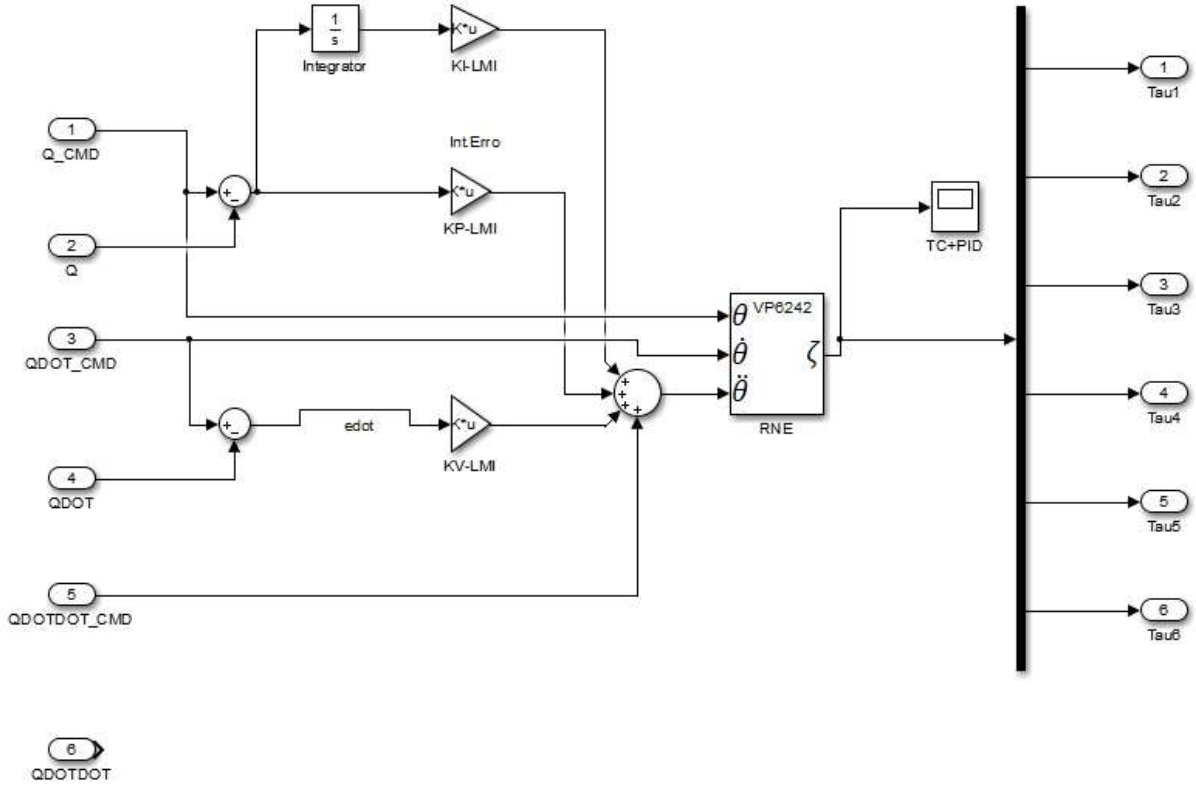

Figura 3.7 - Diagrama de Acionamento de Torque Computado com Compensador PID (Próprio Autor)

Na Figura 3.7 as entradas de 1 a 6 são respectivamente: posição desejada, posição atual, velocidade desejada, velocidade atual, aceleração desejada e aceleração atual. Todos os dados são compostos por um vetor com 6 elementos (sendo uma para cada junta). Posteriormente, os sinais de erro, integral do erro e derivada do erro são gerados para cada junta e são passados por ganhos do controlador. O bloco *"RNE"* (que foi definido na seção 2.1.1) possui informações da dinâmica indireta do robô, e nesse bloco são entrados respectivamente: posição desejada, velocidade desejada, e na entrada referente a aceleração angular desejada. O sinal "ζ" da saída do bloco "RNE" são os torques computados e que devem sem aplicados no manupulador.

Um controlador por torque computado com compensador PID foi desenvolvido para o manipulador Denso VP6242 utilizando as propriedades de projetos aqui descritas. Os passos para o projeto e os resultados obtidos são descritos na parte de resultados obtidos (Seção 7.1).

3.3 CONSIDERAÇÕES FINAIS SOBRE CONTROLE POR TORQUE COMPUTADO

Nas seções anteriores foram desenvolvidas duas topologias de controladores por torque computado, ambas topologias são muito eficientes quando se possui modelagens precisas do dispositivo. Contudo, em aplicações reais existe uma dificuldade muito elevada de extrair um modelo preciso de um dispositivo robótico, pois alguns dos parâmetros dos manipuladores são de difícil obtenção, e além do mais, devido ao desgaste mecânico de peças, é comum que um robô em uso já não tenha as mesmas especificações que tinha quando saiu de fábrica (ou de quando foi feito o levantamento paramétrico do dispositivo). Assim sendo, as considerações de que $\tilde{M}(\theta) = \tilde{V}(\theta, \dot{\theta}) = \tilde{F}(\dot{\theta}) = \tilde{G}(\theta) = 0$ feitas para obter o controlador (3.4) e garantir a função característica de cada junta nem sempre são verdadeiras. Caso não seja possível garantir que a afirmação acima seja verdadeira, as componentes da dinâmica não modelada $(\tilde{M}(\theta), \tilde{V}(\theta, \dot{\theta}), \tilde{F}(\dot{\theta}), \tilde{G}(\theta))$ irão causar torques que são interpretados como distúrbios do sistema. Quanto maior for a magnitude do distúrbio, maior será a influência que será notada na saída do sistema, e na presença de dinâmicas não modeladas as condições (3.21) não são suficientes para garantir a estabilidade do sistema.

Algumas alternativas são expostas na literatura para solucionar o caso em que não é obtido o modelo preciso de um dispositivo robótico. Em Lewis, Dawson e Abdallah (2004) são propostos controladores capazes de garantirem a estabilidade e a convergência do erro para algumas topologias de controle por torque computado com modelo impreciso, contudo é necessário garantir o limite máximo que a variação paramétrica possui. Os mesmos autores sugerem que uma maneira de diminuir o efeito que as dinâmicas não modeladas afetam o sistema seria optar por elevados ganhos de realimentação de estados (K_p e K_v); optar por trajetórias suaves, além de diminuir a magnitude das dinâmicas não modeladas.

Outra maneira bastante difundida na literatura (Lewis, Dawson e Abdallah 2004, Spong e Vidyasagar 1989 e Sciavicco e Siciliano 2000) para garantir a estabilidade e melhorar o desempenho de manipuladores é utilizar controle adaptativo. Nessa abordagem são utilizadas técnicas adaptativas que modificam a planta modelada que executa o torque computado a fim de executar uma melhor linearização por realimentação. Contudo, além do controle adaptativo não necessariamente convergir para os parâmetros reais da planta, a topologia clássica de controle adaptativo demanda de um rico conhecimento da modelagem dinâmica simbólica do dispositivo.

As topologias de controle robustas acima citadas não serão tratadas no escopo desse trabalho, contudo as referências acima citadas são recomendadas para entendimento inicial do conteúdo. Os Capítulos seguintes abordam outra topologia utilizada para tratar das incertezas na planta de manipuladores: a utilização de controladores com estrutura variável e modos deslizantes.

4 CONTROLE POR ESTRUTURA VARIÁVEL

No Capítulo 3, foi abordado o controle por torque computado com compensador PID. Essa classe de controladores utiliza o modelo dinâmico indireto para controlar um manipulador. Neste capítulo será abordada uma arquitetura de controle com maior robustez quanto à incerteza paramétrica e à presença de distúrbios, o controle por estrutura variável.

Como foi abordado na Seção 3.2.1, o sistema dinâmico do erro de trajetória de um manipulador robótico com compensador PID é dado por (4.1):

$$\begin{bmatrix} \boldsymbol{e} \\ \dot{\boldsymbol{e}} \\ \ddot{\boldsymbol{e}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ -\boldsymbol{K_i} & -\boldsymbol{K_p} & -\boldsymbol{K_v} \end{bmatrix} \begin{bmatrix} \boldsymbol{\varepsilon} \\ \boldsymbol{e} \\ \dot{\boldsymbol{e}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} \boldsymbol{u} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} \boldsymbol{w}, \qquad (4.1)$$

$$\boldsymbol{y} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\varepsilon} \\ \boldsymbol{e} \\ \dot{\boldsymbol{e}} \end{bmatrix}.$$

Para o caso de um robô industrial com 6 graus de liberdade cada um dos elementos é uma representação matricial dos respectivos elementos das 6 juntas, isso é, $\boldsymbol{e} = [e_1\ e_2\ e_3\ e_4\ e_5\ e_6]^T$, $\dot{\boldsymbol{e}} = [\dot{e}_1\ \dot{e}_2\ \dot{e}_3\ \dot{e}_4\ \dot{e}_5\ \dot{e}_6]^T$, $\ddot{\boldsymbol{e}} = [\ddot{e}_1\ \ddot{e}_2\ \ddot{e}_3\ \ddot{e}_4\ \ddot{e}_5\ \ddot{e}_6]^T$, $\mathbf{0} = diag(0\ 0\ 0\ 0\ 0\ 0)$, $\mathbf{1} = diag(1\ 1\ 1\ 1\ 1\ 1)$, $\boldsymbol{K_i} = diag\left(k_{i_1}\ k_{i_2}\ k_{i_3}\ k_{i_4}\ k_{i_5}\ k_{i_6}\right)$, $K_p = diag\left(k_{p_1}\ k_{p_2}\ k_{p_3}\ k_{p_4}\ k_{p_5}\ k_{p_6}\right)$, $\boldsymbol{K_v} = diag\left(k_{v_1}\ k_{v_2}\ k_{v_3}\ k_{v_4}\ k_{v_5}\ k_{v_6}\right)$, $\boldsymbol{u} = (u_1\ u_2\ u_3\ u_4\ u_5\ u_6)$, $\boldsymbol{w} = (w_1\ w_2\ w_3\ w_4\ w_5\ w_6)$, $\boldsymbol{y} = (y_1\ y_2\ y_3\ y_4\ y_5\ y_6)$. Desse modo a representação em espaço de estados (4.1) possui matriz $A \in \mathbb{R}^{18x18}$, $B \in \mathbb{R}^{18x6}$ e $C \in \mathbb{R}^{18x18}$.

Em (4.1) os estados $\varepsilon, e, \dot{e}$ são respectivamente a integral do erro de posição angular, o erro de posição angular e o erro da velocidade angular, K_i, K_v e K_p são constantes do compensador PID (mais detalhes no Capítulo 3), u é o sinal de entrada (torque) e y o sinal de saída do sistema (integral da posição do erro angular, erro da posição angular e erro da velocidade angular). O sinal w representa o torque exercido pelas dinâmicas não modeladas ou erro paramétrico. Pela dedução realizada em (3.13) é possível concluir que o torque causado pelas dinâmicas não modeladas e/ou erro paramétrico tem magnitude $w = \widehat{M}^{-1}(\theta)\left[\widetilde{M}(\theta)\ddot{\theta} + \tilde{V}\left(\theta, \dot{\theta}\right)\dot{\theta} + \tilde{F}\left(\dot{\theta}\right) + \tilde{G}(\theta)\right]$, sendo que $\widehat{M}$ é a matriz inercial estimada, $\widetilde{M}$ é o erro de estimação da matriz inercial, $\tilde{V}$ é o erro de estimação da matriz Coriolis e centrífuga, $\tilde{F}$ é o erro de estimação do atrito e $\tilde{G}$ é o erro de estimação da matriz gravitacional. Desse modo, a incerteza paramétrica do manipulador robótico que é abordada nesse Capítulo se resume a um disturbio w existente na representação do sistema dinâmico do erro de trajetória do manipulador robótico.

No decorrer desse capítulo será utilizada a representação em espaço de estados da planta que foi deduzida no capítulo anterior e reapresentada em (4.1). As condições de projeto dos ganhos K_i, K_v e K_p estão representadas na Equação (3.21).

De acordo com Teixeira, Assunção e Aguirre (2007), um sistema com controle por estrutura variável é formado por um conjunto de subsistemas contínuos com chaveamento, de modo que o sinal de controle tem características descontínuas. Essencialmente, um sistema com estrutura variável utiliza uma lei de controle com alta velocidade de chaveamento para levar a trajetória de estados da planta para uma superfície especificada no plano de fase, mantendo-a sobre esta superfície por todo o tempo subsequente. Portanto a dinâmica da planta fica restrita a uma superfície que é convenientemente escolhida de maneira a obter um comportamento desejado. Em Medjebouri e Mehannaoui (2016) são obtidos resultados simulados de uma arquitetura de controle por estrutura variável e modos deslizantes para um manipular robótico industrial. Em Coung e Nan (2016) é abordado a implementação de um controle por estrutura variável e modos deslizantes em um manipulador com dois graus de liberdade.

De acordo com Zinober (1990) e Covacic (2001), o projeto de sistemas com ações de controle descontínuas normalmente se reduz à seleção de superfícies no plano de fase para que a função de controle tenha descontinuidade. Quando certas relações são válidas, um tipo especial de movimento, chamado de modo deslizante pode aparecer. Este pode ser o caso, por exemplo, se, na vizinhança da superfície onde a função de controle tenha descontinuidade, as trajetórias de estados estiverem direcionadas a esta superfície. Uma vez nessa superfície, o sistema evidentemente não pode se mover ao longo de nenhuma trajetória adjacente a esta em qualquer outro instante. Portanto, em resposta a qualquer mudança, um movimento que começa sempre retorna a esta superfície. Consequentemente, no sistema em discussão, a trajetória pode mover-se somente ao longo da superfície descontínua. Esse movimento é convencionalmente chamado de modo deslizante.

Segundo Teixeira, Assunção e Aguirre (2007), Young (1993) e Covacic (2001), em geral o primeiro passo no projeto de um controle por estrutura variável, é a escolha de um subespaço deslizante (superfície de chaveamento). Um modo deslizante estável é conseguido através da seleção adequada de hiperplanos, de maneira que a resposta dinâmica de malha fechada deseja ser alcançada. O segundo passo envolve a seleção de uma lei de controle que assegure o alcance e a permanência do sistema em modo deslizante.

4.1 SISTEMAS ERP

Os sistemas reais positivos (sistemas RP), também conhecidos como sistemas passivos, juntamente com os sistemas estritamente reais positivos (sistemas ERP), têm uma grande relevância no controle de sistemas com incertezas, devido aos importantes resultados disponíveis sobre a estabilidade destes sistemas, como por exemplo a hiperestabilidade assintótica de Popov (Anderson 1968).

Considere uma função de transferência $G(s) = C(sI - A)^{-1}B$, onde

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx, \end{aligned} \tag{4.2}$$

$x \in \mathbb{R}^n, u \in \mathbb{R}^m, y \in \mathbb{R}^p$. Considere também que o sistema seja controlável e observável e que $p = m$.

Considere também as seguintes definições:

Definição 1: (Anderson, 1968) A matriz de transferência $G(s)$ do sistema (4.2) é Real Positiva se as seguintes condições forem satisfeitas:

- Os elementos de $G(s)$ não possuem polos com parte real positiva;
- $G^*(s) = G^T(s^*)$;
- A matriz hermitiana $J(s) = G(s) + G^T(s^*)$ é semi-definida positiva em $Re(s) > 0$, sendo que o asterisco (*) denota o complexo conjugado de um escalar ou o complexo conjugado transposto de um vetor.

Definição 2: (Anderson, 1968) A matriz de transferência $G(s)$ do sistema (4.2) é Estritamente Real Positiva (ERP) se $G(s - \xi)$ for RP para algum $\xi > 0$.

Considere, agora, o sistema (4.3):

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du, \end{aligned} \tag{4.3}$$

sendo que $x \in \mathbb{R}^n, u \in \mathbb{R}^m, y \in \mathbb{R}^m$, que o sistema seja controlável e observável, tal que para todo *T* positivo,

$$\int_0^T u(t)'y(t)dt < \delta[\|x(0)\|]sup\|x(t)\|, 0 \leq t \leq T. \tag{4.4}$$

Em (4.4) δ é uma constante positiva dependente do estado inicial (e independente de *T*). De acordo com Anderson (1968), V. M. Popov formulou na década de 1960 os seguintes resultados:

Definição 3: (Anderson, 1968) O sistema (4.3) é dito hiperestável se, para qualquer $u(.)$ limitado satisfazendo (4.4), a inequação

$$\|x(t)\| \leq K(\|x(0)\| + \delta) \tag{4.5}$$

é satisfeita para alguma constante positiva K e par todo $t \geq 0$.

Definição 4: (Anderson, 1968) O sistema (4.3) é dito assintoticamente hiperestável se, para qualquer $u(.)$ limitado satisfazendo (4.4), a inequação (4.5) é satisfeita e, também,

$$\lim_{t\to\infty} x(t) = 0. \tag{4.6}$$

Em Anderson (1968) foram formulados os seguintes resultados a respeito da hiperestabilidade e hiperestabilidade assintótica de um sistema:

Teorema 1: (Anderson, 1968) A condição necessária e suficiente para que o sistema (4.3) seja hiperestável é que $G(s)$ seja RP.

Teorema 2: (Anderson, 1968) A condição necessária e suficiente para que o sistema (4.3) seja assintóticamente hiperestável é que $G(s)$ seja ERP.

Desse modo, de acordo com Covacic (2001), Covacic (2006) e Teixeira, Assunção e Aguirre (2007), um procedimento para a análise e projeto de sistemas de controle, consiste na manipulação do sistema, com o objetivo de colocá-lo na forma de um sistema ERP, de modo que os resultados sobre a estabilidade destes sistemas possam ser utilizados.

4.2 SÍNTESE DE SISTEMAS ERP

Uma abordagem bastante consolidada para garantir que um sistema seja ERP é a utilização das LMI. De acordo com Covacic (2001), Covacic (2006) e Teixeira, Assunção e Aguirre (2007) projetos baseados em LMIs, inclusive síntese de sistemas ERP utilizando LMIs, possuem algumas vantagens, tais como:

- Uma grande variedade de especificações e restrições de projeto pode ser expressa como LMIs;
- Uma vez formulado em termos de LMI, o problema pode ser solucionado com exatidão por algoritmos de otimização bastante eficientes;
- Enquanto a maioria dos problemas com múltiplas restrições ou objetivos falham em encontrar soluções analíticas em termos de suas equações matriciais, eles são frequentemente factíveis no tratamento por LMI.

Esse trabalho faz uso de condições baseadas em LMIs para garantir que os sistemas sejam ERP. De acordo com Anderson (1968), Covacic (2001), Covacic (2006) e Teixeira, Assunção e Aguirre (2007), dada uma planta linear, invariante no tempo, controlável, observável descrita por (4.2), com $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ e $C \in \mathbb{R}^{p\times n}$, tal que $p = m$ (ou seja, que o sistema tenha a mesma quantidade de entradas e saídas), com $posto(C) = p$, e $posto(B) = posto(CB) = m$ então, com base no método de estabilidade de Lyapunov, é possível obter condições baseado em LMIs que garanta que o sistema descrito na Figura 4.1, com entrada $\tilde{U}(s)$ e saída $Y(s)$, seja ERP.

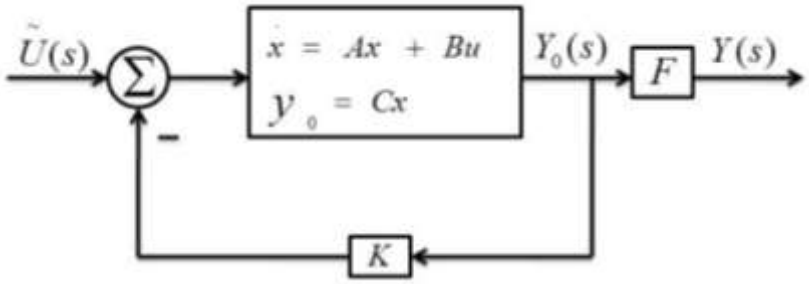

Figura 4.1 - Sistema ERP com Realimentação de Saída (Covacic 2001)

De acordo com Anderson (1968), Covacic (2001), Covacic (2006) e Teixeira, Assunção e Aguirre (2007) o Teorema 3 apresenta condições que garantem que o sistema representado na Figura 4.1 com entrada $\tilde{U}(s)$ e saída $Y(s)$ seja ERP.

Teorema 3: (Teixeira, Lordelo e Assunção 2000 e Lordelo 2000) O sistema (4.2) com entrada $\tilde{U}(s)$, saída $Y(s)$ sujeito as definições 1 – 4 e tal que $p = m$ é ERP se e somente se existirem matrizes $P = P'$, R e F tais que:

$$\begin{gathered} PA + A'P - C^T(R + R^T)C < 0, \\ B'P = FC, \\ P > 0. \end{gathered} \tag{4.7}$$

Além disso, quando (4.7) forem satisfeitas, então a matriz K é dado por: $K = (F^T)^{-1}R$.

4.3 CONTROLE COM ESTRUTURA VARIÁVEL E MODOS DESLIZANTES UTILIZANDO SISTEMAS ERP E LMIS

Como visto em Lordelo (2000), quando certas relações são válidas no projeto de controle por estrutura variável, um tipo especial de movimento chamado de modo deslizante pode acontecer. Esse movimento acontece quando o estado do sistema cruza imediatamente e

rapidamente a superfície de chaveamento, pois todos os movimentos nas vizinhanças da superfície estão direcionadas a esta superfície (Zinober, 1990).

A existência de um modo deslizante requer a estabilidade da trajetória de estados para a superfície de deslizamento no mínimo em uma vizinhança de $\{x(t)|\ s(x(t)) = 0\}$, ou seja, deve aproximar-se da superfície no mínimo assintoticamente. A vizinhança máxima é chamada região de atração. Geometricamente, o vetor tangente ou a derivada no tempo do vetor de estados deve apontar para superfície de deslizamento na região de atração. Isso assemelha-se a um problema de estabilidade generalizado, e o seguinto método de Lyapunov fornece uma ferramenta natural para a análise. Especificamente, a estabilidade para a superfície de chaveamento requer a seleção de uma função de Lyapunov generalizada $V(x,t)$ que é definida positiva e tem uma derivada no tempo negativa na região de atração. (Decarlo, Żak e Mathews 1988) (Covacic 2006).

Considere o sistema abaixo:

$$\begin{aligned} \dot{x} &= Ax + B[u + w(t,x)], \\ y_0 &= Cx, \end{aligned} \tag{4.8}$$

onde $w(t,x)$ representa o conjunto de incertezas definidas no início desse Capítulo, tal que existam constantes positivas a e b tais que $\|w(t,x)\| \leq a\|x\| + b$. Considere também que os estados $x(t)$ não estão disponíveis e que $y_0(x)$ está disponível para medição, $x \in \mathbb{R}^n, u \in \mathbb{R}^m$, $y_0 \in \mathbb{R}^p$, que o sistema seja controlável e observável, que $p = m$ e que existem matrizes $K \in \mathbb{R}^{m\times p}$ e $F \in \mathbb{R}^{m\times p}$ de modo a tornar o sistema em malha fechada representado pela Figura 4.2 e com entrada $\tilde{U}(s)$ e saída $Y(s)$ em um sistema ERP.

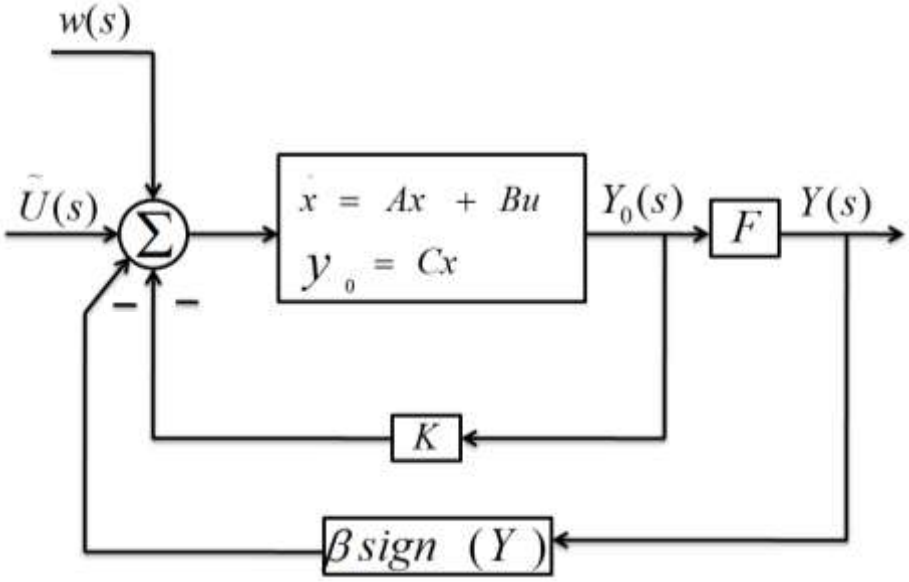

Figura 4.2 - Sistema ERP com Realimentação de Saída e CEV Mediante a Distúrbios/Ruídos (Adaptado de Covacic 2001)

Adotando uma lei de controle $u(t) = -Ky_0 - \beta sign(y)$, tal que $\beta \in \mathbb{R}$, $\beta > a\|x\| + b$ para o sistema (4.8), uma candidata a função de Lyapunov é $V(x) = x^T Px$ é positiva definida e $\dot{V}(x)$ é negativa definida quando o conjunto de LMIs e Igualdade Matricial Linear (do inglês *Linear Matrix Equality – LME)* (4.9) é satisfeito (Covacic 2006):

$$\begin{gathered} A^T P + PA - C^T G^T - GC < 0, \\ B'P = FC, \\ P > 0. \end{gathered} \tag{4.9}$$

Para as inequações (4.9), A, B, C são matrizes da representação do sistema (4.8) em espaço de estados, $P = P^T$, $G = PBK$. A dedução de que (4.8) é um conjunto de equação e inequações que garantem a estabilidade assintótica do sistema (4.8) mediante a lei de controle $u(t) = -Ky_0 - \beta sign(y)$ é encontrada no Apêndice C, bem como em Covacic (2006). Portanto, como $V(x)$ é definida positiva e $\dot{V}(x)$ é definida negativa, o ponto de equilíbrio $x = 0$ do sistema controlado é globalmente assintoticamente estável.

Uma outra possível arquitetura de controle para o sistema (4.8) que também garante a estabilidade do sistema é utilizando a lei de controle $u(t) = -Ky_0 - \beta sign(y) - \alpha y$, tal que $\beta \in \mathbb{R}$ e $\beta > a\|x\| + b$. Essa lei de controle é mostrada na Figura 4.3.

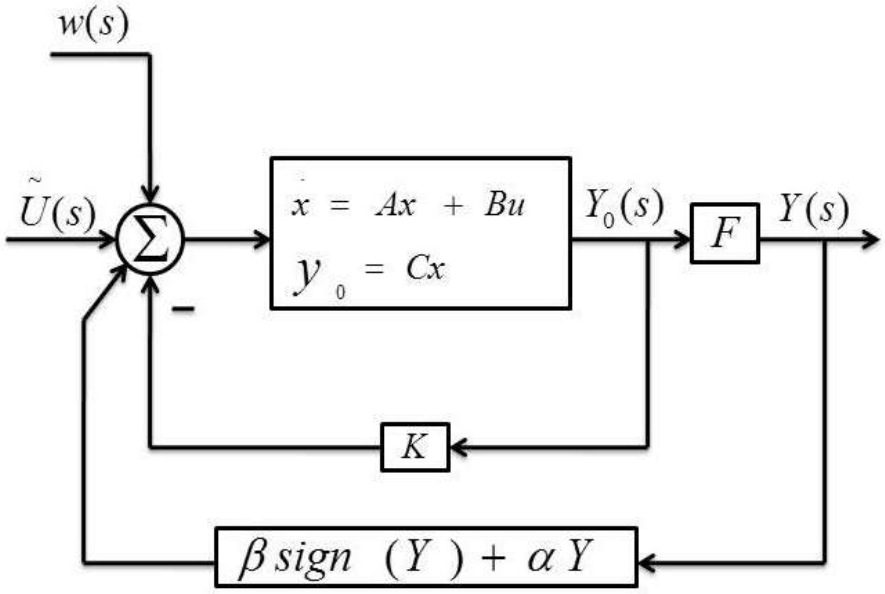

Figura 4.3 – Arquitetura de Controle com Realimentação de Saída e CEV Mediante a Distúrbios/Ruídos (Proprio Autor)

Uma maneira de comprovar a estabilidade desse sistema é comprovando que a candidata a função de Lyapunov do tipo $V(x) = x^T Px$ é definida positiva, e que sua derivada temporal $\dot{V}(x)$é definida negativa. O seguinte conjunto de LMIs e LME garante a condição de estabilidade do sistema descrito acima. A dedução mais detalhada do conjunto de inequações (4.10) é encontrada no apêndice D.

$$\begin{gathered} A^T P + PA - C^T G^T - GC - C^T H^T - HC < 0, \\ B^T P = FC, \\ P > 0. \end{gathered} \tag{4.10}$$

Para as inequações (4.10), A, B, C são matrizes da representação do sistema (4.7) em espaço de estados, $P = P^T$, $G = PBK$, $H = B\alpha K$. Embora a transformação de LME em LMI não seja uma operação simples, essa não será abordada nesse trabalho por não se tratar do foco da pesquisa. Mais informações sobre essa transformação é disponível no Lema 12 abordado em Covacic (2006).

4.3 CONSIDERAÇÕES FINAIS SOBRE CONTROLE POR ESTRUTURA VARIÁVEL

A topologia de controle por estrutura variável e modos deslizantes apresentada nessa Seção garante a estabilidade para o sistema mesmo na presença de um distúrbio $\|w(t,x)\| \leq a\|x\| + b$. Como visto na Seção 3.3 o grande problema do controle por torque computado é que nem sempre é possível garantir que $\tilde{M}(\theta) = \tilde{V}(\theta, \dot{\theta}) = \tilde{F}(\dot{\theta}) = \tilde{G}(\theta) = 0$. Caso existam imperfeições na modelagem, isso acarretará em $\tilde{M}(\theta), \tilde{V}(\theta, \dot{\theta}), \tilde{F}(\dot{\theta}), \tilde{G}(\theta) \neq 0$ e pode ser interpretado como um distúrbio de torque aplicado no sistema $(w(t,x))$. Portanto se o distúrbio de torque presente for dimensionado, é possível utilizar o controle por estrutura variável para garantir a estabilidade do sistema desde que seja utilizado a lei de controle $u(t) = -Ky_0 - \beta sign(y) - \alpha y$ tal que $\beta > a|x| + b$.

5 REDES NEURAIS

Redes neurais é um tema que tem sido bastante utilizado no controle de sistemas dinâmicos, principalmente por sua propriedade de aproximação universal e capacidade de aprendizado. A propriedade de aproximação universal nos garante que sempre existirá um conjunto de pesos sinápticos e função de ativação na qual uma rede neural multicamada aproxime o comportamento de uma função não linear com um erro de aproximação $\in_n$ (erro de aproximação funcional da rede neural), para qualquer valor de $\in_n$ (mesmo que seja difícil encontrar os pesos sinápticos para determinados erro de aproximação funcional) (Lewis, Dawson e Abdallah 2004). Em termos matemáticos o erro de aproximação funcional é dado por: $\in_n = \tilde{f}(x) = f(x) - \hat{f}(x)$. Devido a essas características, as redes neurais artificiais (ou simplesmente redes neurais) podem ser utilizadas para simular o comportamento de sistemas não lineares dinâmicos (Calôba e Aguirre 2007).

Uma rede neural é basicamente formada por um conjunto de neurônios interconectados. A Figura 5.1 representa um neurônio, onde $S_1, S_2, \ldots S_n$ são as entradas, $v_1, v_2, \ldots v_n$ são os pesos sinápticos (que também podem ser representados como uma matriz de pesos sinápticos V), v_0 é o bias, $\sigma(.)$ é a função de ativação do neurônio, e y a saída do neurônio. O bias do neurônio também pode ser adicionado na primeira linha da matriz de pesos sinápticos V desde que a primeira entrada seja constante 1.

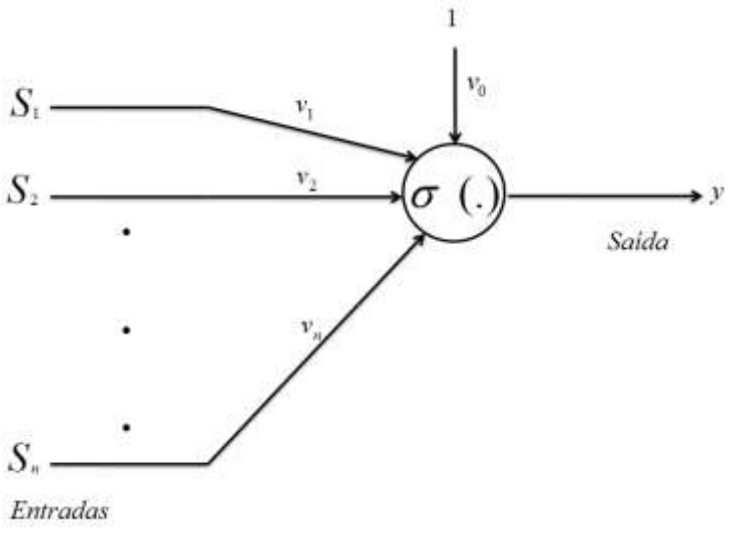

Figura 5.1 - Neurônio Artificial (adaptado de Lewis, Jagannathan e Yesildirek 1999)

Para o neurônio acima, a saída y é dada por: $y = \sigma(\sum_{j=1}^{n} v_j S_j(t) + v_0)$. A função de ativação pode ser substituída por uma grande variedade de funções, e de acordo com Lewis,

Jagannathan e Yesildirek (1999), algumas das principais funções de ativações e suas respectivas expressões matemáticas são descritas na Figura 5.2.

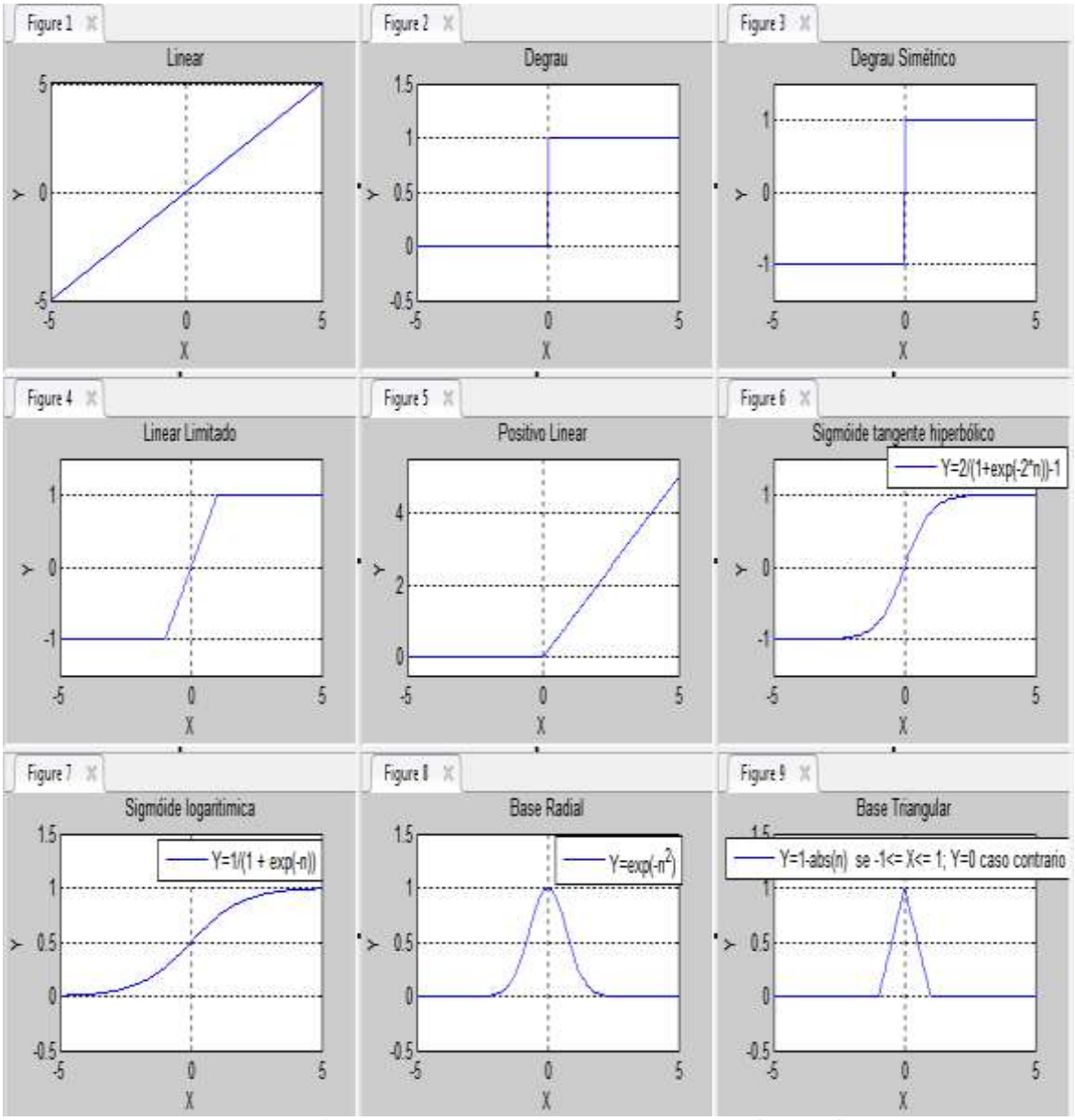

Figura 5.2 - Principais funções de ativação e suas respectivas equações matemáticas (Próprio Autor)

A escolha pela melhor função de ativação varia de acordo com a aplicação desejada para cada rede neural, por exemplo: para o caso de classificação de padrões, normalmente a utilização de funções do tipo “Degrau” e “Degrau Simétrico” é mais eficiente, contudo essas mesmas funções de ativação não são muito aplicadas para emular o comportamento dinâmico de um sistema.

Uma rede neural é composta por Z camadas e cada camada é composta por uma determinada quantidade de neurônios. A Figura 5.3 representa uma rede neural com duas camadas de neurônios e cada camada composta por L e J neurônios.

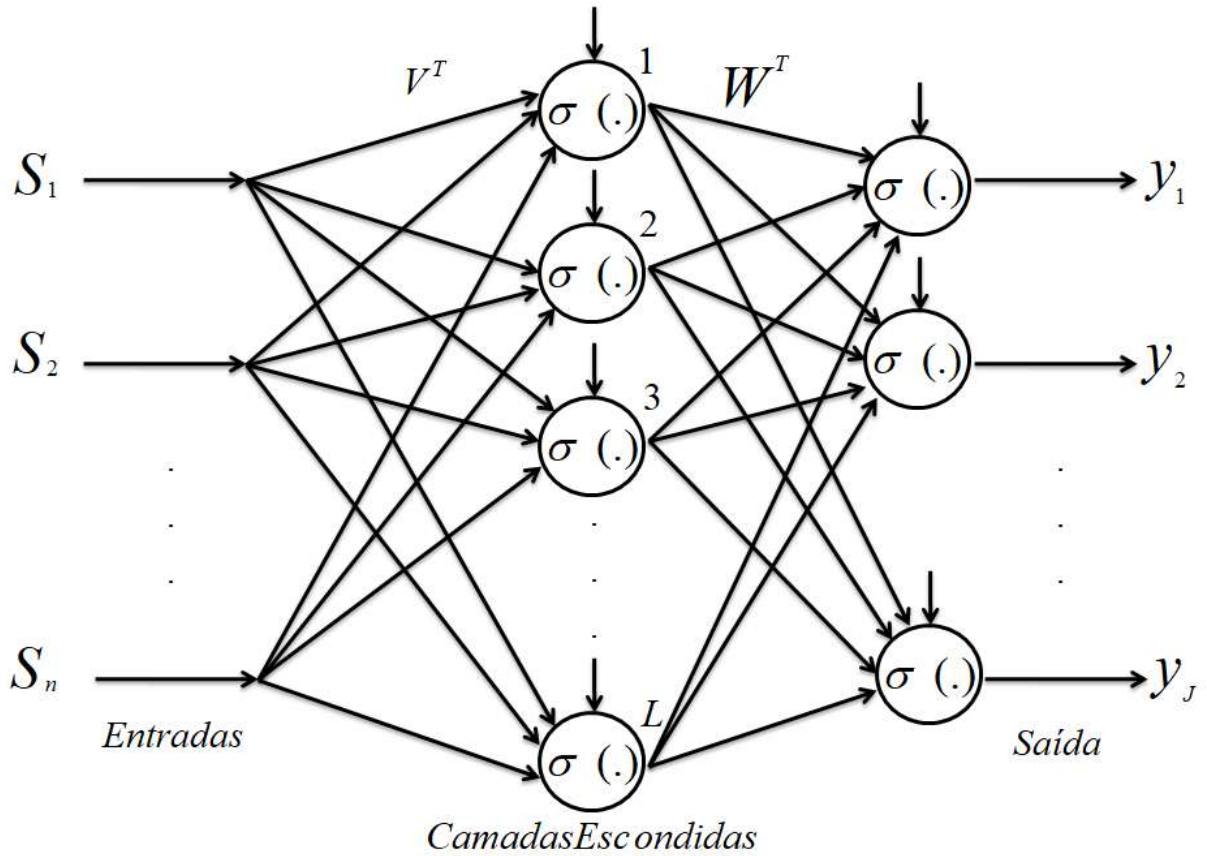

Figura 5.3 - Rede Neural Artificial (Lewis, Jagannathan e Yesildirek 1999)

Para a rede neural acima com n entradas e m saídas, a expressão de saída é dada por: $y = \sigma_J(W^T\sigma_l(V^T S))$, onde σ_J é a função de ativação da ultima camada, σ_l a função de ativação da primeira camada, W^T é a matriz com peso sináptico da ultima camada, V^T é a matriz com peso sináptico da primeira camada, e x é o vetor contendo as entradas da rede.

As topologias de redes neurais multicamadas possuem a propriedade de aproximação universal, contudo um grande desafio é a seleção dos pesos sinápticos apropriados para que uma rede neural se comporte como o desejado. Esse desafio é ainda maior para o caso de redes neurais multicamadas. De acordo com Lewis, Jagannathan e Yesildirek (1999) existem duas maneiras de solucionar esse problema: Analiticamente ou através de algoritmos recursivos. Embora a resposta analítica normalmente possua uma melhor adequação dos dados, para redes neurais mais complexas muitas das vezes não é possível extrair essa resposta, portanto a utilização de algoritmos recursivos torna-se mais vantajosa. Um princípio de algoritmo muito utilizado na adequação de pesos sinápticos de redes neurais é o algoritmo de backpropagation. Esse método basicamente inicia os pesos sinápticos com valores randômicos, e calcula-se o erro entre saída obtida e saída real, posteriormente utiliza-se o gradiente descendente do erro para atualizar os pesos sinápticos a fim de minimizar a função

custo. A função “traingd” do Matlab executa a minimização de uma função custo utilizando gradiente descendente (Mathworks 2017).

5.1 UTILIZAÇÃO DE REDES NEURAIS NO CONTROLE DE MANIPULADORES ROBÓTICOS

Nessa Seção será discutida a utilização de redes neurais no controle dinâmico de manipuladores robóticos. Como discutido na Seção 3.2, uma topologia muito utilizada no controle dinâmico de manipuladores é a utilização de uma técnica chamada de torque computado, que basicamente estima as não linearidades do sistema e aplica um sinal de controle u que idealmente cancela as não linearidades do sistema. A grande desvantagem da utilização das topologias discutidas na Seção 3.2 é que elas exigem a modelagem e os parâmetros dinâmicos do manipulador, e embora a modelagem seja facilmente obtida com o auxílio de softwares, os parâmetros dinâmico dos manipuladores muitas vezes são difíceis de mensurar pelo usuário e em algumas vezes essa dificuldade se estende até mesmo para os fabricantes. Portanto, nessa Seção será descrita uma maneira de utilizar as redes neurais para exercer a técnica de torque computado de um manipulador. A topologia descrita aqui é mais vantajosa do que a topologia descrita na seção 3.2 por não exigir a parametrização dinâmica do manipulador.

De acordo com Lewis, Jagannathan e Yesildirek (1999) o diagrama de acionamento da Figura 5.4 exemplifica a utilização de redes neurais em um loop interno que possui a função de tentar estimar as não linearidades do sistema. Além do loop interno a topologia descrita na Figura 5.4 possui um loop externo que possui uma superfície de chaveamento, em série com um controlador proporcional e com um controlador robusto (que garante a estabilidade do sistema mesmo mediante a presença de um erro de estimação funcional pela parte das redes neurais).

Uma lei de controle robusta que pode ser aplicada no sistema da Figura 5.4 é a lei $v(t) = -\beta sign(r)$. Portanto, a lei de controle atuante no sistema será $\tau = \hat{f}(x) + \alpha r - \beta sign(r)$. E de acordo com a dedução realizada no Capítulo 3, o sistema é estável desde que $\beta > \in_n$, isto é, desde que βseja maior do que o erro de aproximação funcional da rede neural.

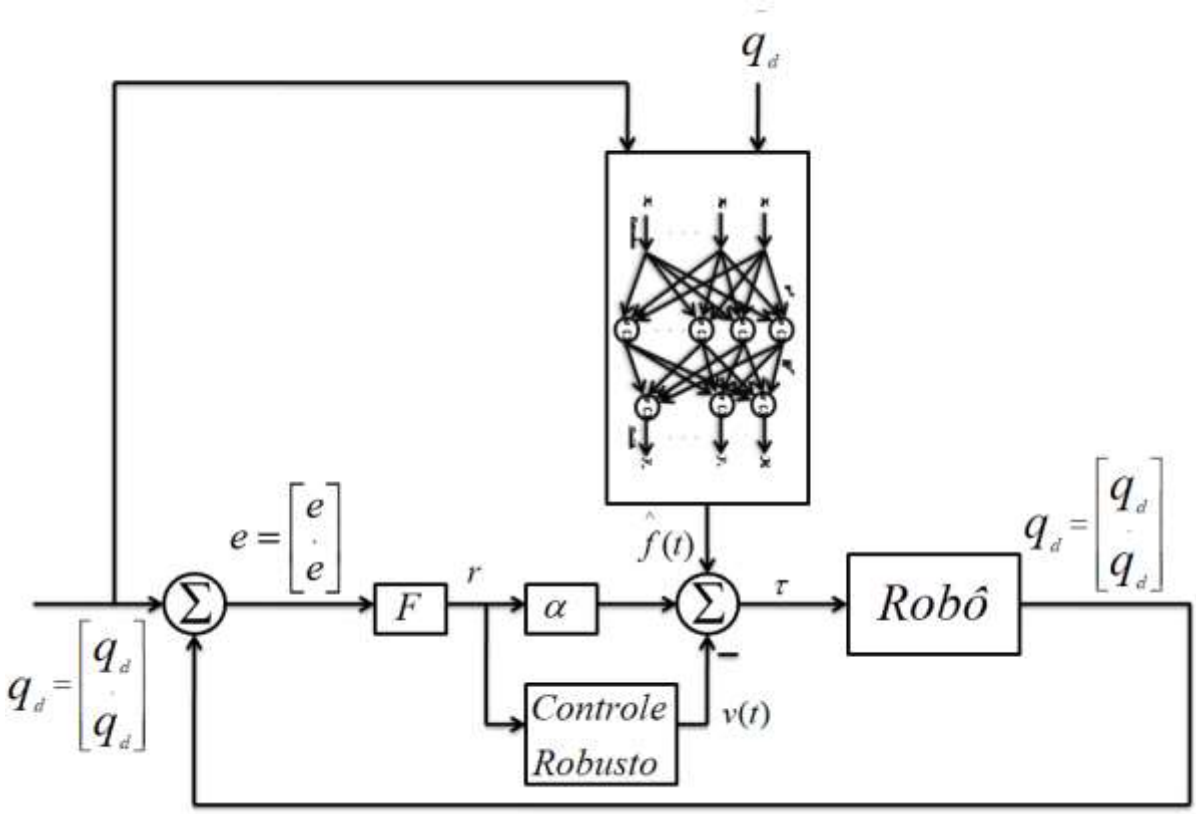

Figura 5.4 – Utilização de Redes Neurais no controle dinâmico de Manipuladores (Lewis, Jagannathan e Yesildirek 1999)

5.2 CONSIDERAÇÕES FINAIS SOBRE REDES NEURAIS

Redes neurais é um tema de grande abrangência para resolução de diversos tipo de problemáticas devido a propriedade de aproximação universal e capacidade de aprendizado que as redes neurais possuem. Esse capítulo abordou a utilização de redes neurais para controle dinâmico de dispositivos robóticos.

Nesse capítulo foram utilizadas redes neurais para efetuar uma tentativa de linearização de um manipulador robótico. Como fruto da imperfeição em linearizar o sistema utilizando redes neurais, distúrbios surgem no sistema. O controlador por estrutura variável e modos deslizantes desenvolvido no Capítulo 4 foi utilizado juntamente com redes neurais para efetuar o controle dinâmico do manipulador robótico. A topologia abordada nesse capítulo permite que um manipulador robótico seja controlado mesmo sem conhecer o modelo dinâmico do mesmo. Uma condição para garantir a estabilidade do controle por estrutura variável e modos deslizantes para controle de sistemas incertos é que exista constantes reais a e b, tais que $\|w(t,x)\| \leq a\|x\| + b$ e que exista um β tal que $\beta > a\|x\| + b$. A comprovação de estabilidade do controle por redes neurais utilizando a lei de controle

$\tau = \hat{f}(x) + \alpha r - \beta sign(r)$ acontece desde que $\beta > \epsilon_n$, isto é, desde que βseja maior do que o erro de aproximação funcional da rede neural (nesse caso a constante real $a = 0$ e $b = \epsilon_n$).

6 MATERIAIS UTILIZADOS

Neste Capítulo, são descritos os dispositivos utilizados para implementar o controle do sistema. A implementação das teorias de controle para o manipulador robótico ocorreu no Laboratório de Controle Avançado, Robótica e Biomédica do Departamento de Engenharia Elétrica da Universidade Estadual de Londrina. A bancada onde foram implementadas as teorias de controle é formada por um manipulador robótico da marca DENSO®, modelo VP6242 equipado com um sensor de força e torque da marca ATI Industrial Automation, modelo FT 14220 Gamma, um driver de acionamento modelo RC7M e um desktop com Matlab/Simulink e QUARC 2.3. A Figura 6.1 mostra a bancada e os principais elementos envolvidos.

Figura 6.1 - Bancada do Laboratório de Controle Avançado, Biomédica e Robótica da Universidade Estadual de Londrina (Próprio Autor)

A bancada da Figura 6.1 foi adquirida com fundos provindo da FINEP que possibilitou o desenvolvimento de novas linhas de pesquisa no laboratório de engenharia biomédica, controle avançado e robótica da Universidade Estadual de Londrina. A Figura 6.1 trata-se da

bancada do primeiro manipulador Denso VP6242 no Brasil que disponibiliza das ferramentas Matlab e Simulink para desenvolver novas tecnologias de acionamento e novas soluções para problemas encontrados no mundo da robótica. Em Hernandes (2014) foi desenvolvido o primeiro projeto relacionado com o robô Denso VP6242 do Laboratório de Controle Avançado, Robótica e Engenharia Biomédica da Universidade Estadual de Londrina, onde foi feita uma breve analise do robô Denso VP6242.

O princípio de funcionamento baseia-se em utilizar o Simulink para desenvolver uma arquitetura de controle que consiga gerar dados de correntes elétricas a ser aplicadas em cada um dos motores do manipulador. Uma vez que os dados são gerados em ambiente virtual, esses são transmitidos para o RC7M através do software QUARC 2.3 (software desenvolvido pela empresa Canadense QUANSER®), e então o RC7M produz e envia uma corrente elétrica requerida para cada um dos motores do robô, produzindo assim a movimentação do manipulador. Por sua vez, o robô Denso envia a posição angular de cada um dos seus encoders ao CR7M, e este envia os mesmos dados ao Simulink. O sensor de força e torque possui interface diretamente com o Simulink. Nas seções 6.1 a 6.3 será detalhado cada um dos principais dispositivos da bancada.

6.1 ROBÔ DENSO VP6242

O manipulador robótico utilizado é o Denso VP6242, mostrado na Figura 6.2, fabricado pela empresa DENSO®. Trata-se de um dispositivo robótico compacto com 6 juntas rotacionais (6 Graus de Liberdade). De acordo com DENSO (2011), o dispositivo possui uma precisão de 0.02mm, pode ser utilizado para manusear cargas de até 2.5kg, e possui um total de 6 encoders, onde cada encoder é posicionado de modo a medir o deslocamento angular de cada junta. O Denso VP6242 se destaca dos demais manipuladores dessa classe por sua alta precisão e velocidade de movimentação, desse modo o dispositivo pode ser utilizado para fins de desenvolvimento de pesquisas de alta precisão, tais como pesquisas em engenharia biomédica (Looi, Yeung, Umasthan e Drake 2013, Lyer, Looi e Frake 2013).

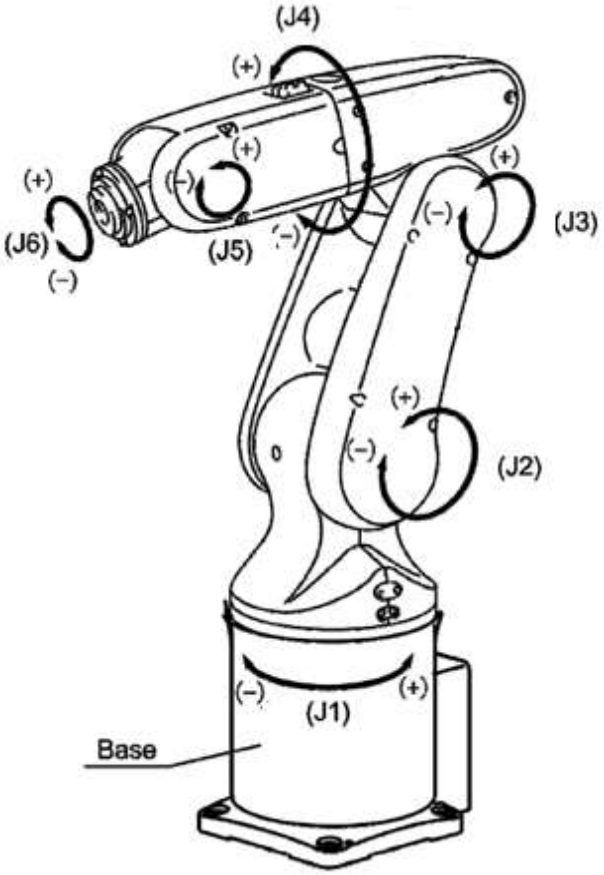

Figura 6.2 - Manipulador DENSO VP6242 (DENSO 2011)

6.2 CONTROLADOR RC7M

O manipulador robótico da Figura 6.2 possui um driver de acionamento também fabricado pela empresa DENSO® chamado controlador RC7M, mostrado na Figura 6.3. O RC7M faz interface entre o robô e um computador, desse modo o driver envia corrente elétrica para os motores e monitora a posição angular de cada junta. O RC7M também pode ser utilizado para fazer a interface entre o manipulador e dispositivos periféricos, tais com controladores lógicos programados, TeachPendent, Mini Pendent entre outros (DENSO 2011).

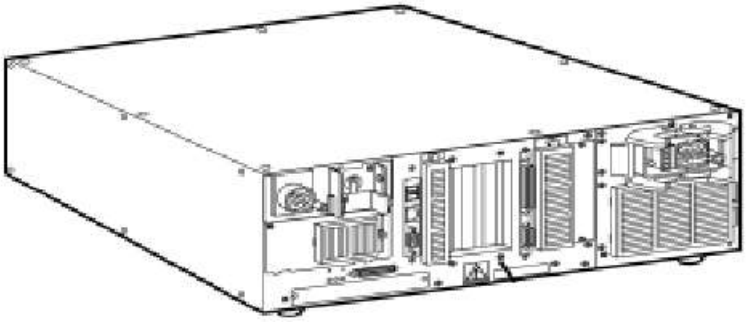

Figura 6.3 - Controlador RC7M (DENSO 2011)

Para a bancada mostrada na Figura 6.1, o RC7M possui como dispositivo periférico o E-Stop que é mostrado na Figura 6.4.

Figura 6.4 - Botão E-Stop (Quanser 2013)

O E-Stop é uma botoeira com retenção de segurança utilizado para parar o robô em casos de emergência. Quando pressionado, esse dispositivo corta toda alimentação do robô.

6.3 INTERFACE SIMULINK/RC7M

A interface entre o computador e o controlador RC7M é feita com o auxílio de blocos desenvolvido pelo programa QUARC (versão 2.3). As ferramentas do QUARC são de alta eficiência para o desenvolvimento e implementação de arquitetura de controle em tempo real para sistemas mecatrônicos.

Dentre os principais blocos desenvolvidos pela QUARC para auxiliar o controle em tempo real do manipulador Denso VP6242, destacam-se os seguintes blocos: Denso Write e Denso Read.

O bloco Denso Write é usado para enviar posição (ou variação de posição) desejada para o robô, ganhos PID individuais de cada junta e sinal de controle feedforward para cada uma das juntas. O controlador CR7M converte o sinal de controle feedforward em amplitude de corrente elétrica alternada a ser aplicado em cada motor CA do manipulador. O usuário deve escolher entre enviar a posição desejada ou enviar a variação de posição desejada (sendo que a escolha por enviar a variação de posição desejada é obtida ao selecionar a opção "*velocity mode"* no bloco Denso Write). É recomendado pelo fabricante que o uso do controle feedforward seja feito com muita cautela, pois um sinal de controle mal projetado pode levar o sistema à instabilidade, podendo ocasionar em danos ao manipulador e aos equipamentos ao seu redor, também é recomendado pelo fabricante que ao utilizar controle feedforward, os ganhos PID individuais de cada junta seja nulo (Quanser 2013a).

O bloco Denso Read é utilizado para efetuar a leitura da posição angular de cada junta (medida em radianos), além de ser possível efetuar a leitura da corrente elétrica em cada motor através do sinal *"effort"*. Os sinais *"status"* e os sinais de *"error"* para cada uma das 6 juntas do manipulador são responsáveis por mostrar a situação de operação do manipulador, e em caso de erro, é possível classificar a natureza do erro. Mais informações sobre os valores e significados de cada sinal *"status"* são encontradas na Tabela 2.3 disponível em Quanser (2013a). De acordo com Quanser (2013a), o bloco Denso Read efetua a leitura do estado de cada variável do robô a cada 1 milisegundo (frequência de 1kHz). A saída "denso" do bloco "Denso Read" deve ser conectada diretamente à entrada "denso" do bloco "Denso Write" e caso a comunicação direta deixe de existir, então o manipulador corta a energia dos motores. A Figura 6.5 mostra os blocos "Denso Read" e "Denso Write".

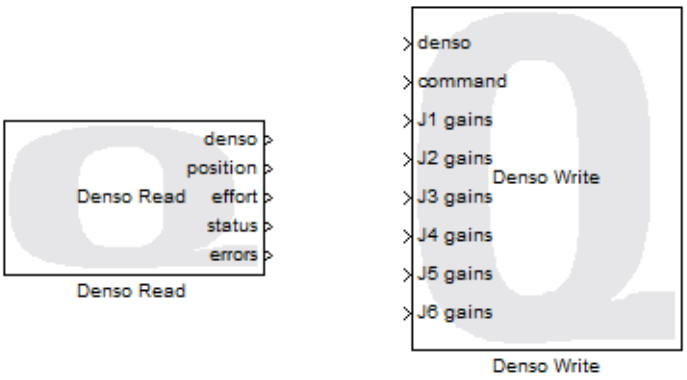

Figura 6.5 - Bloco Denso Read e Denso Write (Próprio Autor)

Desse modo, para implementar uma arquitetura de controle não linear para o manipulador, deve ser utilizado o bloco Denso Write para enviar um sinal de corrente elétrica desejada que produzirá o torque desejado. A conversão entre corrente enviada e torque

produzido pela junta deve ser feita utilizando os dados de constante de torque do motor e ganho de caixa de redução. Essas informações, juntamente com os limites da juntas e a calibração dos encoderes de cada junta são dadas em Quanser (2013a) e mostrado na Tabela 6.1.

Motor/ Encoder	**Relação de Transmissão**	**Calibração de Encoder (count/deg)**	**Constante de torque (Nm/Amp)**	**Limite máximo de junta (graus)**	**Limite mínimo de junta (graus)**
1	120:1	43690.666670	0,38	160	-160
2	160:1	58254.222220	0,38	120	-120
3	120:1	43690.666670	0,22	160	20
4	100:1	36408.888890	0,21	160	-160
5	100:1	36408.888890	0,21	120	-120
6	100:1	36408.888890	0,21	360	-360

Tabela 6.1 - Especificações dos Motores e Juntas para o Manipulador Denso VP6242 (Adaptado de Quanser 2013a)

7 RESULTADOS OBTIDOS

Nesse capítulo será descrita os resultados obtidos de três implementações. As arquiteturas propostas são: controle por torque computado com compensador PID, controle por torque computado com compensador PID e estrutura variável, e controle utilizando redes neuras e estrutura variável. Cada uma das arquiteturas é discutida abaixo.

Em todas as arquiteturas implementadas e discutidas neste capítulo, foi utilizado ou o bloco contendo o modelo dinâmico inverso desenvolvido pelo Robotic Toolbox, ou bloco "S-Function" do Matlab, e ambos os blocos são incompatíveis com a simulação do Simulink configurado em modo externo. Portanto uma solução abordada nesse trabalho para solucionar esse problema foi a utilização de duas janelas de Simulink, sendo uma configurada em modo externo e a outra em modo de simulação. Os blocos Stream Server e Stream Client, ferramenta desenvolvida pela Quanser, foram utilizados para efetuar a troca de informações entre as duas janelas.

Os blocos Stream Server e Stream Client são utilizados como servidores e clientes em um processo de comunicação. Quando configurados propriamente, o servidor envia dados ao cliente e recebe dados do cliente. Por sua vez, o cliente recebe dados do servidor e envia outros dados ao servidor. Para que possa ser efetuado comunicação entre o blocos Server e Client em uma mesma máquina, é necessário que, além das configurações padrões disponível pelo fabricante, os programas a serem comunicados estejam salvos em diretórios diferentes e que cada janela seja aberta em janela diferente do Matlab (Quanser 2017).

7.1 CONTROLE POR TORQUE COMPUTADO COM COMPENSADOR PID

O controle por torque computado com compensador PID foi discutido na Seção 3.2.2. A lei de controle utilizada é dada por $\tau = \widehat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) + \hat{V}\left(\theta,\dot{\theta}\right)\dot{\theta} + \hat{F}\left(\dot{\theta}\right) + \hat{G}(\theta)$, onde $\widehat{M}(\theta), \hat{V}\left(\theta,\dot{\theta}\right), \hat{F}\left(\dot{\theta}\right)$ e $\hat{G}(\theta)$ são obtidos com o auxilio do toolbox de robótica desenvolvido por Peter Corke. No apêndice A é encontrado o código utilizado para obter o modelo estimado do manipulador com o auxílio do Robotic Toolbox. Os ganhos K_v, K_p e K_i foram selecionados de modo a satisfazer as condições dadas pelo conjunto de inequações (3.21). Para tentar diminuir a influencia das incertezas paramétricas no sistema, foi optado empiricamente por constantes elevadas que satisfaçam as condições (3.21) e que ao mesmo tempo não saturem o sinal de controle e que aproxime a resposta do sistema para uma

resposta criticamente amortecida. Após diversos ensaios, os seguintes valores de ganho proporcionaram uma boa resposta ao sistema: $K_i = diag(4\ 4\ 4\ 4\ 4\ 4)$, $K_p = diag(80\ 80\ 80\ 80\ 80\ 80)$ e $K_v = diag(80\ 80\ 80\ 80\ 80\ 80)$. O diagrama padrão da implementação é dado na Figura 7.1.

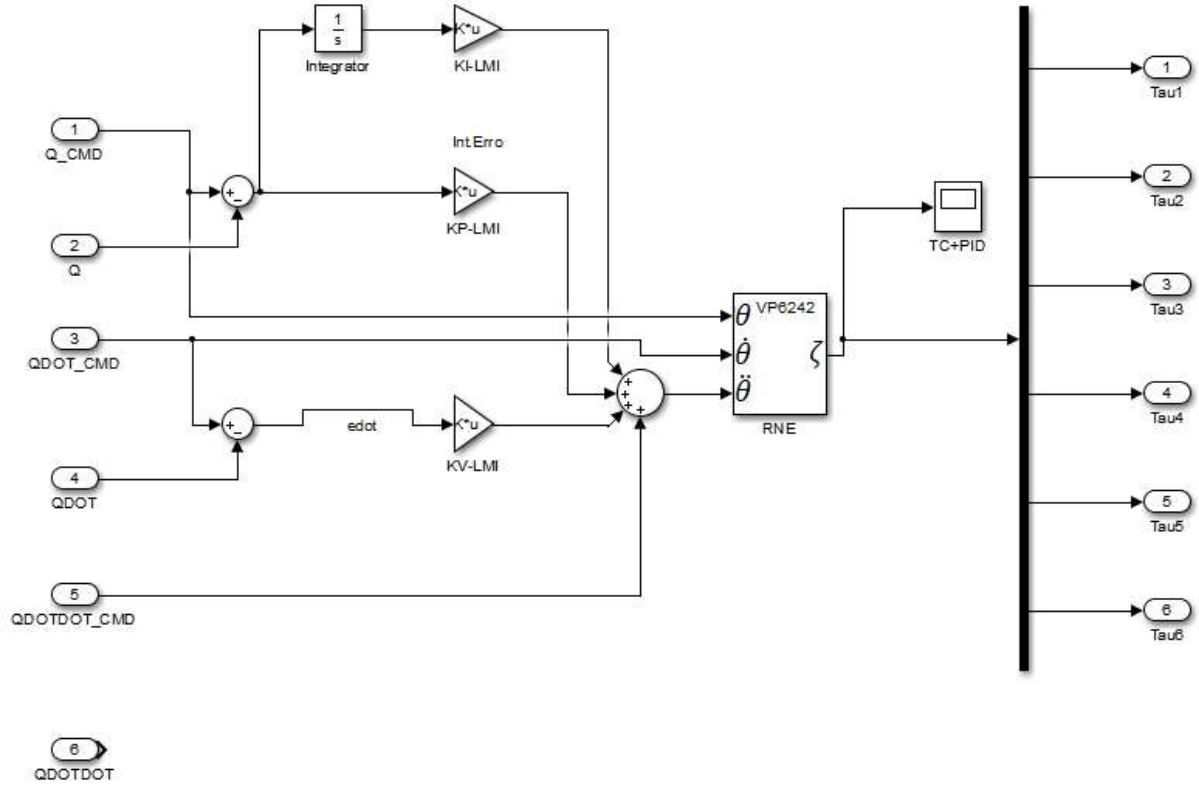

Figura 7.1 - Implementação de Torque Computado com Compensador PID Utilizando o Robotic Toolbox (Próprio Autor)

Na Figura 7.1, o bloco em RNE possui as informações de dinâmica inversa do manipulador Denso VP6242, as entradas de 1 a 6 são as posições, velocidades e acelerações angulares atuais e desejadas, e as matrizes $K_i, K_p e\ K_v$ são os ganhos de compensação. As seis saídas representadas na Figura 7.1 são os respectivos torques desejados a serem aplicados nas juntas.

As Figuras 7.2 e 7.3 mostram o set point e o sinal de erro de trajetória para cada junta. Foram aplicados dois set point no sistema com o intuito de avaliar a rapidez da resposta do sistema. O primeiro set point tem inicio aos 5 segundos e termino no instante 40 segundos, nesse caso cada junta saiu da sua posição de origem e obteve uma variação de 45° durante 2 segundos, posteriormente a subida, cada junta manteve na posição desejada por aproximadamente 11 segundas e por fim retornou à posição de origem em 2 segundos. A Figura 7.2 mostra ao lado de cada set point os respectivos erros de trajetória.

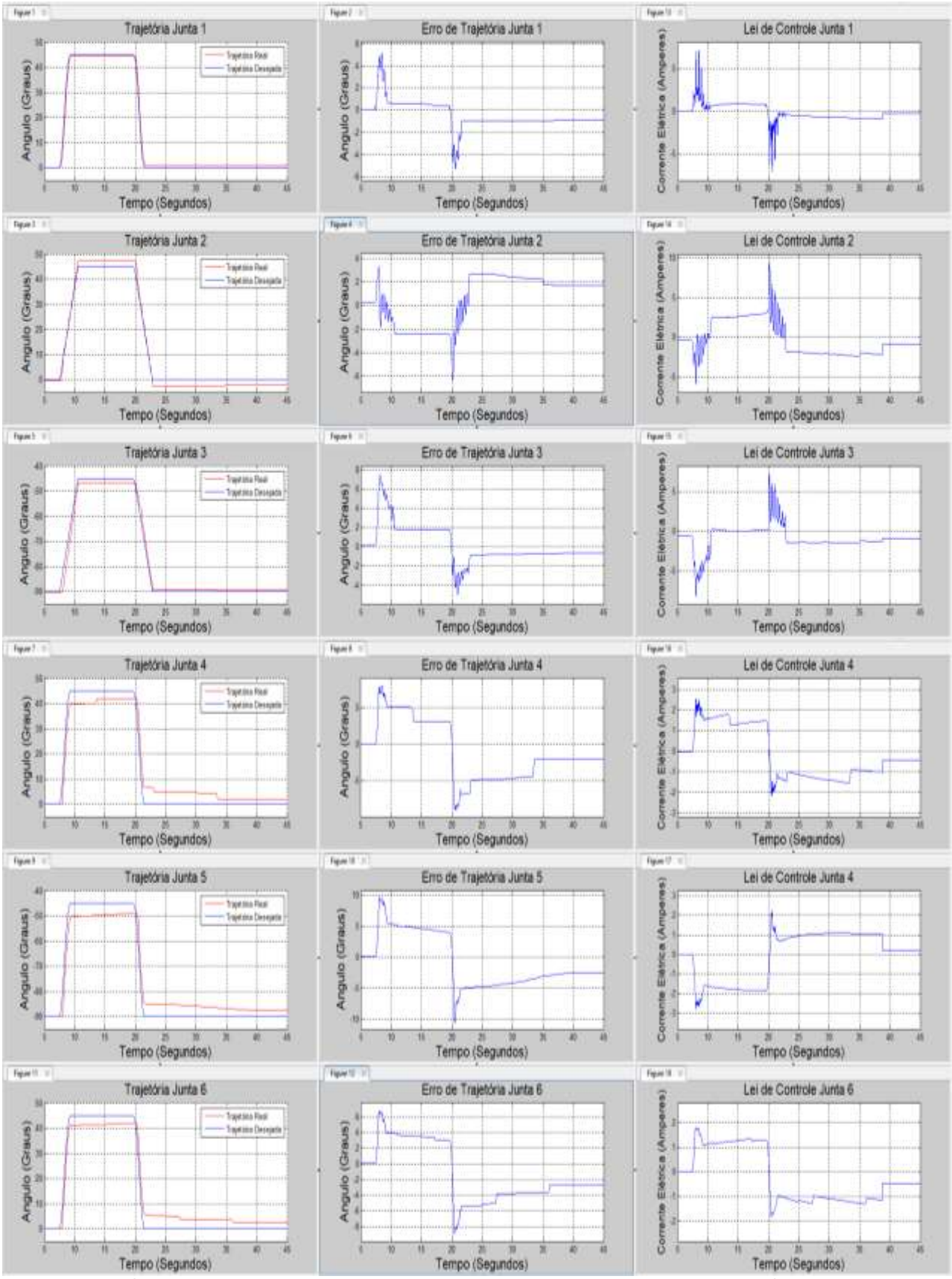

Figura 7.2 - Trajetórias e Erros de Rastreamento para Controle por Torque Computado com Compensador PID e Trajetória de 2 segundos

(Próprio Autor)

Na figura 7.2 podemos notar que todas as juntas tiveram uma resposta estável. As juntas 1 e 3 obtiveram a melhor resposta por ter alcançado os menores picos de erro (pico maximo de erro de aproximadamente 7,5°). Já as juntas 4, 5 e 6 obtiveram picos de erro de aproximadamente 10°. Além disso também é possível notar que as juntas 1 e 3 obtiveram tempo de assentamento pequeno, enquanto as demais juntas demoraram mais de 20 segundos até diminuírem consideravelmente o erro de trajetória.

A figura 7.3 mostra o segundo set point aplicado no sistema. Nessa aplicação o set point tem inicio aos 7 segundos e termino no instante 80 segundos, nesse caso cada junta saiu da sua posição de origem e obteve uma variação de 45° durante 10 segundos, posteriormente a subida, cada junta manteve na posição desejada por aproximadamente 25 segundas e por fim retornou à posição de origem em 10 segundos. As posições de set point tanto de subida como de descida foi gerado de acordo com o interpolador de quinta ordem discutido anteriormente A Figura 7.3 mostra ao lado de cada set point os respectivos erros de trajetória.

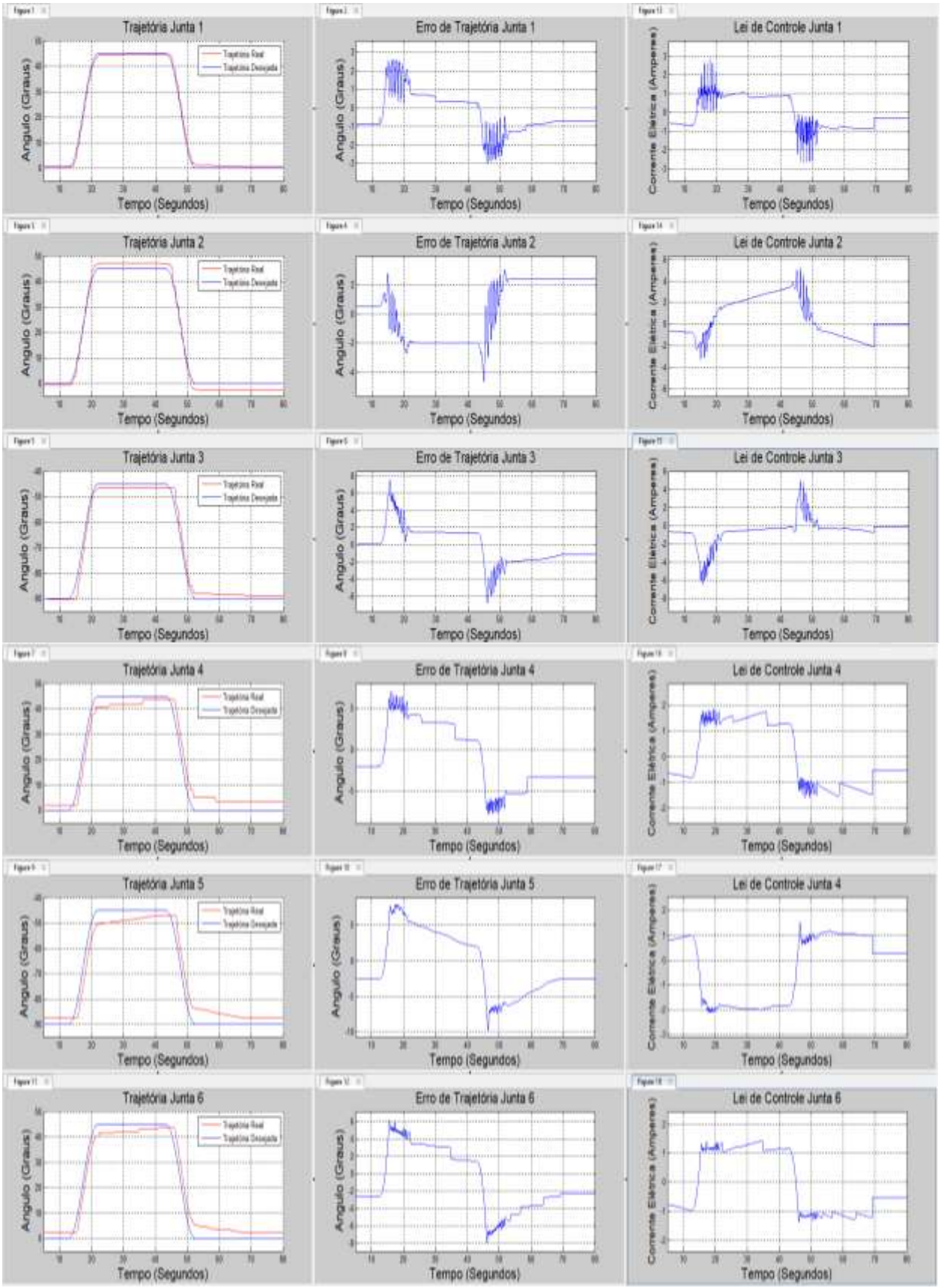

Figura 7.3 - Trajetórias e Erros de Rastreamento para Controle por Torque Computado com Compensador PID e Trajetória de 10 segundos (Próprio Autor)

Os dados mostrados na Figura 7.3 mostram que o controle por torque computado e compensador PID obteve resposta estável e com menor erro de trajetória mediante a trajetórias com transições de posição mais suave do que com transições de posições mais agressivas. Visualmente podemos notar que as juntas 1 e 3 novamente obtiveram as melhores repostas, tanto por possuírem baixo pico de erro (aproximadamente 7,5° no pior caso) como por serem as juntas com menor tempo de assentamento. As juntas 2, 4, 5 e 6 obtiveram respostas lentas, sendo que 30° após termino da transição de posição essas juntas não alcançaram a posição final desejada.

7.2 CONTROLE POR TORQUE COMPUTADO COM COMPENSADOR PID E ESTRUTURA VARIÁVEL

O controle por estrutura variável tem a características de garantir a estabilidade de um sistema mesmo mediante a presença de ruídos. Como discutido no Capítulo 4, o controle por estrutura variável garante a estabilidade de um sistema se o ganho $\beta > |\tau - \hat{\tau}|$. Portanto foi utilizada a seguinte lei de controle $\tau = \widehat{M}(\theta)\left(\ddot{\theta}_d - K_v\dot{e} - K_p e - K_i\varepsilon\right) + \hat{V}\left(\theta, \dot{\theta}\right)\dot{\theta} + \hat{F}\left(\dot{\theta}\right) + \hat{G}(\theta) + \beta sign(Y) + \alpha Y$, onde $\widehat{M}(\theta), \hat{V}\left(\theta, \dot{\theta}\right), \hat{F}(\theta)$ e $\hat{G}(\theta)$ é obtido com o auxilio do toolbox de robótica desenvolvido por Peter Corke. Os ganhos K_v, K_p e K_i foram selecionados de modo a satisfazer as condições dadas pelo conjunto de inequações (3.21). Para tentar diminuir a influencia das incertezas paramétricas no sistema, optou-se empiricamente por constantes elevadas que satisfaçam as condições (3.21) e que ao mesmo tempo não saturem o sinal de controle e que aproxime a resposta do sistema para uma resposta criticamente amortecida. Após diversos ensaios, os seguintes valores de ganho proporcionaram uma boa resposta ao sistema: $K_i = diag(4\ 4\ 4\ 4\ 4\ 4)$, $K_p = diag(80\ 80\ 80\ 80\ 80\ 80)$ e $K_v = diag(80\ 80\ 80\ 80\ 80\ 80)$. Respeitando as restrições apresentadas em (4.10) foi obtido os seguintes valores de F e α:

$$F = \begin{bmatrix} diag(1.73\ 1.73\ 1.73\ 1.73\ 1.73\ 1.73) \\ diag(2.05\ 2.05\ 2.05\ 2.05\ 2.05\ 2.05) \\ diag(0.90\ 0.90\ 0.90\ 0.90\ 0.90\ 0.90) \end{bmatrix}^T, \tag{7.1}$$

$$\alpha = diag(64\ 64\ 64\ 16\ 16\ 8). \tag{7.2}$$

O diagrama padrão da implementação é dado na Figura 7.4, onde o bloco "Level 2 MATLAB S Function" é responsável por fazer a função sinal e multiplicar pelo ganho β conforme descrito no apêndice E.

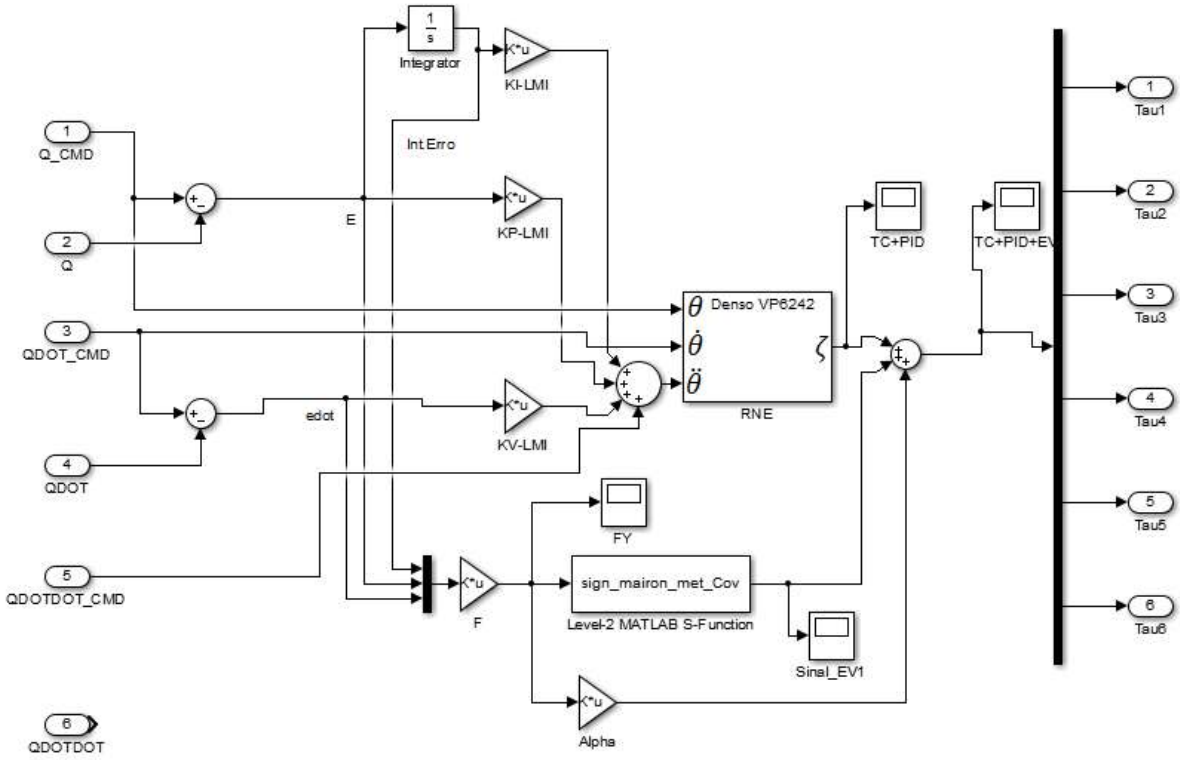

Figura 7.4 - Implementação de Torque Computado com Compensador PID e Estrutura Variável Utilizando o Robotic Toolbox (Próprio Autor)

O sistema (1) mediante a lei de controle é comprovado ser estável, pois as inequações (4.8) são garantidas para

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}, \tag{7.3}$$

sendo: $P_{11} = diag(157.1\ 157.1\ 157.1\ 160.6\ 160.6\ 160.1)$;

$P_{12} = diag(85.2\ 85.2\ 85.2\ 74.5\ 74.5\ 84.6)$;

$P_{13} = diag(1.73\ 1.73\ 1.73\ 1.73\ 1.73\ 1.73)$;

$P_{21} = diag(85.2\ 85.2\ 85.2\ 74.5\ 74.5\ 84.6)$;

$P_{22} = diag(5631.7\ 5631.7\ 5631.7\ 1716.5\ 1716.5\ 1021.8)$;

$P_{23} = \ diag(2.05\ 2.05\ 2.05\ 2.05\ 2.05\ 2.05)$;

$P_{31} = diag(1.73\ 1.73\ 1.73\ 1.73\ 1.73\ 1.73)$;

$P_{32} = diag(2.05\ 2.05\ 2.05\ 2.05\ 2.05\ 2.05)$;

$P_{33} = diag(0.90\ 0.90\ 0.90\ 0.90\ 0.90\ 0.90)$.

Após efetuar a implementação da lei de controle acima, foi constatado que o erro de torque estimado ($\tilde{\tau}$) tinha valores inferiores a [2; 20; 20; 2; 6.67; 4], portanto esses valores foram utilizados como β do sistema.

Para essa topologia também foi utilizado como set point o mesmo interpolador de quinta ordem discutido anteriormente. As Figuras 7.5 e 7.6 mostram o set point, e o sinal de erro de trajetória para cada junta. De maneira análoga à aplicação da Seção 7.1, foi aplicado um set point com inicio no instante 5 segundos e termino no instante 30 segundos, de modo que entre os instante 9 segundos e 11 segundos o set point varia a posição de cada junta em 45° de acordo com a posição gerada pelo interpolador de quinta ordem discutido anteriormente. Posteriormente, do instante 11 segundos até o instante 19 segundos o sistema mantém sem trocar o set point, entre os instante 19 segundos e 21 segundos o set point regressa a posição de origem de acordo com a posição gerada pelo interpolador de quinta ordem discutido anteriormente e por fim o set point mantém a posição de origem entre os instantes 19 e 30 segundos.

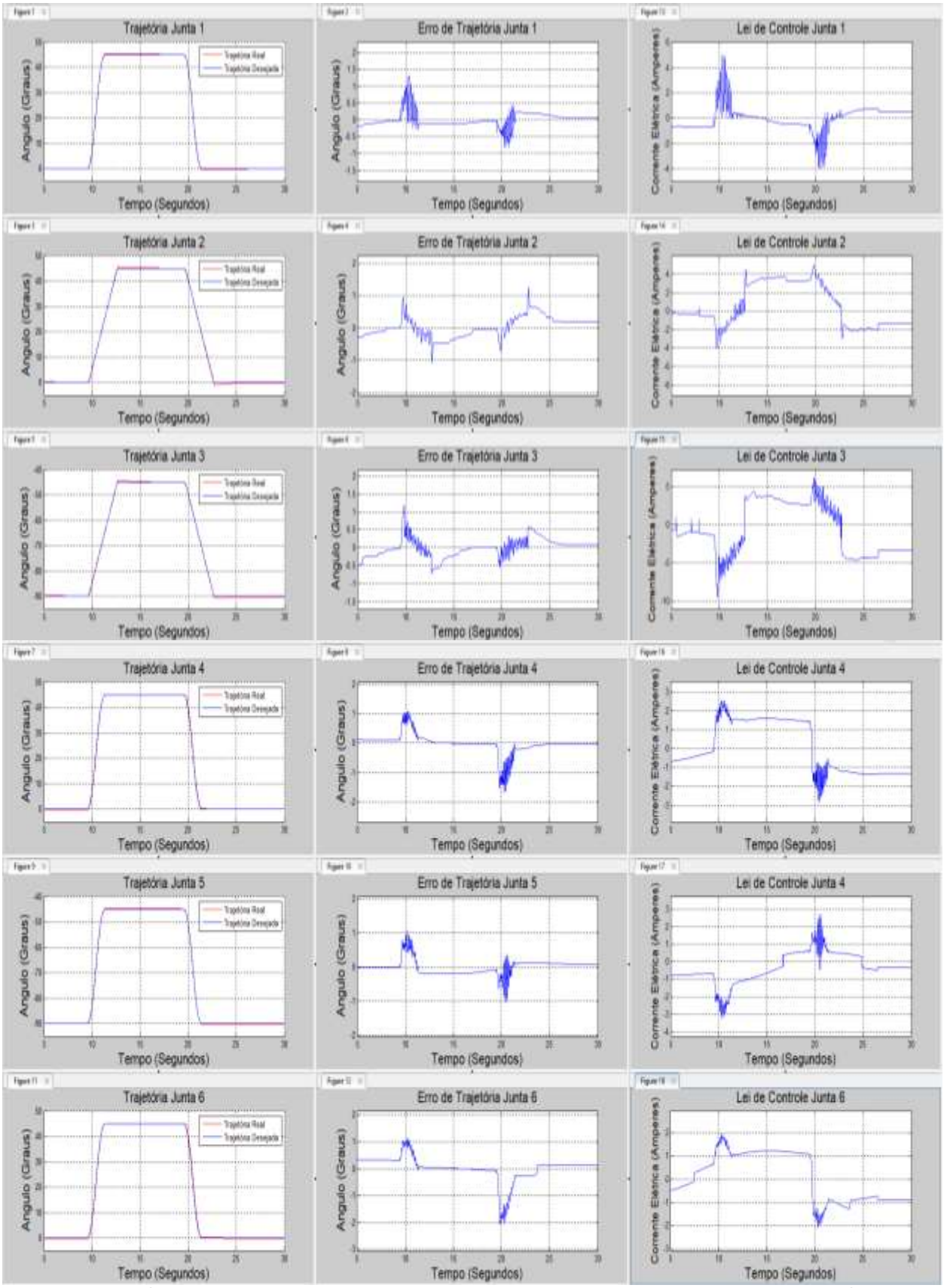

Figura 7.5 - Trajetórias e Erros de Rastreamento para Controle por Torque Computado com Compensador PID e Estrutura Variável e Trajetória de 10 segundos (Próprio Autor)

Na Figura 7.5 podemos notar que todas as juntas tiveram uma resposta estável e com desempenho muito melhor do que o controlador por torque computado e compensador PID. Todos as juntas tiveram pico inferiores a 2.5° e todas as juntas obtiveram tempo de assentamento baixo (inferior a 2.5 segundos).

A Figura 7.6 mostra o segundo set point aplicado no sistema. Nessa aplicação o set point tem inicio aos 7 segundos e termino no instante 60 segundos. Entre o instante 11 segundos e 21 segundos o set point saiu da sua posição de origem e obteve uma variação de 45°, posteriormente cada junta manteve na posição desejada por aproximadamente 20 segundas e por fim, entre os instante 41 segundos e 51 segundos o set point retornou à posição de origem. A Figura 7.6 mostra ao lado de cada set point os respectivos erros de trajetória.

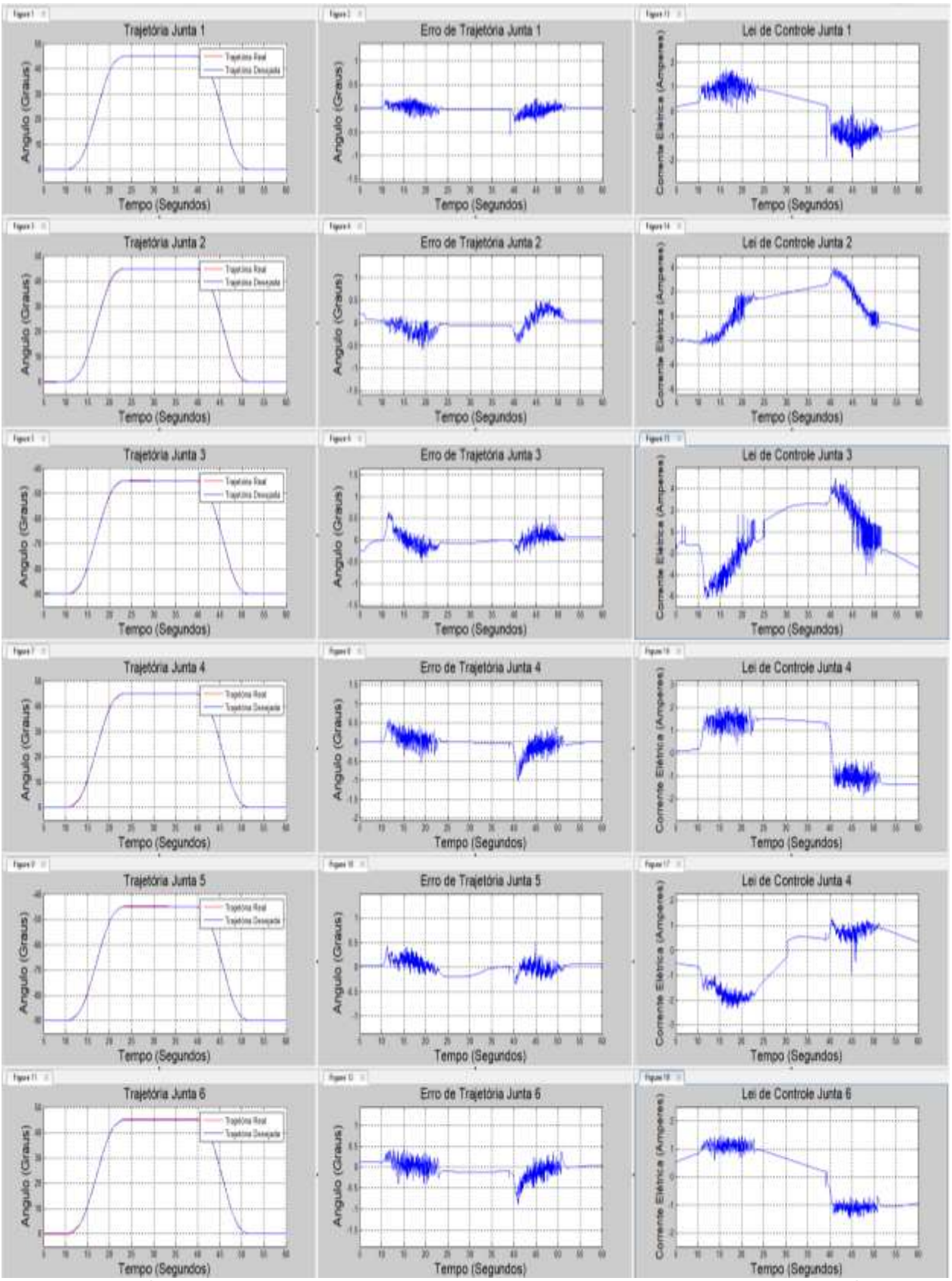

Figura 7.6 - Trajetórias e Erros de Rastreamento para Controle por Torque Computado com Compensador PID e Estrutura Variável e Trajetória de 10 segundos (Próprio Autor)

Na Figura 7.6 podemos notar que todas as juntas tiveram uma resposta estável e com desempenho ainda melhor do que o controlador anterior. Todos as juntas tiveram pico inferiores a 1°, o que significa que o sistema possui um controle de rastreamento com performance muito boa. Comparando as Figuras 7.5 e Figura 7.6 respectivamente com as Figuras 7.2 e 7.3 é possível notar que o controle por estrutura variável não desestabilizou o sistema e teve contribuições relevantes para o controle do robô com modelo dinâmico incerto.

7.3 CONTROLE UTILIZANDO REDES NEURAIS E ESTRUTURA VARIÁVEL

Devido a dificuldade de conhecer os parâmetros dinâmicos de um manipulador, muitas vezes é difícil extrair uma estimativa do modelo dinâmico boa o suficiente para aplicar controle por torque computado. Uma maneira de suprir a necessidade do modelo dinâmico do manipulador é optando pelo uso de redes neurais para que essa possa estimar o torque do manipulador e posteriormente servir como ferramenta que estime o cancelamento das não linearidades do manipulador. Nessa seção será descrita os métodos de treinamento de redes neurais e posteriormente será relatado os resultados do controle do manipulador Denso VP6242 fazendo uso de redes neurais juntamente com estrutura variável.

7.3.1 TREINAMENTO DE REDES NEURAIS PARA O CONTROLE DE MANIPULADORES ROBÓTICOS

Para o presente projeto foi treinado um conjunto de redes neurais que simula o comportamento dinâmico do sistema. Como visto anteriormente, as redes neurais podem ser aplicadas para fazer uma estimativa de linearização por realimentação do sistema. Nessa seção será detalhado aspectos construtivos da rede desenvolvida.

Como visto anteriormente, um manipulador comporta-se semelhante ao modelo dinâmico direto do mesmo, isso é, para manipuladores reais aplica-se um torque (ou sinal de corrente elétrica ou tensão elétrica que está relacionado com o torque resultante dos atuadores) como sinal de entrada, e obtém-se a trajetória executada como sinal de saída. Objetivando treinar uma rede que simule o comportamento dinâmico inverso de um manipulador, foi aplicado um determinado sinal de torque no manipulador e foi lido a trajetória do mesmo. Posteriormente foi treinado um conjunto de redes que mediante a entrada da trajetória obtida, pudesse estimar o torque que foi aplicado.

Devido a grande complexidade em treinar uma rede que estime os torques para o robô, nesse trabalho é utilizado duas redes neurais para exercer essa função. Em ambas as redes, uma vez que os dados para treinamento das redes foram extraídos, foi feito uma normalização dos dados para fins de utilizar as funções de ativação com máximo unitário sem prejudicar a dinâmica do sistema. Os treinamentos foram feitos usufruindo do toolbox *Neural Network Toolbox* versão 7.0.3 do Matlab.

A primeira rede neural é responsável por simular o comportamento das três primeiras juntas. Para obtenção dessa rede foram utilizados como sinal de entrada as posições, velocidades e acelerações angulares das três primeiras juntas, e como alvo da rede foi utilizado o sinal de torque das três primeiras juntas, conforme Figura 7.7.

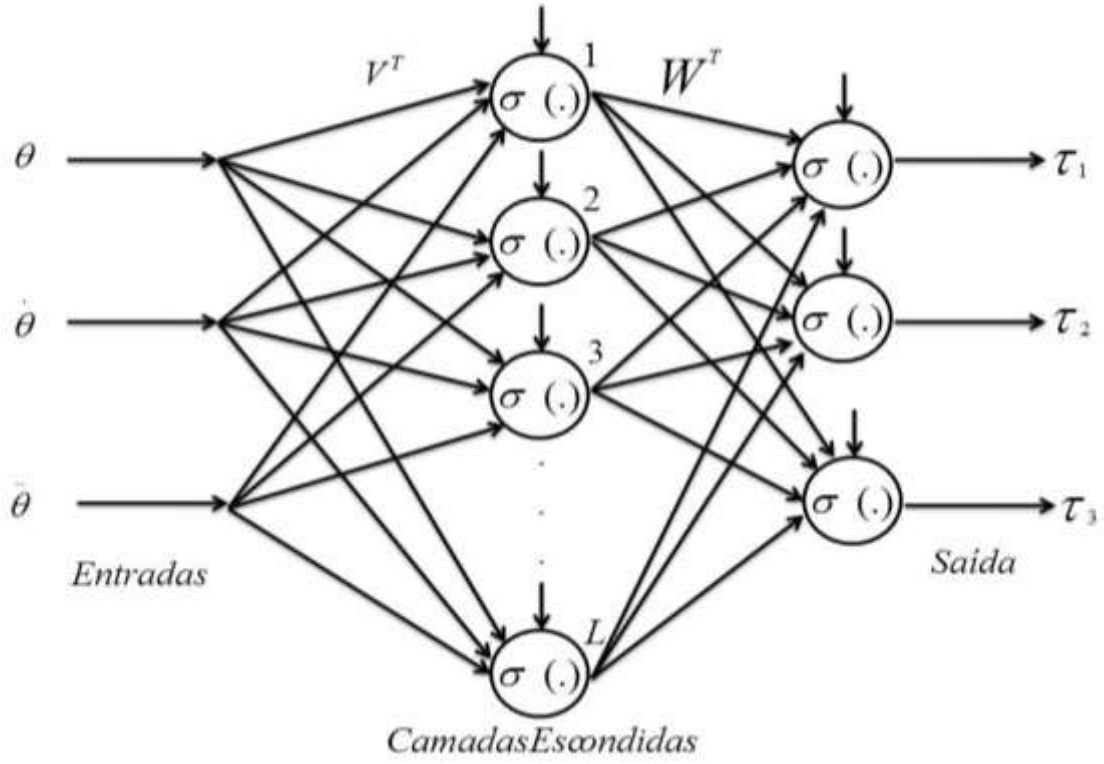

Figura 7.7- Treinamento da Rede Neural 1 (Próprio Autor)

Na Figura 7.7, os sinais $\theta, \dot{\theta}$ e $\ddot{\theta}$ são vetores de dimensão três e representam respectivamente as posições, velocidades e acelerações angulares das três primeiras juntas. Os sinais τ_1, τ_2 e $\tau_3$são grandezas escalares que representam os torques exercidos pela primeira, segunda e terceira junta respectivamente. A Figura 7.8 mostra os sinais que foram utilizados como entradas para treinamento da rede.

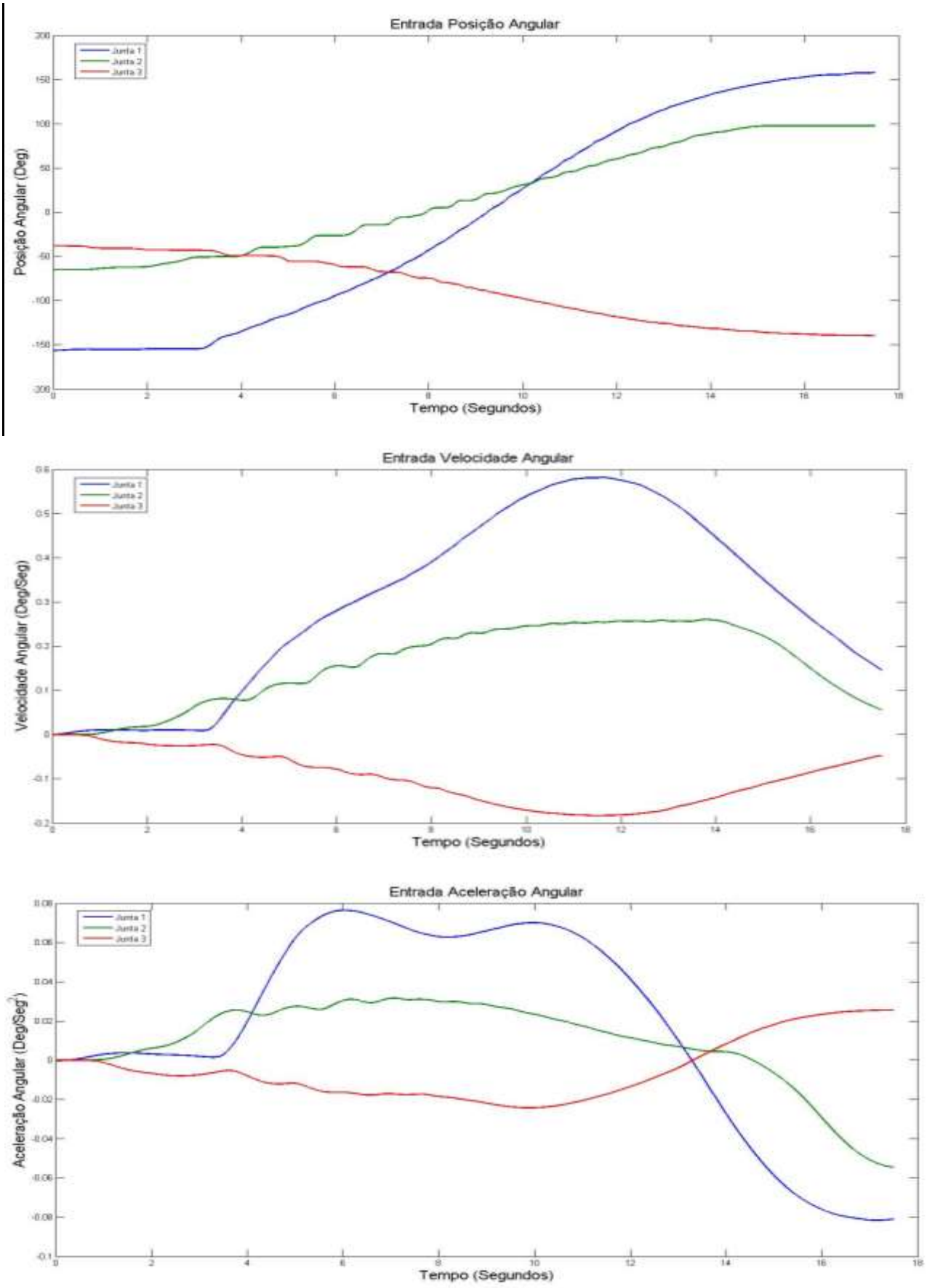

Figura 7.8 - Sinais de Entradas para Rede 1 (Próprio Autor)

Como é possível notar nas Figuras 7.8, os dados de entrada foram obtidos a partir de uma movimentação feita do manipulador, onde o manipulador sai da posição [-150°; -65°; -38°] e atingiu a posição [150°; 100°; -140°] em aproximadamente 18 segundos. Devido ao fato da rede neural possuir três saídas, é necessário que a ultima camada de neurônio possua três neurônios. Após diversa combinação de quantidade de camadas (variando de 2 a 4 camadas) e diversas quantidades de neurônios, foi constatado que a melhor resposta obtida foi com uma rede multicamada com três camadas de neurônios, onde cada camada possui três, seis e três neurônios respectivamente.

A seleção de função de ativação foi feita empiricamente entre as funções de ativação que tenham máximo unitário e que não possua descontinuidades ou variações abruptas. Para essa rede neural foram testados diversas combinações entre as seguintes funções de ativaçãos: linear limitada, sigmóide tangente hiperbólica, sigmóide logarítmica e base radial. Após vários testes executados foi constatado que a melhor resposta foi obtida com funções de ativação do tipo sigmoide tangente hiperbólica para as três juntas.

Para treinamento dessa rede foi observado empiricamente que a utilização de um coeficiente de aprendizado igual a 0.005 juntamente com o método do gradiente descentralizado proporcionava boa convergências com velocidade superior a outras combinações de método de treinamento e coeficiente de aprendizado.

Uma forma de avaliar a eficiência da rede treinada em relação aos dados originais é utilizando o índice de regressão. Quanto mais próximo de 1 o índice de regressão for, significa maior eficiência da rede neural em estimar os dados utilizados como treinamento. Como resultado final da combinação de quantidade de camada de neurônios, quantidade de neurônio por camada, função de ativação, método de treinamento e coeficiente de aprendizado acima citado foi obtido uma rede neural com índice de regressão $R = 0.9899$. A Figura 7.9 mostra os dados utilizados como alvo para treinar as redes (dados reais extraídos do robô) juntamente com os dados estimados pela rede treinada para as três primeiras juntas.

A Figura 7.9 mostra que os dados estimados sobre escrevem os dados reais com uma boa eficiência, isso significa que a rede neural obteve êxito na tarefa de estimar o modelo dinâmico do robô. A seguir serão abordados os parâmetros utilizados para o treinamento da segunda rede neural.

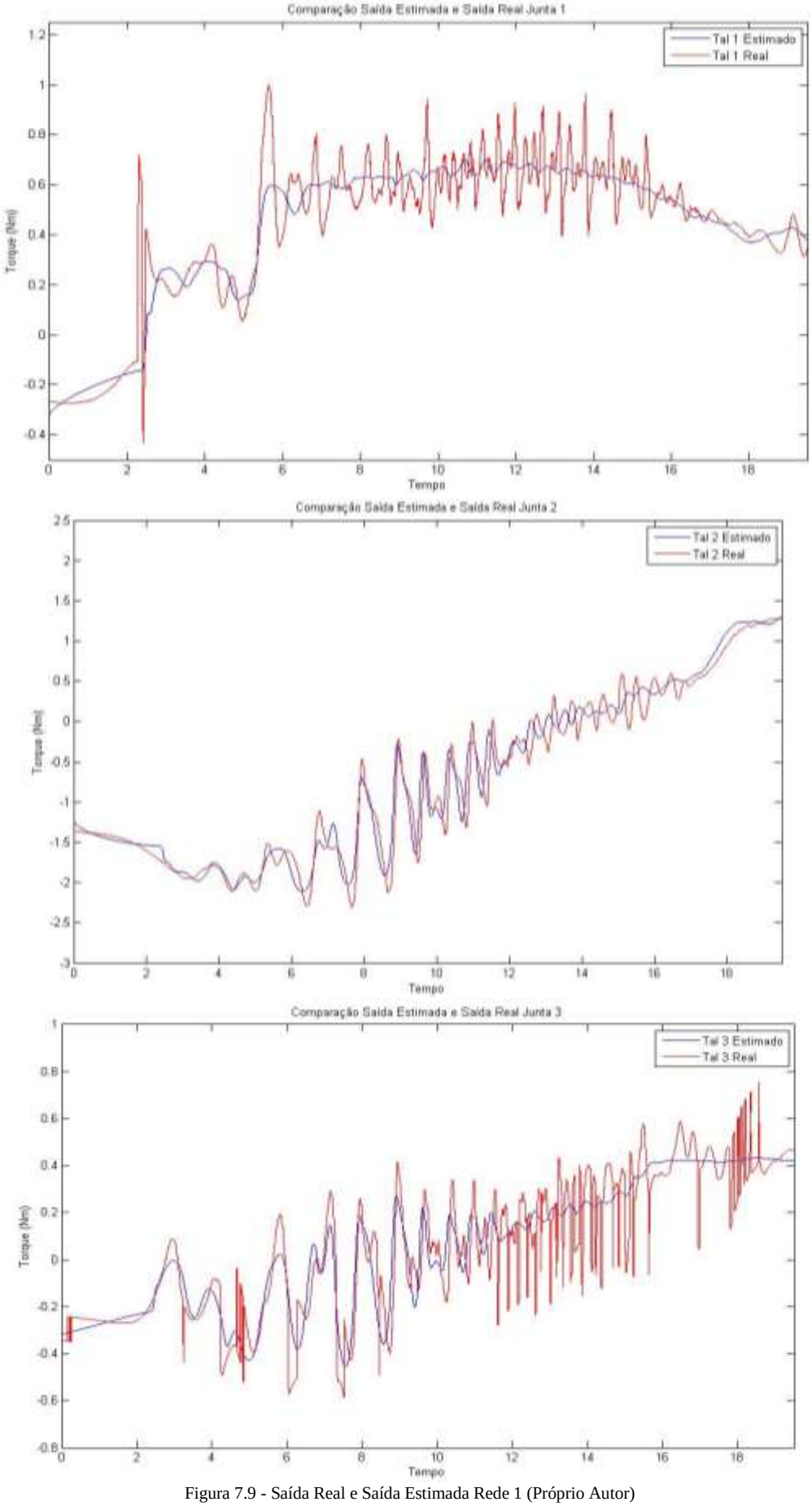

Figura 7.9 - Saída Real e Saída Estimada Rede 1 (Próprio Autor)

A segunda rede neural é responsável por simular o comportamento das três ultimas juntas. Para obtenção dessa rede foram utilizados como sinal de entrada as posições, velocidades e acelerações angulares das três primeiras juntas, e como alvo da rede foi utilizado o sinal de torque das três primeiras juntas, conforme Figura 7.10.

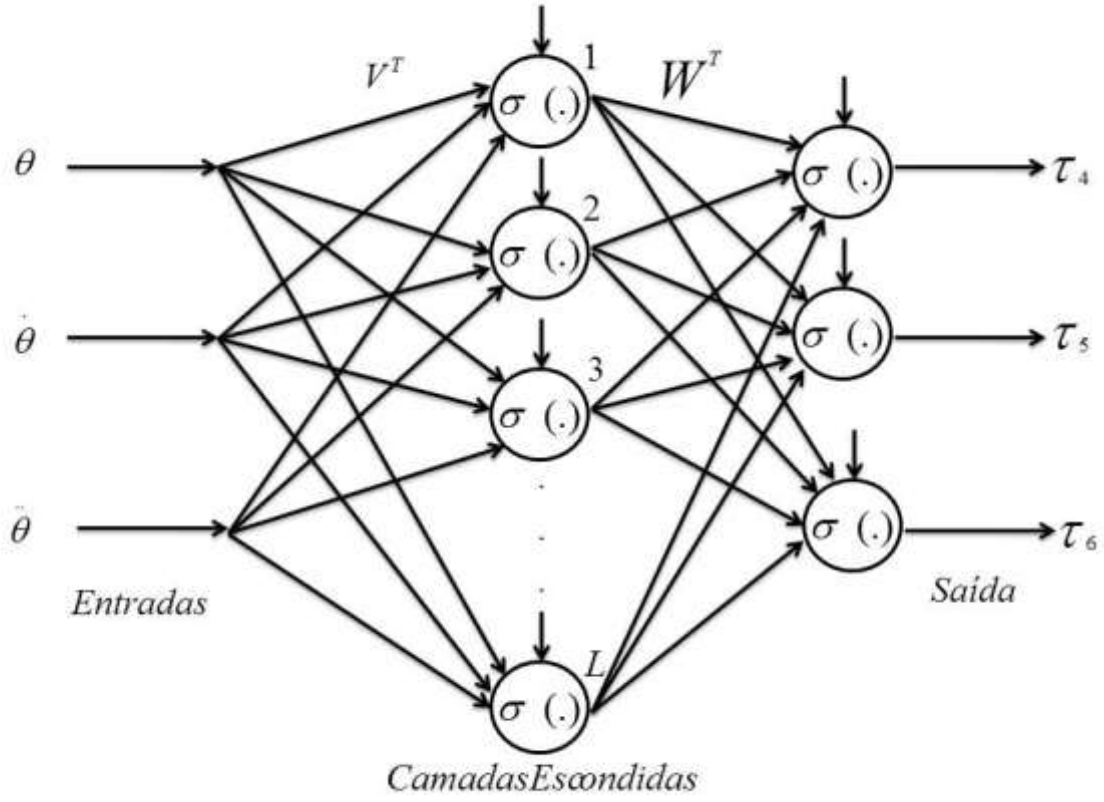

Figura 7.10 - Treinamento da Rede Neural 2 (Próprio Autor)

Na Figura 7.10, os sinais $\theta, \dot{\theta}$ e $\ddot{\theta}$ são vetores de dimensão três e representam respectivamente as posições, velocidades e acelerações angulares das três ultimas juntas. Os sinais τ_4, τ_5 e τ_6 são grandezas escalares que representam os torques exercidos pela quarta, quinta e sexta junta respectivamente. A Figura 7.11 mostra os sinais de entradas da rede treinada.

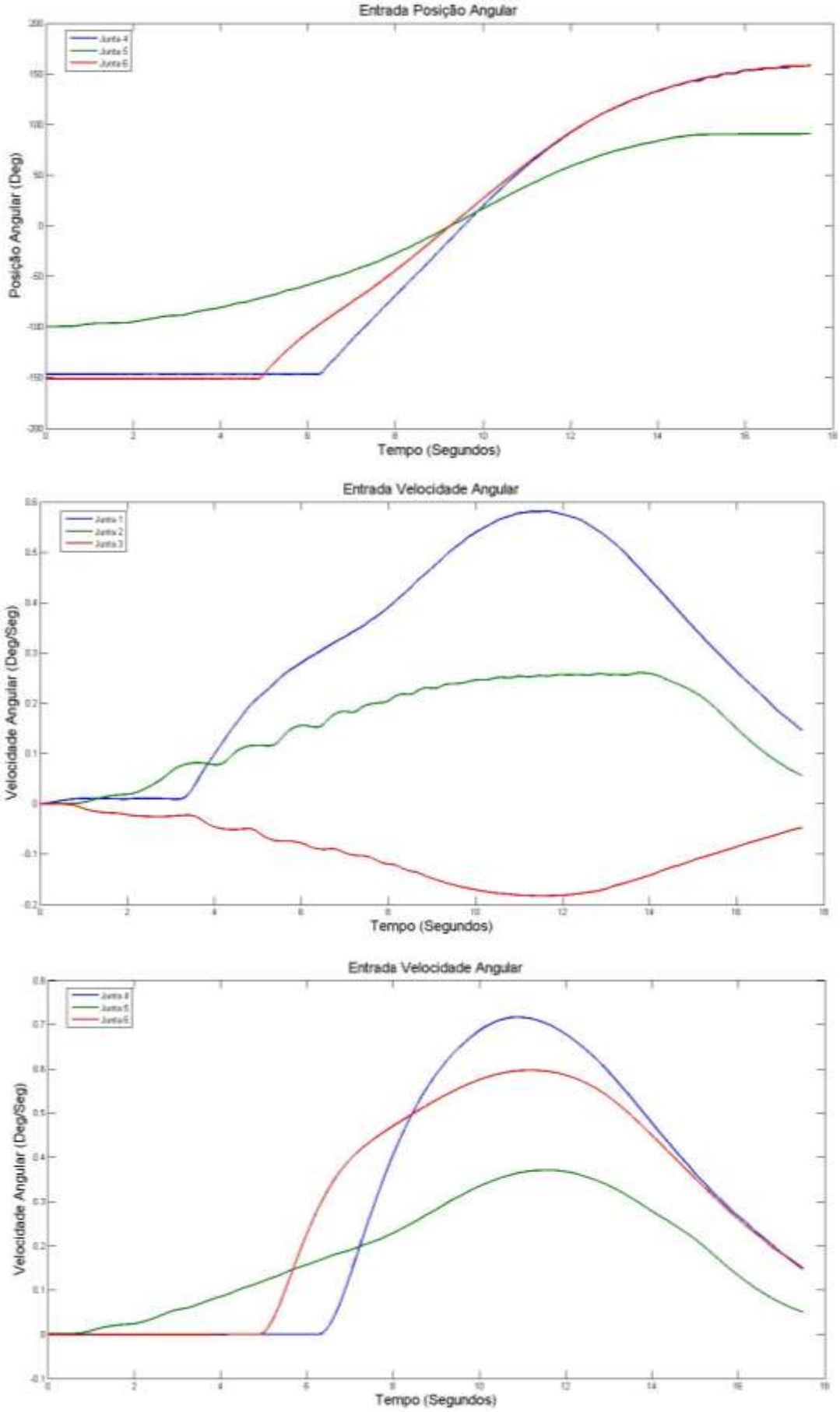

Figura 7.11 - Sinais de Entrada Rede 2 (Próprio Autor)

Como é possível notar na Figura 7.11, os dados de entrada foram obtidos a partir de uma movimentação feita do manipulador, onde o manipulador sai da posição [-160°; 100°; -160°] e atingiu a posição [160°; 100°; 160°] em aproximadamente 18 segundos. De maneira análoga ao treinamento da primeira rede neural, foram feitos diversos testes variando os seguintes parâmetros da rede: quantidade de camada de neurônios (de 2 a 4 camadas); quantidade de neurônio por camadas (desde que a ultima camada tivesse três neurônios); funções de ativação (desde que tivessem máximos unitário e sem descontinuidade ou mudanças de valores abruptas); e valor do coeficiente de aprendizado para o método do gradiente descentralizado.

Após diversos testes, a melhor rede obtida foi utilizada uma rede multicamada com três camadas de neurônios, onde cada camada possui três, doze e três neurônios respectivamente. Essa rede foi treinada utilizando as funções de ativação linear, base radial e linear limitado. Os pesos sinápticos foram obtidos utilizando o método do gradiente descentralizado com coeficiente de aprendizado igual a 0.001. Como resultado foi obtido uma rede com índice de regressão $R = 0.9990$. A Figura 7.12 mostra os dados utilizados como alvo para treinar a rede juntamente com os dados estimados pela rede treinada.

A Figura 7.12 mostra que os dados estimados conseguem sobre escrever os dados reais com eficiência elevadissima, isso significa que a rede neural obteve êxito na tarefa de estimar o modelo dinâmico do robô.

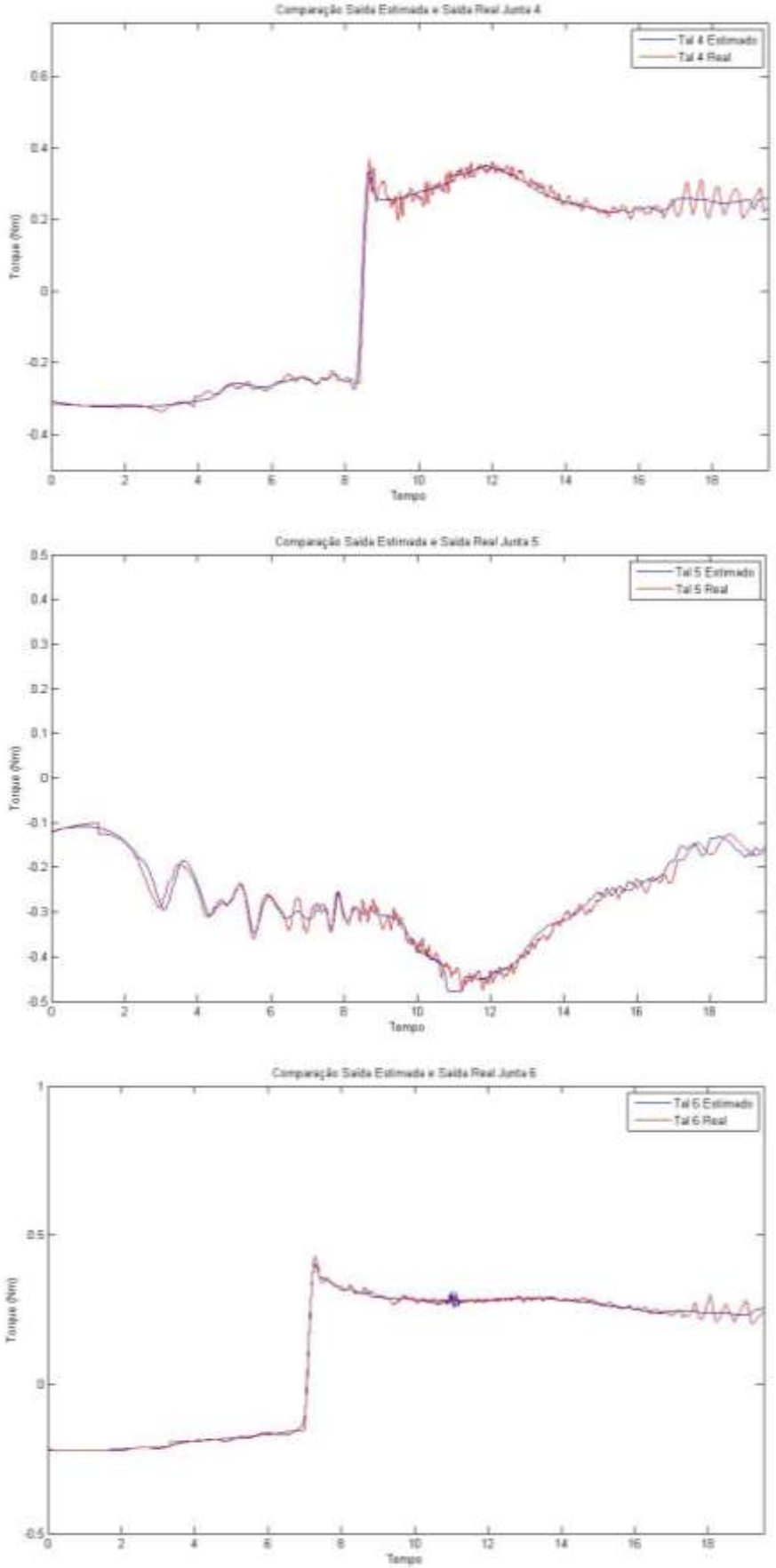

Figura 7.12 - Saída Real e Saída Estimada Rede 2 (Próprio Autor)

7.3.2 PROJETO E IMPLEMENTAÇÃO DE CONTROLE UTILIZANDO REDES NEURAIS E ESTRUTURA VARIÁVEL

As condições de projeto para as redes neurais são apresentadas na Seção 5.2. A implementação do controle utilizando redes neurais e controle por estrutura variável é apresentado na Figura 7.13. A lei de controle utilizada foi $\tau = \hat{f}(x) + \alpha Y - \beta sign(Y)$, tal que $\beta > \epsilon_n$, isso é, beta é maior que o erro de aproximação funcional da rede. Embora a dinâmica do robô seja um sistema acoplado, a utilização de duas redes neurais em paralelo juntamente com o método de controle robusto por estrutura variável e modos deslizantes consegue estabilizar o robô desde que o erro de aproximação universal das redes treinadas seja menor que a constante β da lei de controle, isto é, $\beta > \epsilon_n$. Após efetuar a implementação do estimador de torque, foi constatado empiricamente que o erro de estimação de torque tinha valores inferior a [2; 20; 20; 2; 6.67; 4], portanto esses valores foram utilizados como β do sistema.

A estrutura variável utilizada foi a mesma da implementação anterior, com:

$$F = \begin{bmatrix} diag(1.73\ 1.73\ 1.73\ 1.73\ 1.73\ 1.73) \\ diag(2.05\ 2.05\ 2.05\ 2.05\ 2.05\ 2.05) \\ diag(0.90\ 0.90\ 0.90\ 0.90\ 0.90\ 0.90) \end{bmatrix}^T, \tag{7.4}$$

$$\alpha = diag(64\ 64\ 64\ 16\ 16\ 8). \tag{7.5}$$

O subsistema “INNER_LOOP_NN” possui as duas redes neurais artificiais detalhadas acima.

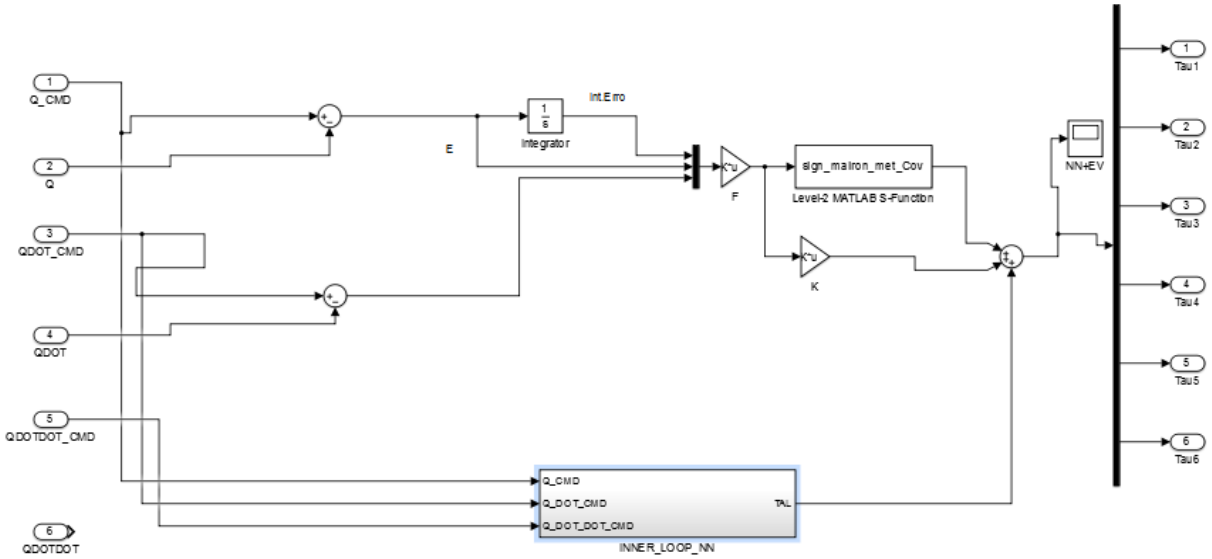

Figura 7.13 - Implementação de Controle com Redes Neurais e Estrutura Variável (Próprio Autor)

Objetivando verificar a resposta dessa topologia, foi utilizado como set point o mesmo interpolador de quinta ordem discutido anteriormente. A Figura 7.14 e Figura 7.15 mostram o set point, e o sinal de erro de trajetória para cada junta. Assim como na aplicação da Seção 7.1 e 7.2, inicialmente foi aplicado um set point com inicia no tempo 5 segundos e termina no tempo 30 segundos, tal que no instante 12 segundos começa a variar a posição de cada junta em 45° durante um intervalo de tempo igual a 2 segundos. Posteriormente o sistema mantém sem trocar o set point do instante 14 segundos até o instante 23 segundos, entre os instante 23 segundos e 25 segundos o set point regressa a posição de origem de acordo com a posição gerada pelo interpolador de quinta ordem discutido anteriormente e por fim o set point mante a posição de origem entre os instantes 25 e 30 segundos. A Figura 7.14 mostra ao lado de cada set point os respectivos erros de trajetória.

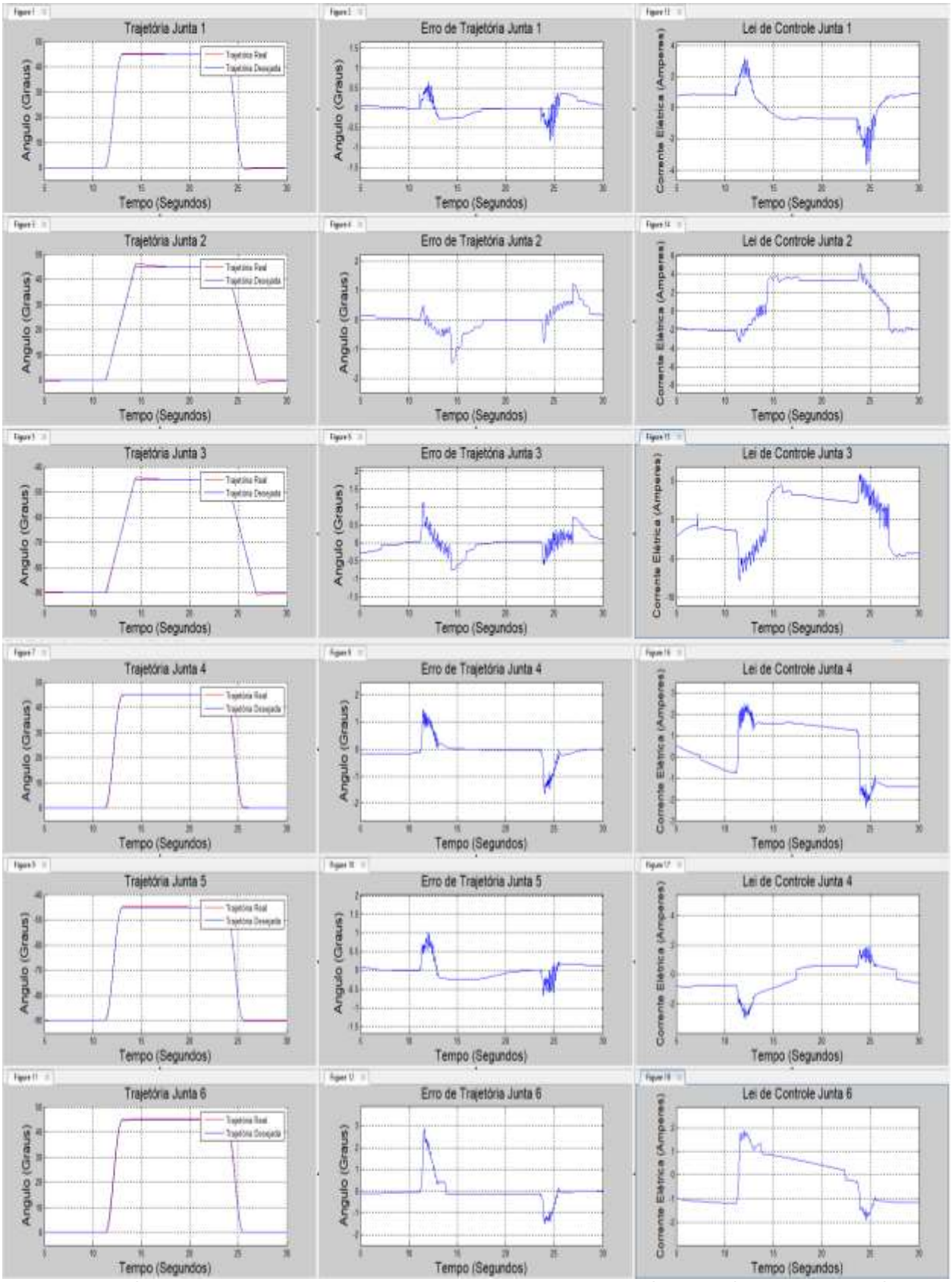

Figura 7.14 - Trajetórias e Erros de Rastreamento para Controle por Redes Neurais e Estrutura Variável e Trajetória de 2 segundos (Próprio Autor)

Na Figura 7.14 podemos notar que todas as juntas tiveram uma resposta estável e com desempenho muito melhor do que o controlador por torque computado e compensador PID. Os piores desempenho foram encontrados nas juntas 4 e 6, tendo picos de erro inferiores a 3°, as demais juntas obtiveram picos de erros inferiores a 1.5°. Todos as juntas tiveram tempo de assentamento baixo e erro de regime permanente praticamente nulo.

A Figura 7.15 mostra o segundo set point aplicado no sistema. Nessa aplicação o set point tem inicio aos 7 segundos e termino no instante 60 segundos. Entre o instante 10 segundos e 20 segundos o set poinr saiu da sua posição de origem e obteve uma variação de 45°, posteriormente cada junta manteve na posição desejada por aproximadamente 15 segundas e por fim, entre os instante 35 segundos e 45 segundos o set point retornou à posição de origem. A Figura 7.15 mostra ao lado de cada set point os respectivos erros de trajetória.

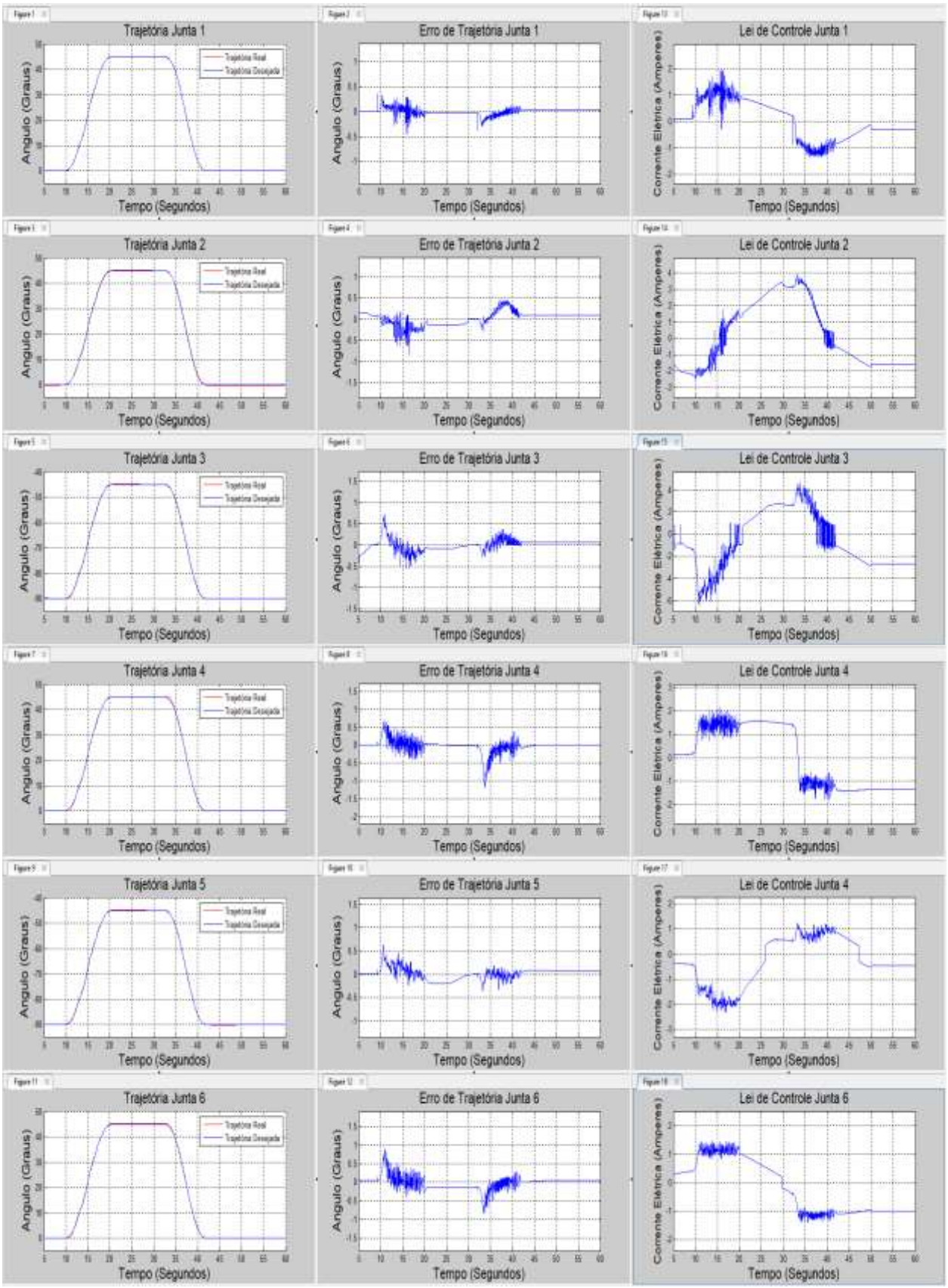

Figura 7.15 - Trajetórias e Erros de Rastreamento para Controle por Redes Neurais e Estrutura Variável e Trajetória de 10 segundos (Próprio Autor)

Na Figura 7.15 podemos notar que todas as juntas tiveram uma resposta estável e com desempenho ainda melhor do que a Figura 7.14. Todos as juntas tiveram pico inferiores a 1°, o que significa que o sistema possui um controle de rastreamento com performance muito boa. Comparando as Figuras 7.14 e 7.15 respectivamente com as Figuras 7.2 e 7.3 é possível notar que o controle por estrutura variável utilizando modelo estimado por redes neurais não desestabilizou o sistema e teve contribuições relevantes para o controle do robô com modelo dinâmico incerto.

Os resultados obtidos das seções 7.3.2 e 7.2 são suficientes para comprovar que o método de controle utilizando controle por estrutura variável é eficiente para controlar sistemas robóticos com modelo dinâmico incerto. A combinação do método de controle proposto com a utilização de redes neurais obteve boa resposta (conforme Figuras 7.14 e 7.15), e essa combinação de métodos pode ser utilizada para controlar sistemas robóticos que não possuem modelo dinâmico.

8 CONCLUSÕES FINAIS

Foi apresentado um método de controle de juntas de um dispositivo robótico com modelo dinâmico incerto utilizando controle por estrutura variável e modos deslizantes com condições baseadas em LMIs. O presente trabalho utilizou duas abordagens diferentes para gerar o modelo dinâmico incerto do sistema: utilização de uma estimativa do modelo dinâmico do robô e utilização de redes neurais treinadas para simular o comportamento dinâmico do dispositivo.

Foram implementado três controladores de junta para o dispositivo robótico: torque computado com compensador PID, torque computado com compensador PID e estrutura variável, e redes neurais com estrutura variável. O primeiro método é o método clássico de controle e foi utilizado para servir com base de comparação do método proposto. O controlador por torque computado com compensador PID apresentou erro de trajetória relativamente alto com alto tempo de assentamento. Esse fato é justificado pela ausência de robustez na arquitetura de controle, e pela modelagem estimada com pouca precisão.

Posteriormente foi proposto um método de controle capaz de controlar o robô mesmo na presença de incertezas no modelo dinâmico do mesmo. Essa método apresenta uma solução para compensar as dinâmicas não modeladas e incertezas paramétricas ao fazer uso do controle por estrutura variável baseado em LMI's. Esse método foi implementado utilizando dois modelos dinâmicos incertos diferentes: O modelo dinâmico obtido da modelagem incerta do sistema e o modelo dinâmico baseado em redes neurais.

A primeira aplicação do método proposto foi utilizando o modelo dinâmico incerto do sistema (mesmo modelo utilizado para projetar o controlador por torque computado e compensador PID básico). Nesse caso, o controlador por estrutura variável baseado em LMI's proposto nesse trabalho é utilizado para compensar o distúrbio causado pelas dinâmicas não modeladas e incertezas paramétricas. Os resultados obtidos dessa implementação foram satisfatórias, de modo que o robô obteve estabilidade, com baixo pico de erro de regime e baixo tempo de assentamento. Contudo esse implementação ainda necessita da utilização do modelo dinâmico do sistema (mesmo que seja incerto), e caso não seja possível obter o modelo dinâmico incerto do robô pode ser dificultoso reproduzir o método proposto utilizando essa aplicação.

A segunda aplicação do método proposto soluciona a necessidade de um modelo dinâmico do sistema ao utilizar redes neurais para executar uma aproximação de cancelamento de não linearidades. Nesse caso, o controlador por estrutura variável baseado

em LMI's proposto nesse trabalho é utilizado para compensar o erro de aproximação funcional da rede, e o resultado obtido também é satisfatório. Os resultados obtidos dessa implementação possuem baixo erro de rastreamento e baixo tempo de assentamento, caracterizando em uma resposta muito melhor do que o controle por torque computado com compensador PID básico. A combinação do método de controle proposto com a utilização de redes neurais pode ser utilizada para controlar sistemas robóticos que não possuem modelo dinâmico.

Por fim, os resultados obtidos das seções 7.3.2 e 7.2 são suficientes para comprovar que o método de controle utilizando controle por estrutura variável baseados em LMIs é eficiente para controlar sistemas robóticos com modelo dinâmico incerto e possui desempenho muito melhor do que o método clássico de controle por torque computado com compensador PID.

8.1 SUGESTÕES PARA PESQUISAS FUTURAS

Segue abaixo algumas sugestões de continuação de trabalho para a presente pesquisa:

- Controle de trajetória em nível cartesiano;
- Utilização de ganhos adaptativos para o controle por estrutura variável;
- Controle misto de posição e força aplicado pela ponteira;
- Instalação e utilização de visão robótica para geração de set point.

REFERÊNCIAS

ANDERSON, B. A Simplified Viewpoint of Hyperstability.**IEEE Transactions on Automatic Control, 13:292-294, 1968.**

CALÔBA, L. P. e AGUIRRE, L. A. **Redes Neurais em Modelagem de Sistemas**. Enciclopédia de Automática: Controle e Automação 3. Primeira edição. 2007.

CHEN, Y-F. MITA, T. e WAKUI, S. A New and Simple Algorithm for Sliding Mode Trajectory Control of the Robot Arm. **IEEE Transactions on Automatic Control 35.7, 1990, p 828-829**

CORKE, P.A Robotic Toolbox for MATLAB**.IEEE Robotics & Automation Magazine, V. 3, N. 1, 1996, p. 24-32.**

CORKE,P. **ROBOTIC, VISION AND CONTROL – FUNDAMENTAL ALGORITHMS IN MATLAB**, VOL 73 – Springer, 2011, 570P.

COUNG, P. e NAN, W. Adaptive Trajectory Tracking Neural Network Control with Robust Compensator for Robotic Manipulator**. Neural Computing and Application. 27.2, 2016, p 525-536.**

COVACIC, M. **Controle Automático com Estrutura Variável Utilizando Sistemas ERP e LMI**, (Dissertação de Mestrado), Universidade Estadual Paulista "Júlio de Mesquita Filho", Ilha Solteira - SP, 2001, 101p.

COVACIC, M. **Síntese de Sistemas ERP Baseado em LMIs e Controle com Estrutura Variável**, (Tese de Doutorado), Universidade Estadual Paulista "Júlio de Mesquita Filho", Ilha Solteira - SP, 2001, 222p.

CRAIG, J. **Introduction to Robotics – Mechanics and Control,** Third Edition – Pearson, 2005, 400p.

DEAN-LEON, E. NAIR, S. e KNOLL, A. User Friendly Matlab-Toolbox for Symbolic Robot Dynamic Modeling Used for Control Design **Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference,.p.2181-2188**

DECARLO, R. A. ŻAK, S. H. MATHEW, G. P. Variable Structure Control of Multivariable Systems: A Tutorial. **Proceedings of IEEE, v. 76, n. 3, p.212-232, 1988.**

DENSO Inc. **DENSO ROBOT VP-G SERIE, GENERAL INFORMATION ABOUT ROBOT,** 2011.

DJURIC, A. e ElMaraghy W. Automatic Separation Method for Generation of Reconfigurable 6R Robot Dynamics Equations. **The International Journal of Advanced Manufacturing Technology,2010, p. 831-842.**

DORF, R. e BISHOP, R. **Modern Control Sstems,** Twelfth Edition, Pearson, 2011, 1082p.

HERNANDES, A. G. **Estudo de Modelagem Robótica,** (Trabalho de Conclusão de Curso), Universidade Estadual de Londrina, Londrina – PR, 2014, 113p.

International Federation of Robotics, **Executive Summary WR 2017 Industrial Robots**, disponivel em: <https://ifr.org/downloads/press/Executive_Summary_WR_2017_Industrial_Robots.pdf>. Acessoem09/Jan/2018.

KHALIL, W. e DOMBRE, E**. Modeling, Identification and Control of Robot –** Butterworth-Heinemann**,**2005, 480P.

KHALIL, W. VIJAYALINGAM, A. KHOMUTENKO,B. MUKHANOV, I. LEMOINE, P. e ECORCHARD, G. OpenSYMORO: An Open-Source Software Package for Symbolic Modelling of Robots. **Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference, p. 1206-1211.**

LEWIS, F, DAWSON, D. e ABDALLAH, C. **Robot Manipulator Control – Theory and Practice,** Second Edition – Marcel Dikker, 2004, 614p.

LEWIS, F, JAGANNATHAN, S, YESILDIREK A. **Neural Network Control of Robot Manipulator and Nonlinear System,** CRC Press, 1998, 433p.

LOOI, T. YEUNG B. UMASTHAN, M. e DRAKE, J. KidsArm – An Image-Guided Pedriatric Anastomosis Robot. **Intelligent Robots and Systems (IROS). 2013 IEEE/RSJ International Conference, p 4105-4110.**

LORDELO, A. D. S. **Controle Automático com Estrutura Variável Utilizando Sistemas ERP e LMI**, (Dissertação de Mestrado), Universidade Estadual Paulista "Júlio de Mesquita Filho", Ilha Solteira - SP, 2000.

LYER, S. LOOI, T. e DRAKE, J. A Single Arm, Single Camera System for Automated Suturing. **Robotics and Automation (ICRA), 2013 IEEE International Conference.**

Mathwors, **trangd – Gradient Descent Backpropagation**, disponivel em: <https://www.mathworks.com/help/control/ref/traingd.html>. Acesso em 30/Dez/2017.

MEDJEBOURI, A. e MEHENNAOUI, L. Adaptive Neuro-Sliding Mode Control of PUMA 560 Robot Manipulator. **Journal of Automation Mobile Robotics and Intelligent Sustems, 2016.**

MURRAY, R, Li, Z e Sastry, S. **A Mathematical Introduction to Robotic Manipulation**, CRC Press, 1994, 456p.

NISE, N. **Control System Engineering** Sixth Edition, John Wiley & Sons, 2011, 926p.

OGATA, K. **Modern Control Engineering,** Fifth Edition, Pearson, 2010, 894p.

QUANSER Inc. **DENSO 6 Axis Robot User Manual**. Ontario, Canadá, 2013.

QUANSER Inc. **DENSO 6 Axis Robot Open-Architecture Setup Guide**. Ontario, Canadá, 2013a.

QUANSER, **Basic Communication**, 2017. disponivel em: <http://quanser-update.azurewebsites.net/quarc/documentation/quarc_communications_basic.html>. Acesso em 07/Jan/2018.

SCIAVICCO, L. e SICILIANO, B**. Modeling and Control of Robot Manipulator –** Springer Science & Business Media, 2000, 378P.

SICILIANO, B. e KHATIB, O. eEditores Springer, **Handbook of Robotics Springer–** Springer, 2008, 1611P.

SICILIANO, B, SCIVIACCO, L, VILLANI, L e ORIOLO, G. **Robotic: Modeling, Planning and Control,** Springer, 2008, 632p.

SPONG E VIDYASAGAR, M. e VIDYASAGAR, M. **Robot Dynamics and Control,** John Wiley & Sons, 1989, 336p.

STOLINE, JEAN-JACQUES E. The Robust Control of Robot Manipulator.**The International Journal of Robotic Research 4.2, 1985, p 49-64.**

TEIXEIRA, M. C. M, ASSUNÇÃO, E. e AGUIRRE, L. A. **Extensões para Sistemas Não-Lineares**. Enciclopédia de Automática: Controle e Automação 1. Primeira edição. 2007, p. 218-243.

TEIXEIRA, M. C. M, LORDELO, A. D. S. e ASSUNÇÃO, E. On LMI based design of SPR systems and output variable structure controllers **Advances in Variable Structure Systems: Analysis, Integration and Applications. p. 199-208, 2000.**

TSENG, C. S. E CHEN, B. S. Multiobjective PID Control Design in Uncertain Robotic System Using Neural Network Elimination Scheme. **IEEE Transiction on Systems, Man, and Cybernetics. V 31, n6 (2001), p. 632-644**

YOUNG, K. K. D, Variable Structure Control for Robotic and Aerospace Applications: Studies in Automation and Control. **Amsterdam Elsevier Science Publisher B. V, 1993**

ZINOBER, A. S. I. Determinist Control of Uncertain Systems. **London IEEE Control Engineering Series, 1990.**

Apêndice A. Modelagem Dinâmica do Manipulador Denso VP6242

O código a seguir foi utilizado no toolbox desenvolvido por Peter Corke (Corke 1996, Corke 2011) para extrair a modelagem dinâmica. No código abaxo o coeficiente de atrito viscoso tem valor Bi, coeficiente de atrito de Colomb tem valor TCi+ e TCi-, os atuadores tem momento de inécia Ji, os elos tem massa Mi centro de gravidade PCix, PCiy e PCiz e possuem momento de inércia dado por Ixxi, Iyyi e Izzi, onde i varia de 1 a 6 e todos os coeficientes precisam ser determinados numericamente (e não simbolicamente).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                Desenvolvido por Mairon Marques                        %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Dynamic Link Viscous Frictions (N.m.s/rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
B_1 = B1;
B_2 = B2;
B_3 = B3;
B_4 = B4;
B_5 = B5;
B_6 = B6;

%% Coulomb Link Frictions (N.m.s/rad) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Tc_1= [TC1+,TC1-];
Tc_2= [TC2+,TC2-];
Tc_3= [TC3+,TC3-];
Tc_4= [TC4+,TC4-];
Tc_5= [TC5+,TC5-];
Tc_6= [TC6+,TC6-];

%% Actuator Gear Ratio %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
G_1 = 120;
G_2 = 160;
G_3 = 120;
G_4 = 100;
G_5 = 100;
G_6 = 100;

%% Actuator Motor Inertia (Kg.m^2)%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J_1 = J1;
J_2 = J2;
J_3 = J3;
J_4 = J4;
J_5 = J5;
J_6 = J6;

%% Link Mass (Kg) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%
m_1 = M1;
m_2 = M2;
m_3 = M3;
```

```
m_4 = M4;
m_5 = M5;
m_6 = M6;

%% Link Center Of Gravity (m)%%
%%%%%%%%%%%%%%%%%%%
Pc_1 = [PC1x PC1y PC1z];
Pc_2 = [PC2x PC2y PC2z];
Pc_3 = [PC3x PC3y PC3z];
Pc_4 = [PC4x PC4y PC4z];
Pc_5 = [PC5x PC5y PC5z];
Pc_6 = [PC6x PC6y PC6z];

%% Inertia Matrix of Link about link COF (Kg.m^2) %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
I_1 = [Ixx1 Iyy1 Izz1];
I_2 = [Ixx2 Iyy2 Izz2];
I_3 = [Ixx3 Iyy3 Izz3];
I_4 = [Ixx4 Iyy4 Izz4];
I_5 = [Ixx5 Iyy5 Izz5];
I_6 = [Ixx6 Iyy6 Izz6];

%% Joint Variable Limits [min max] (rad) %%
%%%%%%%%%%%%%%%%%%%%%%%
J_1_lim = [-160 160]*pi/180;
J_2_lim = [-65 120]*pi/180;
J_3_lim = [-160 -38]*pi/180;
J_4_lim = [-160 160]*pi/180;
J_5_lim = [-100 100]*pi/180;
J_6_lim = [-160 160]*pi/180;

%% Joint Variable offset (rad) %%
%%%%%%%%%%%%%%%%%%
J_1_offset = 0;
J_2_offset = pi/2;
J_3_offset = -pi/2;
J_4_offset = 0;
J_5_offset = 0;
J_6_offset = 0;

%% Link Offset (m) %%
%%%%%%%%%%%%%
d_1 = 0.125;
d_2 = 0;
d_3 = 0;
d_4 = 0.197;
d_5 = 0;
d_6 = 0.07;

%% Link Lenght (m) %%
%%%%%%%%%%%%%
a_1 = 0;
a_2 = 0.210;
a_3 = -0.088;
a_4 = 0;
a_5 = 0;
a_6 = 0;
%% Link twist Angle (rad) %%
%%%%%%%%%%%%%%%
```

```
alpha_1 = pi/2;
alpha_2 = 0;
alpha_3 = -pi/2;
alpha_4 = pi/2;
alpha_5 = -pi/2;
alpha_6 = 0;

%% Definition of Links %%
%%%%%%%%%%%%%%%
L(1) = Link('d', d_1, 'a', a_1, 'alpha', alpha_1, 'standard', 'offset', J_1_offset, 'qlim', J_1_lim, 'I', I_1, 'r', Pc_1, 'm',
m_1, 'G', G_1, 'Tc', Tc_1, 'B', B_1, 'Jm', J_1);

L(2) = Link('d', d_2, 'a', a_2, 'alpha', alpha_2, 'standard', 'offset', J_2_offset, 'qlim', J_2_lim, 'I', I_2, 'r', Pc_2, 'm',
m_2, 'G', G_2, 'Tc',
Tc_2, 'B', B_2, 'Jm', J_2);

L(3) = Link('d', d_3, 'a', a_3, 'alpha', alpha_3, 'standard', 'offset', J_3_offset, 'qlim', J_3_lim, 'I', I_3, 'r', Pc_3, 'm',
m_3, 'G', G_3, 'Tc', Tc_3, 'B', B_3, 'Jm', J_3);

L(4) = Link('d', d_4, 'a', a_4, 'alpha', alpha_4, 'standard', 'offset', J_4_offset, 'qlim', J_4_lim, 'I', I_4, 'r', Pc_4, 'm',
m_4, 'G', G_4, 'Tc', Tc_4, 'B', B_4, 'Jm', J_4);

L(5) = Link('d', d_5, 'a', a_5, 'alpha', alpha_5, 'standard', 'offset', J_5_offset, 'qlim', J_5_lim, 'I', I_5, 'r', Pc_5, 'm',
m_5, 'G', G_5, 'Tc', Tc_5, 'B', B_5, 'Jm', J_5);

L(6) = Link('d', d_6, 'a', a_6, 'alpha', alpha_6, 'standard', 'offset', J_6_offset, 'qlim', J_6_lim, 'I', I_6, 'r', Pc_6, 'm',
m_6, 'G', G_6, 'Tc', Tc_6, 'B', B_6, 'Jm', J_6);

%% Definition of Robot %%
%%%%%%%%%%%%%%%
Denso = SerialLink(L, 'name', 'Denso VP6242', 'manufacturer', 'Denso', 'comment', '6 DOF manipulator');
```

Apêndice B. Algoritmo para Geração de Trajetória

```
function trajetoria(block)
setup(block);
end
function setup(block)

%% DEFINIÇÃO DO NÚMERO DE ENTRADAS E SAÍDAS DO BLOCO

block.NumInputPorts = 6;
block.NumOutputPorts = 1;

block.SetPreCompInpPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

block.InputPort(1).Dimensions=1;
block.InputPort(2).Dimensions=6;
block.InputPort(3).Dimensions=6;
block.InputPort(4).Dimensions=1;
block.InputPort(5).Dimensions=6;
block.InputPort(6).Dimensions=6;


block.OutputPort(1).Dimensions=19;

%% DEFINIÇÃO DO SAMPLE TIME DO BLOCO - 0.002 SEGUNDOS

block.SampleTimes = [0.002 0];
block.SetAccelRunOnTLC(true);

 block.RegBlockMethod('Outputs', @Output);

end

%% BLOCO ONDE A FUNÇÃO É DECLARADA


function Output(block)

Tempo=block.InputPort(1).Data;%% FAZ LEITURA DA ENTRADA 1 - Tempo
Qf=block.InputPort(2).Data;%% FAZ LEITURA DA ENTRADA 2 - Posição Desejada Após Tempo
flagQf=block.InputPort(3).Data;%% FAZ LEITURA DA ENTRADA 3 - Posição Desejada no Instante Anterior
flag=block.InputPort(4).Data;%% FAZ LEITURA DA ENTRADA 4 - Flag que simboliza a necessidade de
incremento em posição ou não (flag<1 significa que necessita incrementar)
DeltaQ1=block.InputPort(5).Data;%% FAZ LEITURA DA ENTRADA 5 - Varição de Posição Desejada
Q0=block.InputPort(6).Data;%% FAZ LEITURA DA ENTRADA 6  - Posiçao inicial (antes da variação de
trajetória)
Sample_time=0.002;
Q=Qf;

if Qf ~= flagQf        %Loop que detecta uma variação na posição desejada
   DeltaQ1=Qf-flagQf;      %Variação de posiçâo a ser executada
   flag=0;              %Ativação de Flag que simboliza que o proximo ponto da trajetória tem que ser um
incremento da posição atual
   Q0=flagQf;            %Salva a posição Inicial antes de iniciar a trajetória
end

if Qf==flagQ    %Loop quando não houve variação de posição desejada
   if flag<1       %Situação quando a Flag indica que necessita gerar o próximo ponto da trajetória
      Q(1)=6*DeltaQ1(1)*flag.^5-15*DeltaQ1(1)*flag.^4+10*DeltaQ1(1)*flag.^3; %%%%%%%%%%%%%%
      Q(2)=6*DeltaQ1(2)*flag.^5-15*DeltaQ1(2)*flag.^4+10*DeltaQ1(2)*flag.^3; %Q significa o Incremento
```

```
    Q(3)=6*DeltaQ1(3)*flag.^5-15*DeltaQ1(3)*flag.^4+10*DeltaQ1(3)*flag.^3; %que deve ser feito na
    Q(4)=6*DeltaQ1(4)*flag.^5-15*DeltaQ1(4)*flag.^4+10*DeltaQ1(4)*flag.^3; %trajetoria de acordo com o
    Q(5)=6*DeltaQ1(5)*flag.^5-15*DeltaQ1(5)*flag.^4+10*DeltaQ1(5)*flag.^3; % interpolador de 5° ordem
    Q(6)=6*DeltaQ1(6)*flag.^5-15*DeltaQ1(6)*flag.^4+10*DeltaQ1(6)*flag.^3; %%%%%%%%%%%%%%
    Q=[Q0(1)+Q(1),Q0(2)+Q(2),Q0(3)+Q(3),Q0(4)+Q(4),Q0(5)+Q(5),Q0(6)+Q(6)]' %O incremento é feito na
posição inicial

  end

  if flag >= 1    %Situação quando a flag indica que não necessita incremeto da posição atual
     Q=Qf;          %Manter a posição
  end

flag=flag+Sample_time/Tempo; %Incremento do Flag. Após uma quantia de "Tempo" segundos a flag>1 e
portanto apenas mantem a posição
end


block.OutputPort(1).Data = [flag;Q;DeltaQ1;Q0]; %Saída do Código

end
```

Apêndice C. Dedução 1

Esse apêndice apresenta a dedução do conjunto de LMI's (4.9). Considere o sistema abaixo:

$$\begin{matrix} \dot{x} = Ax + B[u + w(t,x)] \\ y_0 = Cx \end{matrix}, \tag{C1}$$

mediante a lei de controle

$$u(t) = -Ky_0 - \beta sign(y). \tag{C2}$$

Considere que existam constantes reais a e b, tal que satisfaçam: $\|w(t,x)\| \leq a\|x\| + b$. Portanto, o sistema em malha fechada é presentado por: $\dot{x} = Ax - BKCx - B\beta sign(y) + Bw$. A função de Lyapunov $V = x'Px$ é positiva definida desde que:

$$P > 0. \tag{C3}$$

A derivada temporal da função de Lyapunov, $\dot{V}$, é então dado por:

$$\dot{V} = \dot{x}'Px + x'P\dot{x}. \tag{C4}$$

Substituindo $\dot{x} = Ax + B[-Ky_0 - \beta sign(y) + w(t,x)]$ em (C4), obtemos:

$$\begin{aligned} \dot{V} = {} & [Ax - BKCX - B\beta sign(y) + Bw]'Px \\ & + x'P[Ax - BKCX - B\beta sign(y) + Bw]. \end{aligned} \tag{C5}$$

Reajeitando (C5) chegamos em (C6):

$$\begin{aligned} \dot{V} &= x'A'Px - x'C'K'B'Px + x'PAx - x'PBKCx - [B\beta sign(y) - Bw]'Px \\ &\quad - x'P[B\beta sign(y) - Bw]; \\ \dot{V} &= x'[A'P + PA - C'K'B'P - PBKC]x - B'[\beta sign(y) - w]Px \\ &\quad - x'PB[\beta sign(y) - w]; \\ \dot{V} &= x'[A'P + PA - C'K'B'P - PBKC]x - 2x'PB[\beta sign(y) - w]. \end{aligned} \tag{C6}$$

De (C6) e de acordo com Covacic (2006) podemos extrair que $\dot{V}$ é negativo definido

se:

$$\begin{matrix} A'P + PA - C'K'B'P - PBKC < 0 \\ 2x'PB[\beta sign(y) - w] > 0 \end{matrix} \tag{C7}$$

Embora o conjunto de LMI's (C7)garanta a estabilidade assintótica do sistema (C1) mediante a lei de controle $(C2)$, (C7) não garante que o sistema seja ERP. A garantia de que o sistema seja ERP é encontrada em Anderson (1968). Para um sistema em que $D = 0$, uma condição suficiente para garantir que o sistema seja ERP é garantir que $PB = C$. Uma adaptação para que o sistema com entradas $\widetilde{U}(S)$ e saída $Y(S)$ representado na Figura (4.1) seja ERP foi realizada em Teixeira, Lordelo e Assunção (2000) e garante que o sistema representado na Figura (4.1) é ERP se :

$$B'P = FC \tag{C8}$$

Considerando que $B'P = FC$ e a dedução (5.9) de Covacic (2006), a condição $2x'PB[\beta sign(y) - w] > 0$ é satisfeita se (C.9) for satisfeita.

$$\beta > a\|x\| + b \tag{C9}$$

Assim, se $\beta > a\|x\| + b$, o conjunto de LMI's (C10) garante que o sistema seja ERP e que $V(x)$ é definido positivo e $\dot{V}(x)$ é definida negativa, e, portanto o ponto de equilíbrio $x = 0$ do sistema controlado é globalmente assintoticamente estável.

$$\begin{gathered} A'P + PA - C'K'B'P - PBKC < 0 \\ B'P = FC \\ P > 0 \end{gathered} \tag{C10}$$

Apêndice D. Dedução 2

Esse apêndice apresenta a dedução do conjunto de LMI's (4.10). Considere o sistema abaixo:

$$\begin{matrix} \dot{x} = Ax + B[u + w(t,x)] \\ y_0 = Cx \end{matrix}, \tag{D1}$$

mediante a lei de controle

$$u(t) = -Ky_0 - \beta\alpha sign(y) - \alpha y. \tag{D2}$$

Considere que existam constantes reais a e b, tal que satisfaçam: $\|w(t,x)\| \leq a\|x\| + b$. Portanto, o sistema em malha fechada é presentado por: $\dot{x} = Ax - BKCx - B\beta sign(y) - B\alpha FCx + Bw$. A função de Lyapunov $V = x'Px$ é positiva definida desde que:

$$P > 0. \tag{D3}$$

A derivada temporal da função de Lyapunov, $\dot{V}$, é então dado por:

$$\dot{V} = \dot{x}'Px + x'P\dot{x} \tag{D4}$$

Substituindo $\dot{x} = Ax - BKCx - B\beta sign(y) - B\alpha FCx + Bw$ em (D4) obtemos:

$$\begin{aligned} \dot{V} = {} & [Ax - BKCX - B\beta sign(y) - B\alpha FCx + Bw]'Px \\ & + x'P[Ax - BKCX - B\beta sign(y) - B\alpha FCx + Bw] \end{aligned} \tag{D5}$$

Reajeitando (D5) chegamos em (D6)

$$\begin{aligned} \dot{V} = {} & x'A'Px - x'C'K'B'Px - x'C'F'\alpha'B'Px + x'PAx - x'PBKCx - x'PB\alpha FCx \\ & - [B\beta sign(y) - Bw]'Px - x'P[B\beta sign(y) - Bw] \\ \dot{V} = {} & x'[A'P + PA - C'K'B'P - PBKC - C'F'\alpha'B'P - PB\alpha FC]x \\ & - B'[\beta sign(y) - w]Px - x'PB[\beta sign(y) - w] \\ \dot{V} = {} & x'[A'P + PA - C'K'B'P - PBKC - C'F'\alpha'B'P - PB\alpha FC]x \\ & - 2x'PB[\beta sign(y) - w] \end{aligned} \tag{D6}$$

De (C6) podemos extrair que $\dot{V}$ é negativo definido se:

$$\begin{matrix} A'P + PA - C'K'B'P - PBKC - C'F'\alpha'B'P - PB\alpha FC < 0 \\ 2x'PB[\beta sign(y) - w] > 0 \end{matrix} \tag{D7}$$

De maneira análoga a dedução no apêndice C, embora o conjunto de LMI's (D7) garanta a estabilidade assintótica do sistema (D1) mediante a lei de controle (D2), (D7) não garante que o sistema seja ERP. A garantia de que o sistema seja ERP é encontrada em Anderson (1968). Para um sistema em que $D = 0$, uma condição suficiente para garantir que o sistema seja ERP é garantir que $PB = C$. Uma adaptação para que o sistema com entradas $\widetilde{U}(S)$ e saída $Y(S)$ representado na Figura (4.1) seja ERP foi realizada em Teixeira, Lordelo e Assunção (2000) e garante que o sistema representado na Figura (4.1) é ERP se:

$$B'P = FC. \tag{D8}$$

Considerando que $B'P = FC$, a dedução (5.9) de Covacic (2006), e de maneira análoga a dedução do Apendice C, a condição $2x'PB[\beta sign(y) - w] > 0$ é satisfeita se (D.9) for satisfeita.

$$\beta > a\|x\| + b \tag{D9}$$

Assim,se $\beta > a\|x\| + b$, o conjunto de LMIs (D10) garante que garantem que o sistema seja ERP e que $V(x)$ é definido positivo e $\dot{V}(x)$ é definida negativa, e, portanto o ponto de equilíbrio $x = 0$ do sistema controlado é globalmente assintoticamente estável.

$$\begin{gathered} A'P + PA - C'K'B'P - PBKC - C'F'\alpha'B'P - PB\alpha FC < 0 \\ B'P = FC \\ P > 0 \end{gathered} \tag{D10}$$

Apêndice E. Bloco Level 2 MATLAB S Function – Figura (7.4)

```
function sign_mairon(block)
setup(block);
end
function setup(block)

%% DEFINIÇÃO DO NÚMERO DE ENTRADAS E SAÍDAS DO BLOCO

block.NumInputPorts = 1;
block.NumOutputPorts = 1;

block.SetPreCompInpPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

block.InputPort(1).Dimensions=6;
block.OutputPort(1).Dimensions=6;
%% DEFINIÇÃO DO SAMPLE TIME DO BLOCO - 0.001 SEGUNDOS
block.SampleTimes = [0.001 0];
block.SetAccelRunOnTLC(true);
block.RegBlockMethod('Outputs', @Output);
end

%% BLOCO ONDE A FUNÇÃO É DECLARADA

function Output(block)

Fy=block.InputPort(1).Data;%% FAZ LEITURA DA ENTRADA 1
epsilon=0.0000000000001;
BETTA=[2;20;20;2;20/3;4];

%% APROXIMAÇÃO DA FUNÇÃO SINAL
Signal_Fy=Fy./(abs(Fy)+epsilon);
%%

Saida_EV=BETTA.*Signal_Fy;

block.OutputPort(1).Data = Saida_EV;
end
```

Apêndice F. Trabalhos Publicados

Capítulo de livro:

- **MARQUES, M. F.**; Gaino R. ; Covacic M. R. ; Ana Djuric . Inovação e Tecnologia - II Edição. II. ed. Londrina: Faculdade de Tecnologia SENAI Londrina, 2016. 365p .

Trabalhos Apresentados em congressos

- Magan, M. V. ; **MARQUES, M. F.** ; Covacic M. R. ; GENTILIN, F. A. ; Gaino R. . MIMO Control by Decoupling Theory using Robust PID Controllers applied in Level and Temperature Model. In: ICCMA - International Conference on Control, Mechatronics and Automation, 2017, Edmonton - Canadá. ICCMA, 2017.

- **MARQUES, M. F.**; Magan, M. V. ; Gaino R. ; Covacic M. R. . Control of Uncertain System Represented By Polytope Using Enhaced Lyapunov Function. In: IEEE CHILECON, 2017, Pucón - Chile. IEEE CHILECON, 2017.

- Magan, M. V. ; **MARQUES, M. F.** ; NUNES, W. R. B. M. ; Covacic M. R. ; Gaino R. . SÍNTESE DE CONTROLADOR POR ESTRUTURA VARIÁVEL E MODOS DESLIZANTES PARA O CONTROLE DE POSIÇÃO ANGULAR DO JOELHO DO PACIENTE PARAPLÉGICO. In: CONFERÊNCIA BRASILEIRA DE DINÂMICA, CONTROLE E APLICAÇÕES, 2017, São José do Rio Preto. DINCOM, 2017

Trabalhos em Revisão

- **MARQUES, M. F**; Magan, M. V. ; Gaino R. ; Covacic M. R., VARIABLE STRUCTURED CONTROL APPLIED IN ROBOTIC ARM WITH UNCERTAIN DYNAMIC MODEL, Journal of Intelligent & Robotic Systems, Submetido em Maio de 2018.

Apêndice G. Pré Projeto

Modelagem do Manipulador Robótico Denso

Mairon Figueiredo Marques

RESUMO

Visando a reprodução de movimentos em robôs industriais cada vez mais perfeitos e eficientes, as indústrias robóticas mundiais veem gradativamente desenvolvendo novos atuadores e sensores, aprimorando softwares e otimizando modelos robóticos já propostos. Objetiva-se com esse projeto o desenvolvimento da modelagem de um manipulador robótico e a possível implementação em um dispositivo robótico de caráter industrial.

Para tal missão será levanto os parâmetros de Denavit-Hartenberg (D-H) para o dispositivo robótico da marca DENSO com 6 graus de liberdade que o laboratório do departamento de engenharia elétrica da Universidade Estadual de Londrina possui. Posteriormente será derivado as matrizes rotacionais, equações da cinemática direta e então um estudo das equações de cinemática inversa será apresentada. Uma análise de singularidade será feita e caso seja necessário, a implementação de limites físicos para o manipulador será efetuado a fim de proteger o manipulador de situações singulares. Uma vez que toda a parte cinemática estiver desenvolvida, será desenvolvido o modelo dinâmico dos atuadores do dispositivo, e então o modelo dinâmico dos links serão efetuados utilizando o método recursivo de Newton-Euler. Um sistema de realimentação será projetado, e um controlador será desenvolvido para cada uma das juntas. Por fim será feito a simulação completa do sistema utilizando a plataforma Simulink e decorrente de resultados positivos o modelo será implementado no manipulador robótico DENSO.

Introdução

Sendo uma das maiores fabricantes de dispositivos robóticos no mundo, a DENSO é conhecida por produzir dispositivos robóticos com 4, 5 e até 6 graus de liberdade que são rápidos, confiáveis e duradores [5]. Objetivando a reprodução de movimentos cada vez mais perfeitos e eficientes, as indústrias robóticas mundiais veem gradativamente desenvolvendo novos atuadores e sensores, aprimorando softwares e otimizando modelos robóticos já propostos.

Visando o controle de qualquer dispositivo robótico, é necessário que primeiro se tenha um modelo confiável do sistema que permita que seja implementado ajustes de controle. Com isso em mente, a modelagem de sistemas robóticos toma proporções importantes para o bom funcionamento de todo e qualquer dispositivo robótico.

Deseja-se nesse projeto fazer a modelagem cinemática e dinâmica de um manipulador robótico com 6 graus de liberdade. O modelo será feito para o manipulador robótico da marca DENSO que o departamento de engenharia elétrica da Universidade Estadual de Londrina possui.

Fundamentação Teórica

O modelo cinemático de um manipulador robótico são as equações que descrevem as relações físicas entre as juntas. Um modelo cinemático completo deve se ter inicio na definição dos parâmetros de Denavit-Hartenberg (parâmetros D-H), em seguida define-se as matrizes de transformação, encontra-se as relações da cinemática direta, posteriormente deve-se derivar as equações de cinemática inversa, podendo ainda fazer uma análise das singularidades do manipulador e uma interpretação para cada caso singular. [1,2,3]

O modelo dinâmico de um manipulador tem por característica descrever as forças e torques existentes e atuantes no dispositivo. Para a definição de um modelo dinâmico é necessário fazer a modelagem dos atuadores robóticos, posteriormente através do método de Newton-Euler ou Lagrange, pode-se definir as equações dinâmicas do manipulador.

O presente projeto propõe o desenvolvimento de uma modelagem cinemática e dinâmica completa de um manipulador robótico. Após modelo definido, um controlador, possivelmente PID, será desenvolvido para cada link do projeto. O modelo será simulado através da plataforma Simulink e decorrente dos resultados, será implementado em

dispositivos LEGO ou no manipulador robótico com 6 graus de liberdade DENSO (adquirido pela Universidade Estadual de Londrina).

Objetivos Gerais e Específicos

Objetiva-se com esse projeto o desenvolvimento da modelagem de um manipulador robótico e a possível implementação em um dispositivo robótico de caráter industrial.

Especificamente, objetiva-se: levantamento dos parâmetros de Denevit-Hartenberg de um manipulador robótico; desenvolvimentos das matrizes de rotação; levantamento das equações de cinemática direta e inversa; derivação das equações dinâmicas; projeção de um controlador PID; simulação do sistema; e possível implementação em dispositivo físico.

Metodologia

Visando a execução do projeto, primeiro será levanto os parâmetros D-H do dispositivo em questão. Posteriormente será desenvolvido as matrizes rotacionais, equações da cinemática direta e então um estudo das equações de cinemática inversa será apresentada. Uma análise de singularidade será feita e caso seja necessário, a implementação de limites físicos para o manipulador será efetuado a fim de proteger o manipulador de situações singulares.

Uma vez que toda a parte cinemática estiver desenvolvida, será desenvolvido o modelo dinâmico dos atuadores do dispositivo, e então o modelo dinâmico dos links serão efetuados utilizando o método recursivo de Newton-Euler. Um sistema de realimentação será projetado, e um controlador será desenvolvido para cada uma das juntas. Por fim será feito a simulação completa do sistema utilizando a plataforma Simulink e decorrente de resultados positivos o modelo será implementado no manipulador robótico DENSO detido pelo laboratório de controle avançado, robótica e biomédica do departamento de engenharia elétrica da Universidade Estadual de Londrina.

Resultados Esperados

Espera-se com essa pesquisa o desenvolvimento da modelagem completa de um manipulador robótico industrial, com simulações e possíveis validações efetuada, podendo ser implementado no robô Denso detido pelo laboratório do departamento de engenharia elétrica da Universidade Estadual de Londrina.

Afinidade do Candidato

Trata-se de um candidato ex-participante do programa Ciência Sem Fronteiras, onde teve a oportunidade de se engajar em estudos de robótica avançado com publicações de caráter internacional na área. Possui experiência em modelagem de dispositivos robóticos industrias e grande potencial de absorver informações provindas de fornecedores internacionais.

O candidato possui afinidade com a área de controle de sistemas após efetuar graduação em engenharia de controle e automação, possui entendimento com o orientador professor Dr. Marcio Covacic, e compartilhamento dos estudos com modelagem de manipuladores com o professor Dr. RuberleiGaino.

Conclusão

Visto a disponibilidade do manipulador robótico DENSO pela Universidade Estadual de Londrina e a ausência de mão de obra qualificada disponível para o desenvolvimento e avanço cientifico do dispositivo, através desse projeto encontra-se uma ótima oportunidade de executar a modelagem e controle do dispositivo. Assim sendo, espera-se acelerar o processo de desenvolvimento cientifico referente à esse dispositivo.

Trata-se de uma modelagem que abrirá uma variedade de futuros projetos relacionados ao dispositivo. Visto que o candidato tem qualificação técnica e experiência para a execução do projeto, espera-se alcançar com sucesso as simulações do modelo cinemático, dinâmico e ações de controle para o manipulador e posteriormente a implementação do projeto no dispositivo.