Luiz Henrique Silva Lelis

Implementação de técnica de controle em um manipulador robótico utilizando o conceito de *Hardware in the Loop*

Belo Horizonte

2019, v1

Luiz Henrique Silva Lelis

Implementação de técnica de controle em um manipulador robótico utilizando o conceito de *Hardware in the Loop*

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Minas Gerais para obtenção do Título de Bacharel em Engenharia de Controle e Automação pela Escola de Engenharia da Universidade Federal de Minas Gerais.

Universidade Federal de Minas Gerais – UFMG

Escola de Engenharia

Orientador: Luciano Antonio Frezzatto Santos

Belo Horizonte

2019, v1

Lista de ilustrações

Sumário

1 Introdução

Um sistema pode ser visto como um processo com sinais de entrada que são transformados ou induzidos a responder de alguma forma resultando em outros sinais de saída (OPPENHEIN; WILLSKY, 1997). O intuito do controlador é manipular o sistema com a finalidade de obter um sinal de saída que siga uma referência pré estabelecida. Para isso, é necessário, primeiramente, obter um modelo que descreva o comportamento do sistema e, em seguida, projetar uma estratégia de controle a partir desse modelo. Dentre as técnicas utilizadas para modelagem cabe destacar a descrição do sistema por meio de equações diferenciais lineares (OGATA, 2010) ou a identificação de subsistemas (AGUIRRE, 2015).

No controle em malha aberta, a saída não exerce nenhuma influência sobre o sinal de controle. Dessa forma, um sinal de controle é aplicado à entrada de uma planta ou de um processo de modo que a variável controlada atinja um valor pré-estabelecido; entretanto, esse valor resultante na saída não é utilizado para modificar a entrada. O problema deste tipo de controle é que o sistema pode mudar seu ponto de operação , por exemplo, ao ocorrerem perturbações. Caso isso ocorra, a saída não terá o valor estabelecido anteriormente.

Ao fechar a malha, o sinal de saída passa a influenciar diretamente a ação de controle. Assim, o sistema passa a contar com uma malha de realimentação e o sinal observado na entrada do controlador é, agora, o erro entre a referência do sistema e a saída mensurada. O controlador, então, tende a minimizar o erro de modo a garantir que a saída do sistema seja igual a referência, como ilustrado na Figura 1. “Controlar a saída de uma planta ou de um processo por realimentação significa aplicar na sua entrada, após conveniente amplificação, o sinal resultante da diferença entre o valor desejado e o valor medido da saída (CASTRUCCI; BITTAR; SALES, 2011, p. 3).”

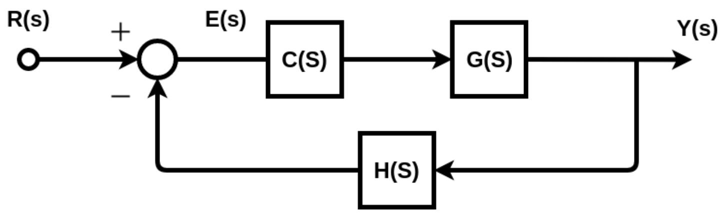
O controle em malha fechada utiliza a informação de como a variável controlada evolui para determinar o sinal de controle aplicado ao processo. Determinado o modelo da planta a ser controlada, o controlador é sintonizado de forma a atender especificações de projeto fornecidas, tais como: sobressinal máximo, tempo de acomodação, tempo de subida e a constante de tempo (no caso de sistemas de primeira ordem). Geralmente, uma vez sintonizado o controlador e validado seu desempenho em malha fechada da planta realimentada, passa-se à implementação prática do controle sem que haja uma prévia validação do comportamento do controlador quando implementado em dispositivo físico.

Uma das formas de se controlar uma planta é através da inclusão de um hardware real na malha de controle. Essa é a idéia básica da simulação *Hardware in the Loop*

(HIL), uma técnica bem estabelecida usada em projeto e avaliação de sistemas de controle (BACIC, 2005).

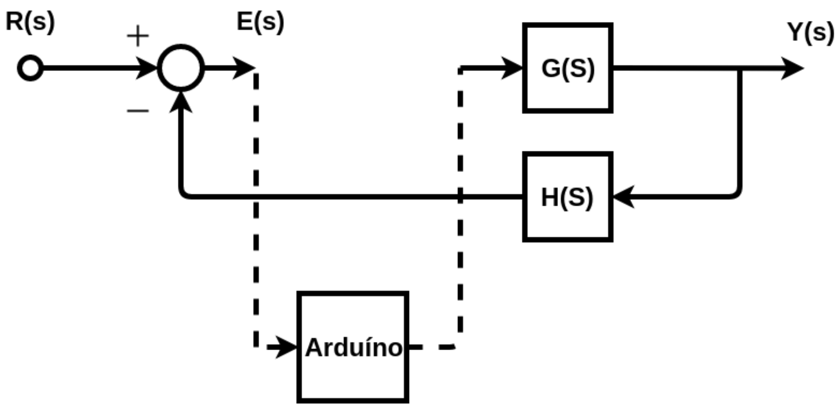
Essa técnica consiste em implementar um controlador previamente projetado e validado em um dispositivo físico e substituir o bloco de controle $C(s)$ (vide Figura 1) por esse dispositivo, como apresentado na Figura 2. Ou seja, o sinal de erro $E(s)$ é transmitido para um dispositivo físico externo, o qual é responsável por determinar a ação de controle a ser aplicada ao modelo da planta em estudo $G(s)$. Em seguida, o sinal de controle calculado é enviado de volta para o ambiente de simulação, no qual a saída $Y(s)$, teórica, é determinada. O intuito é verificar se a estratégia de controle e o desempenho associado permanecem adequados antes de realizar o controle diretamente na planta real.

Figura 1 – Diagrama de blocos do sistema em malha fechada



Fonte: Do autor

Figura 2 – Diagrama de blocos do sistema em malha fechada - *Hardware in the Loop*



Fonte: Do autor

A metodologia para síntese do controlador, validação e implementação na planta real possui três etapas distintas de execução. A etapa um consiste em levantar o modelo da planta que pretende-se controlar, e depois sintonizar um controlador em ambiente de simulação que atenda às condições de projeto previamente especificadas. A segunda etapa propõe substituir o controlador virtual por sua implementação em dispositivo físico e validar a implementação em conjunto com o ambiente de simulação. Por fim, a etapa três consiste na utilização do controlador implementado no dispositivo físico comunicando diretamente com a planta real.

1.1 Motivação e Justificativa

Quando os processos possuem uma dinâmica mais rápida que a execução do algoritmo para a ação de controle, surge um problema computacional complexo. Cumprir a restrição de reduzir o tempo de resposta exige um uso maior de simulações em tempo real (ISERMANN; SCHAFFNIT; SINSEL, 1999). Nos sistemas de tempo real, as tarefas consideradas críticas devem executar dentro um período de tempo estabelecido. A implementação de um *Hardware* em uma malha de controle, conforme o método HIL propõe, possui um ganho em relação às simulações com sistemas operacionais comuns pelo motivo exposto acima.

Atualmente, as simulações HIL são utilizadas mais recorrentemente para o desenvolvimento de novos componentes e atuadores em vários campos diferentes (BOUSCAYROL, 2008). As aplicações ao longo da história têm demonstrado a variabilidade do uso da metologia, aplicações estas como, sistemas eletrônicos de potência (ROTHSTEIN; SIEKMANN; STAUDT, 2017), sistemas de controle de vôo (KARPENKO; SEPEHRI, 2006), engenharia de tráfego (BULLOCK et al., 2004) e robótica móvel (JAIN; KAMALI, 2016).

1.2 Objetivos do Projeto

O objetivo principal do presente projeto é sintonizar e validar estratégias de controle para o sistema dinâmico de um manipulador robótico utilizando a técnica *Hardware in the Loop* (HIL).

1.3 Local de Realização

Departamento de Engenharia Eletrônica da UFMG (DELT/UFMG). O departamento situa-se dentro do campus Pampulha da UFMG na Escola de Engenharia. O campus fica na Av. Presidente Antônio Carlos 6627 - Pampulha, Belo Horizonte - MG, 31270-901.

O departamento foi criado em 1969 e tem contribuído para a formação dos engenheiros eletricistas e engenheiros de controle e automação da UFMG. O DELT é referência no cenário nacional e é, também, o Departamento da UFMG com maior participação no curso de graduação em Engenharia de Controle e Automação.

1.4 Estrutura da Monografia

A monografia está dividida em cinco capítulos. Este capítulo apresentou uma introdução ao projeto e o local onde o trabalho foi realizado. O Capítulo 2 faz uma revisão bibliográfica sobre os principais temas trabalhados no decorrer do projeto, como a modelagem e controle de sistemas dinâmicos lineares, os conceitos da técnica *Hardware in the Loop* (HIL) e os princípios básicos do manipulador robótico. O Capítulo 3 aborda a metodologia de desenvolvimento do trabalho, ele descreve como o projeto foi desenvolvido explicitando o passo a passo das tarefas executadas para que o mesmo fosse concluído. Os resultados experimentais são apresentados no Capítulo 4: primeiramente, o modelo da planta é obtido, depois um controlador é modelado em ambiente de simulação, o controlador é aplicado em um dispositivo físico para controlar a planta simulada, e por fim, a planta simulada é substituída pela planta real. Finalmente, no capítulo 5, tem-se a conclusão da monografia com algumas sugestões para trabalhos futuros e dificuldades encontradas na realização do projeto.

2 Revisão Bibliográfica

Esta monografia pretende validar estratégias de controle em um manipulador robótico por meio da técnica *Hardware in the Loop* (HIL). Por esse motivo, este capítulo aborda uma breve revisão dos principais aspectos que envolvem o tema proposto: manipuladores robóticos, sistemas de controle e HIL. Por clareza, esses tópicos são apresentados em seções distintas.

2.1 Manipuladores Robóticos

Um manipulador robótico pode ser definido como um mecanismo reprogramável e multifuncional que é desenvolvido para mover materiais, peças e ferramentas (MURPHY, 2000). Mecanismo este que é composto por elos e juntas mecânicas. Apesar disso, o manipulador não pode ser visto apenas como uma série de elos (ou *links*) em cascata. Para Spong, Hutchinson e Vidyasagar (2005), o manipulador robótico é composto por um braço mecânico, pela ferramenta no fim do braço (também chamada ferramenta de trabalho), pela fonte de energia externa, pelos sensores externos e internos, pela interface de comunicação com o sistema e pelo controle do microcontrolador.

Na construção do manipulador, os elos são conectados por meio de juntas formando a cadeia cinemática. Segundo Paul (1981), ao incorporar coordenadas em cada elo do manipulador, usando transformação homogênea, é possível descrever a posição relativa e a orientação entre elas. A transformação homogênea trata-se da transformação de coordenadas que descreve a posição e a orientação do eixo da ferramenta de trabalho em relação à base (eixo 0) (SICILIANO et al., 2008).

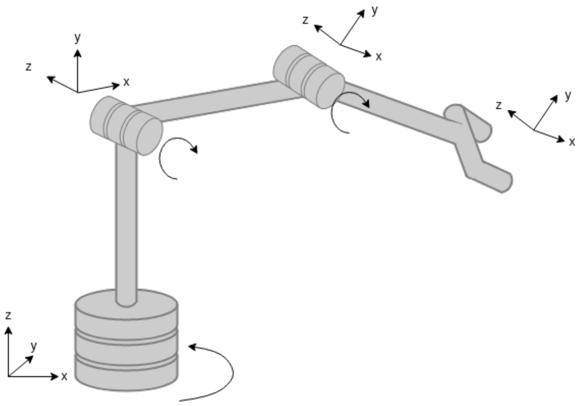
As juntas podem ser tanto de revolução quanto prismáticas (PAUL, 1981). As juntas de revolução são aquelas que permitem um movimento de rotação entre um elo e outro. Por outro lado, as prismáticas são as que possibilitam apenas um movimento linear entre os elos.

A forma geométrica para se classificar os manipuladores é dada pela disposição das juntas na cadeia cinemática. Segundo Spong, Hutchinson e Vidyasagar (2005), a maioria dos manipuladores se enquadra em uma das categorias a seguir (em que R corresponde a uma junta de revolução e P uma junta prismática): articulada (RRR), esférica (RRP), SCARA (RRP), cilíndrica (RPP), ou Cartesiana (PPP).

O grau de liberdade (DOF - *degree-of-freedom*) é um parâmetro fundamental para a configuração espacial do manipulador robótico. É esse parâmetro que define qual a dimensão do espaço de configuração, ou seja, um manipulador possui n graus de liberdade caso sua configuração seja minimamente especificada por n parâmetros (SPONG; HUTCHINSON; VIDYASAGAR, 2005). Para Spong, Hutchinson e Vidyasagar (2005), a maioria dos manipuladores industriais atualmente possuem seis ou menos graus de liberdade.

É importante ressaltar que a planta utilizada neste trabalho é um manipulador com três juntas revolutas. O manipulador real, possui outras duas juntas, uma para o punho e outra para a garra, entretanto estas foram desconsideradas para a modelagem e para o controle. Esse manipulador está representado na Figura 3 e também é conhecido na literatura como manipulador de cotovelos (*elbow manipulator*), articulado, revoluto, antropomórfico (*anthropomorphic manipulator*) ou manipulador RRR.

Figura 3 – Planta utilizada - manipulador de três juntas revolutas



Fonte: Do autor

2.2 Sistemas de Controle

Segundo Phillips e Nagle (2007), um controlador é necessário em uma planta para processar um sinal de erro de forma a atender certas especificações pré-definidas. Esse sinal de erro é dado pela diferença entre a resposta do sistema, determinada por um sensor, e a trajetória desejada. Dentre as especificações mais comuns em controle de sistemas dinâmicos lineares estão: rejeição do distúrbio, erro em estado estacionário e a resposta transiente (OGATA, 2010).

As variedades de controle se dão conforme os tipos de sinais existentes. Sinais analógicos são aqueles que apresentam valor em qualquer instante de tempo, os sinais discretos apresentam valores em instantes múltiplos do tempo de amostragem, e por fim, os sinais digitais são aqueles amostrados no tempo tendo sua amplitude representada por um número limitado de *bits*, ou seja, a amplitude sofre o efeito da quantização (CASTRUCCI; BITTAR; SALES, 2011). Geralmente, no projeto de controladores digitais, primeiro é realizado o projeto do controlador analógico, o qual é convertido, então, em digital para execução computacional. Por outro lado, existem também metodologias para a síntese direta de controladores digitais, as quais podem fornecer melhores resultados quando comparadas ao método indireto. Todavia, essas últimas são menos utilizadas.

Nos dias atuais, o controlador mais utilizado na indústria é o controlador PID. Segundo Ogata (2010), mais da metade dos controladores industriais empregam o controle PID ou variantes. O seu sucesso está ligado diretamente a sua concepção robusta e sua aplicabilidade geral a maioria dos sistemas. O seu nome advém das componentes que definem a lei de controle, dada na Equação (2.1), a qual é composta pelas ações proporcional, integradora e derivativa (CASTRUCCI; BITTAR; SALES, 2011):

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right) \tag{2.1}$$

com:

- *u(t)*: sinal de saída do controlador, ou variável manipulada;
- *e(t)*: sinal de entrada do controlador, ou erro entre a resposta do sistema e o sinal de referência;
- K_p, T_i, T_d: parâmetros de ajustes do PID.

No escopo desta monografia, o controle PID será utilizado em conjunto com a técnica HIL apresentada na próxima subseção. Apesar disso, outras técnicas de controle podem ser igualmente empregadas seguindo a metodologia apresentada na monografia.

2.3 Técnica *Hardware in the Loop (HIL)*

A idéia básica da técnica HIL é a inclusão de uma parte do *hardware* real no loop de simulação durante o desenvolvimento do sistema (BACIC, 2005). Isto é, a técnica consiste em inserir um dispositivo físico na malha de controle de uma simulação. Nessa técnica, uma parte do sistema é integrada a uma outra parte que está sendo simulada em tempo real (ABOURIDA; DUFOUR; BELANGER, 2005).

Os primeiros usos da técnica HIL estão relacionados com as simulações de vôo (ISERMANN; SCHAFFNIT; SINSEL, 1999). Utilizando essa técnica, a NASA realizou simulações de alta fidelidade para o desenvolvimento de tecnologias de aeronaves altamente manobráveis (EVANS; SCHILLING, 1984). Outras aplicações dessa técnica vieram posteriormente com os testes dinâmicos de componentes de veículos, como, por exemplo, suspensão e corpo do carro (ISERMANN; SCHAFFNIT; SINSEL, 1999).

Para Abourida, Dufour e Belanger (2005), a técnica *Hardware-in-the-Loop* (HIL) é fundamental para simulações em tempo real; não para simular o sistema completo em tempo real, mas sim para conectar uma parte do sistema a um modelo digital em tempo real. Além disso, essa técnica de simulação tem como desafio o alcance da precisão de um modelo aceitável com um tempo de simulação digital viável (ABOURIDA; DUFOUR; BELANGER, 2005). Isso porque alguns sistemas (aqueles altamente não-lineares) precisam de uma frequência de amostragem muito alta para alcançarem uma precisão aceitável.

A técnica HIL será utilizada nesta monografia através do uso de um computador de placa única de tamanho reduzido (*Raspberry*). A *Raspberry* representa o *Hardware* do método, e ela será inserida na malha de controle após a etapa de modelagem e controle no ambiente simulado.

3 Metodologia

Neste capítulo, serão apresentadas as técnicas utilizadas para solução do problema. O projeto foi realizado em um ano e sua concepção foi dividida em três frentes, nomeadamente: desenvolvimento da solução para o modelo da planta com controlador simulado, controle da planta simulada com o controlador implementado em dispositivo físico (microcontrolador) e a implementação final em dispositivo real. As etapas pormenorizadas do projeto seguem abaixo:

1. Desenvolvimento da solução para o modelo da planta com controlador simulado
    a) Definição e modelagem da planta de estudo.
    b) Construção do ambiente de simulação.
    c) Validação do modelo.
    d) Implementação e validação da estratégia de controle.
2. Controle da planta simulada com o controlador em dispositivo físico
    a) Discretização da planta de estudo seguido da obtenção da sua equação de diferenças.
    b) Discretização do controlador seguido da obtenção da sua equação de diferenças.
    c) Configuração do ambiente de testes com a planta rodando no computador, e o controlador rodando no microcontrolador (comunicação $UART$).
3. Implementação em dispositivo real
    a) Configuração do ambiente com a planta real e o microcontrolador (comunicação $UART$).
    b) Implementação da estratégia de controle no microcontrolador.
    c) Testes de validação da estratégia de controle na planta física..

3.1 Desenvolvimento da solução para o modelo da planta

A modelagem dos manipuladores robóticos visa descrever como os elos e juntas estão configurados fisicamente para tornar possível a configuração de sua orientação e sua posição deles (PAUL, 1981). Com isso, ao passar uma trajetória de referência que a ferramenta de trabalho deve descrever, as demais juntas se reajustam de modo a garantir que a ferramenta sempre esteja corretamente posicionada. A estratégia de controle deve, então, estar adequadamente sintonizada para garantir o seguimento acurado da trajetória.

No presente trabalho são realizadas duas modelagens para o manipulador: a modelagem cinemática e a modelagem dinâmica. A modelagem cinemática visa descrever a amplitude de movimento das juntas robóticas, ao passo que a modelagem dinâmica busca considerar as forças e torques que produzem o movimento, descrevendo, explicitamente, a relação entre força e movimento (SPONG; HUTCHINSON; VIDYASAGAR, 2005).

A principal ferramenta utilizada para se obter o modelo cinemático direto de um manipulador robótico é a convenção de *Denavit-Hartenberg* (DH) (PAUL, 1981) . A modelagem dinâmica, por sua vez, pode ser realizada por meio das equações de Euler-Lagrange, que correspondem a um método baseado na energia do sistema (LYNCH; PARK, 2017).

3.1.1 Convenção de *Denavit-Hartenberg*

Considera-se que o manipulador a ser modelado é de cadeia aberta, ou seja, um manipulador que o número de graus de liberdade seja igual ao número de articulações ativas. Além disso, ele deve ser constituído por $n + 1$ elos conectados por n juntas, onde o elo 0 é convencionalmente fixado ao solo. Assim sendo, segundo Siciliano et al. (2008), a equação cinemática direta para o manipulador pode ser calculada a partir de:

$$T_n^0 = A_1^0(q_1)A_2^1(q_2)\cdots A_n^{n-1}(q_n) \tag{3.1}$$

A relação (3.1) se refere à transformação de coordenadas descrevendo a posição e a orientação do eixo n em relação à base (eixo 0) (SICILIANO et al., 2008). Segundo Spong, Hutchinson e Vidyasagar (2005), na convenção DH, cada transformação homogênea A_i representada em (3.1) equivale ao produto de quatro transformações básicas, apresentado a seguir:

$$\begin{aligned} A_i &= Rot_{z,\theta_i} Trans_{z,\theta_i} Trans_{x,\alpha_i} Rot_{x,\alpha_i} \\ &= \begin{bmatrix} \cos(\theta_i) & -\mathrm{sen}(\theta_i) & 0 & 0 \\ \mathrm{sen}(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &\times \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_i) & -\mathrm{sen}(\alpha_i) & 0 \\ 0 & \mathrm{sen}(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

$$= \begin{bmatrix} \cos(\theta_i) & -\text{sen}(\theta_i)\cos(\alpha_i) & \text{sen}(\theta_i)\text{sen}(\alpha_i) & a_i\cos(\theta_i) \\ \text{sen}(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\text{sen}(\alpha_i) & a_i\text{sen}(\theta_i) \\ 0 & \text{sen}(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

A matriz final encontrada em (3.2) é chamada de Matriz de Trasformação Homogênea. Os quatro parâmetros da relação (3.2) representam: tamanho do elo (a_i), deslocamento do elo (d_i), giro do elo (α_i), ângulo da junta (θ_i). De acordo com Paul (1981), os parâmetros são obtidos através do procedimento a seguir:

- Rotacionar x_i em torno do eixo z_i um ângulo θ_i;
- Transladar ao longo do eixo z_i uma distância d_i;
- Transladar ao longo de x_{i+1} uma distância a_i;
- Rotacionar z_i em torno de x_{i+1} o ângulo de torção α_i

Assim sendo, a modelagem cinemática é obtida através da multiplicação de diversas matrizes de transformação homogêneas individuais, conforme (3.1). Com isso, para um manipulador com três graus de liberdade, a transformação de coordenadas do elemento terminal em relação a base é dada por $T_3^0 = A_1 A_2 A_3$.

3.1.1.1 Cinemática Diferencial e o Jacobiano

A cinemática diferencial é responsável por fornecer a relação entre as velocidades das juntas e a correspondente velocidade final linear e angular da ferramenta de trabalho (SICILIANO et al., 2008). Essa relação é descrita por uma matriz denominada jacobiana geométrica, que depende da configuração do manipulador. O Jacobiano Analítico, por outro lado, é expresso por meio da diferenciação da função cinemática direta com relação às variáveis das juntas. A obtenção da matriz Jacobiana é fundamental para determinar as equações de movimento do manipulador robótico.

Segundo Spong, Hutchinson e Vidyasagar (2005), considerando um manipulador com n graus de liberdade, a equação da cinemática direta pode ser escrita na forma:

$$T_n^0(q) = \begin{bmatrix} R_n^0(q) & o_n^0(q) \\ 0 & 1 \end{bmatrix} \tag{3.3}$$

A equação (3.3) é a mesma que (3.1), onde $q = [q_1, ..., q_n]^T$ é o vetor das variáveis das juntas, $R_n^0(q)$ a matriz de rotação, $o_n^0(q)$ o vetor de translação, 0 a perspectiva e 1 o fator de escala. As relações da velocidade linear v_n^0 e a velocidade angular ω_n^0 em função das velocidades das juntas é linear (SICILIANO et al., 2008) e pode ser expressa por:

$$v_n^0 = J_v \dot{q} \tag{3.4}$$

$$\omega_n^0 = J_\omega \dot{q} \tag{3.5}$$

onde J_v e J_ω são matrizes $3 \times n$. É possível ainda, reescrever (3.4) e (3.5) da seguinte forma:

$$\xi = J\dot{q} \tag{3.6}$$

$$\xi = \begin{bmatrix} v_n^0 \\ \omega_n^0 \end{bmatrix} \quad \text{e} \quad J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix} \tag{3.7}$$

O vetor ξ também pode ser chamado de velocidade do corpo (SPONG; HUTCHINSON; VIDYASAGAR, 2005) e é importante notar que ele não é a derivada de uma variável de posição. A matriz J é chamado de **Jacobiano** e trata-se de uma matriz $6 \times n$.

Combinando as partes angular e linear, segundo Siciliano et al. (2008), a metade de cima da matriz do Jacobiano é dada por:

$$J_{v_i} = \begin{cases} z_{i-1} \times (o_n - o_{i-1}) & \text{para a } i\text{-ésima junta revoluta} \\ z_{i-1} & \text{para a } i\text{-ésima junta prismática} \end{cases} \tag{3.8}$$

A segunda metade, ou a metade baixo da matriz do Jacobiano é dada por:

$$J_{\omega_i} = \begin{cases} z_{i-1} & \text{para a } i\text{-ésima junta revoluta} \\ 0 & \text{para a } i\text{-ésima junta prismática} \end{cases} \tag{3.9}$$

Juntando ambas as metades da matriz do Jacobiano obtém-se para a junta revoluta a equação (3.10) e para a junta prismática (3.11):

$$J_i = \begin{bmatrix} z_{i-1} \times (o_n - o_{i-1}) \\ z_{i-1} \end{bmatrix} \tag{3.10}$$

$$J_i = \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} \tag{3.11}$$

Assim sendo, as únicas ferramentas necessárias para calcular o Jacobiano são os vetores unitários z_i e as coordenadas das origens $o_1, ..., o_n$. As coordenadas para z_i são dadas pelos três primeiros elementos da terceira coluna de T_i^0, enquanto as coordenadas de

o_n são dadas pelos três primeiros elementos da quarta coluna de T_i^0 . Dessa forma, apenas a terceira e a quarta colunas da matriz de homogeneidade T são necessárias para obter o Jacobiano. (SPONG; HUTCHINSON; VIDYASAGAR, 2005)

3.1.2 Equações de *Euler-Lagrange*

Com o conjunto de coordenadas generalizadas independentes q_j, $j = 1, \ldots, n$, onde n representa os graus de liberdade do manipulador, o Lagrangiano do sistema é definido pela relação (3.12) (SPONG; HUTCHINSON; VIDYASAGAR, 2005), onde K representa a energia cinética e P a energia potencial do sistema:

$$L = K - P \tag{3.12}$$

Segundo Spong, Hutchinson e Vidyasagar (2005), em geral, as equações de *Euler-Lagrange* aplicadas a um sistema de n coordenadas podem ser representadas na forma da Equação (3.13), onde a força generalizada τ_i representa as forças externas e torques não deriváveis de uma função potencial:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \tag{3.13}$$

Conforme apresentado acima, as equações de *Euler-Lagrange* podem ser usadas para derivar as equações dinâmicas de maneira direta. É possível computar esses termos para um manipulador robótico de n elos por meio das fórmulas da energia cinética e da energia potencial usando as variáveis de articulação obtidas a partir da modelagem de *Denavit-Hartenberg* como coordenadas generalizadas (SPONG; HUTCHINSON; VIDYASAGAR, 2005).

3.1.2.1 Energia cinética para um manipulador de n elos

Segundo Spong, Hutchinson e Vidyasagar (2005) a energia cinética é dada pela soma de dois termos: a energia de translação, obtida concentrando toda a massa do objeto no centro de massa, e a energia cinética rotacional em torno do centro de massa. Assim, a energia cinética do manipulador é dada por:

$$K = \frac{1}{2}mv^T v + \frac{1}{2}\omega^T \Gamma \omega \tag{3.14}$$

onde m é a massa do objeto, v e ω são os vetores de velocidade linear e angular, respectivamente, e Γ é uma matriz simétrica 3×3 chamada Tensor de Inércia. O Tensor de Inércia é relacionado ao quadro de referência inercial do manipulador. Dessa forma, é possível relacionar o tensor de inércia com a matriz de rotação através de uma transformação de similaridade:

$$\Gamma = RIR^T \tag{3.15}$$

onde R é a matriz de rotação $R_n^0(q)$ obtida em (3.4), e I é uma matriz que não depende do movimento do objeto. Cada elemento dessa matriz é calculado através de integrais sobre as regiões do espaço ocupados por todas as partes do corpo rígido:

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \tag{3.16}$$

$$\begin{aligned} I_{xx} &= \iiint (y^2 + z^2)\rho(x,y,z)dxdydz \\ I_{yy} &= \iiint (x^2 + z^2)\rho(x,y,z)dxdydz \\ I_{zz} &= \iiint (x^2 + y^2)\rho(x,y,z)dxdydz \\ I_{xy} = I_{yx} &= -\iiint xy\rho(x,y,z)dxdydz \\ I_{xz} = I_{zx} &= -\iiint xz\rho(x,y,z)dxdydz \\ I_{yz} = I_{zy} &= -\iiint yz\rho(x,y,z)dxdydz \end{aligned} \tag{3.17}$$

Através do Tensor de Inércia, do jacobiano, da matriz de rotação e da massa de cada parte do braço obtém-se a energia cinética do manipulador:

$$K = \frac{1}{2}\dot{q}^T D(q)\dot{q} \tag{3.18}$$

sendo:

$$D(q) = \sum_{i=1}^{n}[m_i J_{vi}(q)^T J_{vi}(q) + J_{\omega i}(q)^T R_i(q) I_i(q) R_i(q)^T J_{\omega i}(q)] \tag{3.19}$$

3.1.2.2 Energia potencial para um manipulador de n elos

A energia potencial do manipulador de n elos é dada pela soma da energia potencial individual de cada parte envolvida. A única fonte de energia potencial do manipulador é a gravidade, assumindo que a massa total de cada elemento do manipulador está concentrada no seu centro de massa.

Segundo Spong, Hutchinson e Vidyasagar (2005), a energia potencial é uma função apenas das coordenadas generalizadas e não de suas derivadas, assim, a energia potencial depende da configuração do robô e independe da velocidade:

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} g^T r_{ci} m_i \tag{3.20}$$

onde g é o vetor que dá a direção da gravidade no referencial inercial, e r_{ci} fornece as coordenadas do centro de massa do elo i. Sendo assim, a matriz da energia potencial é dada por:

$$g(q) = \phi_k = \frac{\partial P}{\partial q_k} \tag{3.21}$$

3.1.2.3 Equações de movimento

Aplicando o que foi exposto acima, as equações de *Euler-Lagrange* (3.13) podem ser expressas segundo:

$$\sum_{i} d_{kj}(q)\ddot{q}_j + \sum_{i,j} c_{ijk}(q)\dot{q}_i\dot{q}_j + \phi_k = \tau_k \quad , \quad k = 1, \ldots, n \tag{3.22}$$

ou na forma matricial:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{3.23}$$

onde $D(q)$ e $g(q)$ representam as matrizes da energia cinética e potencial, respectivamente, e $C(q,\dot{q})$ representa uma matriz construída com os chamados Símbolos de Christoffel, definidos matematicamente por Spong, Hutchinson e Vidyasagar (2005) através da equação:

$$c_{ijk} = \frac{1}{2}\left\{\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right\} \tag{3.24}$$

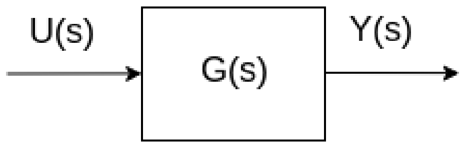
Como o manipulador RRR possui três juntas revolutas, e cada junta revoluta é descrita por uma matriz 3×3, serão necessários 27 Símbolos de Christoffel diferentes.

3.2 Ensaio em malha aberta para a obtenção do modelo da planta

O modelo matemático de um sistema é definido como um conjunto de equações que representa a sua dinâmica (OGATA, 2010). Esses modelos são sempre idealizações do comportamento real e são válidos para excitações dentro de certos limites de amplitude e frequência (CASTRUCCI; BITTAR; SALES, 2011).

Uma das formas de se obter o modelo de um determinado sistema $G(s)$, dentro de certos limites de amplitude e frequência, é através dos seus dados de entrada $U(s)$ e saída $Y(s)$, vide Figura 4. Ao aplicar na entrada uma referência conveniente (degrau, impulso, senoide), ignorando os fenômenos internos, obtém-se uma determinada saída que ao ser avaliada ao longo do tempo, ficam aparentes os parâmetros da equação diferencial procurada (CASTRUCCI; BITTAR; SALES, 2011).

Figura 4 – Diagrama do ensaio em malha aberta



Fonte: Do autor

A resposta temporal de um sistema de controle é composto da resposta transitória, aquela que compreende o tempo entre o estado inicial e o final, e a resposta estacionária que corresponde ao comportamento do sinal de saída quando t tende ao infinito (OGATA, 2010). Dependendo do comportamento das respostas do sistema, ele pode ser classificado como de primeira ordem (3.25), de segunda ordem (3.26) ou como sistema de ordem superior.

$$\frac{Y(s)}{U(s)} = \frac{1}{\tau s + 1} \tag{3.25}$$

$$\frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{3.26}$$

onde:

- τ: constante de tempo do sistema ($\tau > 0$);
- ω_n: frequência natural não amortecida ($\omega_n > 0$);
- ξ: coeficiente de amortecimento.

A constante de tempo do sistema de primeira ordem é definida como o instante em que a resposta do sistema atingiu 63,2% de sua variação final (CASTRUCCI; BITTAR; SALES, 2011). Por outro lado, para os sistemas de segunda ordem, o que define o ξ é o sobressinal máximo do sistema, e o ω_n pode ser definido através das constantes de tempo do sistema de segunda ordem, como por exemplo o tempo de acomodação. O tempo de acomodação para uma faixa de $\pm$ 2% em torno do valor final é dado por (3.27):

$$t_s \cong \frac{4}{\xi\omega_n} \tag{3.27}$$

3.3 Projeto de sistemas de controle pelo método do lugar das raízes

O lugar das raízes é um método simples que possibilita a representação gráfica das raízes da equação característica para todos os valores do ganho, ou qualquer outro parâmetro da função de transferência de malha aberta (OGATA, 2010). O projeto de controle pelo método do lugar das raízes é realizado através da adição de zeros e polos à função de transferência de malha aberta do sistema, forçando o lugar das raízes a passar pelos polos de malha fechada desejados.

No presente projeto, foram levantados três modelos para planta, um para cada uma das juntas (base, ombro, cotovelo). Para cada uma delas foi projetado um controlador independe (conhecido na literatura como controle de juntas independes). Os projetos foram realizado através do método do lugar das raízes, com o auxílio da ferramenta *sisotool* do *Matlab*.

3.4 Configuração do ambiente de simulação

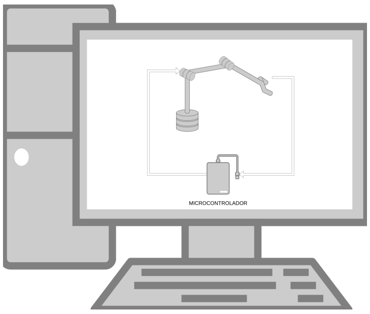
A partir dos modelos cinemáticos e dinâmicos obtidos, um ambiente de simulação é concebido na plataforma *Simulink/Matlab* (Figura 5a). Por meio desse ambiente, diferentes técnicas de controle podem ser sintetizadas e validadas para o manipulador estudado. Posteriormente, é feita a implementação da estratégia de controle sintetizada em dispositivo físico (microcontrolador), sendo que a validação é realizada no ambiente de simulação desenvolvido por meio da técnica (HIL), vide Figura 5b. Finalmente, a última etapa corresponde à implementação das estratégias de controle no manipulador robótico estudado (planta física). Como ilustrado na Figura 5c, nessa etapa o microntrolador é conectado ao manipulador de forma a controlá-lo, fazendo-o seguir uma trajetória pré-especificada. Assim, verifica-se se, de fato, a resposta do sistema físico em malha fechada corresponde à resposta obtida por meio de simulação utilizando a técnica HIL.

3.5 Resumo do Capítulo

Este capítulo apresentou de forma sucinta a metodologia de projeto a ser seguida para alcançar os objetivos propostos neste trabalho. No próximo capítulo são realizadas as modelagens cinemática e dinâmica do manipulador, a partir das quais o ambiente de simulação será construído.

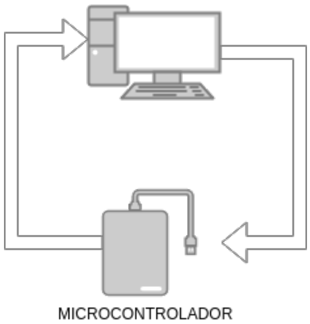
Figura 5 – Diagramas das etapas da técnica *Hardware-in-the-loop*

(a) Planta e controlador simulados



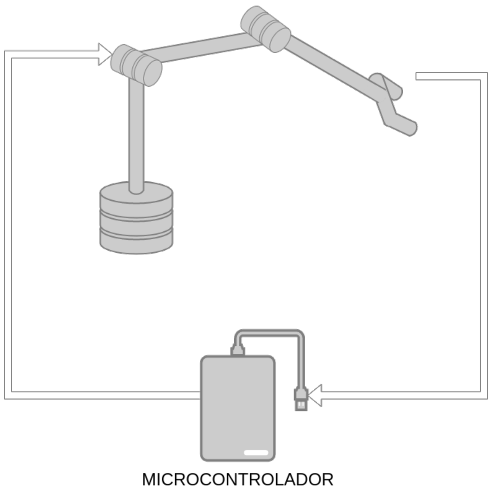
Fonte: Do autor

(b) Solução para o modelo simulado da planta



Fonte: Do autor

(c) Solução para a planta física



Fonte: Do autor

4 Resultados

Neste capítulo, os resultados para o projeto descritos no capítulo anterior serão apresentados. A obtenção do modelo da planta foi dada de duas formas diferentes: através da convenção de *Denavit-Hartenberg* e equações de *Euler-Lagrange* e através do ensaio em malha aberta. Para o primeiro método, foram feitas algumas considerações, estas seguem abaixo:

1. O elo do cotovelo e o elo do ombro são barras de comprimento L e massa m, e possuem o eixo de rotação no fim da barra;

2. O elo da base é um cilindro sólido de raio r, altura h e massa m;

3. A distribuição de massa de cada elo com a sua junta respectiva é simétrica em relação à estrutura do corpo. Ou seja, a massa está uniformemente distribuída ao longo do corpo.

A modelagem através da convenção de *Denavit-Hartenberg* e equações de *Euler-Lagrange* foi feita de forma programática através do MATLAB e segue no Anexo A. Os modelos da planta usados para projetar os constroladores foram obtidos através do ensaio em malha aberta e seguem na próxima seção. Três modelos foram levantados, um para a base, um para o ombro e outro para o cotovelo, e para cada um deles foi projetado um controlador, esse processo é conhecido na literatura como controle de juntas independentes.

4.1 Ensaio em malha aberta

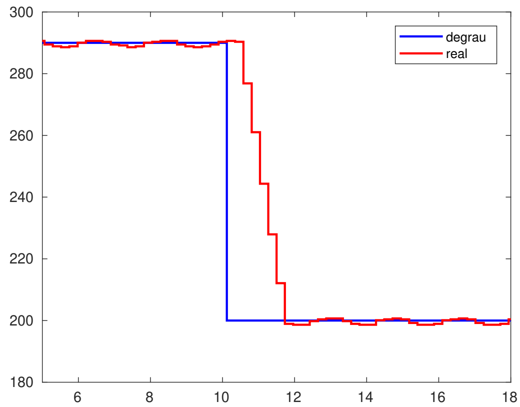
A comunicação com os servomotores foi feita inicialmente com um computador através da porta serial com o protocolo *UART*. A linguagem usada para a execução do ensaio em malha aberta foi o *python*, e seu código completo encontra-se em (LELIS, 2019d).

Para cada uma das juntas, foi feita uma montagem semelhante à Figura 4. Foi aplicada na entrada de cada um dos servos uma referência em degrau para o ângulo θ $(°C)$. Inicialmente, o sistema tinha sido configurado para uma amostragem de $0,01s$, de acordo com a resposta obtida, observou-se uma superamostragem dos dados, com isso, o tempo de amostragem foi alterado para $0,23s$. As saídas θ $(°C)$ obtidas ao longo do tempo seguem na Figura 6.

O modelo de cada junta poderia ser aproximado para uma modelo de primeira ordem, mas os servomotores possuem um controlador interno de configuração eletrônica

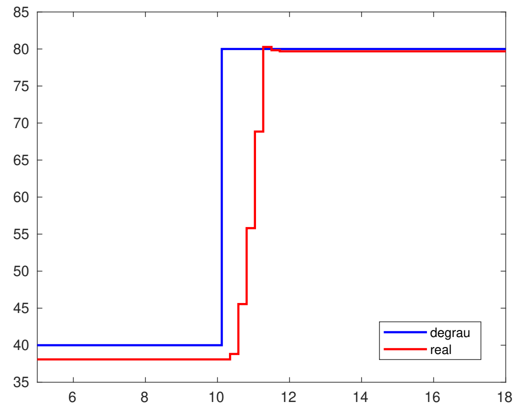
Figura 6 – Gráficos da entrada e resposta para o ensaio em MA de cada uma das juntas

(a) Base



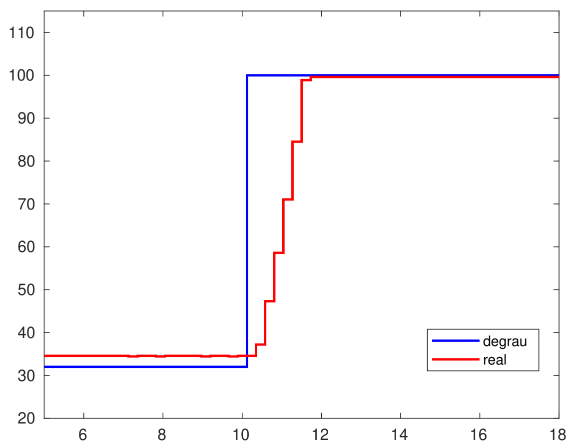
Fonte: Do autor

(b) Ombro



Fonte: Do autor

(c) Cotovelo



Fonte: Do autor

para o ângulo. Por esse motivo, o mais apropriado foi aproximar cada junta para um modelo de segunda ordem, conforme (3.26).

Observando as Figuras 6a e 6c, observa-se um comportamento próximo do amortecimento crítico. Quando ocorre esse tipo de comportamento, pode-se fazer a aproximação: $\xi = 1$ (OGATA, 2010), (CASTRUCCI; BITTAR; SALES, 2011). Por outro lado, em 6b, observa-se um pequeno sobressinal, o que resulta em um ξ menor, aproximando-se para este caso: $\xi = 0,8$.

O tempo de acomodação, definido em (3.27), variou para todas as juntas uma quantidade semelhante $0,9s < t_s < 1,3s$. De acordo com testes visuais para a validação do modelo, e pelas Figuras 6a, 6b e 6c, foi considerado para a base, ombro e cotovelo respectivamente: $t_s = 1,0952s$, $t_s = 0,9333s$, $t_s = 1,0952s$. Dessa forma, substituindo em (3.27) as constantes encontradas por aproximação, obtém-se para o ω_n da base, ombro e cotovelo respectivamente: $\omega_n = 3.6522rad/s$, $\omega_n = 5.3571rad/s$, $\omega_n = 3.6522rad/s$. O valor das variáveis encontradas para cada uma das juntas segue na Tabela 1

Tabela 1 – Constantes encontradas para o modelo de segunda ordem

Junta	ξ	t_s (s)	ω_n (rad/s)
Base	1	1,0952	3,6522
Ombro	0,8	0,9333	5,3571
Cotovelo	1	1,0952	3,6522

4.2 Fase 1 da técnica HIL

Como foi exposto na seção anterior, as constantes foram encontradas por aproximações de acordo com o que foi observado no ensaio em malha aberta. A partir disso, as funções de transferência para a base, ombro e cotovelo foram obtidas através da Equação 3.26 e seguem respectivamente em (4.1), (4.2) e (4.3):

$$G(s) = \frac{13,3384}{s^2 + 7,3043s + 13,3384} \tag{4.1}$$

$$G(s) = \frac{0,9965 \cdot 28,69898}{s^2 + 8,5714s + 28,69898} \tag{4.2}$$

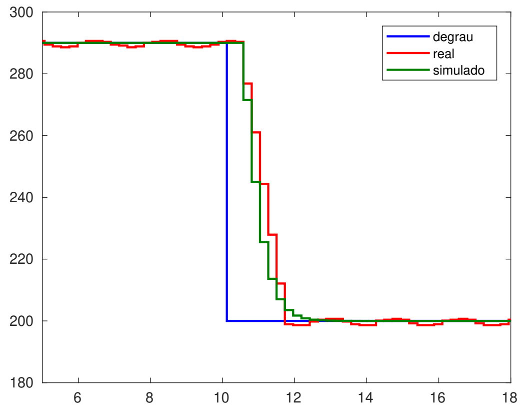
$$G(s) = \frac{13,3384}{s^2 + 7,3043s + 13,3384} \tag{4.3}$$

As funções de transferência obtidas representam uma simulação da planta do projeto. O mesmo degrau aplicado na planta real, foi aplicado nas funções de transferência para a validação dos modelos. Os resultados obtidos seguem na Figura 7a, 7b e 7c.

Obtidos os modelos, a próxima etapa da fase 1 da técnica HIL foi o projeto dos controladores pelo método do lugar das raízes. Considerando que as especificações do projeto de controle seriam atendidas caso houvesse pouco sobressinal na resposta, e o erro em estado estacionário igual a zero, mesmo impactando negativamente no tempo de acomodação, um controlador PI para cada junta seria suficiente. O controlador PI corresponde a Equação 2.1 sem a ação derivativa. Assim sendo, na análise do lugar das raízes (através da ferramenta *sisotool* do *Matlab*), foi colocado um polo de malha fechada na origem e um zero no eixo real para o controlador de cada uma das juntas. O ganho foi ajustado de tal forma que os polos de malha fechada ficassem o mais próximo possível do eixo real, assim a resposta não apresentaria oscilação (ou quase não apresentaria) ao fechar a malha. Outra consideração importante observada foi modelar o controlador de tal forma

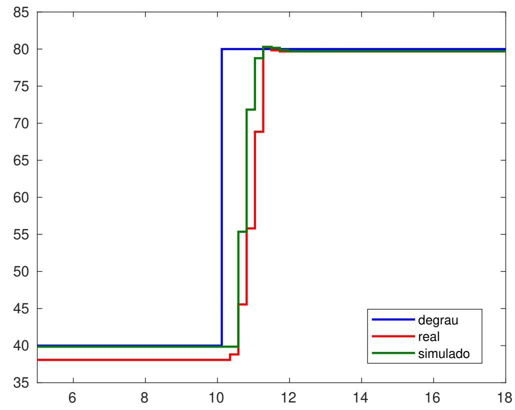
Figura 7 – Gráficos da entrada e resposta do modelo obtido para cada uma das juntas

(a) Base



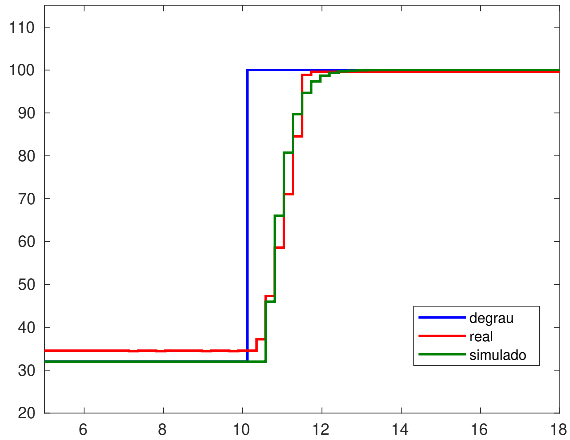
Fonte: Do autor

(b) Ombro



Fonte: Do autor

(c) Cotovelo



Fonte: Do autor

que mesmo depois da discretização, ele não apresentasse zeros não mínimos. O resultado final obtido para os controladores da base, ombro e cotovelo seguem respsectivamente em (4.4), (4.5) e (4.6):

$$C(s) = \frac{0,152s + 0,8}{s} \quad ou \quad C(s) = 0,152 + \frac{0,8}{s} \tag{4.4}$$

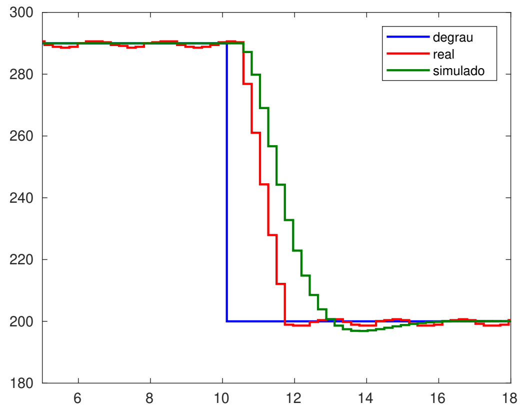
$$C(s) = \frac{0,413s + 0,9229}{s} \quad ou \quad C(s) = 0,413 + \frac{0,9229}{s} \tag{4.5}$$

$$C(s) = \frac{0,152s + 0,8}{s} \quad ou \quad C(s) = 0,152 + \frac{0,8}{s} \tag{4.6}$$

O código simulado completo com as funções de transferência e os controladores encontra-se em (LELIS, 2019a). Ao fechar as malhas, os resultados obtidos seguem na Figura 8a, na Figura 8b e na Figura 8c.

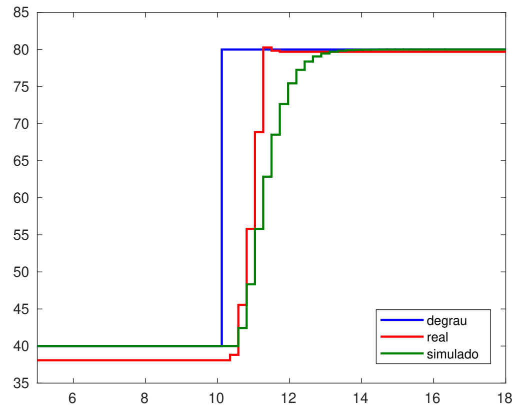
Figura 8 – Gráficos das respostas ao degrau em malha fechada - HIL Fase 1

(a) Base



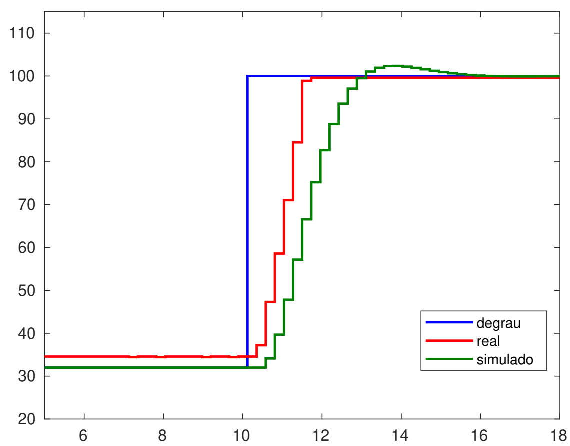
Fonte: Do autor

(b) Ombro



Fonte: Do autor

(c) Cotovelo



Fonte: Do autor

Com o primeiro controlador projetado para a base e para o cotovelo, ao fechar a malha, a resposta não apresentava sobresinal como mostrado na Figura 8a e na Figura 8c. Apesar disso, ao passar para o domínio discreto, o controlador passava a ter um zero não mínimo. E como foi citado anteriormente, o zero não mínimo é um comportamento indesejado para o sistema, por esse motivo, o controlador de ambas as juntas foi alterado.

4.3 Fase 2 da técnica HIL

Como o sinal da resposta da planta sofre uma discretização ao comunicar com a placa de aquisição de dados do computador e da *Raspberry*, torna-se necessária a discretização do controlador e da planta simulada. A discretização foi feita através do comando *c2d( )* do *Matlab* através do método do segurador de ordem zero e com um tempo de amostragem de $0,23s$. O resultado para cada uma das juntas segue abaixo respeitando a ordem de configuração do manipulador:

$$G(z) = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245} \quad e \quad C(z) = \frac{0,152z + 0,032}{z - 1} \tag{4.7}$$

$$G(z) = \frac{0,3876z - 0,198}{z^2 - 0,5515z + 0,1393} \quad e \quad C(z) = \frac{0,1572z + 0,06513}{z - 1} \tag{4.8}$$

$$G(z) = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245} \quad e \quad C(z) = \frac{0,152z + 0,032}{z - 1} \tag{4.9}$$

O próximo passo foi a obtenção da equação de diferenças para cada uma das plantas e controladores. Para isso, foi feito o que segue para a planta da base:

$$G(z) = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245}$$

$$\frac{Y(z)}{U(z)} = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245}$$

$$\frac{Y(z)}{U(z)} = \frac{0,8273/z - 0,6543/z^2}{1 - 0,9515/z + 0,1245/z^2}$$

$$Y(z) - \frac{0,9515}{z}\,Y(z) + \frac{0,1245}{z^2}\,Y(z) = \frac{0,8273}{z}\,U(z) - \frac{0,6543}{z^2}\,U(z)$$

$$y(k) = 0,8273\,u(k-1) - 0,6543\,u(k-2) + 0,9515\,y(k-1) - 0,1245\,y(k-2) \tag{4.10}$$

para o controlador da base:

$$C(z) = \frac{0,152z + 0,032}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,152z + 0,032}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,152 + 0,032/z}{1 - 1/z}$$

$$U(z) - \frac{U(z)}{z} = 0,152\ E(z) + \frac{0,032}{z}\ E(z)$$

$$u(k) = u(k-1) + 0,152\ e(k) + 0,032\ e(k-1) \tag{4.11}$$

para a planta do ombro:

$$G(z) = \frac{0,3876z + 0,198}{z^2 - 0,5515z + 0,1393}$$

$$\frac{Y(z)}{U(z)} = \frac{0,3876z + 0,198}{z^2 - 0,5515z + 0,1393}$$

$$\frac{Y(z)}{U(z)} = \frac{0,3876/z + 0,198/z^2}{1 - 0,5515/z + 0,1393/z^2}$$

$$Y(z) - \frac{0,5515}{z}\ Y(z) + \frac{0,1393}{z^2}\ Y(z) = \frac{0,3876}{z}\ U(z) + \frac{0,198}{z^2}\ U(z)$$

$$y(k) = 0,3876\ u(k-1) + 0,198\ u(k-2) + 0,5515\ y(k-1) - 0,1393\ y(k-2) \tag{4.12}$$

para o controlador do ombro:

$$C(z) = \frac{0,1572z + 0,06513}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,1572z + 0,06513}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,1572 + 0,06513/z}{1 - 1/z}$$

$$U(z) - \frac{U(z)}{z} = 0,1572\ E(z) + \frac{0,06513}{z}\ E(z)$$

$$u(k) = u(k-1) + 0,1572\,e(k) + 0,06513\,e(k-1) \tag{4.13}$$

para a planta do cotovelo:

$$G(z) = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245}$$

$$\frac{Y(z)}{U(z)} = \frac{0,8273z - 0,6543}{z^2 - 0,9515z + 0,1245}$$

$$\frac{Y(z)}{U(z)} = \frac{0,8273/z - 0,6543/z^2}{1 - 0,9515/z + 0,1245/z^2}$$

$$Y(z) - \frac{0,9515}{z}\,Y(z) + \frac{0,1245}{z^2}\,Y(z) = \frac{0,8273}{z}\,U(z) - \frac{0,6543}{z^2}\,U(z)$$

$$y(k) = 0,8273\,u(k-1) - 0,6543\,u(k-2) + 0,9515\,y(k-1) - 0,1245\,y(k-2) \tag{4.14}$$

para o controlador do cotovelo:

$$C(z) = \frac{0,152z + 0,032}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,152z + 0,032}{z - 1}$$

$$\frac{U(z)}{E(z)} = \frac{0,152 + 0,032/z}{1 - 1/z}$$

$$U(z) - \frac{U(z)}{z} = 0,152\,E(z) + \frac{0,032}{z}\,E(z)$$

$$u(k) = u(k-1) + 0,152\,e(k) + 0,032\,e(k-1) \tag{4.15}$$

Com as equações de diferença prontas, foi necessário realizar a montagem conforme a Figura 5b, configurar a conexão UART entre o computador e a *Raspberry*, e escrever o código em *python* para a planta simulada e para o controlador que passou a rodar na *Raspberry* (LELIS, 2019b). Com isso, a parte referente à simulação da planta ficou conforme o que segue:

```
# BASE - Funcao de transferencia da planta
y_k[0] = 0.8273*u_k_delay[0] - 0.6543*u_k_delay_2[0]
  + 0.9515*y_k_delay[0] - 0.1245*y_k_delay_2[0]

# SHOULDER - Funcao de transferencia da planta
y_k[1] = 0.3876*u_k_delay[1] + 0.198*u_k_delay_2[1]
  + 0.5515*y_k_delay[1] - 0.1393*y_k_delay_2[1];

# FOREARM - Funcao de transferencia da planta
y_k[2] = 0.8273*u_k_delay[2] - 0.6543*u_k_delay_2[2]
  + 0.9515*y_k_delay[2] - 0.1245*y_k_delay_2[2];
```

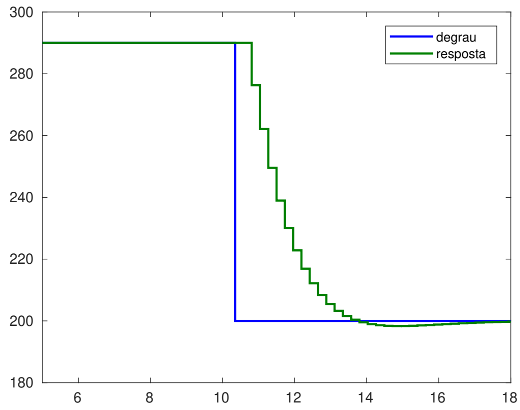
e do lado do controlador na *Raspberry*:

```
# BASE - Funcao de transferencia do controlador
u_k[0] = u_k_delay[0] + 0.152*e_k[0] + 0.032*e_k_delay[0]

# SHOULDER - Funcao de transferencia do controlador
u_k[1] = u_k_delay[1] + 0.1572*e_k[1] + 0.06513*e_k_delay[1]

# FOREARM - Funcao de transferencia do controlador
u_k[2] = u_k_delay[2] + 0.152*e_k[2] + 0.032*e_k_delay[2]
```

Foram aplicadas as mesmas referências em degrau da primeira fase da técnica HIL para cada uma das plantas simuladas. Assim sendo, as saídas obtidas ao longo do tempo para a fase 2 da técnica HIL seguem nas Figuras 10a, 10b e 10c.

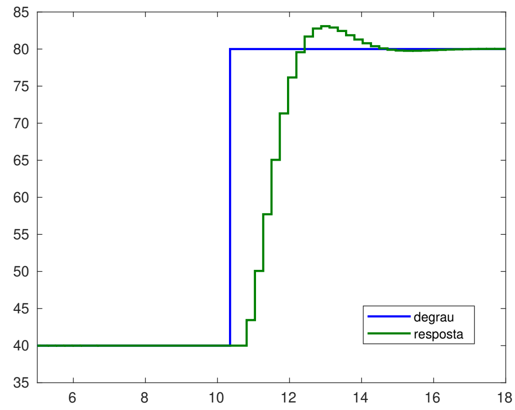
Figura 9 – Gráficos das respostas ao degrau em malha fechada - HIL Fase 2

(a) Base



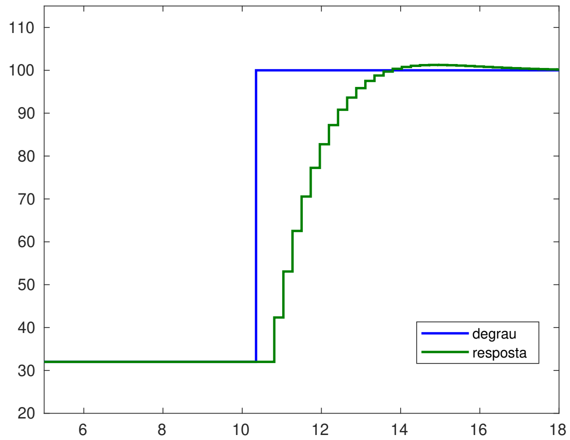
Fonte: Do autor

(b) Ombro



Fonte: Do autor

(c) Cotovelo



Fonte: Do autor

4.4 Fase 3 da técnica HIL

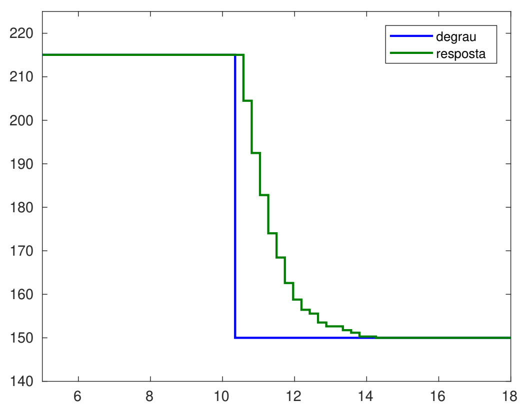
A última etapa da técnica HIL consiste em pegar o controlador que foi configurado na *Raspberry* da etapa anterior e inseri-lo na malha de controle da planta do manipulador, conforme a Figura 5c. Além disso, foi necessário configurar um ambiente de comunicação serial UART com os servomotores do manipulador, para isso, uma biblioteca em *python* foi utilizada. A malha foi fechada no próprio código do controlador, a parte do código referente à malha fechada segue abaixo, e o código completo econtra-se em (LELIS, 2019c).

Foram aplicadas referências em degrau, mas com leves diferenças para as amplitudes das etapas da técnica HIL anteriores. Assim sendo, as saídas obtidas ao longo do tempo para a fase 3 da técnica HIL seguem nas Figuras 10a, 10b e 10c.

```python
    def get_U_k(self, y_k):

        # Calcula lei de controle
        r_k = self.reference
        e_k = r_k - y_k

        # Controle de posicao
        u_k = self.u_k_delay + self.Kp*e_k + self.Ki*self.e_k_delay

        return u_k, e_k
```

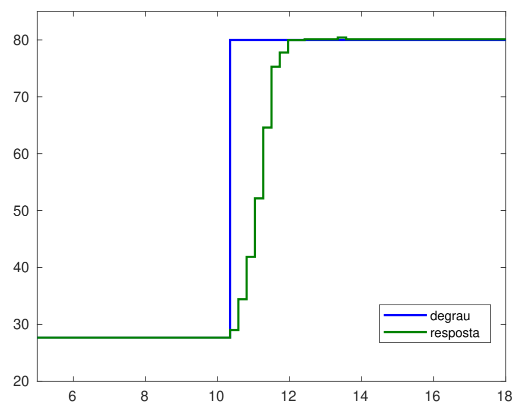
Figura 10 – Gráficos das respostas ao degrau em malha fechada - HIL Fase 3

(a) Base



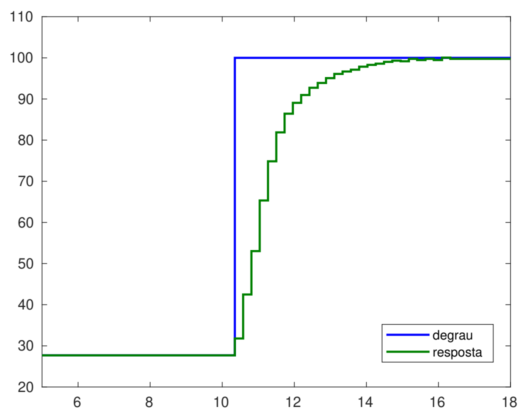
Fonte: Do autor

(b) Ombro



Fonte: Do autor

(c) Cotovelo



Fonte: Do autor

4.5 Resumo do Capítulo

Este capítulo apresentou os resultados alcançados no projeto, apresentando a memória de cálculo, explicitando como o sistema foi iplementado e expondo os resultados através dos gráficos. No próximo capítulo é apresentado o desfecho do projeto, como os resultados estão relacionados, são expostos os problemas encontrados e são apresentadas algumas sugestões para uma futura evolução do projeto.

5 Conclusão

A técnica HIL, que consiste em inserir um dispositivo físico na malha de controle, foi dividida em três etapas neste projeto. O resultado obtido em cada uma das etapas foi exposto através dos gráficos, e o objetivo é que o padrão se repita entre as etapas. Além disso, a sintonia e validação da estratégia de controle para a planta em questão é fundamental para o êxito da conclusão do projeto.

Analisando o exposto no capítulo anterior, observa-se que as considerações feitas para simplificar a modelagem *Denavit-Hartenberg* e as equações de *Euler-Lagrange* tornaram o desenvolvimento da solução para o modelo da planta fisicamente inviável. A solução mais coerente encontrada foi o ensaio em malha aberta para a obtenção do modelo de cada junta do manipulador, seguido do controle de junta independente.

A Figura 7 evidencia que a modelagem teve um resultado bastante coerente com a planta real, as curvas se sobrepuseram e os parâmetros como o tempo de acomodação e o sobressinal coincidiram com o esperado. O projeto de controle pelo método do lugar das raízes teve como resultado um controlador que tornou a dinâmica da planta mais lenta. Por outro lado, o controlador PI fez o sistema ficar mais robusto, rejeitando distúrbios ao fechar a malha além de tornar a transição dos estados mais suave (é possível notar a diferença entre a Figura 6 e a Figura 10). Além disso, o padrão da resposta do sistema ao longo do tempo se repetiu entre as etapas do método HIL. Sendo assim, o conjunto dos tópicos citados acima resultou na sintonia e validação da estratégia de controle.

Por fim, o desafio futuro envolve tanto a planta do manipulador quanto a configuração *Hardware in the loop*. Para o manipulador é necessário reduzir ou remover a trepidação observada principalmente na base. Essa trepidação foi reduzida ao validar a estratégia de controle (conforme a Figura 10 mostra), mas ela ainda ocorre. Em relação à técnica HIL, é de grande importância optar por um *hardware* de tempo real, o que não foi atendido com a *Raspberry*. A solução atual mais coerente seria modificar o *kernel Linux* da *Raspberry* para tornar o tempo de resposta aos eventos um tempo pré-definido.

Referências

ABOURIDA, S.; DUFOUR, C.; BELANGER, J. Real-time and hardware-in-the-loop simulation of electric drives and power electronics: Process, problems and solutions. *The 2005 International Power Eletronics Conference*, 2005. Citado 2 vezes nas páginas 11 e 12.

AGUIRRE, L. *Introdução à Identificação de Sistemas*. [S.l.: s.n.], 2015. ISBN 978-85-423-0079-6. Citado na página 5.

BACIC, M. On hardware-in-the-loop simulation. *Proceedings of the 44th IEEE Conference on Decision and Control*, p. 3194–3198, Dec 2005. ISSN 0191-2216. Citado 2 vezes nas páginas 6 e 11.

BOUSCAYROL, A. Different types of hardware-in-the-loop simulation for electric drives. *IEEE International Symposium on Industrial Electronics*, p. 2146 – 2151, 08 2008. Citado na página 7.

BULLOCK, D. et al. Hardware-in-the-loop simulation. *Transportation Research Part C Emerging Technologies*, v. 12, p. 73–89, 02 2004. Citado na página 7.

CASTRUCCI, P.; BITTAR, A.; SALES, R. *Controle Automatico*. Rio de Janeiro, RJ: Editora GEN/LTC, 2011. Citado 6 vezes nas páginas 5, 10, 11, 19, 20 e 24.

EVANS, M. B.; SCHILLING, L. J. The role of simulation in the development and flight test of the himat vehicle. 05 1984. Citado na página 11.

ISERMANN, R.; SCHAFFNIT, J.; SINSEL, S. Hardware-in-the-loop simulation for the design and testing of engine-control systems. *Control Engineering Practice*, v. 7, p. 643–653, 1999. Citado 2 vezes nas páginas 7 e 11.

JAIN, S.; KAMALI, C. Hardware in the loop simulation for a mini uav. *IFAC-PapersOnLine*, v. 49, n. 1, p. 700 – 705, 2016. ISSN 2405-8963. 4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2016. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2405896316301380>. Citado na página 7.

KARPENKO, M.; SEPEHRI, N. Hardware-in-the-loop simulator for research on fault tolerant control of electrohydraulic flight control systems. p. 7 pp.–, June 2006. ISSN 0743-1619. Citado na página 7.

LELIS, L. H. S. *Etapa 1 do método Hardware in the Loop*. [S.l.]: GitHub, 2019. <https://github.com/luizhlelis/rrrManipulatorControl/tree/master/hil_simulation/phase1_simulated/model_2>. Citado na página 27.

LELIS, L. H. S. *Etapa 2 do método Hardware in the Loop*. [S.l.]: GitHub, 2019. <https://github.com/luizhlelis/rrrManipulatorControl/tree/master/hil_simulation/phase2_control_simulation>. Citado na página 31.

LELIS, L. H. S. *Etapa 3 do método Hardware in the Loop*. [S.l.]: GitHub, 2019. <https://github.com/luizhlelis/rrrManipulatorControl/tree/master/hil_simulation/phase3_plant_implament>. Citado na página 32.

LELIS, L. H. S. *Obtendo o modelo através do ensaio em malha aberta*. [S.l.]: GitHub, 2019. <https://github.com/luizhlelis/rrrManipulatorControl/tree/master/hil_simulation/collectingData>. Citado na página 23.

LYNCH, K. M.; PARK, F. C. *Modern Robotics: Mechanics, Planning, and Control*. 1st. ed. New York, NY, USA: Cambridge University Press, 2017. ISBN 1107156300, 9781107156302. Citado na página 14.

MURPHY, R. R. *Introduction to AI Robotics*. 1st. ed. Cambridge, MA, USA: MIT Press, 2000. ISBN 0262133830. Citado na página 9.

OGATA, K. *Modern Control Engineering*. 5th. ed. Prentice Hall, 2010. (Instrumentation and controls series). ISBN 9780136156734. Disponível em: <https://books.google.com.br/books?id=Wu5GpNAelzkC>. Citado 7 vezes nas páginas 5, 10, 11, 19, 20, 21 e 24.

OPPENHEIN, A.; WILLSKY, A. *Signals & systems*. Englewood Cliffs, N.J: Prentice-Hall.: Pearson Education, 1997. Citado na página 5.

PAUL, R. *Robot Manipulators: Mathematics, Programming, and Control : the Computer Control of Robot Manipulators*. [S.l.]: MIT Press, 1981. (MIT Press series in artificial intelligence). ISBN 9780262160827. Citado 4 vezes nas páginas 9, 13, 14 e 15.

PHILLIPS, C. L.; NAGLE, H. T. *Digital Control System Analysis and Design*. 4th. ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007. ISBN 0130812226, 9780130812223. Citado na página 10.

ROTHSTEIN, A.; SIEKMANN, L.; STAUDT, V. Combined-hardware-in-the-loop system and its test-bench verification. *2017 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM) 2017 Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP)*, p. 581–586, May 2017. Citado na página 7.

SICILIANO, B. et al. *Robotics: Modelling, Planning and Control*. Springer London, 2008. (Advanced Textbooks in Control and Signal Processing). ISBN 9781846286421. Disponível em: <https://books.google.com.br/books?id=VsTOQOnQjCAC>. Citado 4 vezes nas páginas 9, 14, 15 e 16.

SPONG, M.; HUTCHINSON, S.; VIDYASAGAR, M. *Robot Modeling and Control*. Wiley, 2005. ISBN 9780471649908. Disponível em: <https://books.google.com.br/books?id=wGapQAAACAAJ>. Citado 8 vezes nas páginas 9, 10, 14, 15, 16, 17, 18 e 19.

Anexos

ANEXO A – Obtenção do modelo através da convenção de *Denavit-Hartenberg* e equações de *Euler-Lagrange*

```
clc;
close all;

% --------------------- 1) DENAVIT - HATENBERG ------------------------
% Rodar o Matlab com um usuario com alto privilegio
% 1.1) Carregando a matriz de Denavit-Hatenberg

dh = [
    0,  0,      0,      0;
    0,  0.085,  0.065,  -pi/2;
    0,  0,      0.172,  0;
    0,  0,      0.235,  0];
dh_representation = SerialLink(dh)

% Carregando angulos iniciais das juntas (radianos)
q = [0, 0, 0, 0]

% Representacao grafica do manipulador
dh_representation.plot(q)
dh_representation.teach

% 1.2) Matrizes homogeneas

% Especificidades do manipulador em estudo
syms theta_0 theta_1 theta_2 theta_3;

alpha_0 = 0;
alpha_1 = -pi/2;
alpha_2 = 0;
alpha_3 = 0;

a0 = 0;
a1 = 0.065;
```

```
a2 = 0.172;
a3 = 0.235;

d0 = 0;
d1 = 0.085;
d2 = 0;
d3 = 0;

syms s0 c0 s1 c1 s2 c2 s3 c3
% Usar para exibir resultado em funcao de theta (resultado simbolico)
A_0 =    [
            [c0      -s0*1     s0*0    a0*c0];
            [s0      c0*1      -c0*0   a0*s0];
            [0       0         1       d0    ];
            [0       0         0       1     ];
         ];
A_1 =    [
            [c1      -s1*0     s1*-1    a1*c1];
            [s1      c1*0      -c1*-1   a1*s1];
            [0       -1         0       d1    ];
            [0       0          0       1     ];
         ];
A_2 =    [
            [c2      -s2*1    s2*0    a2*c2];
            [s2      c2*1      -c2*0   a2*s2];
            [0       0         1       d2    ];
            [0       0         0       1     ];
         ];
A_3 =    [
            [c3      -s3*1     s3*0    a3*c3];
            [s3      c3*1      -c3*0   a3*s3];
            [0       0         1       d3    ];
            [0       0         0       1     ];
         ];

% Matriz homogenea resultante - z0 = [0 0 1]' e o0 = [0 0 0]'
T_0_1 = A_1;
T_0_2 = A_1*A_2;
T_0_3 = A_1*A_2*A_3;
Tresp = vpa(T_0_3,3);

R_1 = T_0_1(1:3,1:3);
```

```
R_2 = T_0_2(1:3,1:3);
R_3 = T_0_3(1:3,1:3);

% -------------------- 2) EULER - LAGRANGE ----------------------------

% 2.1) Jacobiano

% z_i = terceira coluna / o_i = quarta coluna
z_0 = A_0(1:3,3);
o_0 = A_0(1:3,4);

z_1 = T_0_1(1:3,3);
o_1 = T_0_1(1:3,4);

z_2 = T_0_2(1:3,3);
o_2 = T_0_2(1:3,4);

z_3 = T_0_3(1:3,3);
o_3 = T_0_3(1:3,4);

% Jacobianos para as juntas revolutas (SPONG)
J_parc_1 =  [
                cross(z_0,(o_3-o_0));
                z_0;
            ];
J_parc_2 =  [
                cross(z_1,(o_3-o_1));
                z_1;
            ];
J_parc_3 =  [
                cross(z_2,(o_3-o_2));
                z_2;
            ];

% Resultados
J_1 =   [
            J_parc_1     zeros(6,2);
        ];
J_2 =   [
            J_parc_1     J_parc_2     zeros(6,1);
        ];
J_3 =   [
```

```
            J_parc_1     J_parc_2     J_parc_3;
        ];

J_1 = vpa(J_1,3);
J_2 = vpa(J_2,3);
J_3 = vpa(J_3,3)

% ----------------------------- Matriz D(q) ---------------------------

% Elbow manipulator / articulated / revolute / anthropomorphic / RRR

% As tres primeiras linhas da matriz J correspondem a velocidade
% linear e as tres ultimas a velocidade angular
J_v1 = J_1(1:3,:);
J_w1 = J_1(4:6,:);
J_v2 = J_2(1:3,:);
J_w2 = J_2(4:6,:);
J_v3 = J_3(1:3,:);
J_w3 = J_3(4:6,:);

% Barra de comprimento L e massa m (Eixo de rotacao no fim da barra)
% Iy = Iz = 1/3*m*L^2, onde o eixo x eh o unico paralelo a barra

% Caso a distribuicao de massa do corpo seja simetrica em relacao a
% estrutura do corpo, entao os produtos cruzados de inercia sao
% identicos a zero
m_motor = .055;
m_elo = .010;
Raio_1 = .085;
g = 9.8;
L_1 = .085;
L_2 = .172;
L_3 = .235;

m_1 = m_elo;
m_2 = m_elo+m_motor*2;
m_3 = m_elo+m_motor*4;

I_1 =   [
            0               0               0;
            0               0               0;
            0               0               1/2*m_1*Raio_1^2;
```

```
        ]
I_2 =   [
            0               0               0;
            0               0               0;
            0               0               1/3*m_2*L_2^2;
        ]
I_3 =   [
            0               0               0;
            0               0               0;
            0               0               1/3*m_3*L_3^2;
        ]

D1 = m_1*(J_v1).'*J_v1+J_w1.'*R_1*I_1*R_1.'*J_w1;
D2 = m_2*(J_v2).'*J_v2+J_w2.'*R_2*I_2*R_2.'*J_w2;
D3 = m_3*(J_v3).'*J_v3+J_w3.'*R_3*I_3*R_3.'*J_w3;

D = vpa(D1,3) + vpa(D2,3) + vpa(D3,3);

% -------------------- Simbolos de Christoffel -----------------------

% Derivadas: sen(x) = cos(x) / cos(x) = -sen(x)

syms theta_1 theta_2 theta_3
D_11 = (0.065*sin(theta_1) + 0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3))*(0.015*sin(theta_1)
    + 0.04*cos(theta_2)*sin(theta_1) +
    0.054*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.054*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    (0.065*sin(theta_1) + 0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3))*(7.8e-3*sin(theta_1)
    + 0.021*cos(theta_2)*sin(theta_1) +
    0.028*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.028*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    (0.065*cos(theta_1) + 0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))*(6.5e-4*cos(theta_1)
    + 1.7e-3*cos(theta_1)*cos(theta_2) +
    2.4e-3*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    2.4e-3*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    (0.065*cos(theta_1) + 0.17*cos(theta_1)*cos(theta_2) +
```

```
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))*(0.015*cos(theta_1)
    + 0.04*cos(theta_1)*cos(theta_2) +
    0.054*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.054*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    (0.065*cos(theta_1) + 0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))*(7.8e-3*cos(theta_1)
    + 0.021*cos(theta_1)*cos(theta_2) +
    0.028*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.028*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    (6.5e-4*sin(theta_1) + 1.7e-3*cos(theta_2)*sin(theta_1) +
    2.4e-3*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    2.4e-3*sin(theta_1)*sin(theta_2)*sin(theta_3))*(0.065*sin(theta_1)
    + 0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3));

D_12 = 1.0*cos(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.015*sin(theta_1) +
    0.04*cos(theta_2)*sin(theta_1) +
    0.054*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.054*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    1.0*cos(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(7.8e-3*sin(theta_1) +
    0.021*cos(theta_2)*sin(theta_1) +
    0.028*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.028*sin(theta_1)*sin(theta_2)*sin(theta_3)) -
    1.0*sin(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.015*cos(theta_1) +
    0.04*cos(theta_1)*cos(theta_2) +
    0.054*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.054*cos(theta_1)*sin(theta_2)*sin(theta_3)) -
    1.0*sin(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(7.8e-3*cos(theta_1) +
    0.021*cos(theta_1)*cos(theta_2) +
    0.028*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.028*cos(theta_1)*sin(theta_2)*sin(theta_3));
```

```
D_13 = 1.0*cos(theta_1)*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.015*sin(theta_1) +
    0.04*cos(theta_2)*sin(theta_1) +
    0.054*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.054*sin(theta_1)*sin(theta_2)*sin(theta_3)) -
    1.0*sin(theta_1)*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.015*cos(theta_1) +
    0.04*cos(theta_1)*cos(theta_2) +
    0.054*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.054*cos(theta_1)*sin(theta_2)*sin(theta_3));

D_21 = 0.35*cos(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.065*sin(theta_1) +
    0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) -
    0.35*sin(theta_1)*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.065*cos(theta_1) +
    0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3));

D_22 = (0.23*sin(theta_1)*(0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    0.23*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) *
    (1.0*sin(theta_1)*(0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    1.0*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) +
    (0.12*sin(theta_1)*(0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    0.12*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
```

```
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) *
    (1.0*sin(theta_1)*(0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    1.0*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) +
    cos(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) + 0.35*cos(theta_1)^2*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))^2 +
    1.0*sin(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) + cos(theta_1)^2*(1.2e-3*cos(theta_1)^2 +
    1.2e-3*sin(theta_1)^2) + 0.35*sin(theta_1)^2*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) + 0.24*cos(theta_3)*sin(theta_2))^2 +
    1.0*sin(theta_1)^2*(1.2e-3*cos(theta_1)^2 + 1.2e-3*sin(theta_1)^2);

D_23 = (0.23*sin(theta_1)*(0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    0.23*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) *
    (1.0*cos(theta_1)*(0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    1.0*sin(theta_1)*(0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3))) +
    cos(theta_1)^2*(4.2e-3*cos(theta_1)^2 + 4.2e-3*sin(theta_1)^2) +
    1.0*sin(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) +
    0.23*cos(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) + 0.24*cos(theta_3)*sin(theta_2)) +
    0.23*sin(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) + 0.24*cos(theta_3)*sin(theta_2));

D_31 = 0.23*cos(theta_1)*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.065*sin(theta_1) +
    0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) -
```

```
    0.23*sin(theta_1)*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.065*cos(theta_1) +
    0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3));

D_32 =
    (0.23*cos(theta_1)*(0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    0.23*sin(theta_1)*(0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)))*(1.0*sin(theta_1) *
    (0.17*cos(theta_2)*sin(theta_1) +
    0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)) +
    1.0*cos(theta_1)*(0.17*cos(theta_1)*cos(theta_2) +
    0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3))) +
    cos(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) +
    1.0*sin(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) +
    0.23*cos(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) + 0.24*cos(theta_3)*sin(theta_2)) +
    0.23*sin(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))*(0.17*sin(theta_2) +
    0.24*cos(theta_2)*sin(theta_3) + 0.24*cos(theta_3)*sin(theta_2));

D_33 = 0.23*cos(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))^2 +
    0.23*sin(theta_1)^2*(0.24*cos(theta_2)*sin(theta_3) +
    0.24*cos(theta_3)*sin(theta_2))^2 +
    cos(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) +
    1.0*sin(theta_1)^2*(4.2e-3*cos(theta_1)^2 +
    4.2e-3*sin(theta_1)^2) +
    (0.23*cos(theta_1)*(0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
    0.23*sin(theta_1)*(0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3))) *
    (1.0*cos(theta_1)*(0.24*cos(theta_1)*cos(theta_2)*cos(theta_3) -
    0.24*cos(theta_1)*sin(theta_2)*sin(theta_3)) +
```

```
    1.0*sin(theta_1)*(0.24*cos(theta_2)*cos(theta_3)*sin(theta_1) -
    0.24*sin(theta_1)*sin(theta_2)*sin(theta_3)));

del_d11_theta1 = diff(D_11,theta_1);
del_d11_theta2 = diff(D_11,theta_2);
del_d11_theta3 = diff(D_11,theta_3);

del_d12_theta1 = diff(D_12,theta_1);
del_d12_theta2 = diff(D_12,theta_2);
del_d12_theta3 = diff(D_12,theta_3);

del_d13_theta1 = diff(D_13,theta_1);
del_d13_theta2 = diff(D_13,theta_2);
del_d13_theta3 = diff(D_13,theta_3);

del_d21_theta1 = diff(D_21,theta_1);
del_d21_theta2 = diff(D_21,theta_2);
del_d21_theta3 = diff(D_21,theta_3);

del_d22_theta1 = diff(D_22,theta_1);
del_d22_theta2 = diff(D_22,theta_2);
del_d22_theta3 = diff(D_22,theta_3);

del_d23_theta1 = diff(D_23,theta_1);
del_d23_theta2 = diff(D_23,theta_2);
del_d23_theta3 = diff(D_23,theta_3);

del_d31_theta1 = diff(D_31,theta_1);
del_d31_theta2 = diff(D_31,theta_2);
del_d31_theta3 = diff(D_31,theta_3);

del_d32_theta1 = diff(D_32,theta_1);
del_d32_theta2 = diff(D_32,theta_2);
del_d32_theta3 = diff(D_32,theta_3);

del_d33_theta1 = diff(D_33,theta_1);
del_d33_theta2 = diff(D_33,theta_2);
del_d33_theta3 = diff(D_33,theta_3);

c_111 = .5*(del_d11_theta1 + del_d11_theta1 - del_d11_theta1);
c_112 = .5*(del_d21_theta1 + del_d21_theta1 - del_d11_theta2);
c_113 = .5*(del_d31_theta1 + del_d31_theta1 - del_d11_theta3);
```

```
c_121 = .5*(del_d12_theta1 + del_d11_theta2 - del_d12_theta1);
c_122 = .5*(del_d22_theta1 + del_d21_theta2 - del_d12_theta2);
c_123 = .5*(del_d32_theta1 + del_d31_theta2 - del_d12_theta3);

c_131 = .5*(del_d13_theta1 + del_d11_theta3 - del_d13_theta1);
c_132 = .5*(del_d23_theta1 + del_d21_theta3 - del_d13_theta2);
c_133 = .5*(del_d33_theta1 + del_d31_theta3 - del_d13_theta3);

c_211 = .5*(del_d11_theta2 + del_d11_theta1 - del_d11_theta1);
c_212 = .5*(del_d21_theta2 + del_d21_theta1 - del_d11_theta2);
c_213 = .5*(del_d31_theta2 + del_d31_theta1 - del_d11_theta3);

c_221 = .5*(del_d12_theta2 + del_d11_theta2 - del_d12_theta1);
c_222 = .5*(del_d22_theta2 + del_d21_theta2 - del_d12_theta2);
c_223 = .5*(del_d32_theta2 + del_d31_theta2 - del_d12_theta3);

c_231 = .5*(del_d13_theta2 + del_d11_theta3 - del_d13_theta1);
c_232 = .5*(del_d23_theta2 + del_d21_theta3 - del_d13_theta2);
c_233 = .5*(del_d33_theta2 + del_d31_theta3 - del_d13_theta3);

c_311 = .5*(del_d11_theta3 + del_d11_theta1 - del_d11_theta1);
c_312 = .5*(del_d21_theta3 + del_d21_theta1 - del_d11_theta2);
c_313 = .5*(del_d31_theta3 + del_d31_theta1 - del_d11_theta3);

c_321 = .5*(del_d12_theta3 + del_d11_theta2 - del_d12_theta1);
c_322 = .5*(del_d22_theta3 + del_d21_theta2 - del_d12_theta2);
c_323 = .5*(del_d32_theta3 + del_d31_theta2 - del_d12_theta3);

c_331 = .5*(del_d13_theta3 + del_d11_theta3 - del_d13_theta1);
c_332 = .5*(del_d23_theta3 + del_d21_theta3 - del_d13_theta2);
c_333 = .5*(del_d33_theta3 + del_d31_theta3 - del_d13_theta3);

% -------------------------- Energia Potencial --------------------

phi_1 = 0;
phi_2 = m_2*g*L_2*cos(theta2);
phi_3 = m_3*g*L_3*cos(theta2 + theta3);
```