

Gustavo Lemos Schwartz
Luiz Felipe Mantovani Rodrigues

**Análise de desempenho de controle embarcado
em microcontrolador e em
Hardware-In-the-Loop de um manipulador
robótico.**

Campos dos Goytacazes - RJ

2018

Gustavo Lemos Schwartz
Luiz Felipe Mantovani Rodrigues

**Análise de desempenho de controle embarcado em
microcontrolador e em *Hardware-In-the-Loop* de um
manipulador robótico.**

Trabalho de conclusão de curso apresentado
ao Instituto Federal de Educação, Ciência e
Tecnologia Fluminense como requisito par-
cial para conclusão do curso de Bacharelado
em Engenharia de Controle e Automação.

Instituto Federal Fluminense – IFF
Campus Campos Centro
Engenharia de Controle e Automação

Orientador: M.Sc. Edson Simões dos Santos

Campos dos Goytacazes - RJ
2018

Biblioteca Anton Dakitsch
CIP - Catalogação na Publicação

S399a

Schwartz, Gustavo Lemos, Rodrigues, Luiz Felipe Mantovani
Análise de desempenho de controle embarcado em microcontrolador e
em Hardware-In-the-Loop de um manipulador robótico. / Rodrigues, Luiz
Felipe Mantovani Schwartz, Gustavo Lemos - 2018.
85 f.; il. color.

Orientador: Edson Simões dos Santos

Trabalho de conclusão de curso (graduação) -- Instituto Federal de
Educação, Ciência e Tecnologia Fluminense, Campus Campos Centro,
Curso de Bacharelado em Engenharia de Controle e Automação, Campos dos
Goytacazes, RJ, 2018.

Referências: f. 78 a 79.

1. Robótica. 2. Espaço de Estados. 3. Hardware-In-the-Loop. 4. Controle
Embarcado. I. Santos, Edson Simões dos, orient. II. Título.

Gustavo Lemos Schwartz
Luiz Felipe Mantovani Rodrigues

Análise de desempenho de controle embarcado em microcontrolador e em *Hardware-In-the-Loop* de um manipulador robótico.

Trabalho de conclusão de curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia Fluminense como requisito parcial para conclusão do curso de Bacharelado em Engenharia de Controle e Automação.

M.Sc. Edson Simões dos Santos
INSTITUTO FEDERAL
FLUMINENSE
Orientador

D.Sc. Rogerio Atem de Carvalho
INSTITUTO FEDERAL
FLUMINENSE
Convidado 1

D.Sc. William da Silva Vianna
INSTITUTO FEDERAL
FLUMINENSE
Convidado 2

Campos dos Goytacazes - RJ
2018

"Não são os títulos que ilustram os homens, são os homens que ilustram os títulos."

(Niccolo Machiavelli)

Agradecimentos

Primeiramente a Deus, que nos deu força para seguir adiante mesmo nos momentos mais difíceis, tanto dentro quanto fora do meio discente, e nos proporcionou momentos maravilhosos de grande aprendizado em nossa vida.

Aos nossos pais e irmãos, que deram total apoio e confiança nessa trajetória.

Aos nossos amigos, Rodrigo Brasil, Murilo Serpa, Fábio Ramos, Tomás Marquez e Bruna Leonardo, companheiros de universidade e amigos para todas as horas que tornaram a nossa trajetória mais prazerosa e descontraída.

Ao Professor e amigo Edson Simões dos Santos, pelos ensinamentos, orientação e confiança em nosso trabalho.

Ao Instituto Federal Fluminense, seu corpo docente e direção.

E a todos que de alguma forma contribuíram com nossa formação e crescimento pessoal, nossos mais sinceros agradecimentos.

*"O caminho do homem justo é cercado por todos os lados,
pela tirania dos homens maus e iniquidade dos egoístas.
Abençoado àquele que em nome da caridade e da boa vontade
conduz os fracos através do vale das trevas,
leva consigo seus irmãos e acha a última ovelha desgarrada.
E eu atingirei com raiva furiosa e vingança grandiosa
àqueles que tentarem envenenar e destruir meus irmãos.
E você saberá que meu nome é Senhor quando minha lei se abater sobre vós."*

(Bíblia Sagrada, Ezequiel 25:17)

Resumo

Com a evolução tecnológica da indústria, os processos industriais vêm se tornando cada vez mais complexos, exigindo que as estratégias de controle utilizadas sejam as mais confiáveis possíveis. No momento de escolher a estratégia de controle mais adequada à um projeto deve-se levar em consideração o tipo de *Hardware* utilizado, sendo necessário que se faça uma análise de seus prós e contras, uma vez que um mesmo algoritmo de controle pode se comportar de maneira diferente dependendo da plataforma escolhida. Este trabalho visa comparar as respostas de um mesmo controlador em duas abordagens distintas: utilizando *Hardware-In-the-Loop*, sendo processado por um computador pessoal; e embarcado em placa controladora Arduino. A planta objeto do controle foi um manipulador robótico do modelo RD5 NT da fabricante Didacta Italia. Através do *software* Matlab® e algumas de suas ferramentas como o Simulink™ foram levantadas as curvas dos atuadores das juntas do robô, estimadas as respectivas funções de transferência e representado o sistema através de equações de estado. Em seguida, um controlador baseado no modelo foi criado e testado nas duas diferentes abordagens de controle. Após diversos testes ficou evidente a diferença das respostas do sistema através dos gráficos e da análise visual, além de algumas limitações do controle em HIL em comparação ao embarcado.

Palavras-chaves: Arduino, *Hardware-In-the-Loop*, Controle Embarcado, Robótica, Controle por Realimentação de Estados, Observador de Estado

Abstract

With industry technological evolution, industrial processes has become even more complex, requiring that the control strategies used be as reliable as possible. When choosing the more suitable control strategy for a project, the tipe of hardware must be considered, making necessary to evaluate its pros and cons, since the behaviour of a control algorithm may differ depending on the choosen hardware. This paper compares the responses of the same control algorithm in two different approaches: Hardware-In-the-Loop, using a personal computer to process the algorithm; and embedded in Arduino controller board. The controlled plant was a robotic manipulator model RD5 NT from manufacturer Didaacta Italia. Through software Matlab® and some of its tools such as Simulink™, the curves of the joints motors were aquired, their transfer functions estimated and the system represented by equations of state. Furthermore, a control algorithm based on the model was developed and tested on the two different control approaches. After several tests, the difference among the system responses became evident through the graphs and visual analysis, showing also some limitations of HIL control towards embedded control.

Key words: Arduino, *Hardware-In-the-Loop*, Embedded Control, Robotics, Full State Feedback Control, State Observer

Listas de ilustrações

Figura 1 – Equação Geral de Espaço de Estados	18
Figura 2 – Forma Canônica controlável das Matrizes de Espaço de Estados	19
Figura 3 – Matriz de Transformação de Base	20
Figura 4 – Forma Genérica das Matrizes da Forma Canônica Controlável	20
Figura 5 – Acréscimo da Diferença Entre Saída Medida e Saída Estimada no Polinômio Genérico	21
Figura 6 – Placa Arduino UNO	22
Figura 7 – Placa Arduino MEGA 2560	23
Figura 8 – Manipulador Robótico RD5 NT	25
Figura 9 – Diagrama de blocos para levantamento das curvas dos motores das juntas	26
Figura 10 – Curvas Real (Vermelho) e Estimada (Azul) da posição da base	27
Figura 11 – Função de transferência estimada da posição da base	28
Figura 12 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da base	28
Figura 13 – Função de transferência estimada da velocidade da base	28
Figura 14 – Curvas Real (Vermelho) e Estimada (Azul) da posição da 1 ^a junta	29
Figura 15 – Função de transferência estimada da posição da 1 ^a junta	29
Figura 16 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da 1 ^a junta	30
Figura 17 – Função de transferência estimada da velocidade da 1 ^a junta	30
Figura 18 – Curvas Real (Vermelho) e Estimada (Azul) da posição da 2 ^a junta	31
Figura 19 – Função de transferência estimada da posição da 2 ^a junta	31
Figura 20 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da 2 ^a junta	32
Figura 21 – Função de transferência estimada da velocidade da 2 ^a junta	32
Figura 22 – Curvas Real (Vermelho) e Estimada (Azul) da posição do punho	33
Figura 23 – Função de transferência estimada da posição do punho	33
Figura 24 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade do punho	34
Figura 25 – Função de transferência estimada da velocidade do punho	34
Figura 26 – Componentes das matrizes (Base)	35
Figura 27 – Matrizes de Estado (Base)	35
Figura 28 – Matrizes Após a Mudança de Base (Base)	36
Figura 29 – Componentes das matrizes (1 ^a junta)	36
Figura 30 – Matrizes de Estado (1 ^a junta)	36
Figura 31 – Matrizes Após a Mudança de Base (1 ^a junta)	37
Figura 32 – Componentes das matrizes (2 ^a junta)	37
Figura 33 – Matrizes de Estado (2 ^a junta)	37
Figura 34 – Matrizes Após a Mudança de Base (2 ^a junta)	38
Figura 35 – Componentes das matrizes (Punho)	38

Figura 36 – Matrizes de Estado (Punho)	38
Figura 37 – Matrizes Após a Mudança de Base (Punho)	39
Figura 38 – Diagrama de blocos HIL	41
Figura 39 – <i>Setpoint</i> de velocidade	42
Figura 40 – Função exponencial do SP de velocidade	42
Figura 41 – Diagrama de blocos de leitura dos sensores HIL	43
Figura 42 – Diagrama de blocos da BASE - HIL	44
Figura 43 – Integrador da posição da Base	45
Figura 44 – Integrador da velocidade da Base	45
Figura 45 – Vetor de controle K após adição do Integrador - Base	45
Figura 46 – Função de Transferência da posição da Base compensada	46
Figura 47 – Função de Transferência da velocidade da Base compensada	46
Figura 48 – Diagrama de blocos da 1 ^a Junta - HIL	47
Figura 49 – Integrador da posição da 1 ^a Junta	48
Figura 50 – Integrador da velocidade da 1 ^a Junta	48
Figura 51 – Vetor de controle K após adição do Integrador - 1 ^a Junta	48
Figura 52 – Função de Transferência da posição da 1 ^a Junta compensada	49
Figura 53 – Função de Transferência da velocidade da 1 ^a Junta compensada	49
Figura 54 – Diagrama de blocos da 2 ^a Junta - HIL	50
Figura 55 – Integrador da posição da 2 ^a Junta	51
Figura 56 – Integrador da velocidade da 2 ^a Junta	51
Figura 57 – Vetor de controle K após adição do Integrador - 2 ^a Junta	51
Figura 58 – Função de Transferência da posição da 2 ^a Junta compensada	52
Figura 59 – Função de Transferência da velocidade da 2 ^a Junta compensada	52
Figura 60 – Diagrama de blocos do Punho - HIL	53
Figura 61 – Integrador da posição do Punho	53
Figura 62 – Integrador da velocidade do Punho	54
Figura 63 – Vetor de controle K após adição do Integrador - Punho	54
Figura 64 – Função de Transferência da posição do Punho compensada	54
Figura 65 – Função de Transferência da velocidade do Punho compensada	54
Figura 66 – Uso de memória da placa Arduino MEGA	55
Figura 67 – Diagrama de blocos - Controle embarcado	56
Figura 68 – Diagrama de blocos da leitura do controle embarcado	57
Figura 69 – Pulses de sincronia com $T_s=0.05s$	59
Figura 70 – Pulses de sincronia com $T_s=0.04s$	59
Figura 71 – Novo diagrama de blocos embarcado	60
Figura 72 – Novo diagrama de blocos HIL	60
Figura 73 – Inicialização dos algoritmos	61
Figura 74 – Posição da Base - HIL ($T_s=0.05s$)	62

Figura 75 – Posição da Base - HIL (Ts=0.04s)	62
Figura 76 – Posição da Base - Embarcado (Ts=0.05s)	63
Figura 77 – Posição da Base - Embarcado (Ts=0.04s)	64
Figura 78 – Posição da Base - Embarcado (Ts=0.005s)	64
Figura 79 – Posição da Base - Embarcado (Ts=0.002s)	65
Figura 80 – Posição da Base - Embarcado (Ts=0.001s)	65
Figura 81 – Posição da 1 ^a Junta - HIL (Ts=0.05s)	66
Figura 82 – Posição da 1 ^a Junta - HIL (Ts=0.04s)	66
Figura 83 – Posição da 1 ^a Junta - Embarcado (Ts=0.05s)	67
Figura 84 – Posição da 1 ^a Junta - Embarcado (Ts=0.04s)	67
Figura 85 – Posição da 1 ^a Junta - Embarcado (Ts=0.005s)	68
Figura 86 – Posição da 1 ^a Junta - Embarcado (Ts=0.002s)	68
Figura 87 – Posição da 2 ^a Junta - HIL (Ts=0.05s)	69
Figura 88 – Posição da 2 ^a Junta - HIL (Ts=0.04s)	69
Figura 89 – Posição da 2 ^a Junta - Embarcado (Ts=0.05s)	70
Figura 90 – Posição da 2 ^a Junta - Embarcado (Ts=0.04s)	70
Figura 91 – Posição da 2 ^a Junta - Embarcado (Ts=0.005s)	71
Figura 92 – Posição da 2 ^a Junta - Embarcado (Ts=0.002s)	71
Figura 93 – Posição da 2 ^a Junta - Embarcado (Ts=0.001s)	72
Figura 94 – Posição do Punho - HIL (Ts=0.05s)	72
Figura 95 – Posição do Punho - HIL (Ts=0.04s)	73
Figura 96 – Posição do Punho - Embarcado (Ts=0.05s)	73
Figura 97 – Posição do Punho - Embarcado (Ts=0.04s)	74
Figura 98 – Posição do Punho - Embarcado (Ts=0.005s)	74
Figura 99 – Posição do Punho - Embarcado (Ts=0.002s)	75
Figura 100 – Posição do Punho - Embarcado (Ts=0.001s)	75
Figura 101 – Localização do <i>System Identification Tool</i> no MATLAB	81
Figura 102 – Importando dados no SIT	82
Figura 103 – Configurando dados no SIT	82
Figura 104 – Tratamento dos dados no SIT	83
Figura 105 – Estimação de modelo do sistema no SIT	83
Figura 106 – Parâmetros de estimação de Função de Transferência no SIT	84
Figura 107 – Curva estimada pelo SIT	84
Figura 108 – Função de Transferência estimada pelo SIT	85

Lista de tabelas

Tabela 1 – Especificações da placa Arduino UNO	23
Tabela 2 – Especificações da placa Arduino MEGA 2560	24
Tabela 3 – Amplitudes de Movimento das juntas	24

Lista de abreviaturas e siglas

g	gramas;
HIL	<i>Hardware-in-the-loop</i> ;
Hz	Hertz;
KB	kilobytes;
mA	miliampère;
MHz	megaherz;
mm	milímetros;
PC	Computador Pessoal;
PID	Controlador Proporcional-Integral-Derivativo;
PWM	<i>Pulse-Width Modulation</i> (Modulação por largura de pulso);
s	Segundos;
SIT	<i>System Identification Tool</i> ;
SP	<i>Setpoint</i> (Valor de referência);
Ts	<i>Sample Time</i> (Período de amostragem);
V	Volts;
Vcc	Volts em Corrente Contínua;

Sumário

1	INTRODUÇÃO	15
1.1	Objetivo Geral	15
1.2	Objetivos Específicos	16
1.3	Justificativa	16
1.4	Disposição e organização dos capítulos	16
2	CONCEITUAÇÃO TEÓRICA	18
2.1	Espaço de Estados	18
2.1.1	Matriz de Mudança de Base	19
2.1.2	Observador de Estados	21
2.2	<i>Harware-In-the-Loop (HIL)</i>	21
2.3	Microcontrolador	22
2.4	Manipulador robótico	24
3	MODELAGEM	26
3.1	Levantamento das curvas	26
3.2	Modelo em Espaço de Estados	35
4	CONTROLE	40
4.1	Controle em HIL	40
4.2	Controle embarcado	55
5	ANÁLISE	58
5.1	Sincronismo	58
5.2	Resultados finais	62
6	CONSIDERAÇÕES FINAIS	76
6.1	Conclusão	76
6.2	Sugestões para trabalhos futuros	77
	REFERÊNCIAS	78
	APÊNDICES	80
	APÊNDICE A – <i>SYSTEM IDENTIFICATION TOOL</i>	81

1 Introdução

Com o advento de novas tecnologias pós Indústria 4.0 e o desenvolvimento de conceitos como Tecnologia das Coisas, Revolução Digital, entre outros, o homem moderno se encontra em constante busca sobre ferramentas que facilitem e automatizem tarefas do seu cotidiano.

Em se tratando de robótica especificamente, sua origem é vinculada ao ambiente industrial. No século XX, as linhas de produção baseadas no modelo Fordista utilizavam máquinas projetadas especificamente para uma só função, a isso se dá o nome de Automação Rígida. Com o avanço da robótica industrial, passou-se a utilizar uma mesma máquina para realizar múltiplas funções, esta sendo chamada de Automação Flexível. (ROMANO, 2002)

Hoje a robótica avançou para resolver problemas mais comuns na vida das pessoas, aproximando, assim, as máquinas dos lares e escritórios. Atualmente é possível encontrar robôs em qualquer ambiente realizando as mais variadas tarefas como limpar o chão, transportar ou mover objetos.

Os manipuladores ou braços robóticos são ótimos exemplos de ferramentas que auxiliam tarefas normais do cotidiano, e são, também, robôs didáticos largamente utilizados para o ensino da robótica e da programação. Sua construção é relativamente simples, sendo formado basicamente por juntas, elos, sensores e atuadores.

Já seu controle é complexo, pois um sistema de controle monovariáveis como um PID, por exemplo, torna difícil seu controle de movimentos, que dependem de múltiplas variáveis, como posição e velocidade, com precisão. Pode-se utilizar, por exemplo, um modelo em Espaço de estados, o qual possibilita utilizar um vetor de controle, enviando um sinal de correção proporcional a mais de uma variável.

Um dos paradigmas da robótica é a relativa diferença nas velocidades de transmissão e processamento de dados entre um controle implementado diretamente em *hardware* e o controle realizado em *software* por um computador não-dedicado.

1.1 Objetivo Geral

Realizar o controle de um manipulador robótico através da realimentação de estados, utilizando um Observador de estado, implementando-o diretamente em *hardware* e, em seguida, em *Hardware-In-the-Loop* com um PC para análise de resultados.

1.2 Objetivos Específicos

- Obter as curvas reais referentes às juntas do manipulador robótico e estimá-las em funções de transferência;
- Desenvolver um controlador utilizando um modelo em espaço de estados baseado nas funções de transferência;
- Utilizar a placa Arduino como plataforma de embarque para o controle diretamente em *hardware* e como placa de aquisição de dados no controle em *Hardware-In-the-Loop* utilizando o *software* de simulação e controle Matlab/Simulink;
- Realizar testes que comprovem as diferenças entre as abordagens de controle;

1.3 Justificativa

Pela importância do controle moderno e suas diversas aplicações, e pela possibilidade de se implementar algoritmos de controle nas mais diversas formas, faz-se necessário escolher o tipo de controle e o tipo de *Hardware* a ser utilizado. Nesta necessidade, vê-se muitas aplicações didáticas que poderiam ser feitas de maneira mais simples e com melhor desempenho com plataformas de fácil acesso, na ausência de onerosos equipamentos industriais.

Embora conceitos como *Hardware-In-the-Loop* sejam largamente utilizados na indústria, como práticas de teste antes de se embarcar um algoritmo de controle em um *Hardware* dedicado, ainda há carência de conhecimento nesta área em ambientes de ensino.

1.4 Disposição e organização dos capítulos

Capítulo 1 - Introdução: Este capítulo apresenta uma contextualização do tema abordado pelo trabalho, trazendo informações gerais sobre como este foi desenvolvido, assim como seus objetivos e justificativa.

Capítulo 2 - Conceituação Teórica: Este capítulo apresenta a teoria envolvida neste trabalho, envolvendo Espaço de Estados, Observador de Estado e *Hardware-In-the-Loop* bem como as plataformas utilizadas.

Capítulo 3 - Modelagem: Este capítulo demonstra como foi feito o levantamento das curvas dos atuadores da manipulador e a representação em Espaço de Estados.

Capítulo 4 - Controle: Este capítulo explica os algoritmos de controle utilizados, assim como a metodologia envolvida em cada abordagem.

Capítulo 5 - Análise: Este capítulo apresenta a análise sobre a comparação das diferentes abordagens de controle testadas.

Capítulo 6 - Considerações Finais: Este capítulo apresenta as conclusões do trabalho e sugestões para trabalhos futuros.

Referências: Lista as fontes consultadas que serviram como base para o trabalho.

Apêndice A - *System Identification Tool*: Este apêndice apresenta uma simples explicação de como foi utilizado a ferramenta *System Identification Tool* para a estimativa das Funções de Transferência.

2 Conceituação Teórica

Este capítulo apresentará a teoria base deste estudo, bem como os componentes utilizados neste.

2.1 Espaço de Estados

Espaço de estados é o espaço de n dimensões em que os eixos de coordenadas são os eixos x_1, x_2, \dots, x_n . Nestes moldes, qualquer estado pode ser representado por um ponto no espaço de estados (SEQUEIRA, 2003). Para melhor entendimento do conceito de espaço de estados é necessário que previamente sejam conhecidos outros conceitos importantes, que são:

Estado – O estado de um sistema dinâmico é o menor conjunto de variáveis que colhido para um instante de tempo $t = t_0$, junto à uma entrada para $t \geq t_0$ determina completamente a resposta do sistema em qualquer instante regido por $t \geq t_0$ (ANGELICO; SCALASSARA; VARGAS, 2015).

Variáveis de estado – As variáveis de estado de um sistema dinâmico são o menor conjunto de variáveis que determinam o estado do sistema dinâmico, então, se pelo menos n variáveis $x_1(t), x_2(t), \dots, x_n(t)$ são necessárias para descrever completamente o comportamento de um sistema dinâmico, então estas variáveis são um conjunto de variáveis de estado (SEQUEIRA, 2003).

Vetor de estado – É um vetor que determina de maneira unissonante o estado de um sistema $x(t)$ para qualquer $t \geq t_0$ quando especificada a entrada $u(t)$ para $t \geq t_0$. Este vetor é composto pelas n variáveis de estados que são necessárias para descrever completamente o comportamento do sistema (SEQUEIRA, 2003).

Em posse da função de transferência relacionando a entrada e a saída de um sistema, é possível transportá-la para um modelo em espaço de estados utilizando-se da equação da Figura 1.

Figura 1 – Equação Geral de Espaço de Estados

$$G(s) = \frac{B_0 s^n + B_1 s^{n-1} + B_2 s^{n-2} + \dots + B_n}{s^n + A_1 s^{n-1} + A_2 s^{n-2} + \dots + A_n}$$

Fonte: (LEMOS, 2015).

Em que os elementos $B_0, B_1, B_2, \dots, B_n$ e A_1, A_2, \dots, A_n devem ser analisados de

acordo com a função de transferência em questão, permitindo organizar as matrizes do modelo seguindo a estrutura da Figura 2.

Figura 2 – Forma Canônica controlável das Matrizes de Espaço de Estados

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_{n-1}(t) \\ \dot{x}_{n-2}(t) \end{bmatrix} &= \begin{bmatrix} 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \cdot & \cdot & \ddots & \cdot \\ \cdot & \cdot & \ddots & \cdot \\ 0 & 0 & \dots & 1 \\ -A_n & -A_{n-1} & \dots & -A_1 \end{bmatrix} \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{n-1}(t) \\ x_{n-2}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= [B_n - A_n B_0 \quad B_{n-1} - A_{n-1} B_0 \quad \dots \quad B_1 - A_1 B_0] \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} + B_0 u(t) \end{aligned}$$

Fonte: (LEMOS, 2015).

Em que:

- **A** é a matriz de estado
- **B** é a matriz de entrada
- **C** é a matriz de saída
- **D** é a matriz de transmissão direta

Dessa maneira pode-se representar um sistema multivariáveis por uma equação de estados.

2.1.1 Matriz de Mudança de Base

Utilizando a mudança de base, pode-se representar o mesmo sistema numa base diferente. Assim, escolhe-se representar o modelo numa base em que a matriz C esteja no formato identidade, de modo que os estados sejam iguais às saídas.

Para realizar a mudança de base é preciso encontrar a matriz de transformação (ou matriz T). Através desta serão determinadas as novas matrizes de estado na forma canônica (GONÇALVES; SODRÉ, 2006).

Primeiramente, deve-se adotar a nova matriz de saída como sendo a matriz identidade, chamada de Cn. A matriz de transformação pode ser obtida através da equação

$T = C^{-1} \cdot C_n$. Portanto, após encontrar a inversa da matriz de saída (C), calcula-se a matriz T , que por sua vez é a matriz C^{-1} , visto que qualquer matriz multiplicada por uma matriz identidade não sofrerá alteração. A composição da matriz T pode ser visualizada na Figura 3.

Figura 3 – Matriz de Transformação de Base

$$T = C^{-1} = [B_n - A_n B_0 \quad B_{n-1} - A_{n-1} B_0 \quad \dots \quad B_1 - A_1 B_0]^{-1}$$

Fonte: (GONÇALVES; SODRÉ, 2006).

Para encontrar as demais matrizes, as seguintes fórmulas são utilizadas:

- $A_n = T^{-1} \cdot A \cdot T$
- $B_n = T^{-1} \cdot B$

A forma genérica das matrizes da forma canônica controlável é exibida na Figura 4.

Figura 4 – Forma Genérica das Matrizes da Forma Canônica Controlável

$$\begin{array}{c} \textbf{An} \\ \left[\begin{array}{c} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_{n-1}(t) \\ \dot{x}_{n-2}(t) \end{array} \right] = \left[\begin{array}{ccccc} 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \\ -A_n & -A_{n-1} & \dots & -A_1 \end{array} \right] \cdot \left[\begin{array}{c} x_1(t) \\ x_2(t) \\ \vdots \\ x_{n-1}(t) \\ x_{n-2}(t) \end{array} \right] + \left[\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{array} \right] u(t) \end{array}$$

$$\begin{array}{c} \textbf{Cn} \\ y(t) = \left[\begin{array}{cccc} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 \end{array} \right] \cdot \left[\begin{array}{c} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{array} \right] \end{array}$$

Fonte: (GONÇALVES; SODRÉ, 2006).

Estas novas matrizes serão utilizadas para os cálculos de controle, uma vez que, além de simplificar os cálculos envolvidos, a medição direta dos estados do sistema é de suma importância para um controle eficiente.

2.1.2 Observador de Estados

Para que um sistema possa ter seu estado inicial determinado corretamente é necessário que este seja observável, ou seja, se qualquer condição inicial puder ser obtida conhecendo-se as entradas e saídas do sistema para todo instante de tempo t entre 0 e T (CABRAL, 2018).

O esquema que estima o estado a partir da entrada e da saída é denominado de Observador de Estado (SILVEIRA, 2010). Deste modo, é necessário que um observador de estado seja construído para estimar variáveis do sistema que não possuam sensores para a medição direta.

Em sistemas que a obtenção dos estados está atrelada ao uso do polinômio genérico de estados, um problema recorrente é o fato de que, uma vez aplicado na planta real, qualquer perturbação do estado no sistema real, iria afastá-lo do estado do modelo, uma vez que não existe nenhum mecanismo de correção (BARÃO, 2000).

A fim de corrigir este problema, adiciona-se a diferença entre a saída medida e a saída estimada para que o modelo seja corrigido com este sinal. A equação obtida após a aplicação deste método é mostrada na Figura 5.

Figura 5 – Acréscimo da Diferença Entre Saída Medida e Saída Estimada no Polinômio Genérico

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - \hat{y})$$

(BARÃO, 2000)

Onde L é um vetor coluna de dimensão correspondente às variáveis do sistema. O vetor L deve ser selecionado de forma que se obtenha um bom comportamento para o erro de estimação, isto é, que o erro converja rapidamente para zero (BARÃO, 2000). Para efeitos de visualização, as aplicações adiante irão chamar o vetor L de Ke.

Neste trabalho, os valores de Ke serão definidos de acordo com a resposta do sistema, com sua escolha baseada em testes empíricos.

2.2 *Harware-In-the-Loop (HIL)*

Uma simulação HIL refere-se a uma simulação que inclui ou é capaz de incluir qualquer parte de hardware do sistema simulado em tempo real (ALBUQUERQUE, 2007).

O conceito de *Hardware-in-the-loop* refere-se à uma simulação onde alguns componentes envolvidos na mesma são substituídos por equipamentos reais. Um dos principais usos para esta ferramenta é a realização de testes para parametrização de controladores de forma segura e barata (PALMA, 2006).

Sistemas em HIL são, usualmente, realizados com controladores reais em plantas simuladas para testar a resposta destes controladores sem danificar a planta real. Neste trabalho, porém, foi feito o oposto, testamos um controlador virtual em uma planta real (TUTUNJI; SALEEM, 2015).

Neste projeto, a parte física do sistema foi composta pelo manipulador robótico, pela interface de potência e pelas placas Arduino, enquanto o *software* Matlab foi utilizado para programar o algoritmo de controle.

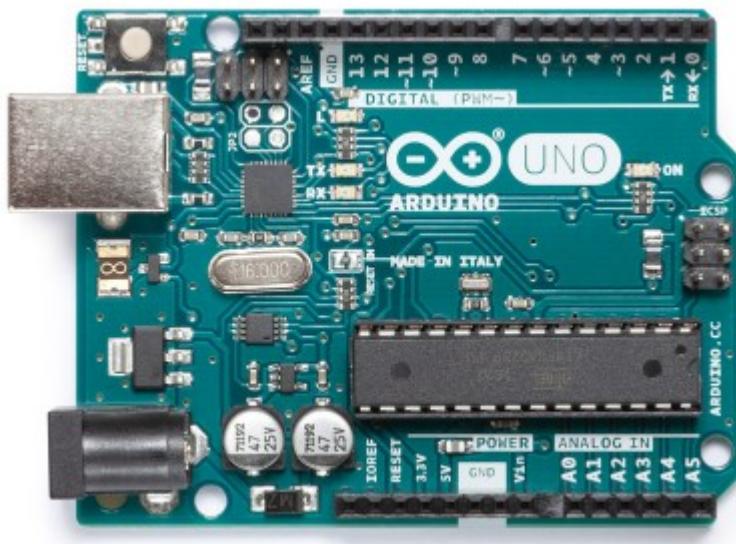
2.3 Microcontrolador

Foram utilizadas duas placas microprocessadas Arduino, uma para o controle embarcado e outra como interface de aquisição de dados entre computador e manipulador robótico.

Segundo o fabricante, “Arduino é uma plataforma eletrônica de código aberto baseada em hardware e software fáceis de usar” (ARDUINO, 2018).

Foi utilizada uma placa Arduino UNO, ilustrada na Figura 6, para leitura dos sensores do braço robótico a fim de coletar os dados para análise. As especificações da placa se encontram na Tabela 1.

Figura 6 – Placa Arduino UNO



Fonte: <https://store.arduino.cc/usa/arduino-uno-rev3>

Tabela 1 – Especificações da placa Arduino UNO

Microcontrolador	ATmega328P
Tensão de operação	5V
Tensão de entrada recomendada	7-12V
Tensão de entrada limite	6-20V
Pinos de E/S Digitais	14 (6 com saída PWM)
Pinos de entrada analógica	6
Corrente DC por pino de E/S	20 mA
Memória Flash	32 KB (dos quais 0.5 KB utilizados para inicialização)
Velocidade de Clock	16 MHz
Comprimento	68,6 mm
Largura	53,4 mm
Peso	25 g

Fonte: <https://store.arduino.cc/usa/arduino-uno-rev3>

Foi utilizada, também, uma placa Arduino MEGA 2560, ilustrada na Figura 7, como *hardware* dedicado no controle embarcado e como interface de aquisição de dados no controle HIL. Suas especificações são exibidas na Tabela 2.

Figura 7 – Placa Arduino MEGA 2560



Fonte: <https://store.arduino.cc/usa/arduino-mega-2560-rev3>

Tabela 2 – Especificações da placa Arduino MEGA 2560

Microcontrolador	ATmega2560
Tensão de operação	5V
Tensão de entrada recomendada	7-12V
Tensão de entrada limite	6-20V
Pinos de E/S Digitais	54 (15 com saída PWM)
Pinos de entrada analógica	16
Corrente DC por pino de E/S	20 mA
Memória Flash	256 KB (dos quais 8 KB utilizados para inicialização)
Velocidade de Clock	16 MHz
Comprimento	101,52 mm
Largura	53,3 mm
Peso	37 g

Fonte: <https://store.arduino.cc/usa/arduino-mega-2560-rev3>

2.4 Manipulador robótico

Este manipulador, modelo RD5 NT fabricado pela empresa Didacta Italia, é constituído de 5 eixos e foi desenvolvido para fins didáticos, sendo aplicado diretamente na área de estudo da robótica (ITALIA, 2008). Possuindo duas juntas articuladas no corpo mais base e punho, este manipulador apresenta quatro graus de liberdade. São ao todo cinco motores alimentados por uma fonte de 15Vcc: um para a base, dois para as juntas articuladas, um para o movimento do punho e um para abrir e fechar a sua garra. As juntas conferem ao robô a configuração TRR-R, ou seja, o braço é composto por uma junta torcional e duas rotacionais, e o punho é do tipo rotacional (CARRARA, 2015). Para realizar a leitura de suas juntas, são utilizados quatro potenciômetros sem fim que são alimentados por uma fonte de 5Vcc. No quesito comunicação, o manipulador troca dados a partir de um cabo DB25, conectado a uma interface que interliga o manipulador ao Arduino.

Na Tabela 3 são mostradas as amplitudes máximas de movimento das juntas do robô. A Figura 8 mostra o manipulador real que se encontra no Laboratório de Automação Inteligente do Instituto Federal Fluminense - *Campus Campos Centro*.

Tabela 3 – Amplitudes de Movimento das juntas

Articulação	Graus
Base	293°
1ª Junta	107°
2ª Junta	284°
Punho	360°

Fonte: (ITALIA, 2008)

Figura 8 – Manipulador Robótico RD5 NT



Fonte: Autores.

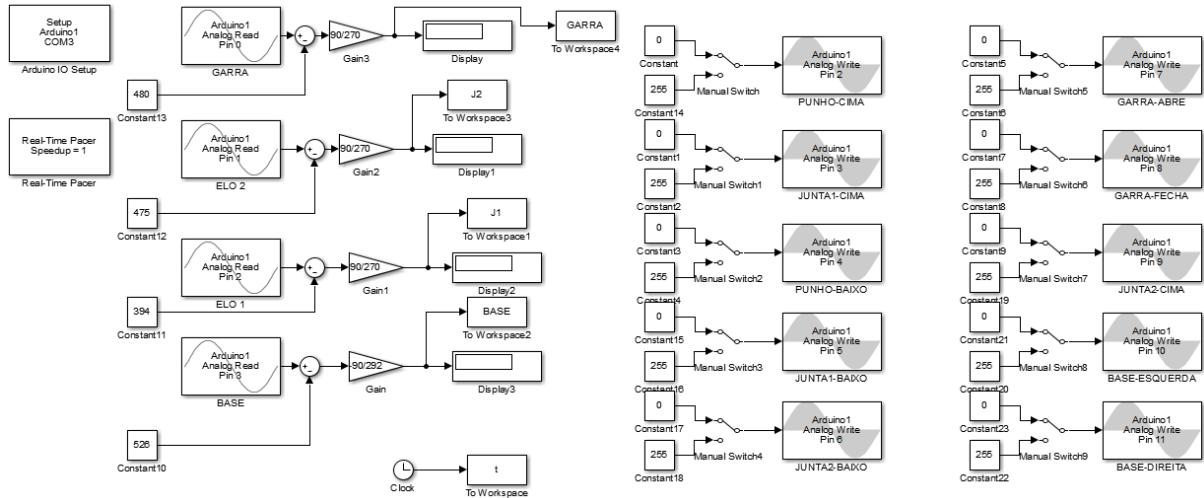
3 Modelagem

Com base na teoria de Espaço de Estados e no conhecimento do *software* Matlab foi feita a modelagem do braço robótico através do método da curva de reação dos atuadores de cada junta, estimando as respectivas funções de transferência e convertendo-as no modelo em Espaço de Estados.

3.1 Levantamento das curvas

Para obter as curvas dos atuadores de cada junta do manipulador foi aplicada potência máxima nestes e obtido os valores de posição angular, na unidade de graus, e de velocidade angular, na unidade de graus por segundo. A Figura 9 mostra o diagrama de blocos que foi utilizado para tal. Os blocos aqui presentes serão detalhados ao longo do trabalho.

Figura 9 – Diagrama de blocos para levantamento das curvas dos motores das juntas



Fonte: Autores.

Utilizando a ferramenta *System Identification Tool* (SIT) do Matlab, detalhada no Apêndice A, realizou-se a estimação das funções de transferência de cada junta do braço robótico.

Esta ferramenta facilita a modelagem estilo caixa preta. Nesta modelagem a resposta do sistema a uma entrada é aproximada a uma função de transferência (SJÖBERG et al., 1995).

Apesar de haver a possibilidade de estimar o modelo de cada junta diretamente em espaço de estados, escolheu-se fazê-lo primeiramente em funções de transferência pois,

somente assim foi possível abranger as duas variáveis desejadas (posição e velocidade) nas mesmas equações de estado. Nota-se ainda que os denominadores das funções de transferência de posição e de velocidade devem ser os mesmos para possibilitar a representação em espaço de estados para uma única entrada utilizando a equação da Figura 1.

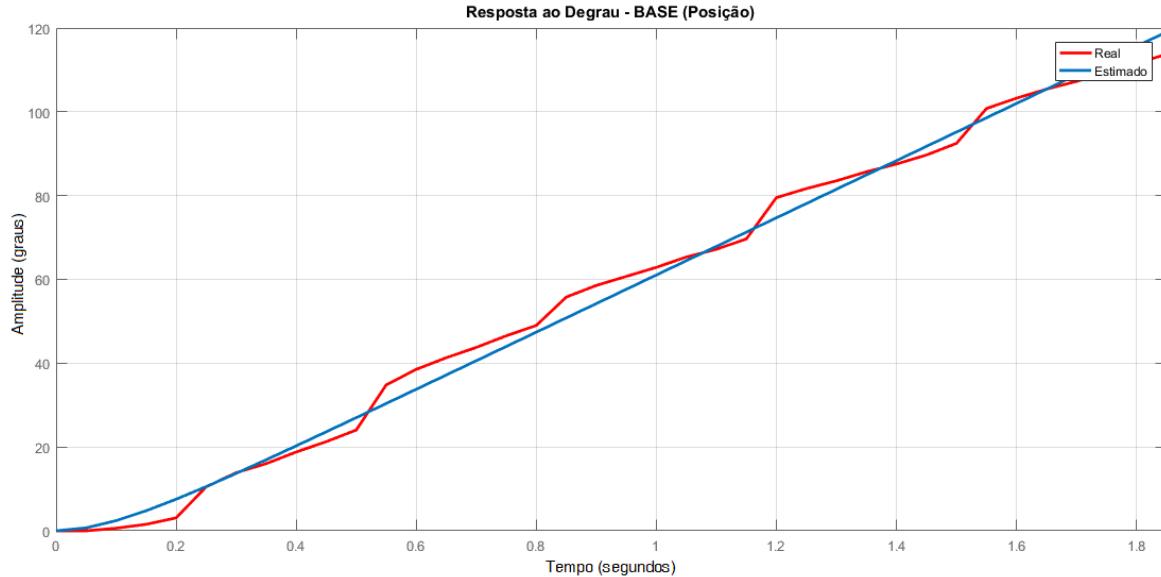
Por questões construtivas do manipulador, alguns pontos de desvio não determinísticos foram notados nos gráficos de posição das variáveis, como destacado na Figura 10. Como os dados de velocidade foram obtidos a partir da derivada da variável posição, os gráficos de velocidade apontaram picos toda vez que aconteciam estes desvios, como em destaque na Figura 12. A estimativa das curvas de velocidade desconsiderou estes picos.

A seguir pode-se conferir os gráficos das curvas obtidas pelos sensores e as curvas estimadas pelo SIT das juntas do manipulador.

- Base:

Para a posição da base as seguintes curvas foram obtidas:

Figura 10 – Curvas Real (Vermelho) e Estimada (Azul) da posição da base



Fonte: Autores.

Com base nos dados do gráfico da Figura 10, foi estimada a função de transferência da posição da base em relação à potência aplicada ao motor mostrada na Figura 11.

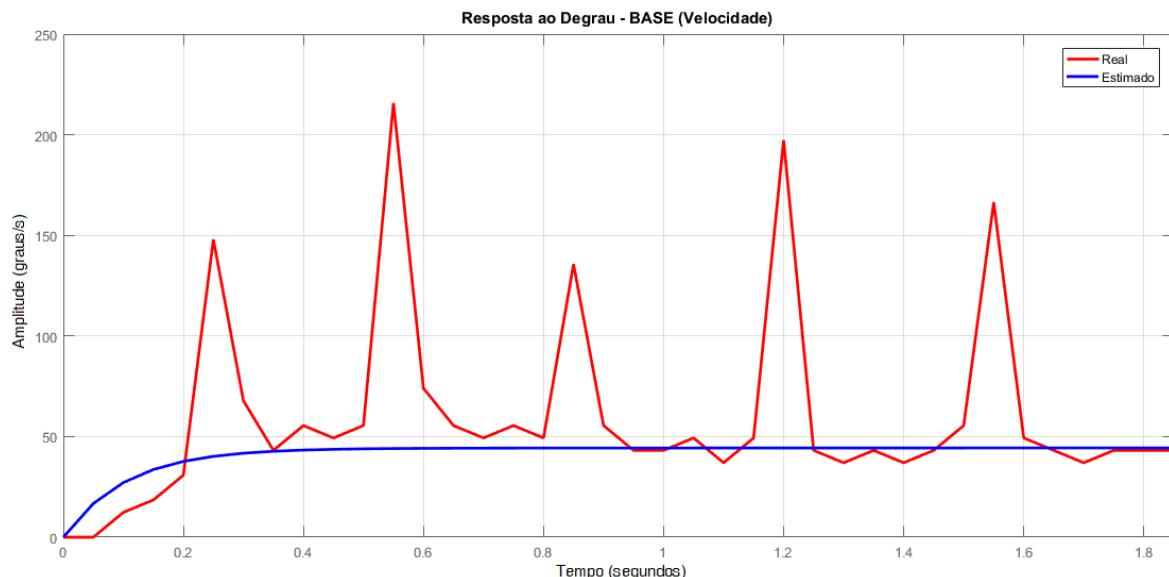
Figura 11 – Função de transferência estimada da posição da base

$$G_p(s) = \frac{2,541}{s^2 + 9,496s}$$

Fonte: Autores.

No mesmo experimento obteve-se a curva da velocidade da base do manipulador, que pode ser verificada na Figura 12. Sua função de transferência estimada para este movimento se encontra na Figura 13.

Figura 12 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da base



Fonte: Autores.

Figura 13 – Função de transferência estimada da velocidade da base

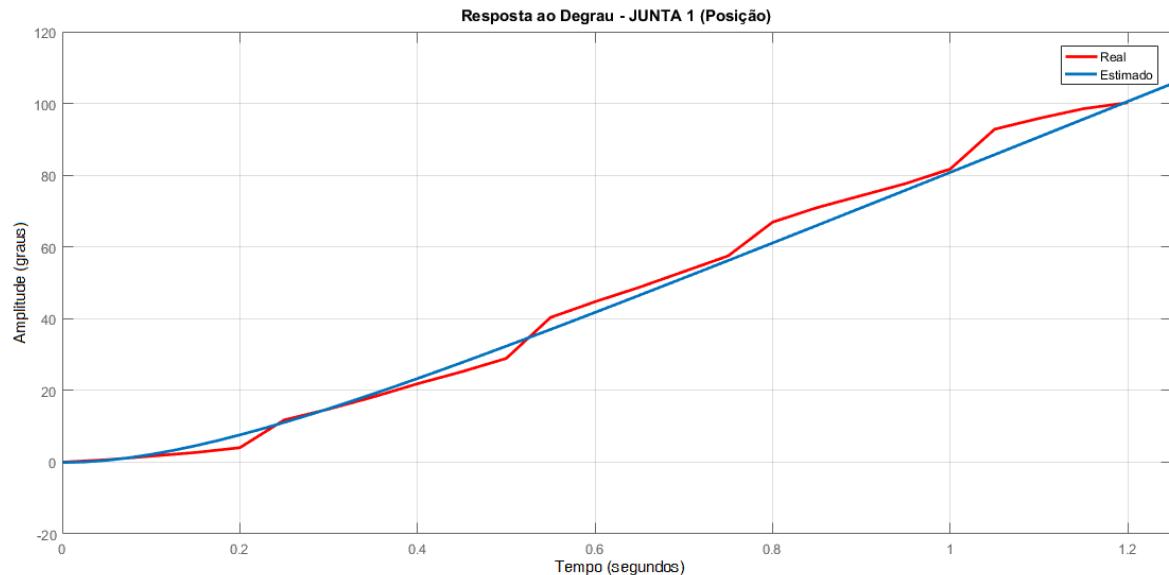
$$G_v(s) = \frac{1,649s + 0,001649}{s^2 + 9,496s}$$

Fonte: Autores.

- 1^a Junta:

Para a posição da 1^a Junta as seguintes curvas foram obtidas:

Figura 14 – Curvas Real (Vermelho) e Estimada (Azul) da posição da 1^a junta



Fonte: Autores.

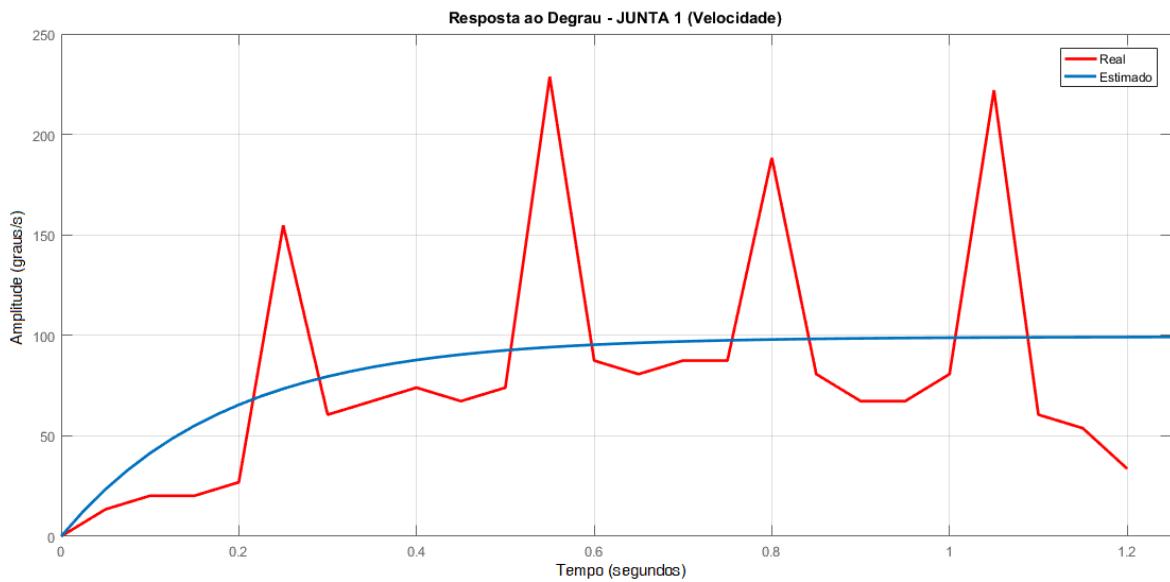
Com base nos dados do gráfico da Figura 14, foi estimada a função de transferência da posição da 1^a Junta em relação à potência aplicada ao motor mostrada na Figura 15.

Figura 15 – Função de transferência estimada da posição da 1^a junta

$$G_p(s) = \frac{-0,0004s^2 - 0,002156s + 2,097}{s^2 + 5,39s}$$

Fonte: Autores.

No mesmo experimento obteve-se a curva da velocidade da 1^a Junta do manipulador, que pode ser verificada na Figura 16. Sua função de transferência estimada para este movimento se encontra na Figura 17.

Figura 16 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da 1^a junta

Fonte: Autores.

Figura 17 – Função de transferência estimada da velocidade da 1^a junta

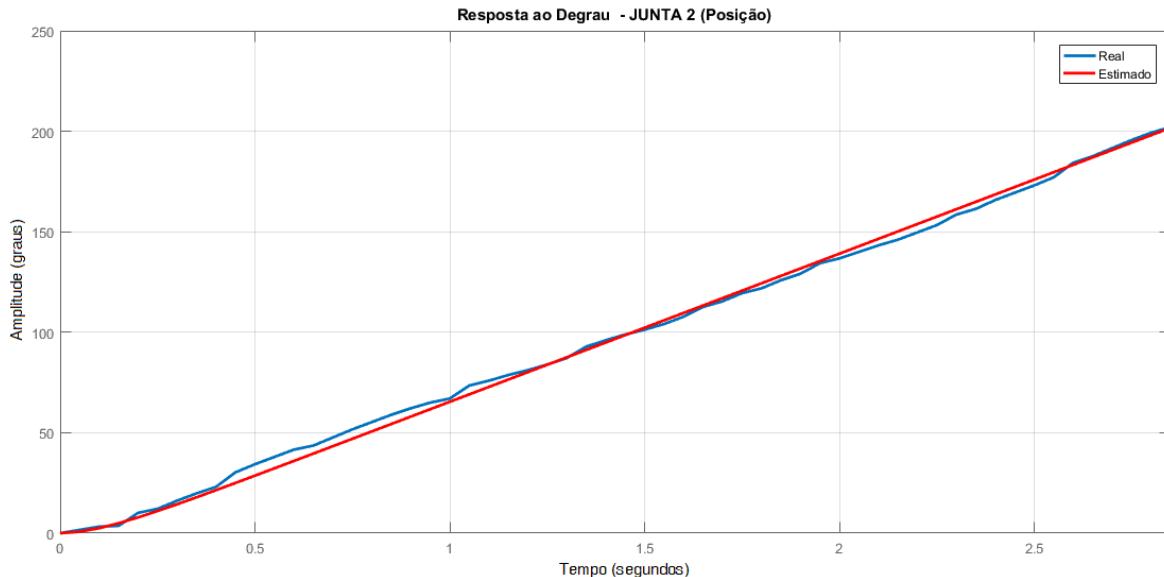
$$Gv(s) = \frac{2,097s + 0,002097}{s^2 + 5,39s}$$

Fonte: Autores.

- 2^a Junta:

Para a posição da 2^a Junta as seguintes curvas foram obtidas:

Figura 18 – Curvas Real (Vermelho) e Estimada (Azul) da posição da 2^a junta



Fonte: Autores.

Com base nos dados do gráfico da Figura 18, foi estimada a função de transferência da posição da 2^a Junta em relação à potência aplicada ao motor mostrada na Figura 19.

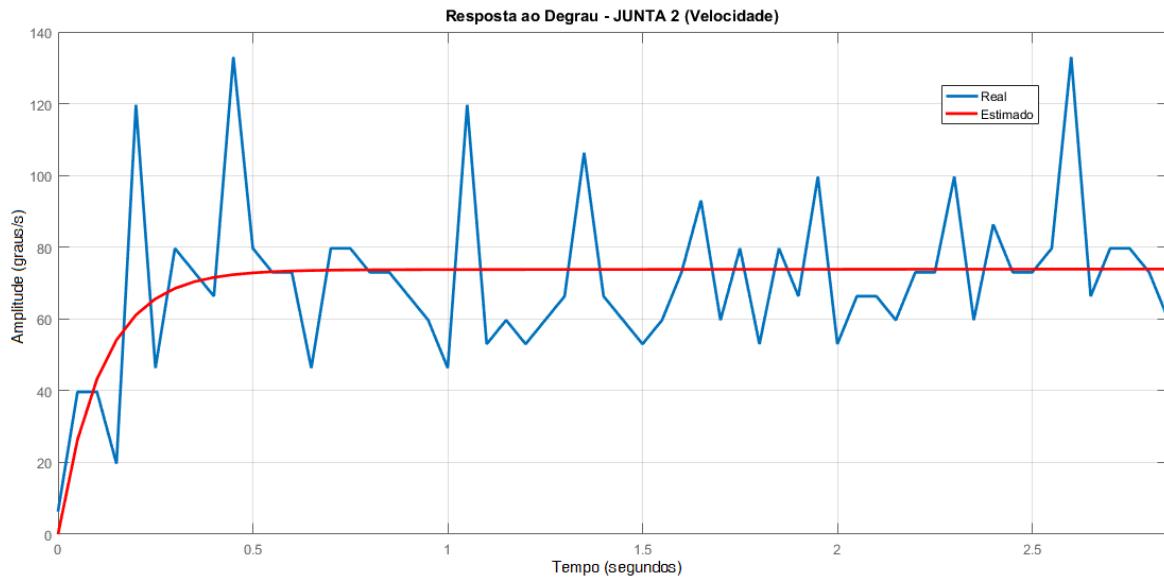
Figura 19 – Função de transferência estimada da posição da 2^a junta

$$G_p(s) = \frac{0,0001s^2 + 0,00092s + 2,555}{s^2 + 8,84s}$$

Fonte: Autores.

No mesmo experimento obteve-se a curva da velocidade da 2^a Junta do manipulador, que pode ser verificada na Figura 20. Sua função de transferência estimada para este movimento se encontra na Figura 21.

Figura 20 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade da 2^a junta



Fonte: Autores.

Figura 21 – Função de transferência estimada da velocidade da 2^a junta

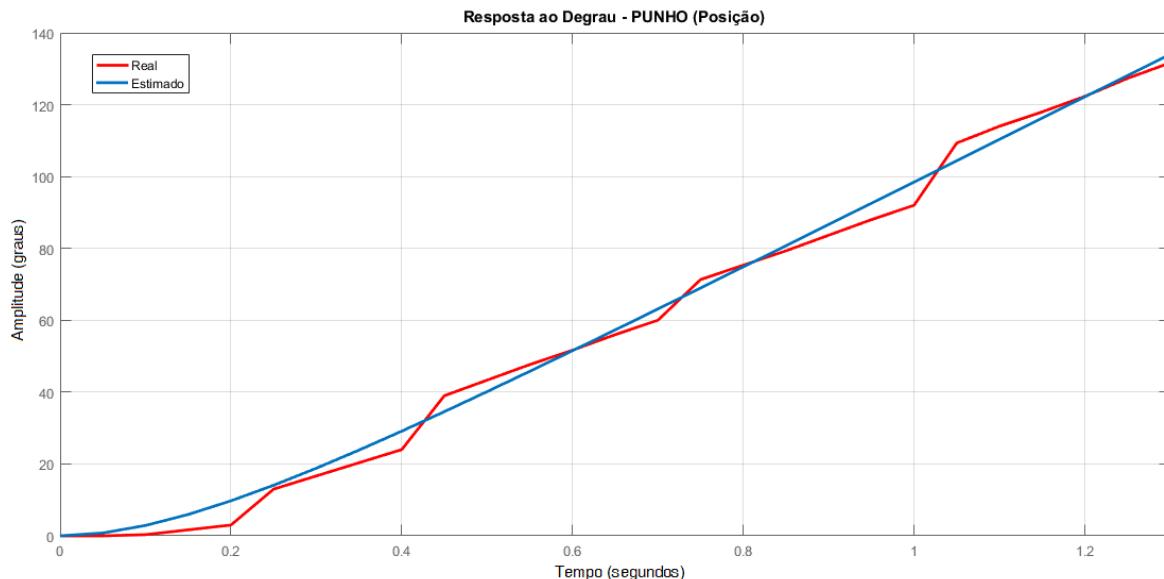
$$Gv(s) = \frac{2,555s + 0,002555}{s^2 + 8,84s}$$

Fonte: Autores.

- Punho:

Para a posição do Punho as seguintes curvas foram obtidas:

Figura 22 – Curvas Real (Vermelho) e Estimada (Azul) da posição do punho



Fonte: Autores.

Com base nos dados do gráfico da Figura 22, foi estimada a função de transferência da posição do Punho em relação à potência aplicada ao motor mostrada na Figura 23.

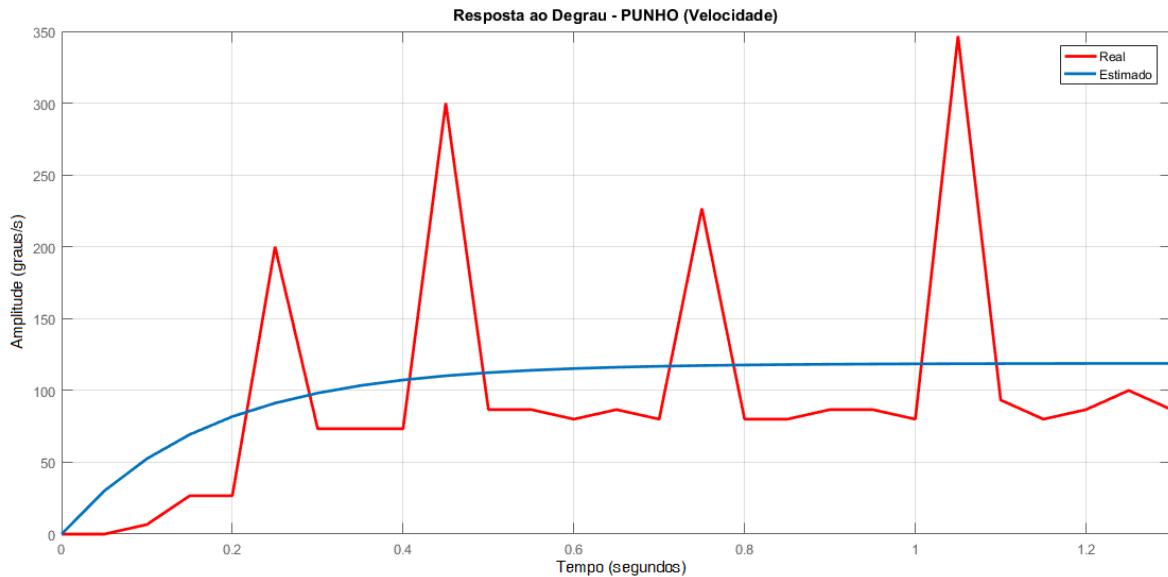
Figura 23 – Função de transferência estimada da posição do punho

$$G_p(s) = \frac{2,719}{s^2 + 5,838s}$$

Fonte: Autores.

No mesmo experimento obteve-se a curva da velocidade do Punho do manipulador, que pode ser verificada na Figura 24. Sua função de transferência estimada para este movimento se encontra na Figura 25.

Figura 24 – Curvas Real (Vermelho) e Estimada (Azul) da velocidade do punho



Fonte: Autores.

Figura 25 – Função de transferência estimada da velocidade do punho

$$Gv(s) = \frac{2,719s + 0,002719}{s^2 + 5,838s}$$

Fonte: Autores.

3.2 Modelo em Espaço de Estados

Aplicando a teoria de espaço de estados nas funções de transferência estimadas anteriormente, representou-se o sistema através de modelos em espaço de estados como apresentado nas figuras de Figura 26 a Figura 37.

- Modelo da Base

A Figura 26 mostra as funções de transferência de posição e velocidade da Base e a composição da matriz C do modelo em espaço de estados.

Figura 26 – Componentes das matrizes (Base)

$$G_p(s) = \frac{2,541}{s^2 + 9,496s}$$

$$G_v(s) = \frac{1,649s + 0,001649}{s^2 + 9,496s}$$

$$y_{1,1} = b_2 - a_2 \cdot b_0 = 2,541$$

$$y_{2,1} = b_2 - a_2 \cdot b_0 = 0,001649$$

$$y_{1,2} = b_1 - a_1 \cdot b_0 = 0$$

$$y_{2,2} = b_1 - a_1 \cdot b_0 = 1,649$$

Fonte: Autores.

A Figura 27 exibe as matrizes da equação de estado e da equação de saída.

Figura 27 – Matrizes de Estado (Base)

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -9,496 \end{bmatrix} ; B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} ; C = \begin{bmatrix} 2,541 & 0 \\ 0,001649 & 1,649 \end{bmatrix} ; D = 0$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -9,496 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 2,541 & 0 \\ 0,001649 & 1,649 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 0U$$

Fonte: Autores.

Utilizando o método de mudança de base por meio da matriz de transformação T obtiveram-se as novas matrizes indicadas na Figura 28.

Figura 28 – Matrizes Após a Mudança de Base (Base)

$$A_n = \begin{bmatrix} -0,001 & 1,5409 \\ 0,0062 & -9,495 \end{bmatrix} ; \quad B_n = \begin{bmatrix} 0 \\ 1,649 \end{bmatrix} ; \quad C_n = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} ; \quad D_n = 0$$

Fonte: Autores.

- Modelo da 1^a Junta

A Figura 29 mostra as funções de transferência de posição e velocidade da 1^a Junta e a composição da matriz C do modelo em espaço de estados.

Figura 29 – Componentes das matrizes (1^a junta)

$$G_p(s) = \frac{-0,0004s^2 - 0,002156s + 2,097}{s^2 + 5,39s} \quad G_v(s) = \frac{2,097s + 0,002097}{s^2 + 5,39s}$$

$$y_{1,1} = b_2 - a_2 \cdot b_0 = 2,097$$

$$y_{2,1} = b_2 - a_2 \cdot b_0 = 0,002097$$

$$y_{1,2} = b_1 - a_1 \cdot b_0 = 0$$

$$y_{2,2} = b_1 - a_1 \cdot b_0 = 2,097$$

Fonte: Autores.

A Figura 30 exibe as matrizes da equação de estado e da equação de saída.

Figura 30 – Matrizes de Estado (1^a junta)

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -5,39 \end{bmatrix} ; \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} ; \quad C = \begin{bmatrix} 2,097 & 0 \\ 0,002097 & 2,097 \end{bmatrix} ; \quad D = 0$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -5,39 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 2,097 & 0 \\ 0,002097 & 2,097 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 0U$$

Fonte: Autores.

Utilizando o método de mudança de base por meio da matriz de transformação T obtiveram-se as novas matrizes indicadas na Figura 31.

Figura 31 – Matrizes Após a Mudança de Base (1^a junta)

$$A_n = \begin{bmatrix} -0,001 & 1 \\ 0,0054 & -5,389 \end{bmatrix}; \quad B_n = \begin{bmatrix} 0 \\ 2,097 \end{bmatrix}; \quad C_n = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad D_n = 0$$

Fonte: Autores.

- Modelo da 2^a Junta

A Figura 32 mostra as funções de transferência de posição e velocidade da 2^a Junta e a composição da matriz C do modelo em espaço de estados.

Figura 32 – Componentes das matrizes (2^a junta)

$$G_p(s) = \frac{0,00001s^2 + 0,00092s + 2,555}{s^2 + 8,84s}$$

$$G_v(s) = \frac{2,555s + 0,002555}{s^2 + 8,84s}$$

$$y_{1,1} = b_2 - a_2 \cdot b_0 = 2,555$$

$$y_{2,1} = b_2 - a_2 \cdot b_0 = 0,002555$$

$$y_{1,2} = b_1 - a_1 \cdot b_0 = 3,6 \times 10^{-5}$$

$$y_{2,2} = b_1 - a_1 \cdot b_0 = 2,555$$

Fonte: Autores.

A Figura 33 exibe as matrizes da equação de estado e da equação de saída.

Figura 33 – Matrizes de Estado (2^a junta)

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -8,84 \end{bmatrix}; \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad C = \begin{bmatrix} 2,555 & 3,6 \times 10^{-5} \\ 0,002555 & 2,555 \end{bmatrix}; \quad D = 0$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -8,84 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 2,555 & 3,6 \times 10^{-5} \\ 0,002555 & 2,555 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 0U$$

Fonte: Autores.

Utilizando o método de mudança de base por meio da matriz de transformação T obtiveram-se as novas matrizes indicadas na Figura 34.

Figura 34 – Matrizes Após a Mudança de Base (2^a junta)

$$A_n = \begin{bmatrix} -0,001 & 0,9999 \\ 0,0088 & -8,839 \end{bmatrix} ; \quad B_n = \begin{bmatrix} 0 \\ 2,555 \end{bmatrix} ; \quad C_n = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} ; \quad D_n = 0$$

Fonte: Autores.

- Modelo do Punho

A Figura 35 mostra as funções de transferência de posição e velocidade do Punho e a composição da matriz C do modelo em espaço de estados.

Figura 35 – Componentes das matrizes (Punho)

$$G_p(s) = \frac{2,719}{s^2 + 5,838s}$$

$$y_{1,1} = b_2 - a_2 \cdot b_0 = 2,719$$

$$y_{2,1} = b_2 - a_2 \cdot b_0 = 0,002719$$

$$y_{1,2} = b_1 - a_1 \cdot b_0 = 0$$

$$y_{2,2} = b_1 - a_1 \cdot b_0 = 2,719$$

Fonte: Autores.

A Figura 36 exibe as matrizes da equação de estado e da equação de saída.

Figura 36 – Matrizes de Estado (Punho)

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -5,838 \end{bmatrix} ; \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} ; \quad C = \begin{bmatrix} 2,719 & 0 \\ 0,002719 & 2,719 \end{bmatrix} ; \quad D = 0$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -5,838 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 2,719 & 0 \\ 0,002719 & 2,719 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 0U$$

Fonte: Autores.

Utilizando o método de mudança de base por meio da matriz de transformação T obtiveram-se as novas matrizes indicadas na Figura 37.

Figura 37 – Matrizes Após a Mudança de Base (Punho)

$$A_n = \begin{bmatrix} -0,001 & 1 \\ 0,0058 & -5,837 \end{bmatrix}; \quad B_n = \begin{bmatrix} 0 \\ 2,719 \end{bmatrix}; \quad C_n = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad D_n = 0$$

Fonte: Autores.

4 Controle

Com base no modelo em Espaço de estados foi projetado um vetor de controle K para controlar posição e velocidade de cada junta. Como o controle ótimo não era o objetivo deste trabalho selecionou-se ganhos de forma empírica para os controladores com o intuito de deixar as respostas das juntas satisfatórias para a comparação dos dois métodos de controle.

Os parâmetros de controle foram ajustados em simulações com o controle em HIL, com $T_s=0.05s$.

Os dois algoritmos de controle aplicados foram exatamente iguais, porém no controle em HIL foi acrescentado o algoritmo de leitura do Arduino UNO ao algoritmo simulado no Simulink.

O computador utilizado nas simulações foi um notebook Acer® modelo Aspire M5-481T com processador Intel® Core™ i5-3317U e a versão do *software* Matlab® foi a R2016a (9.0) 64-bit.

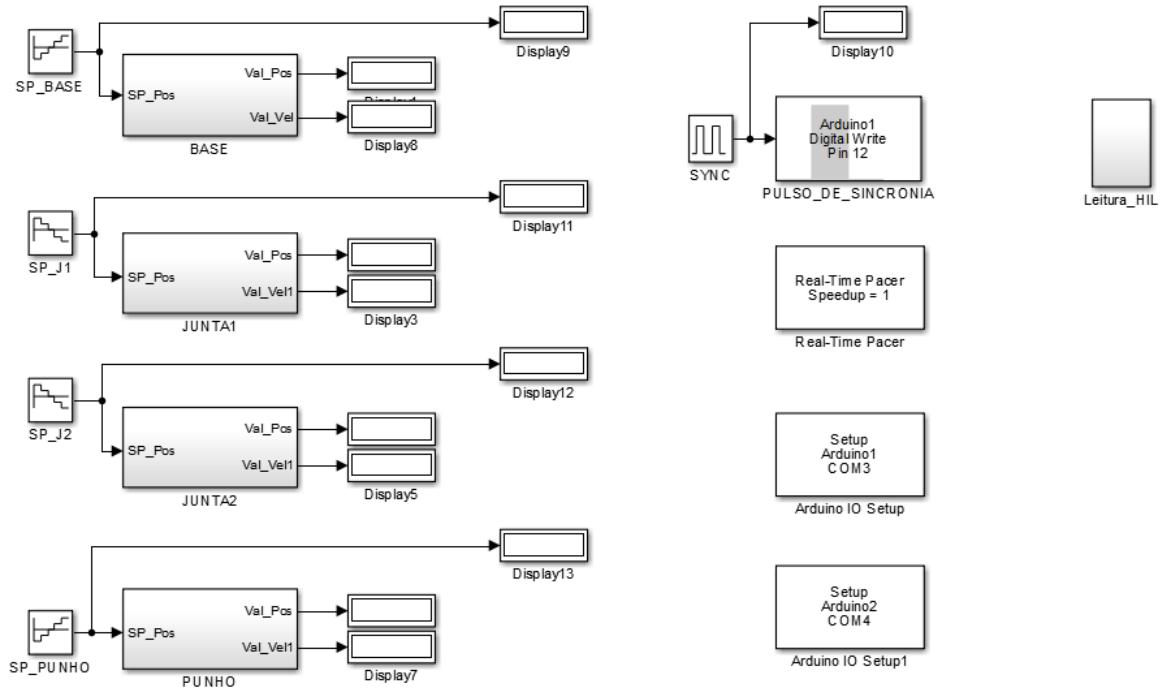
4.1 Controle em HIL

O controle em HIL foi feito com o software Matlab Simulink rodando no Computador Pessoal (PC) e foi utilizado um pacote de blocos de interface de aquisição de dados Simulink-Arduino (entradas e saídas) encontrado na internet (CAMPA, 2016). Estes blocos podem ser visualizados na Figura 38.

Os blocos de nome *ArduinoIO Setup* e *ArduinoIO Setup1* da Figura 38 são parte deste pacote e têm a função de configurar os parâmetros de comunicação para o Matlab fazer leitura ou escrita na placa microcontrolada Arduino através da porta serial. Neste bloco configura-se a porta COM na qual a placa está conectada ao PC.

O bloco de nome *Real-Time Pacer* da Figura 38 também é parte integrante deste pacote e tem a função de simular o algoritmo a um tempo igual ao tempo base do PC.

Figura 38 – Diagrama de blocos HIL

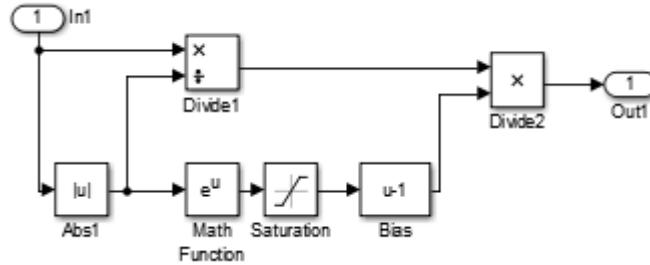


Fonte: Autores.

Utilizamos sequências pré-programadas como referência para o controle de posição. Estas sequências estão representadas pelos blocos de nome *SP_BASE*, *SP_J1*, *SP_J2* e *SP_PUNHO* na Figura 38.

Para um melhor desempenho da velocidade das juntas utilizou-se uma função exponencial relacionada ao erro de posição como referência para a velocidade. Deste modo o SP de velocidade seria maior quando distante da posição desejada, e tenderia a zero na posição ótima através de uma relação não linear. Esta configuração está explicitada no diagrama da Figura 39 e pode ser detalhada como: o erro de posição entra no diagrama pelo bloco *In1*; os blocos *Abs1*, *Divide1* e *Divide2* têm a função de inverter o sentido do SP de acordo com o sentido do erro em relação à referência de posição; o bloco *Math Function* relaciona o erro através de uma função exponencial (e^u , em que u é o sinal de entrada do bloco); os blocos *Saturation* e *Bias* foram colocados para solucionar um erro de *offset* da função exponencial; finalmente, o bloco *Out1* envia o sinal calculado como SP do controle de velocidade.

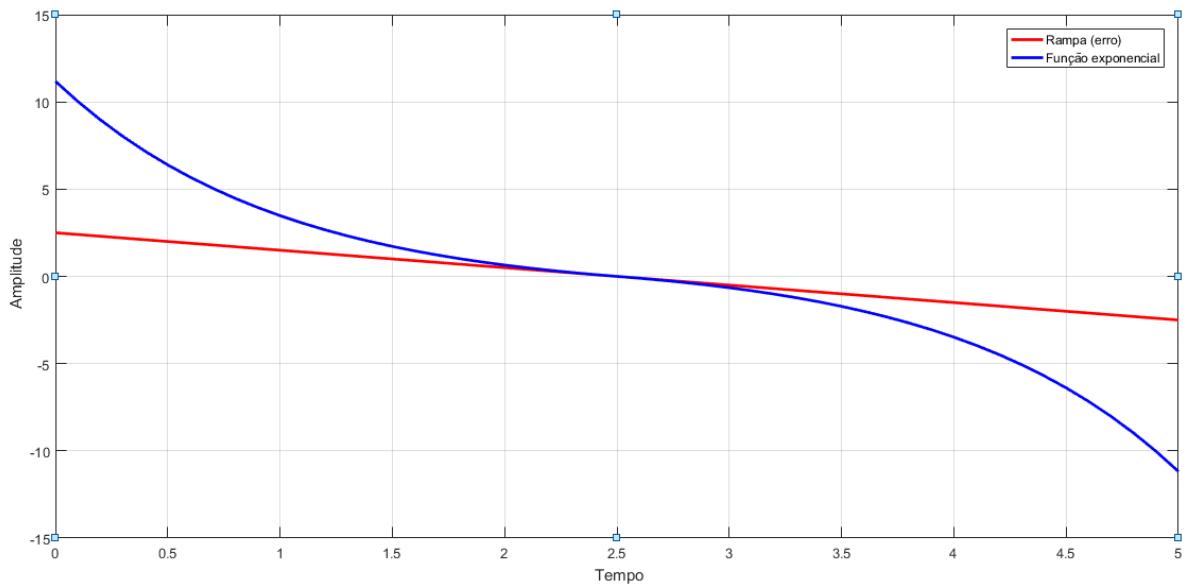
Figura 39 – Setpoint de velocidade



Fonte: Autores.

A Figura 40 apresenta a resposta de um exemplo de utilização da função exponencial supracitada. O gráfico em vermelho simula um sinal de erro em rampa de $-2,5$ a $+2,5$ enquanto o gráfico em azul mostra a resposta da função a esta entrada.

Figura 40 – Função exponencial do SP de velocidade



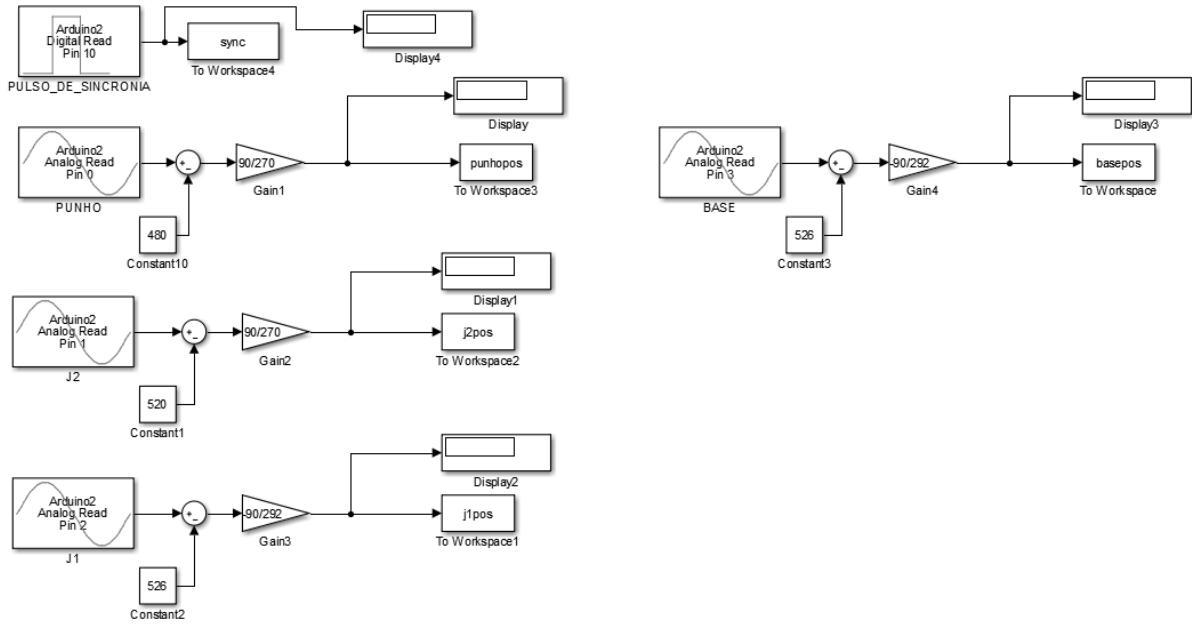
Fonte: Autores.

Para ler os dados dos sensores das juntas foi utilizado uma segunda placa Arduino (UNO) conectado às saídas destes. O algoritmo de leitura capturou os dados lidos por esta placa e os enviou para o espaço de trabalho (*Workspace*) do Matlab através do bloco *To Workspace*. Os blocos de leitura podem ser vistos na Figura 41. O esquema, que se repete para todas as juntas, possui um bloco de leitura analógica da placa Arduino (nomeados como *BASE*, *J1*, *J2* e *PUNHO*), um bloco somador aliado a uma constante (valor de *bias*) e a um ganho para calibrar os valores na unidade angular de graus, além de um *display* para visualização.

Um pulso de sincronia, ainda, foi enviado do Arduino MEGA para o Arduino UNO a fim de sincronizar os dados gráficos das diferentes leituras. Na Figura 38 o bloco

de escrita digital *PULSO_DE_SINCRONIA* e o bloco de sequência repetitiva *SYNC* enviam o sinal a partir da placa Arduino MEGA. Na Figura 41 o bloco de leitura digital *PULSO_DE_SINCRONIA* e seu bloco anexo *To Workspace4* fazem leitura na placa Arduino UNO.

Figura 41 – Diagrama de blocos de leitura dos sensores HIL



Fonte: Autores.

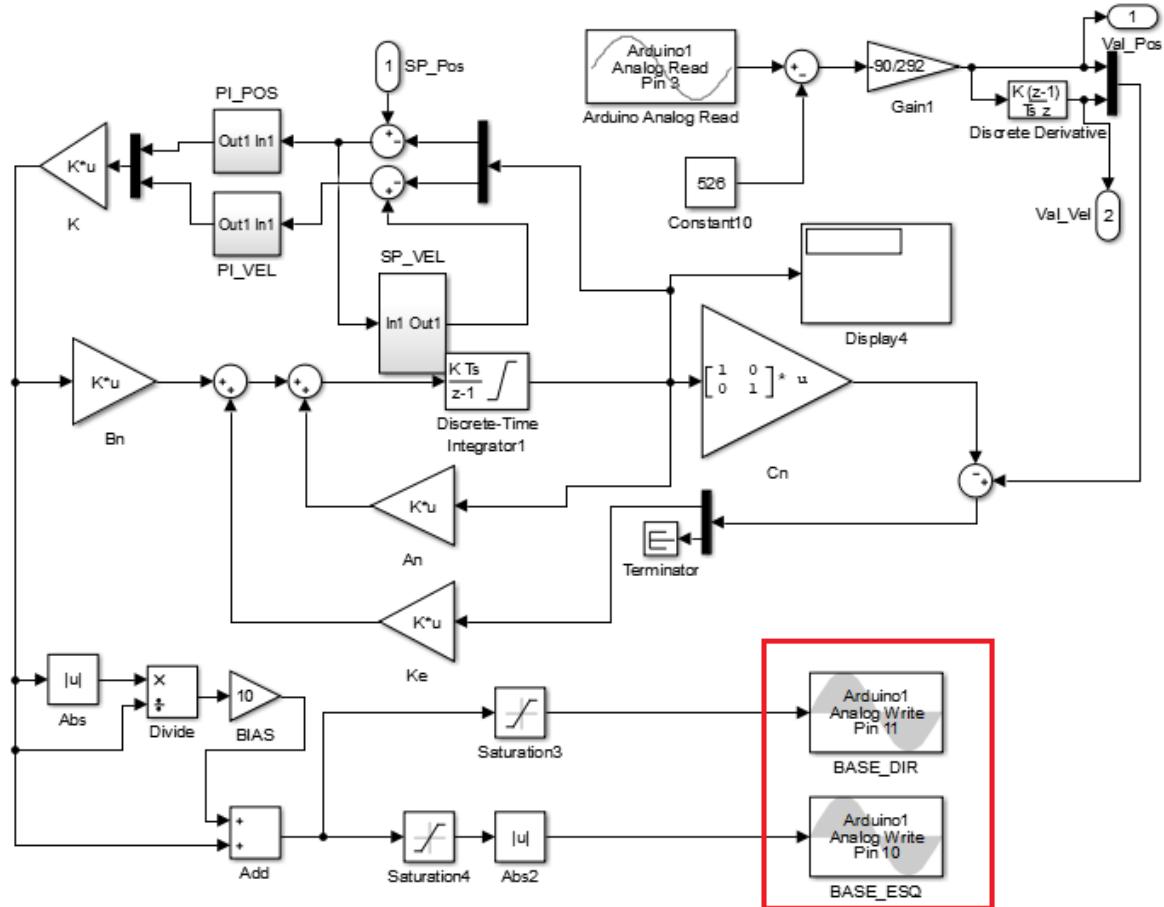
A seguir têm-se os subsistemas de cada junta de forma expandida, exibindo o diagrama de controle utilizado. Nota-se que na entrada dos blocos do vetor do observador de estados (K_e) existe uma divisão de sinais com um terminador. Isto se dá porque o observador de estado trabalha somente com o erro da variável posição descrito como a diferença entre a posição real lida no sensor e a posição estimada pelo observador.

- Base:

A Figura 42 apresenta o diagrama de blocos da Base com o modelo em Espaço de estados, com o vetor de ganho K e o vetor do observador de estados K_e já parametrizados, e o valor de Bias já calibrado. Estão destacados na cor vermelha os blocos de escrita analógica que enviam os sinais para os atuadores. Estes blocos enviam sinais de saída PWM por valores de 0 (valor mínimo de saída) a 255 (valor máximo de saída). Os dois blocos estão relacionados de forma que, quando um deles envia valor 0 e o outro envia valor maior que zero, o motor gira em um sentido, na potência proporcional ao valor maior que zero. Quando invertidos os valores nos blocos o motor gira no sentido inverso.

$K = [10 \ 0];$
 $Ke = [30; 10];$
Bias=10;

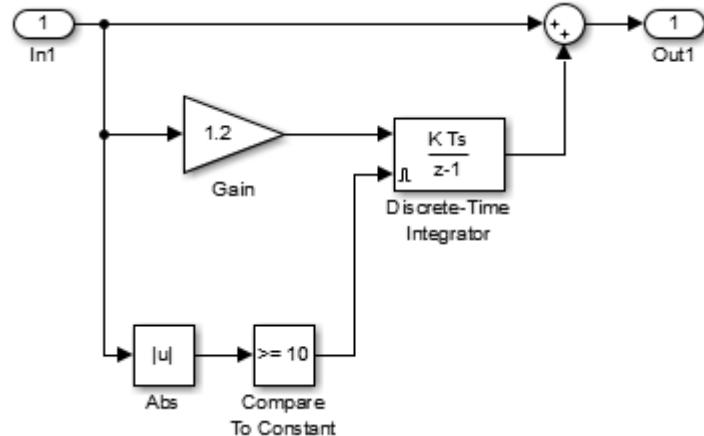
Figura 42 – Diagrama de blocos da BASE - HIL



Fonte: Autores.

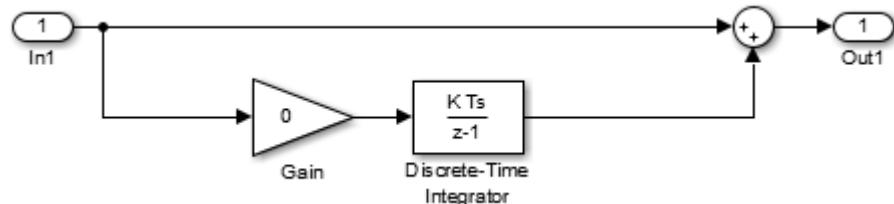
Além do vetor de ganho K foram utilizados integradores para compor o controlador a fim de melhorar o desempenho do controle de posição e de velocidade. O integrador do controle de posição, visualizado na Figura 43, só atuava quando o erro de posição fosse menor que 10 graus, com o objetivo de reduzir o erro em regime permanente. Na base nota-se que não foi necessário ganho de correção da variável velocidade, pois a resposta do sistema já se mostrava satisfatória. Também não se fez necessário o uso de integrador na velocidade, como mostrado na Figura 44.

Figura 43 – Integrador da posição da Base



Fonte: Autores.

Figura 44 – Integrador da velocidade da Base



Fonte: Autores.

Após a inclusão destes integradores, o novo vetor de controle K resultou como apresentado na Figura 45.

Figura 45 – Vetor de controle K após adição do Integrador - Base

$$K = \begin{bmatrix} 10 + \frac{1}{1,2s} & 0 \end{bmatrix}$$

Fonte: Autores.

As funções de transferência da base com suas variáveis compensadas pelo novo vetor de controle são mostradas na Figura 46, em relação à variável posição, e na Figura 47 em relação à variável velocidade.

Figura 46 – Função de Transferência da posição da Base compensada

$$G_p(s) = \frac{2,541}{s^2 + 11,6135s + 25,4094}$$

Fonte: Autores.

Figura 47 – Função de Transferência da velocidade da Base compensada

$$G_v(s) = \frac{1,649s + 0,001649}{s^2 + 11,6135s + 25,4094}$$

Fonte: Autores.

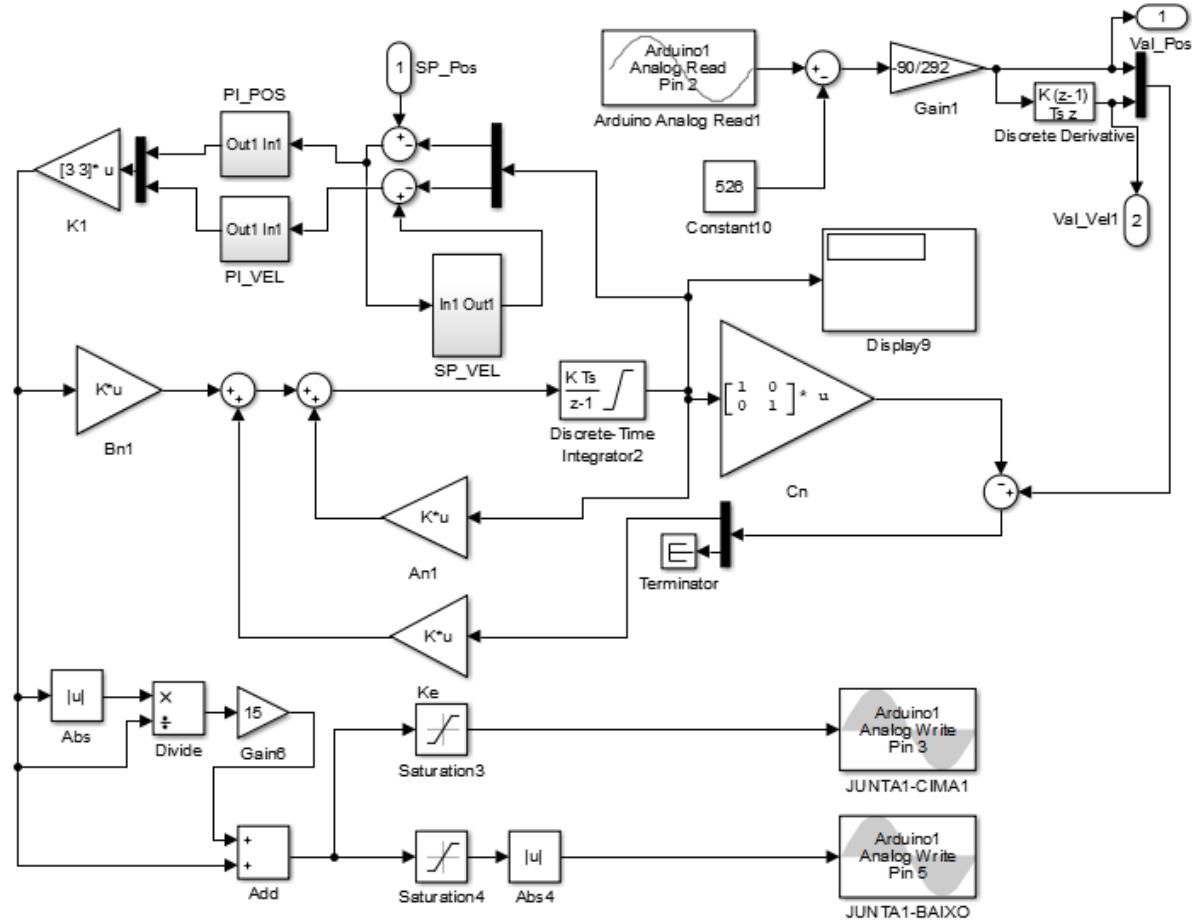
- 1^a Junta:

A Figura 48 apresenta o diagrama de blocos da 1^a Junta com o modelo em Espaço de estados, com o vetor de ganho K e o vetor do observador de estados Ke já parametrizados, e o valor de Bias já calibrado.

K1=[3 3];

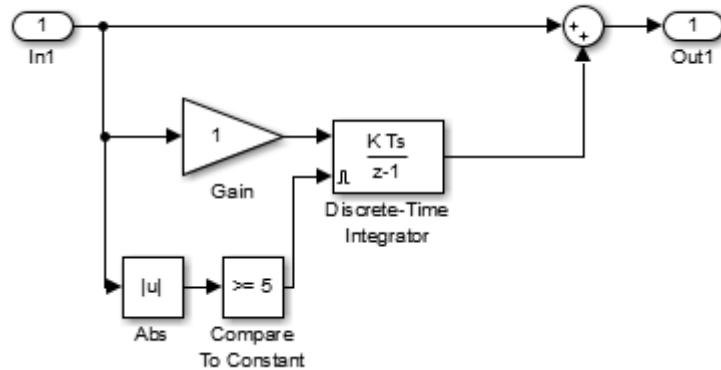
Ke=[20;10];

Bias=15;

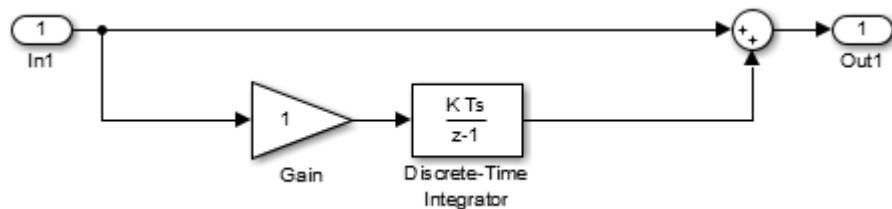
Figura 48 – Diagrama de blocos da 1^a Junta - HIL

Fonte: Autores.

O integrador do controle de posição, visualizado na Figura 49, só atuava quando o erro de posição fosse menor que 5 graus, com o objetivo de reduzir o erro em regime permanente. Na 1^a Junta fez-se necessário o uso de integrador na velocidade, como mostrado na Figura 50.

Figura 49 – Integrador da posição da 1^a Junta

Fonte: Autores.

Figura 50 – Integrador da velocidade da 1^a Junta

Fonte: Autores.

O vetor de controle K está apresentado na Figura 51.

Figura 51 – Vetor de controle K após adição do Integrador - 1^a Junta

$$K = \begin{bmatrix} 3 + \frac{1}{s} & 3 + \frac{1}{s} \end{bmatrix}$$

Fonte: Autores.

As funções de transferência da 1^a Junta com suas variáveis compensadas pelo novo vetor de controle são mostradas na Figura 52, em relação à variável posição, e na Figura 53 em relação à variável velocidade.

Figura 52 – Função de Transferência da posição da 1^a Junta compensada

$$G_p(s) = \frac{-0,0004s^3 - 0,002156s^2 + 2,097s}{s^3 + 11,681s^2 + 8,3943s + 2,0991}$$

Fonte: Autores.

Figura 53 – Função de Transferência da velocidade da 1^a Junta compensada

$$G_v(s) = \frac{-2,097s^2 - 0,002097s}{s^3 + 11,681s^2 + 8,3943s + 2,0991}$$

Fonte: Autores.

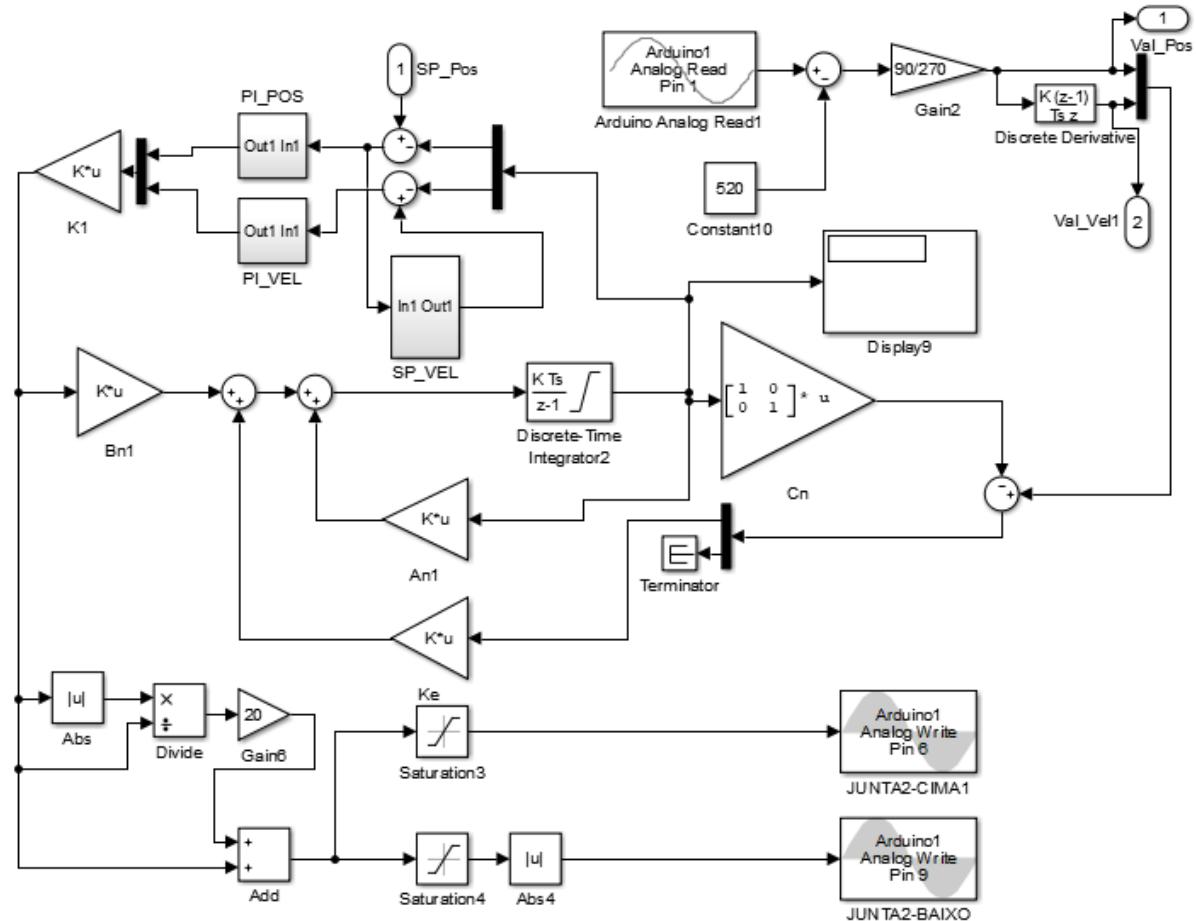
- 2^a Junta:

A Figura 54 apresenta o diagrama de blocos da 2^a Junta com o modelo em Espaço de estados, com o vetor de ganho K e o vetor do observador de estados Ke já parametrizados, e o valor de Bias já calibrado.

K1=[7.5 5];

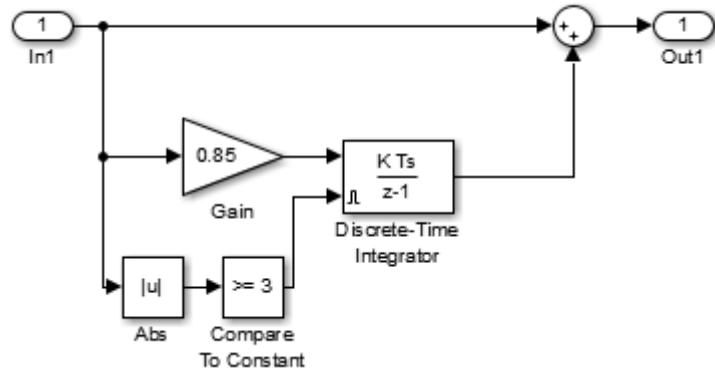
Ke=[30;10];

Bias=20;

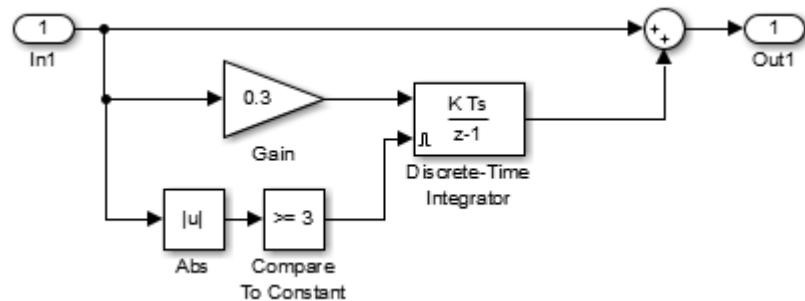
Figura 54 – Diagrama de blocos da 2^a Junta - HIL

Fonte: Autores.

O integrador do controle de posição, visualizado na Figura 55, só atuava quando o erro de posição fosse menor que 3 graus, com o objetivo de reduzir o erro em regime permanente. Na 1^a Junta fez-se necessário o uso de integrador na velocidade somente atuando quando o erro fosse menor que 3 graus, como mostrado na Figura 56.

Figura 55 – Integrador da posição da 2^a Junta

Fonte: Autores.

Figura 56 – Integrador da velocidade da 2^a Junta

Fonte: Autores.

O vetor de controle K está apresentado na Figura 57.

Figura 57 – Vetor de controle K após adição do Integrador - 2^a Junta

$$K = \begin{bmatrix} 7,5 + \frac{1}{0,85s} & 5 + \frac{1}{0,3s} \end{bmatrix}$$

Fonte: Autores.

As funções de transferência da 2^a Junta com suas variáveis compensadas pelo novo vetor de controle são mostradas na Figura 58, em relação à variável posição, e na Figura 59 em relação à variável velocidade.

Figura 58 – Função de Transferência da posição da 2^a Junta compensada

$$G_p(s) = \frac{0,0001s^3 + 0,00092s^2 + 2,555s}{s^3 + 21,6150s^2 + 27,6901s + 3,0141}$$

Fonte: Autores.

Figura 59 – Função de Transferência da velocidade da 2^a Junta compensada

$$G_v(s) = \frac{2,555s^2 + 0,002555s}{s^3 + 21,6150s^2 + 27,6901s + 3,0141}$$

Fonte: Autores.

- Punho:

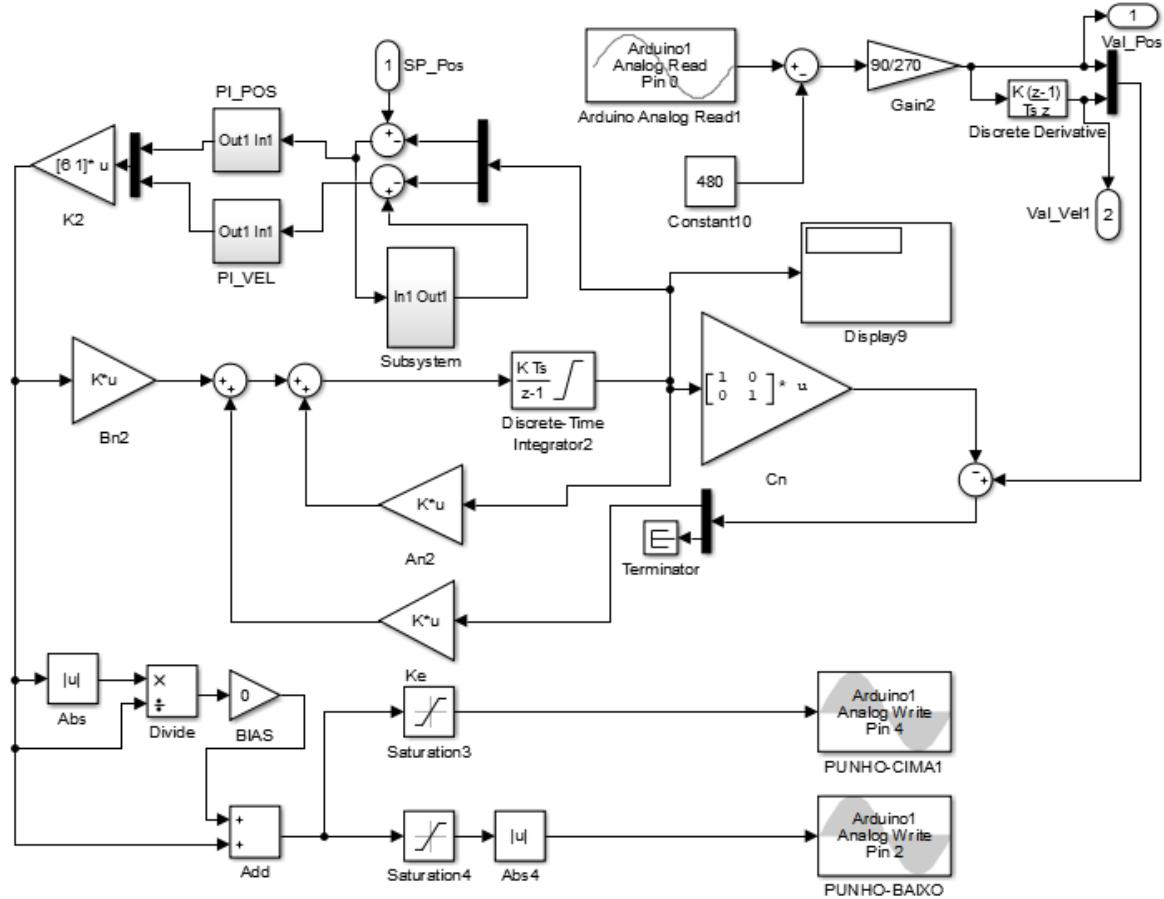
A Figura 60 apresenta o diagrama de blocos do Punho com o modelo em Espaço de estados, com o vetor de ganho K e o vetor do observador de estados Ke já parametrizados, e o valor de Bias já calibrado.

K2=[6 1];

Ke=[30;10];

Bias=0;

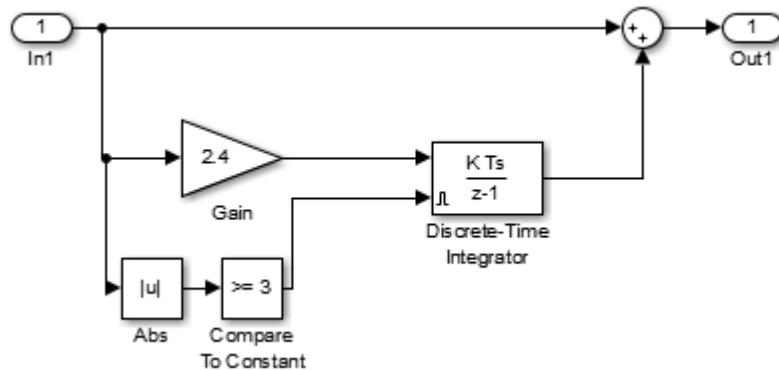
Figura 60 – Diagrama de blocos do Punho - HIL



Fonte: Autores.

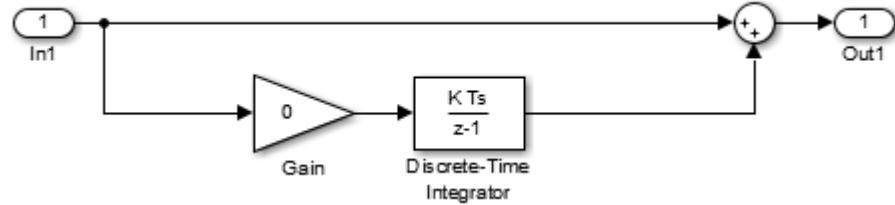
O integrador do controle de posição, visualizado na Figura 61, só atuava quando o erro de posição fosse menor que 3 graus, com o objetivo de reduzir o erro em regime permanente. Na 1^a Junta não foi necessário o uso de integrador na velocidade, como mostrado na Figura 62.

Figura 61 – Integrador da posição do Punho



Fonte: Autores.

Figura 62 – Integrador da velocidade do Punho



Fonte: Autores.

O vetor de controle K está apresentado na Figura 63.

Figura 63 – Vetor de controle K após adição do Integrador - Punho

$$K = \begin{bmatrix} 6 + \frac{1}{2,4s} & 1 \end{bmatrix}$$

Fonte: Autores.

As funções de transferência do Punho com suas variáveis compensadas pelo novo vetor de controle são mostradas na Figura 64, em relação à variável posição, e na Figura 65 em relação à variável velocidade.

Figura 64 – Função de Transferência da posição do Punho compensada

$$G_p(s) = \frac{2,719s}{s^3 + 8,557s^2 + 16,3168s + 1,1329}$$

Fonte: Autores.

Figura 65 – Função de Transferência da velocidade do Punho compensada

$$G_v(s) = \frac{2,719s^2 + 0,002719s}{s^3 + 8,557s^2 + 16,3168s + 1,1329}$$

Fonte: Autores.

4.2 Controle embarcado

No controle embarcado foi utilizado o mesmo algoritmo do controle em HIL apresentado na seção anterior, somente alterando os blocos de conexão com o Arduino para embarcar o algoritmo neste. Na Figura 67 vemos os blocos do algoritmo e o bloco *PULSO_DE_SINCRONIA* da biblioteca oficial do Arduino para Simulink obtido através do próprio software Matlab.

Utilizando a função *build* do Simulink, embarcou-se o código na placa Arduino MEGA.

O código do algoritmo ocupou 19,4% da memória de dados e 6,7% da memória de programa da placa Arduino MEGA. Esta informação foi obtida através do visualizador de diagnóstico do Simulink como pode-se conferir na Figura 66.

Figura 66 – Uso de memória da placa Arduino MEGA

```
AVR Memory Usage
-----
Device: atmega2560

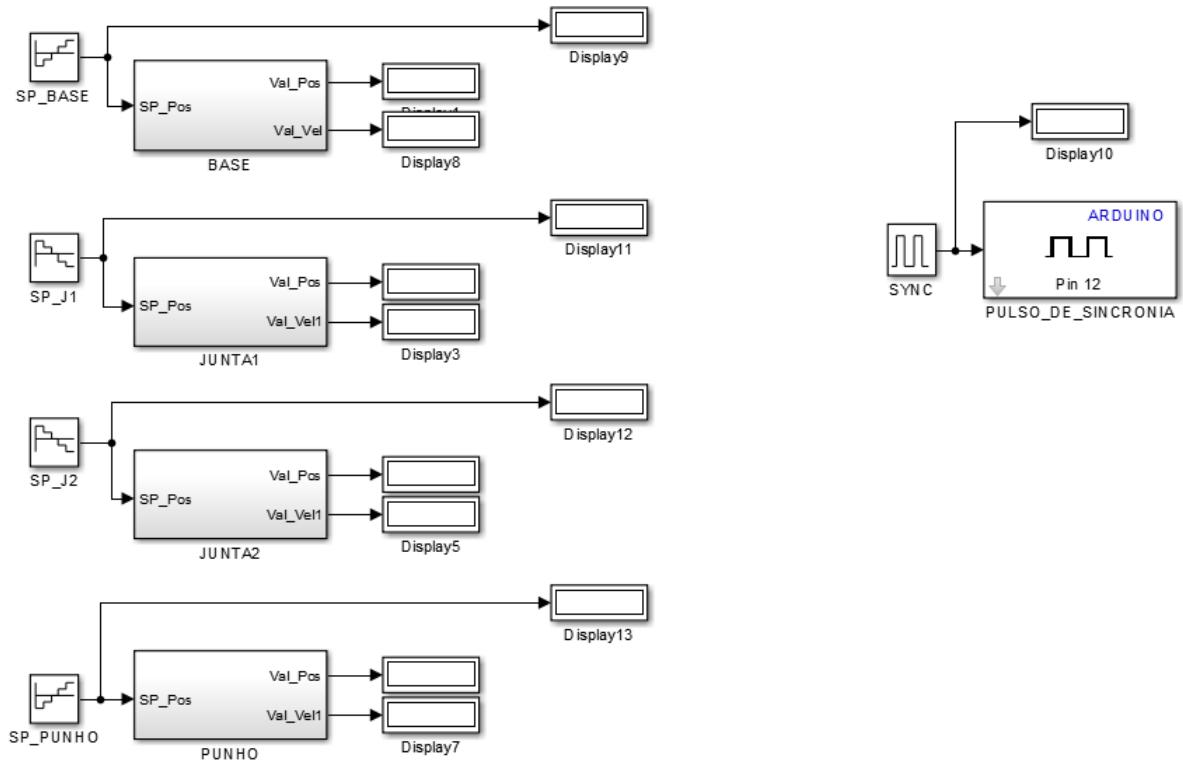
Program: 17444 bytes (6.7% Full)
(.text + .data + .bootloader)

Data:      1592 bytes (19.4% Full)
(.data + .bss + .noinit)
```

Fonte: Autores.

A Figura 67 apresenta o diagrama do algoritmo embarcado.

Figura 67 – Diagrama de blocos - Controle embarcado

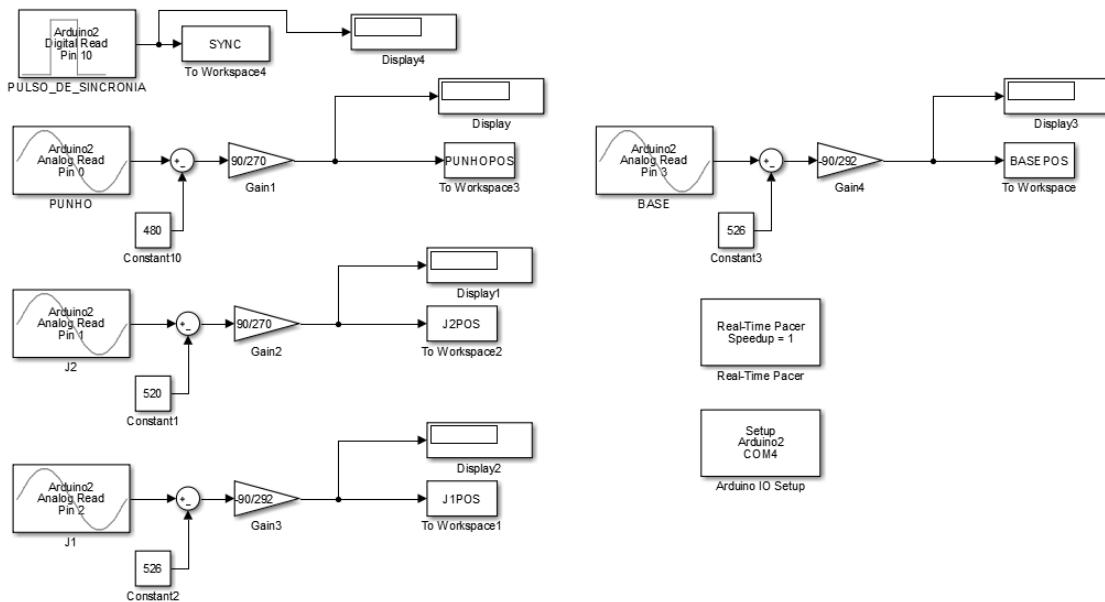


Fonte: Autores.

Também foi retirado o algoritmo de leitura, pois, como este se conectava à placa Arduino UNO, não poderíamos embarcá-lo no MEGA.

A leitura, então, foi feita de forma separada, comunicando o PC com o Arduino UNO assim como foi feito no HIL, após o controle ser embarcado no Arduino MEGA e desconectado do PC. A Figura 68 mostra os blocos de leitura utilizados com a placa Arduino UNO.

Figura 68 – Diagrama de blocos da leitura do controle embarcado



Fonte: Autores.

5 Análise

Neste capítulo os resultados das simulações baseadas nos dois tipos de controle de posição das juntas são analisados graficamente e comparados.

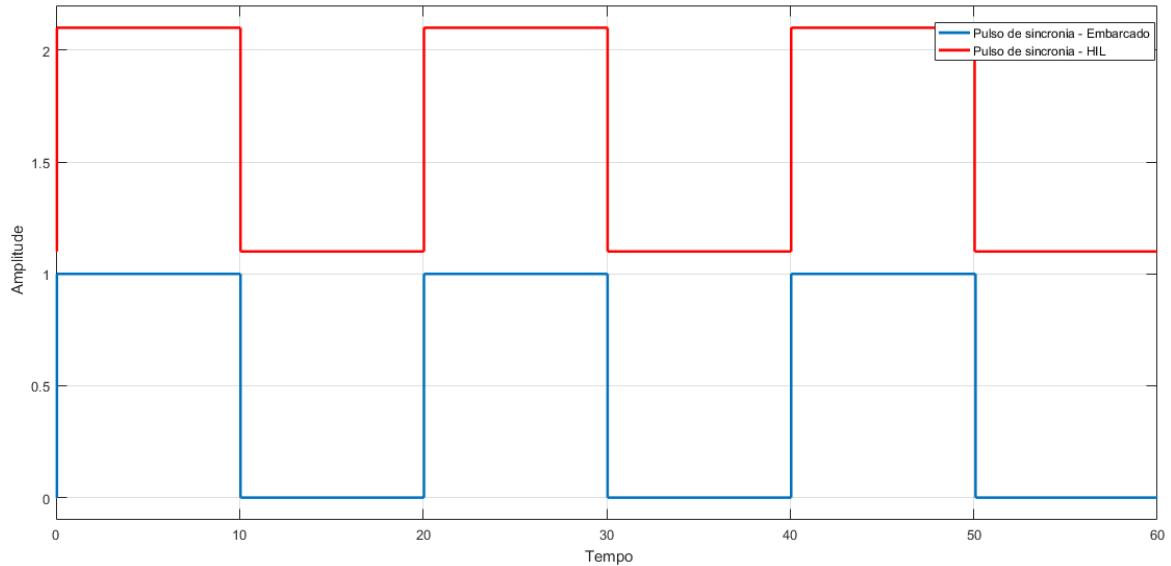
No controle em HIL, simulou-se o algoritmo nos períodos de amostragem (*Sample Times*) de 0.05s e 0.04s pois, com valores menores que estes, o sistema não respondia de forma estável, por limitações de velocidade de comunicação da placa Arduino com o PC via porta serial. Portanto a taxa de atualização de processamento máxima foi de 25Hz, visto que a frequência de amostragem é igual ao inverso do período.

No controle embarcado, por não haver a barreira da comunicação Arduino-PC, o controle foi simulado a *Sample Times* (T_s) bem menores. Porém, a medição dos dados para análise foi feita com a segunda placa Arduino (UNO) em HIL com o PC, sendo este simulado com $T_s=0.05s$.

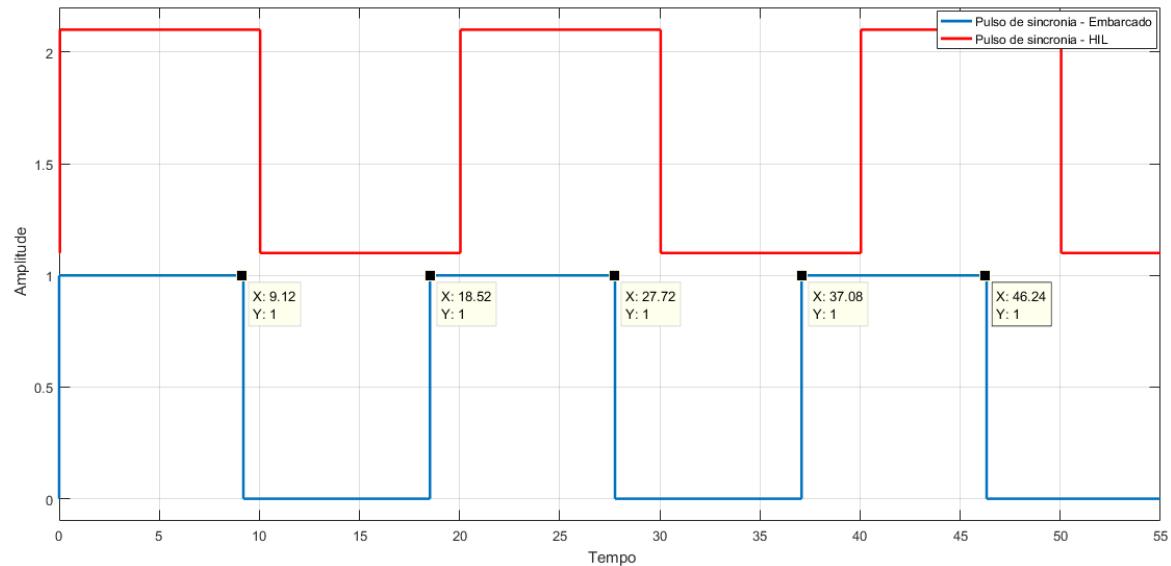
Visualmente, os melhores resultados foram obtidos com um menor valor de $T_s=0.005s$ para o controle embarcado e $T_s=0.05s$ para o controle em HIL, significando uma diferença de dez vezes na taxa de atualização.

5.1 Sincronismo

Notou-se que, com *Sample Times* menores que 0.05s, os pulsos de sincronia advindos da placa Arduino com o controle embarcado (MEGA) perdiam o passo fixo do tempo real no processamento. Este fenômeno pode ser observado nos gráficos da Figura 69 e da Figura 70:

Figura 69 – Pulses de sincronia com $T_s=0.05s$ 

Fonte: Autores.

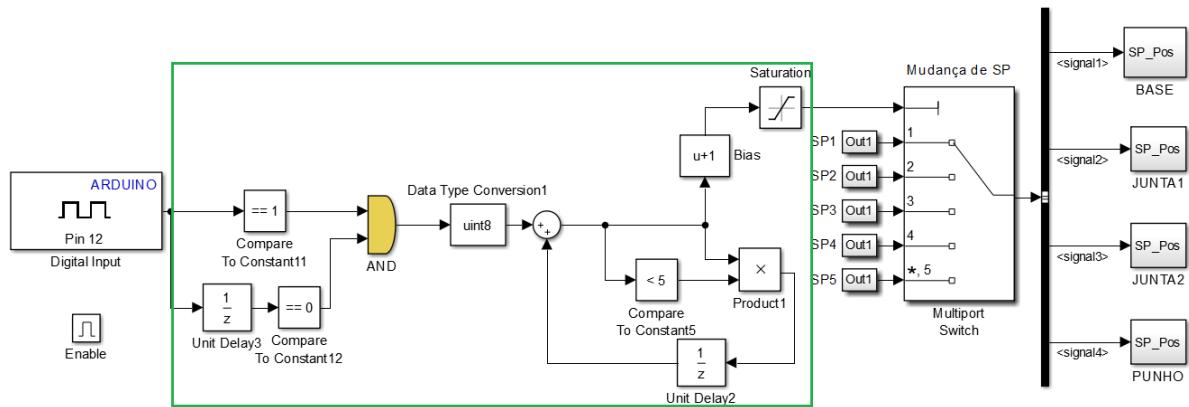
Figura 70 – Pulses de sincronia com $T_s=0.04s$ 

Fonte: Autores.

Esta falta de sincronia impediria a análise gráfica fidedigna dos dados dos sensores.

Para tentar igualar o processamento do tempo nos dois controles, foram acrescidos nos algoritmos novos blocos para gerar a mudança de *Setpoint* de forma externa. A configuração desses blocos pode ser visualizada na Figura 71.

Figura 71 – Novo diagrama de blocos embarcado



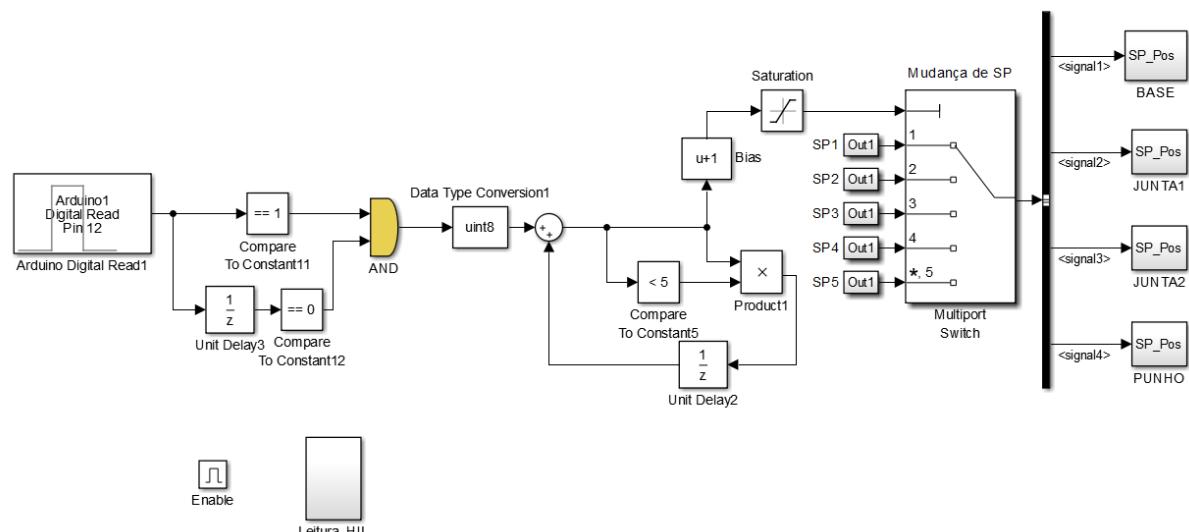
Fonte: Autores.

Os blocos destacados na cor verde na Figura 71 tinham a função de variar o sinal de saída entre 1 e 5 toda vez que recebesse um pulso digital alto na entrada. Esta variação permitiu o bloco de nome *Multiport Switch* intercambiar os valores de referência contidos nos blocos *SP1*, *SP2*, *SP3*, *SP4* e *SP5* para os subsistemas das juntas. As transições de referência podem ser observadas em vermelho nos gráficos da seção 5.2.

Desta forma o controlador não ficaria processando a mudança de *Setpoint*, este somente mudaria as referências pré-programadas para as juntas quando recebesse um sinal digital alto em uma entrada do Arduino MEGA. Enquanto isso, o PC processaria os pulsos para a mudança dos SPs e enviaria o sinal digital através do Arduino UNO.

Apesar da mudança só ter provocado efeito prático no controle embarcado, foi mudado o controle HIL para manter a padronização, como é visto na Figura 72.

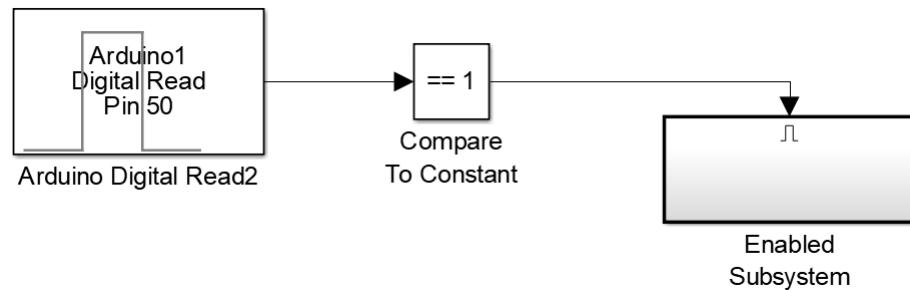
Figura 72 – Novo diagrama de blocos HIL



Fonte: Autores.

Foi adicionado, também, um botão externo para inicializar os algoritmos nas duas placas Arduino no mesmo instante. Para isso utilizou-se o bloco *Enabled Subsystem* do Simulink que iniciaria os algoritmos somente quando o botão levasse uma entrada digital das duas placas Arduino para nível lógico alto, como é exemplificado na Figura 73.

Figura 73 – Inicialização dos algoritmos



Fonte: Autores.

Estas alterações eliminaram o problema do sincronismo do tempo, visto que o sinal de mudança de SP em relação ao tempo passou a ser processado pelo PC.

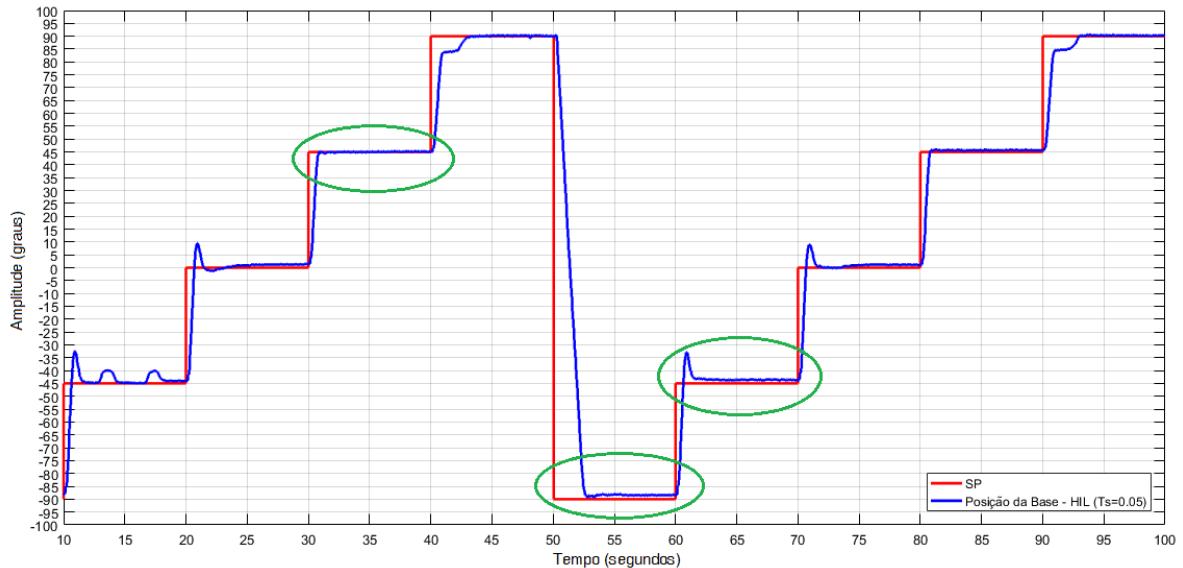
5.2 Resultados finais

- Base:

HIL:

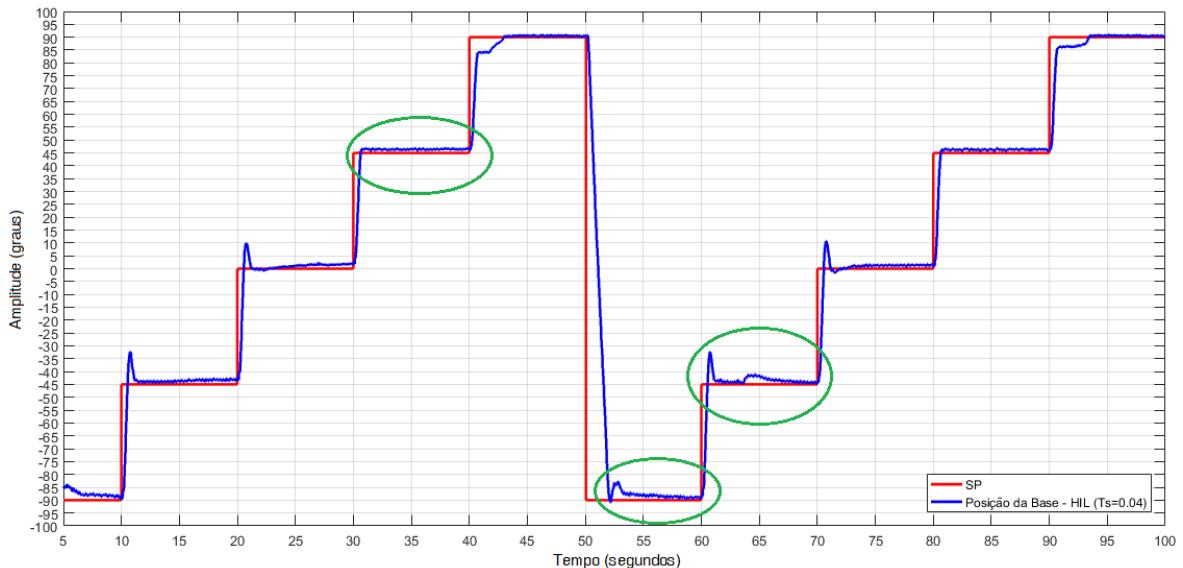
A base, por ser a junta mais estável, sofreu menos com oscilações quando reduzido o T_s de 0.05s para 0.04s. Ainda assim, podemos notar que, no segundo ciclo, a junta oscila mais em $T_s=0.04s$ como destacado nas figuras Figura 74 e Figura 75.

Figura 74 – Posição da Base - HIL ($T_s=0.05s$)



Fonte: Autores.

Figura 75 – Posição da Base - HIL ($T_s=0.04s$)



Fonte: Autores.

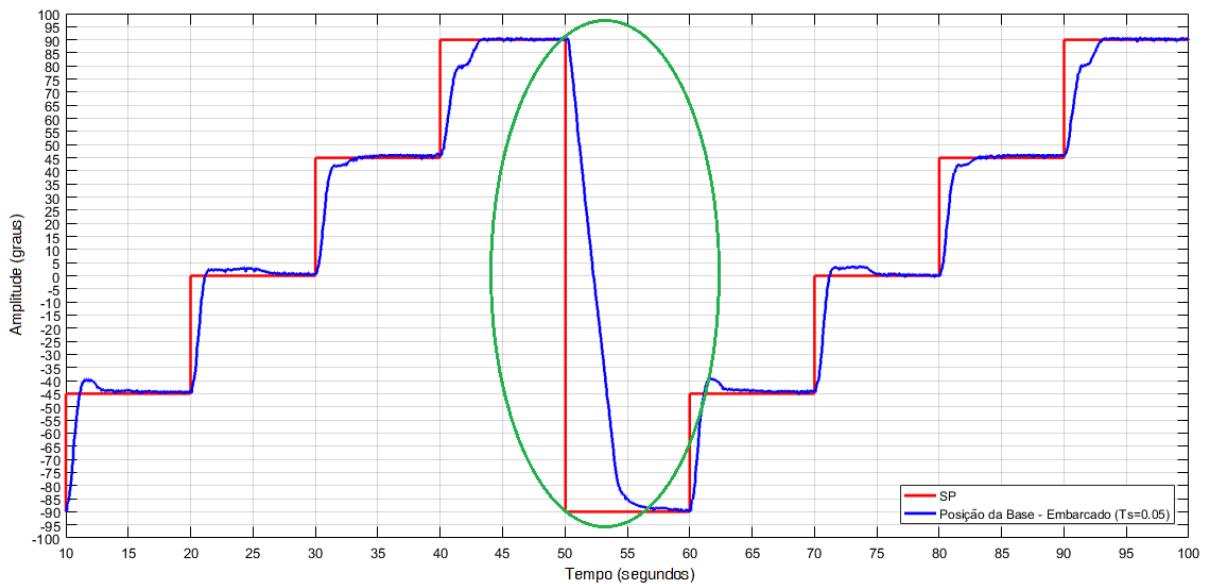
Embarcado:

Com $T_s=0.05s$, o controle em HIL ainda se comportou um pouco melhor do que o controle embarcado com o mesmo T_s , como pode-se notar nas figuras Figura 74 e Figura 76. Porém, quando reduzimos o T_s para 0.04s não notamos no controle embarcado a oscilação que foi provocada no controle HIL (Figura 75 e Figura 77).

Nota-se também que o controlador embarcado demorou um pouco mais para responder na transição de $SP = 90$ ao $SP = -90$, como destacado na Figura 76.

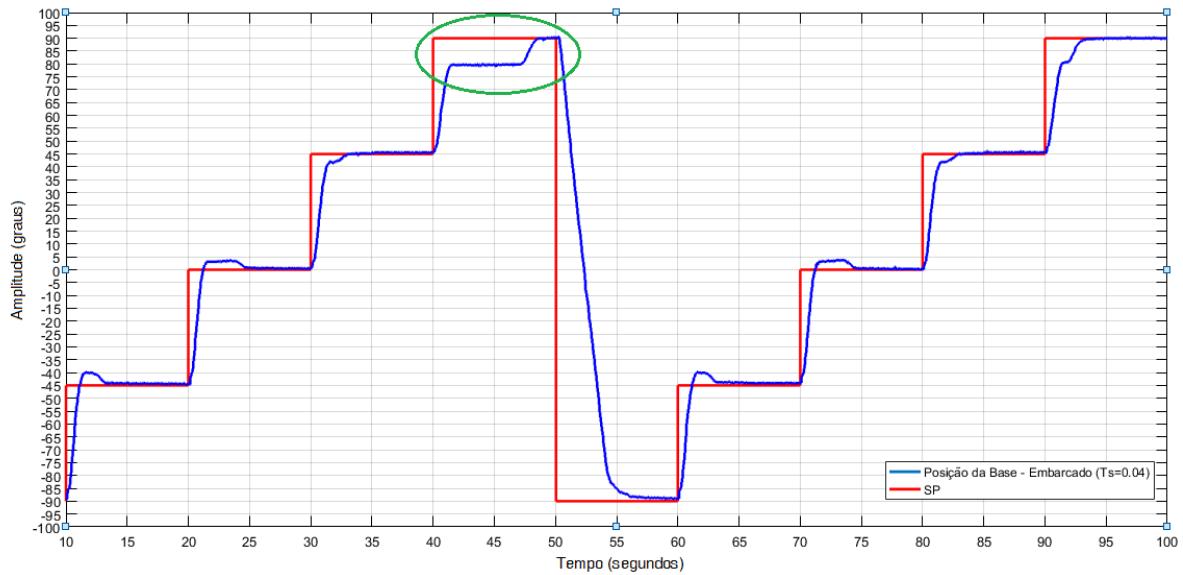
Com valores menores que $T_s=0.04s$, o controle em HIL não pôde ser implementado com sucesso, se tornando deveras instável. Entretanto o controle embarcado foi bem sucedido em valores de T_s de até 0.002s devido à boa estabilidade da junta.

Figura 76 – Posição da Base - Embarcado ($T_s=0.05s$)



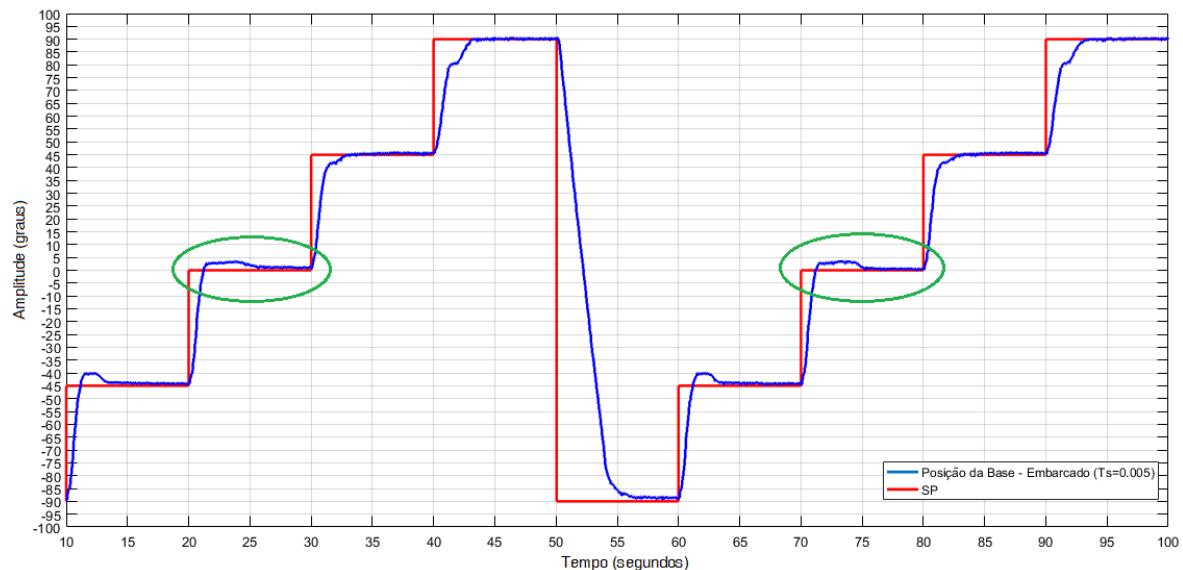
Fonte: Autores.

Em algumas simulações, o primeiro ciclo de mudança de referência se comportou de modo diferente do resto dos ciclos, provocando alguns erros aleatórios, como destacado na Figura 77.

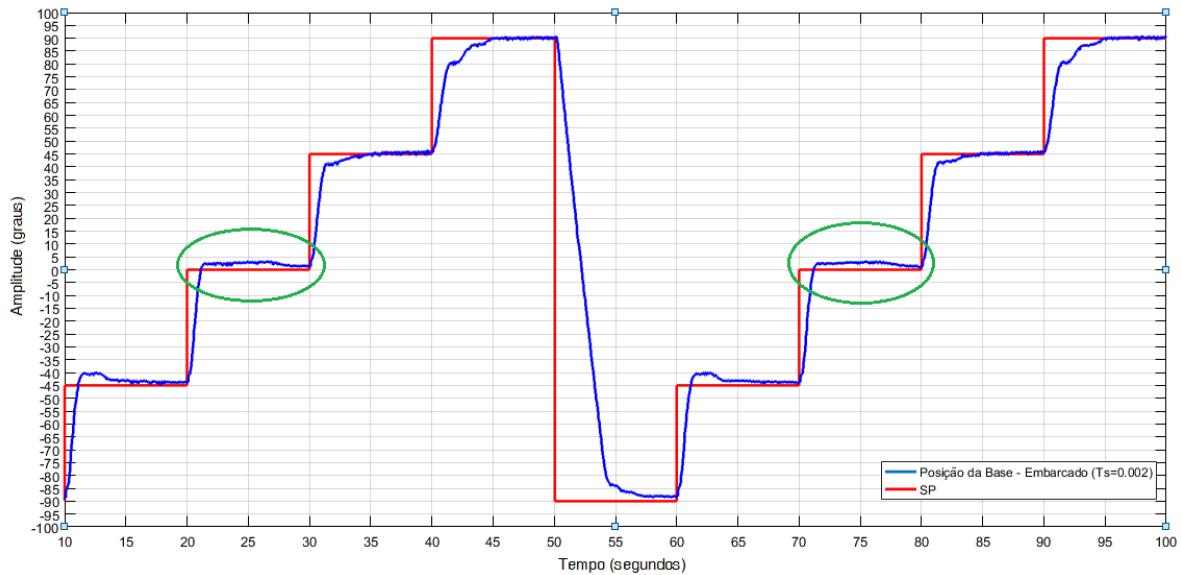
Figura 77 – Posição da Base - Embarcado ($T_s=0.04s$)

Fonte: Autores.

A partir de $T_s=0.002s$ já se nota uma mudança na resposta em pelo menos uma transição, como visto na Figura 78 e na Figura 79.

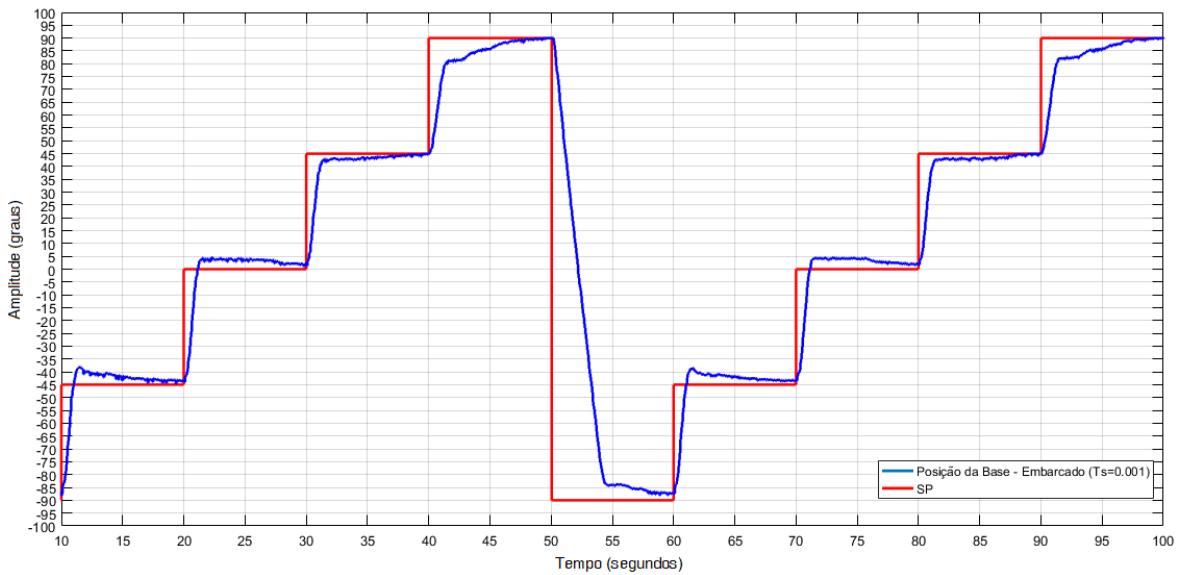
Figura 78 – Posição da Base - Embarcado ($T_s=0.005s$)

Fonte: Autores.

Figura 79 – Posição da Base - Embarcado ($T_s=0.002s$)

Fonte: Autores.

Com $T_s=0.001s$, a resposta de todas as juntas já são afetadas, diminuindo a performance do controlador como mostra a Figura 80.

Figura 80 – Posição da Base - Embarcado ($T_s=0.001s$)

Fonte: Autores.

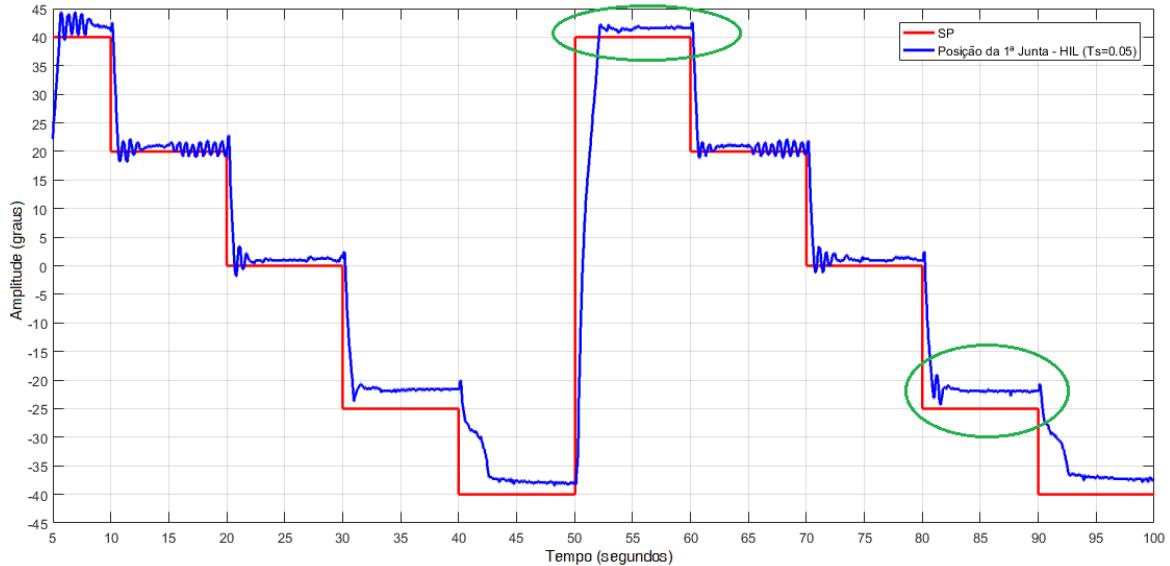
- 1^a Junta:

HIL:

A primeira junta foi a mais instável do conjunto, pois esta era altamente afetada pelo movimento da segunda junta. No controle em HIL, é destacado na Figura 81 e

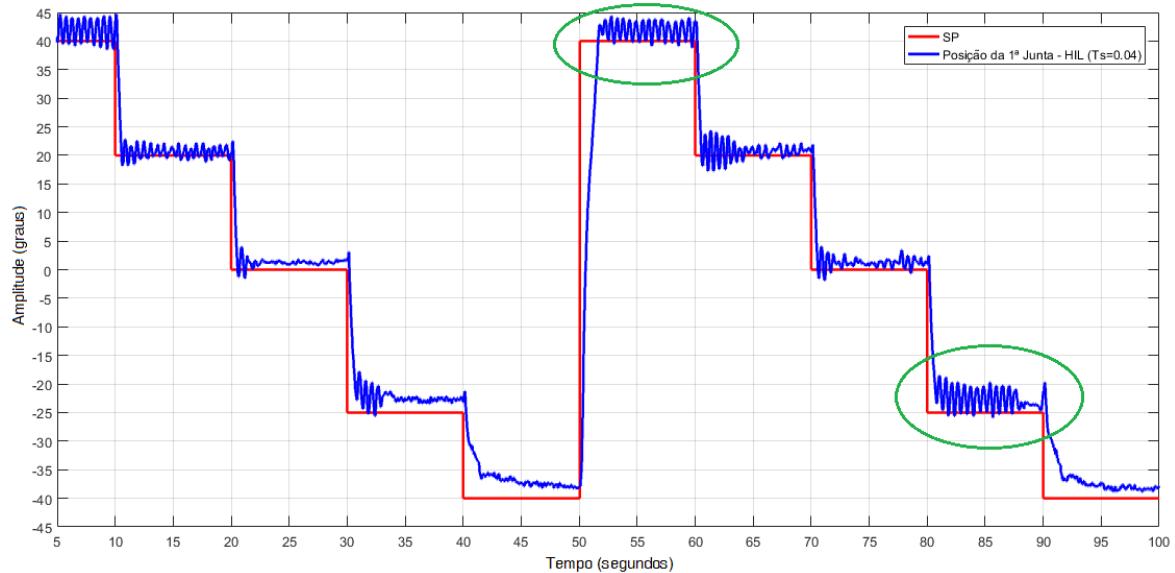
na Figura 82 que a instabilidade só piora quando diminuído o Ts para 0.04s.

Figura 81 – Posição da 1^a Junta - HIL (Ts=0.05s)



Fonte: Autores.

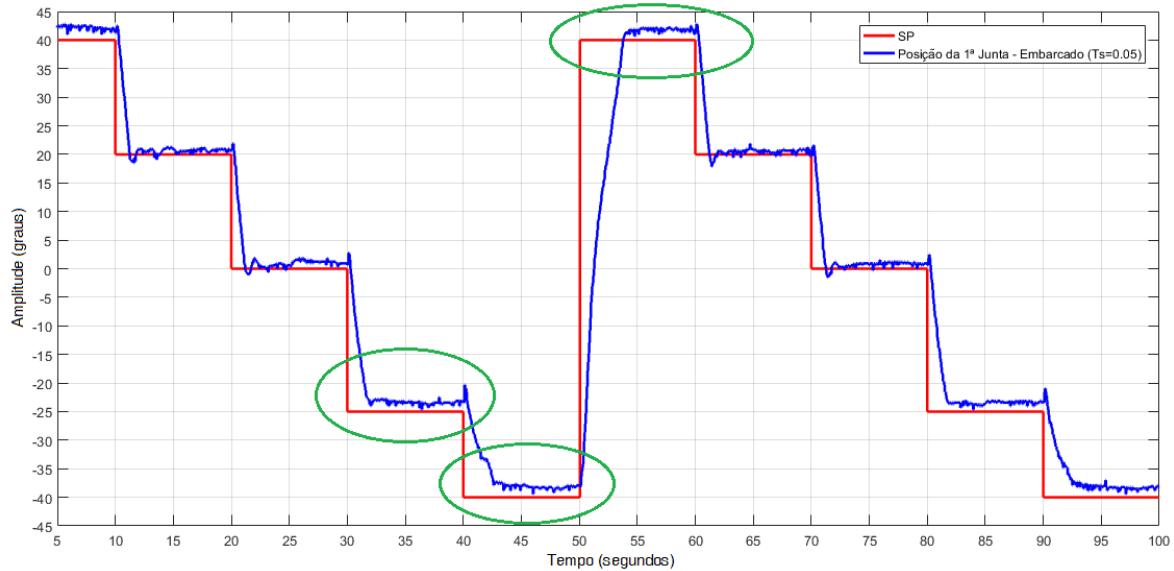
Figura 82 – Posição da 1^a Junta - HIL (Ts=0.04s)



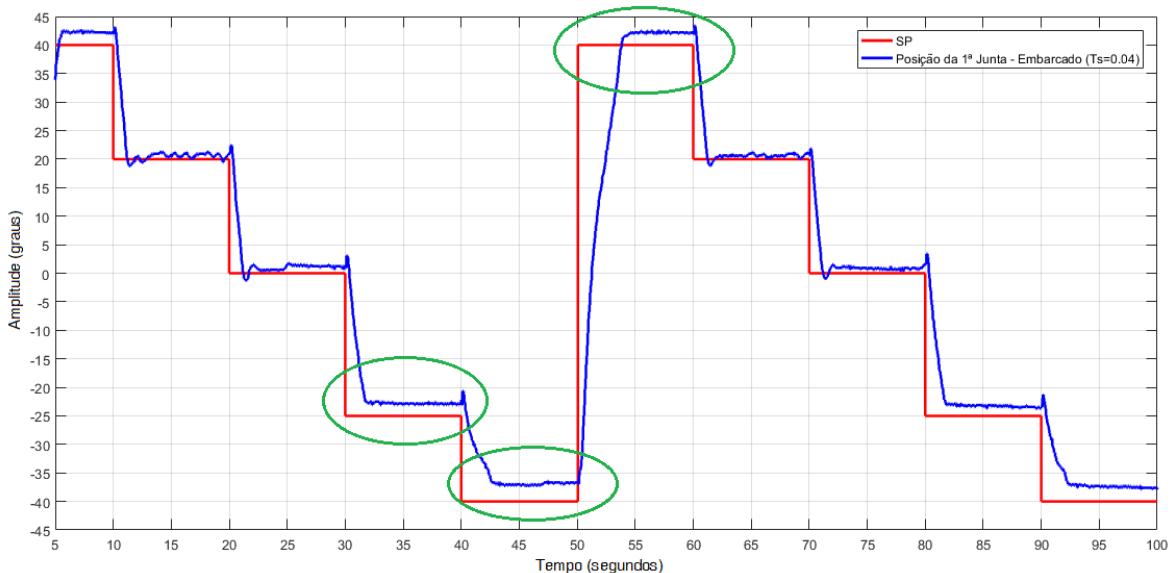
Fonte: Autores.

Embarcado:

No controle embarcado logo se percebe a notável diferença em comparação com o HIL. A junta se torna mais estável, inclusive com Ts=0.04s, como mostrado na Figura 81 em comparação com a Figura 83 e na Figura 82 em comparação com a Figura 84.

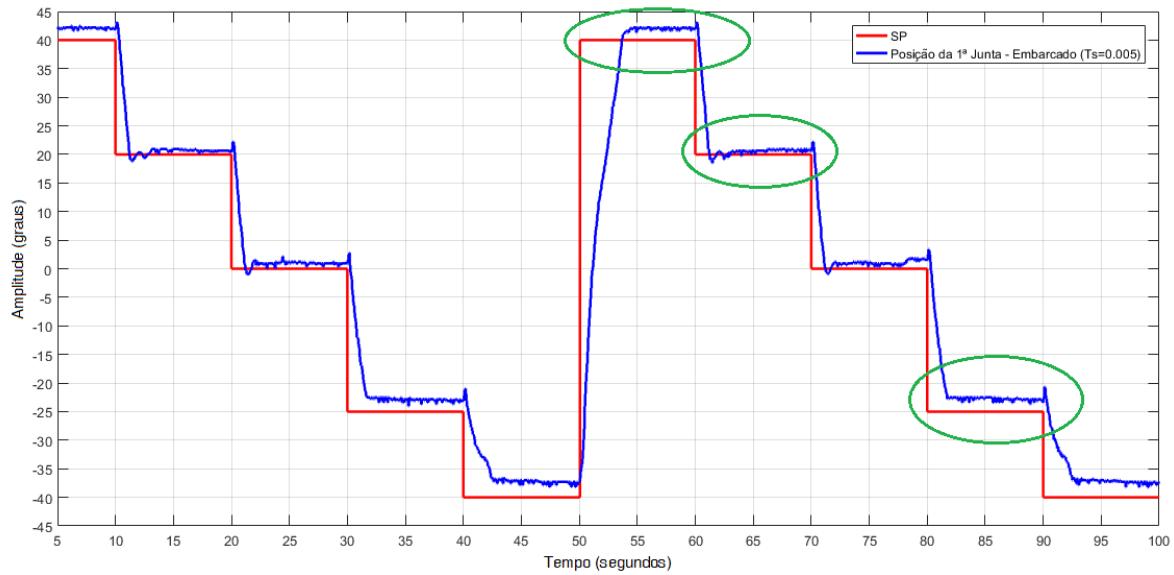
Figura 83 – Posição da 1^a Junta - Embarcado ($T_s=0.05s$)

Fonte: Autores.

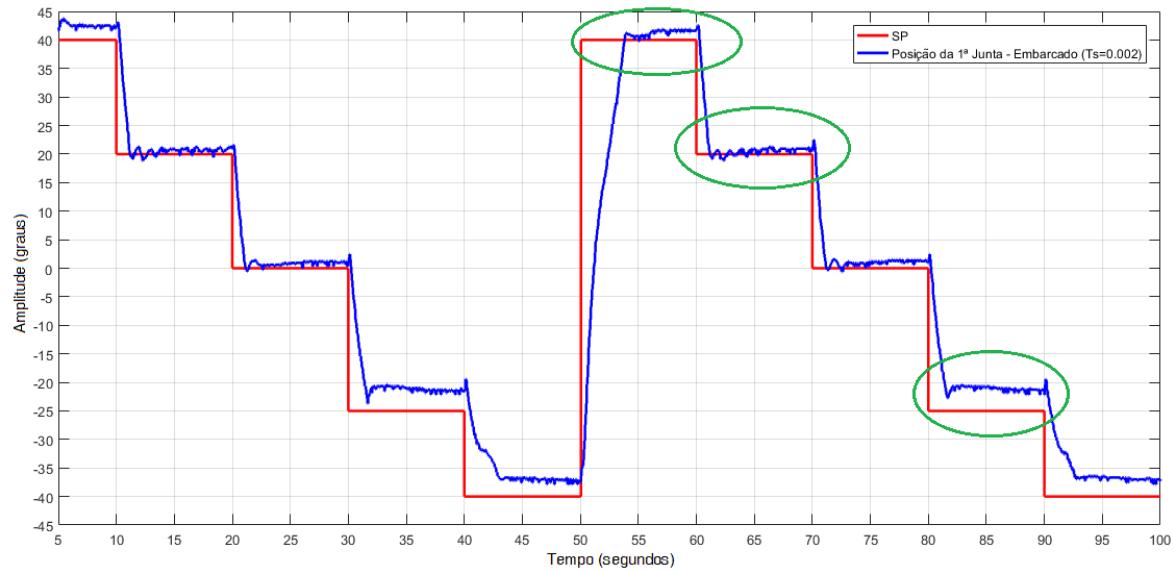
Figura 84 – Posição da 1^a Junta - Embarcado ($T_s=0.04s$)

Fonte: Autores.

A junta se comportou de maneira estável até $T_s=0.002s$, onde a resposta começa a destoar, como se nota na Figura 85 e na Figura 86.

Figura 85 – Posição da 1^a Junta - Embarcado ($T_s=0.005s$)

Fonte: Autores.

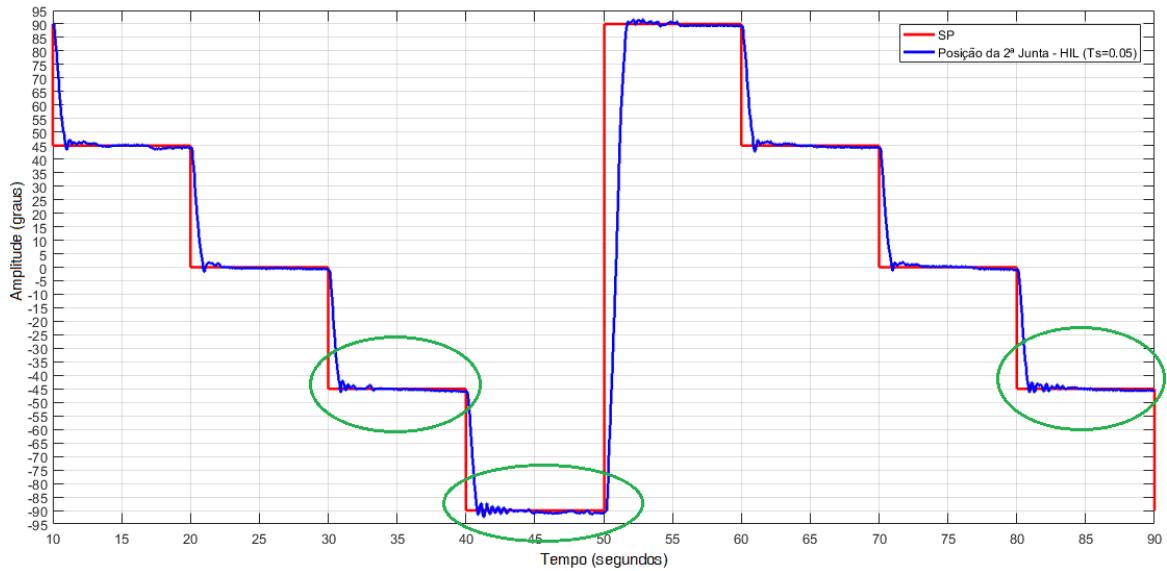
Figura 86 – Posição da 1^a Junta - Embarcado ($T_s=0.002s$)

Fonte: Autores.

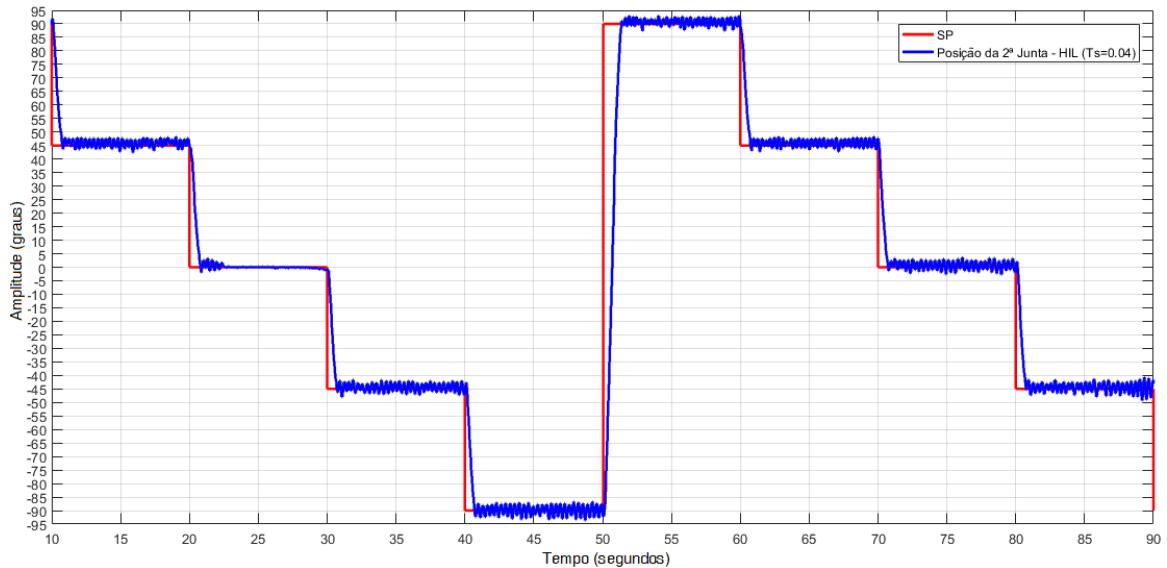
- 2^a Junta:

HIL:

Na segunda junta também percebe-se na Figura 87 e na Figura 88 uma instabilidade instituída por períodos de amostragens inferiores a 0.05s.

Figura 87 – Posição da 2^a Junta - HIL (Ts=0.05s)

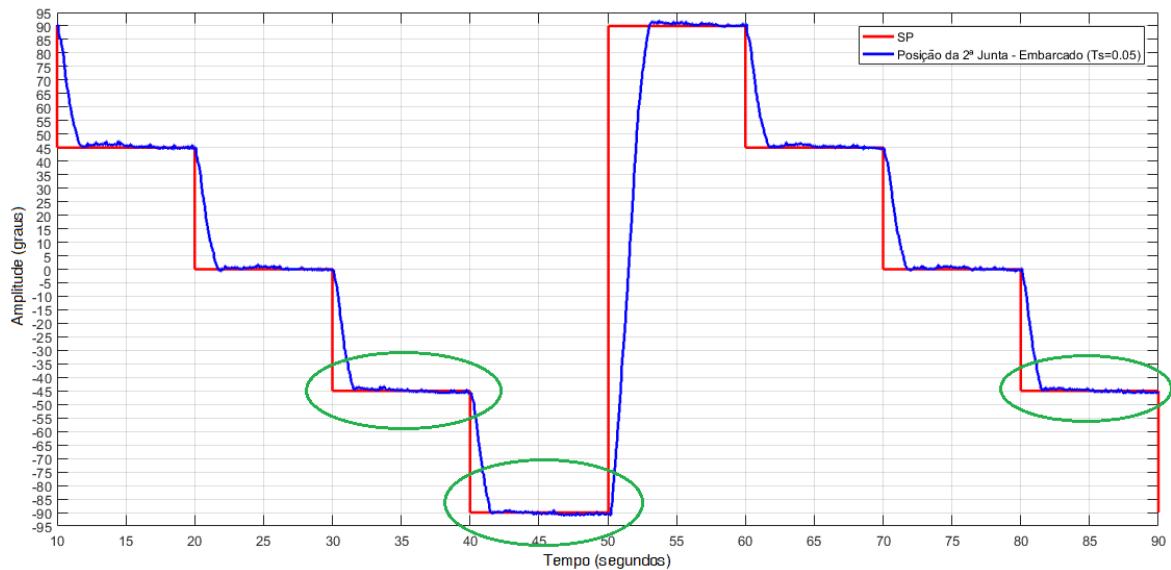
Fonte: Autores.

Figura 88 – Posição da 2^a Junta - HIL (Ts=0.04s)

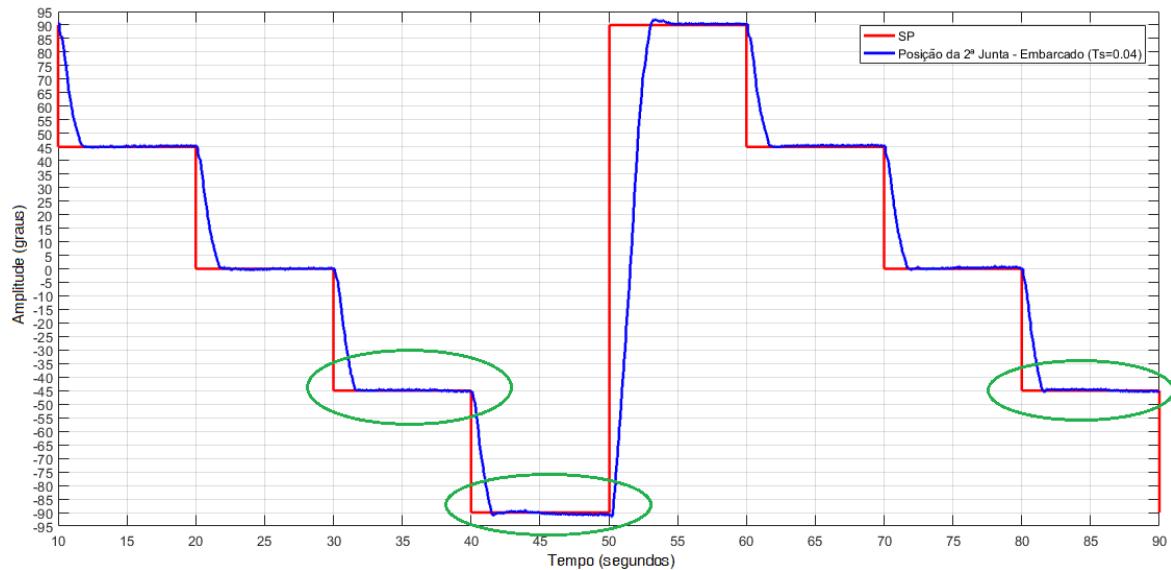
Fonte: Autores.

Embarcado:

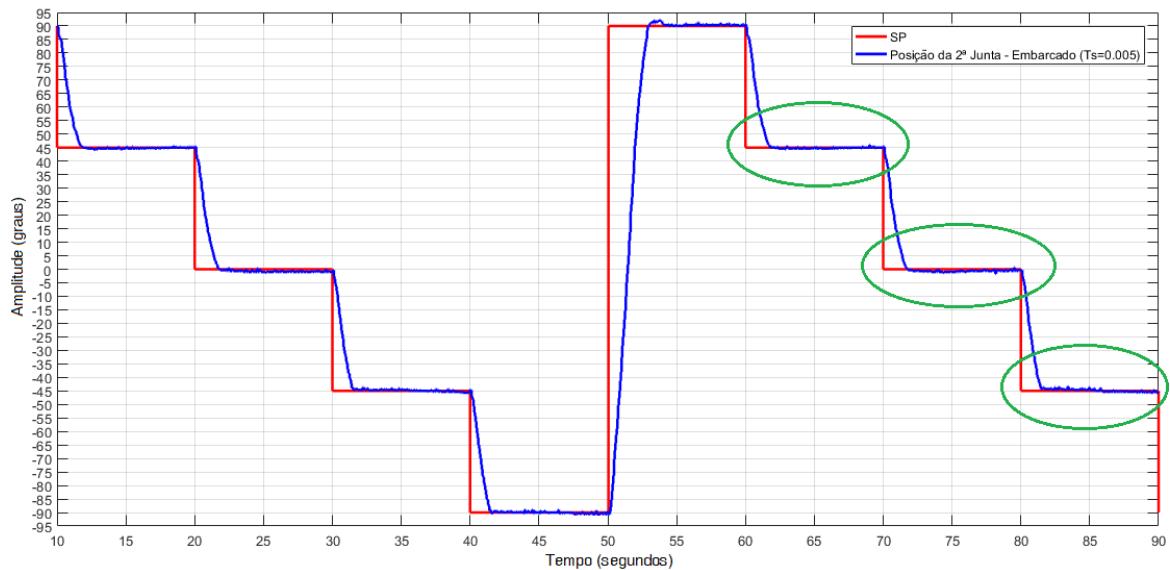
A diferença é nítida com o controle embarcado. A estabilidade é facilmente alcançada já em $T_s=0.05s$ como destacado na Figura 87 e na Figura 89, e ainda é melhorada com períodos inferiores, como mostrado na Figura 90 e na Figura 91, até o $T_s=0.002s$. A partir deste T_s já se vê algumas oscilações a mais, comprovadas na Figura 92. Com $T_s=0.001s$ a piora é significativa, notada na Figura 93.

Figura 89 – Posição da 2^a Junta - Embarcado (Ts=0.05s)

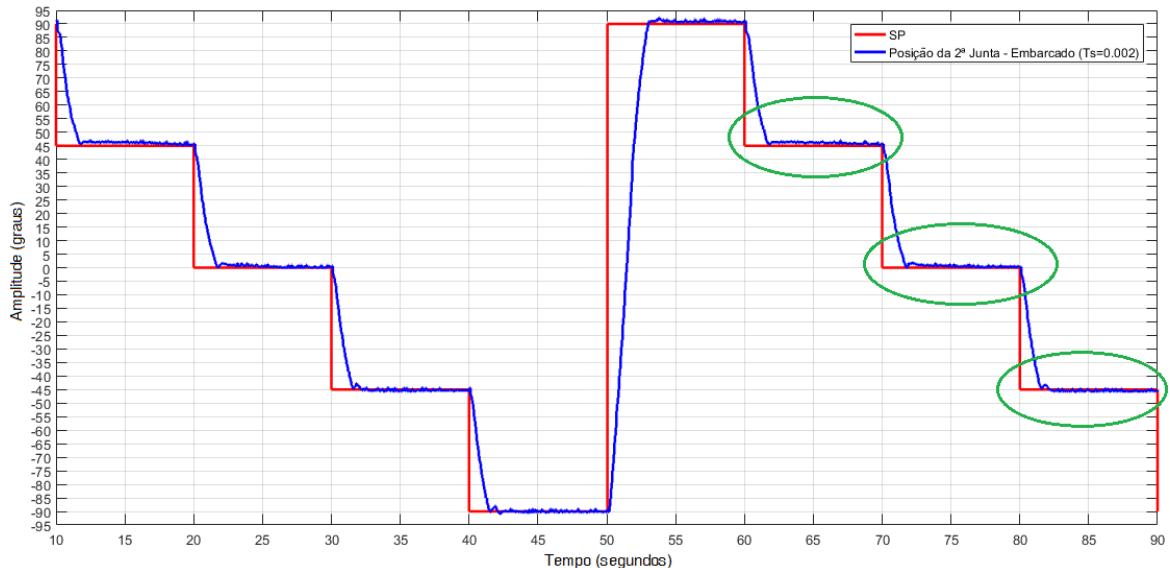
Fonte: Autores.

Figura 90 – Posição da 2^a Junta - Embarcado (Ts=0.04s)

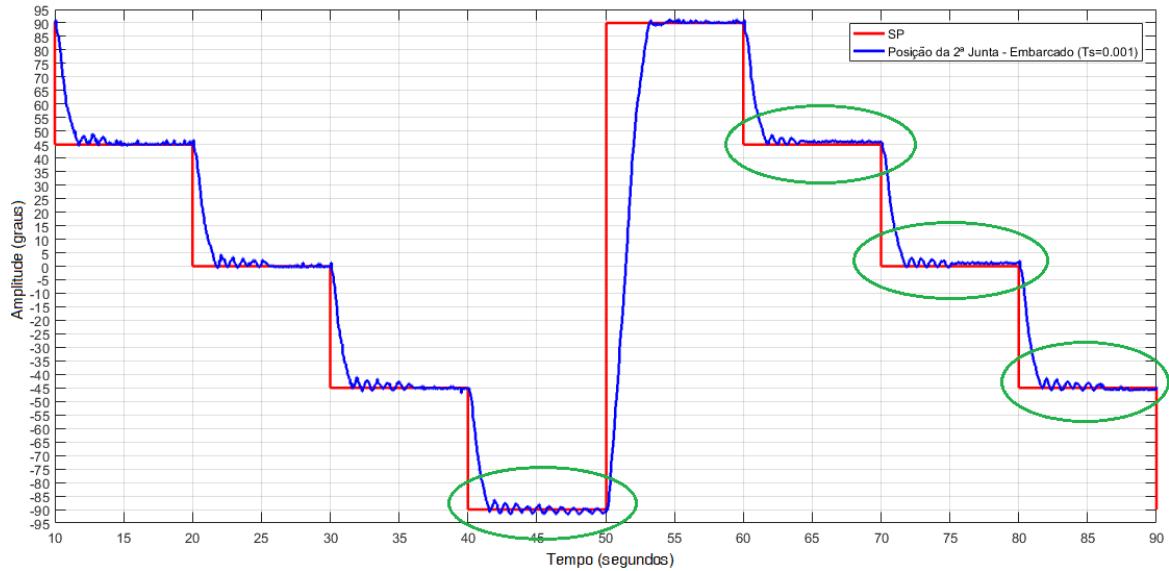
Fonte: Autores.

Figura 91 – Posição da 2^a Junta - Embarcado (Ts=0.005s)

Fonte: Autores.

Figura 92 – Posição da 2^a Junta - Embarcado (Ts=0.002s)

Fonte: Autores.

Figura 93 – Posição da 2^a Junta - Embarcado (Ts=0.001s)

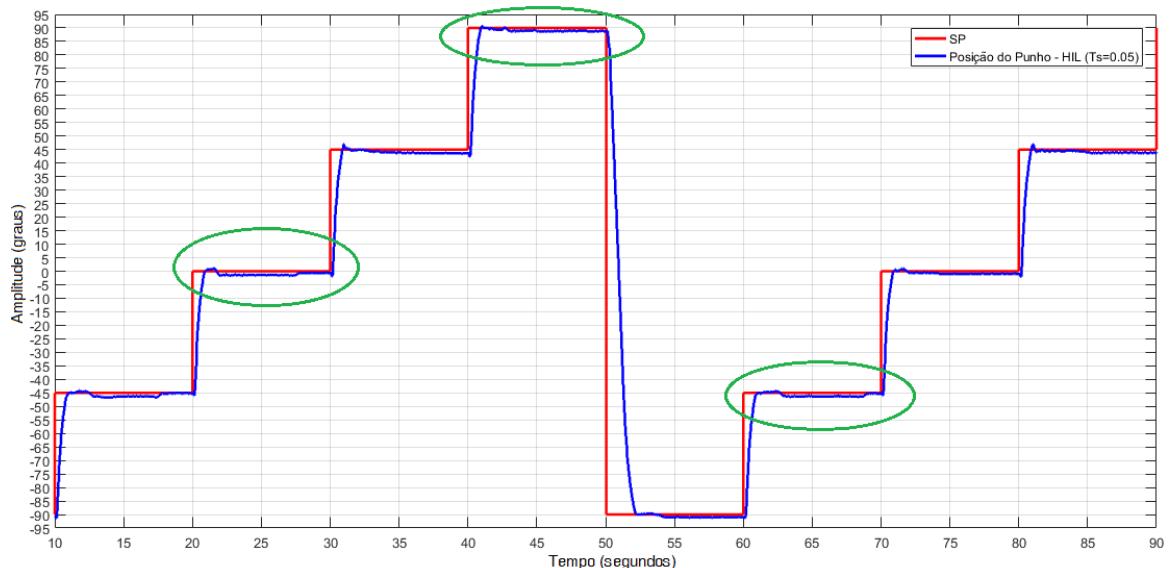
Fonte: Autores.

- Punho:

HIL:

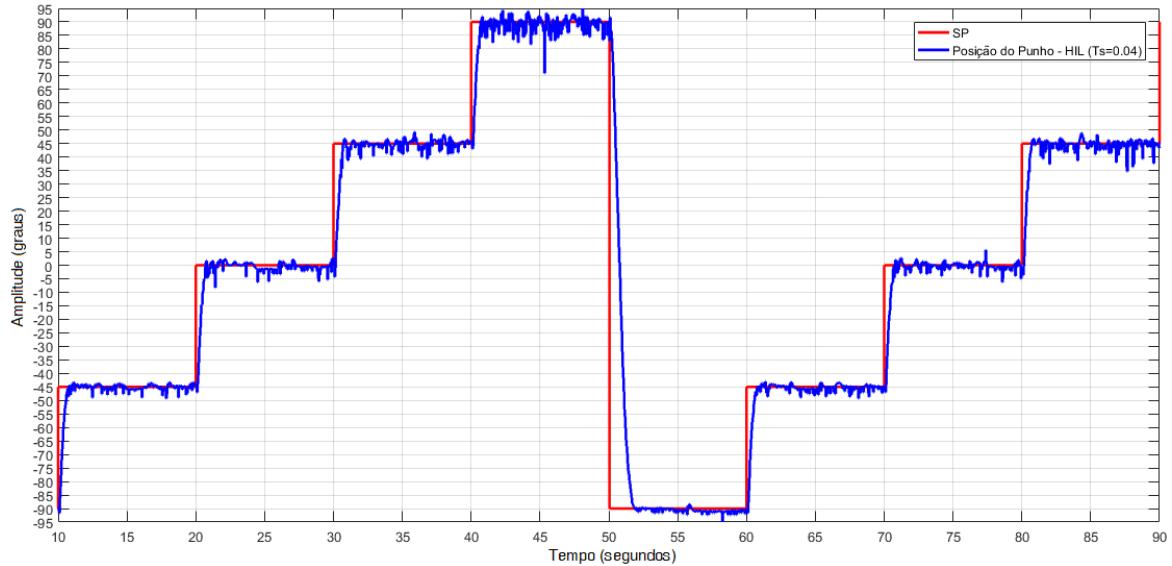
Novamente tem-se que, em HIL, o sistema se instabiliza com períodos abaixo de 0.05s, como evidenciado na Figura 94 e na Figura 95.

Figura 94 – Posição do Punho - HIL (Ts=0.05s)



Fonte: Autores.

Figura 95 – Posição do Punho - HIL (Ts=0.04s)

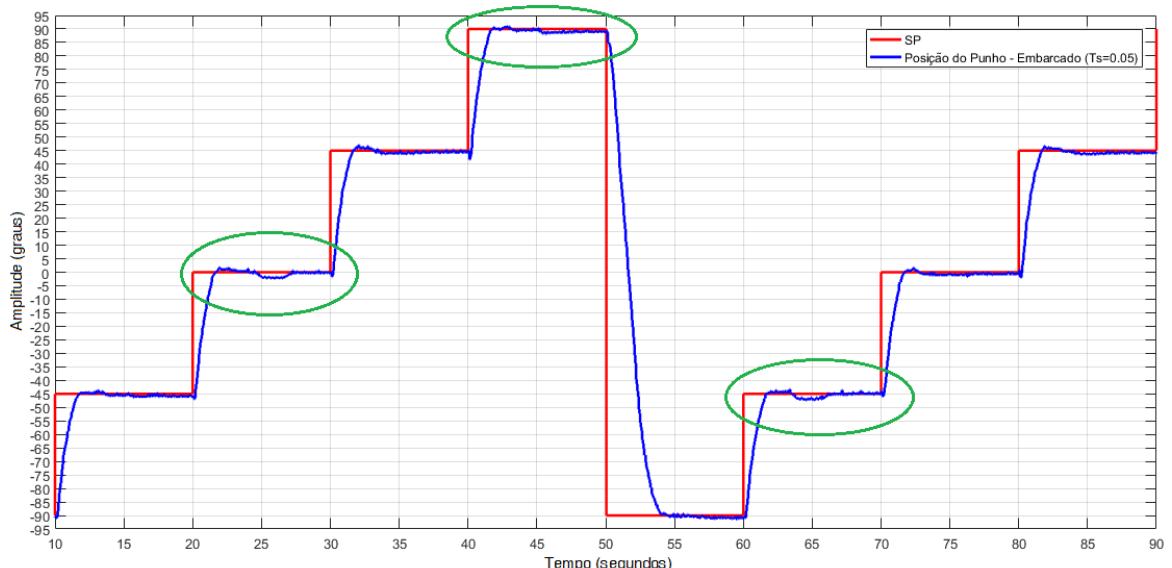


Fonte: Autores.

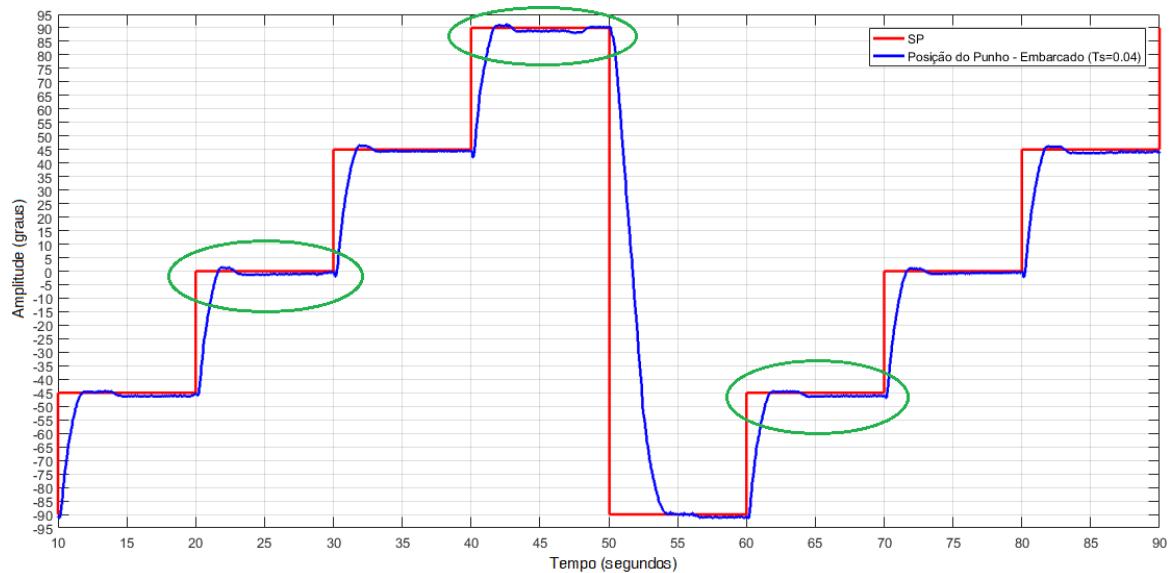
Embarcado:

Em $T_s=0.05s$ o controle em HIL tem uma resposta ligeiramente melhor se comparado ao embarcado, como destacado na Figura 94 e na Figura 96. Porém em *Sample Times* menores, onde o controle HIL não alcança com sucesso, o embarcado consegue diminuir ainda mais a oscilação da junta, em destaque na Figura 97, Figura 98 e Figura 99.

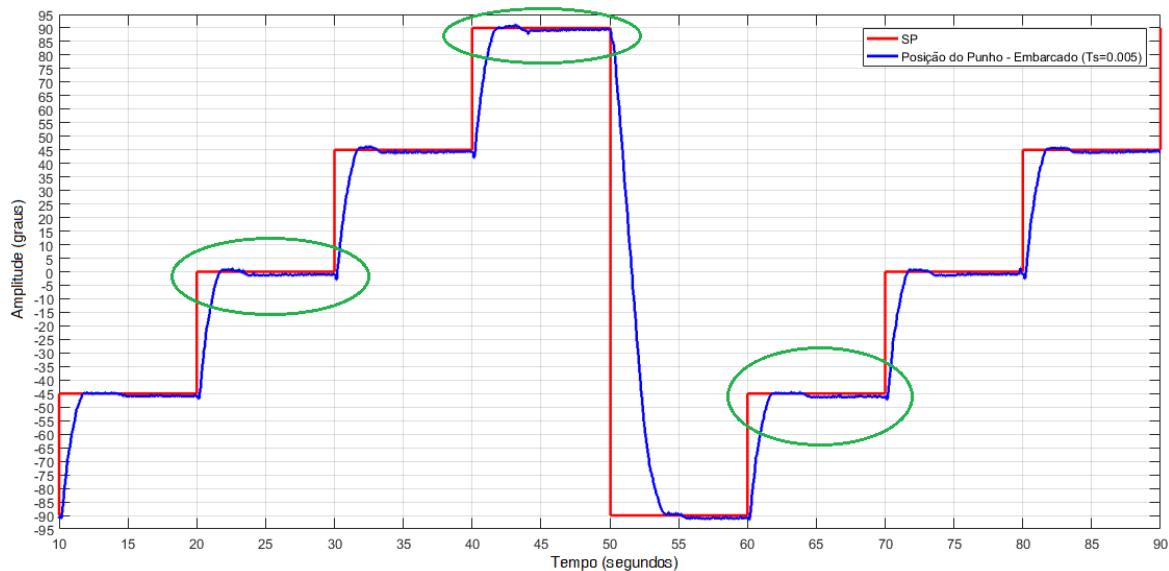
Figura 96 – Posição do Punho - Embarcado (Ts=0.05s)



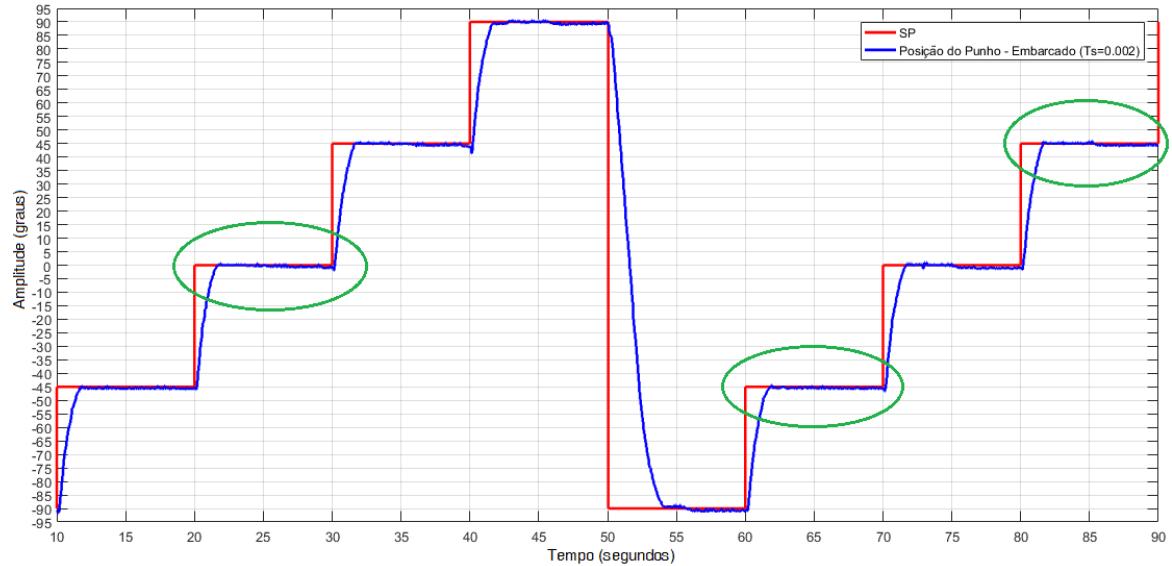
Fonte: Autores.

Figura 97 – Posição do Punho - Embarcado ($T_s=0.04s$)

Fonte: Autores.

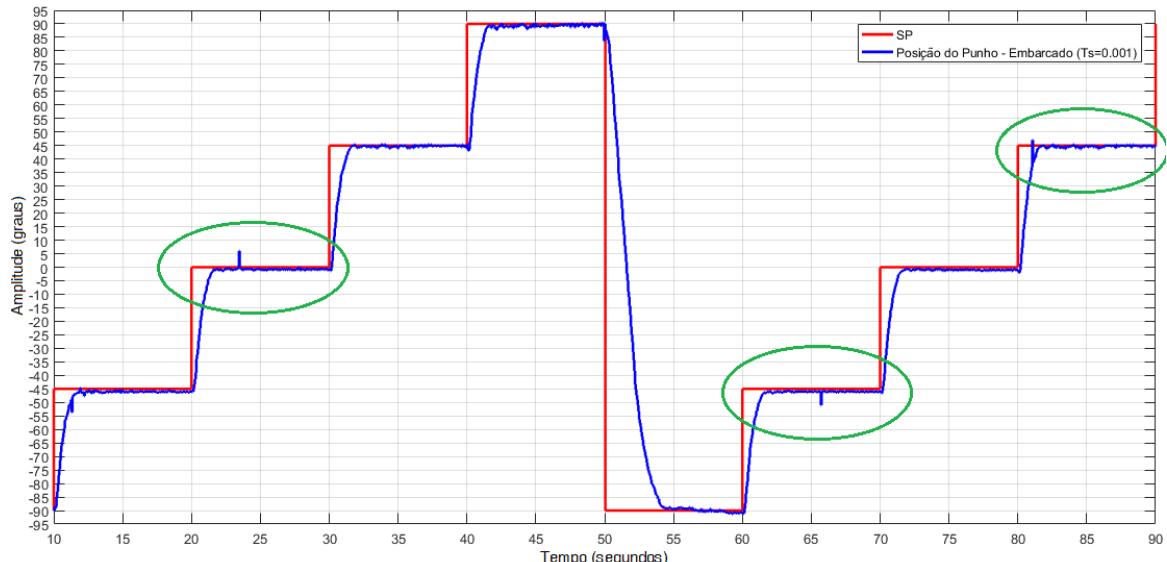
Figura 98 – Posição do Punho - Embarcado ($T_s=0.005s$)

Fonte: Autores.

Figura 99 – Posição do Punho - Embarcado ($T_s=0.002s$)

Fonte: Autores.

Somente a partir de $T_s=0.001s$, apresentado na Figura 100, é que a resposta começa a assumir caráter oscilatório.

Figura 100 – Posição do Punho - Embarcado ($T_s=0.001s$)

Fonte: Autores.

6 Considerações Finais

Este capítulo apresenta a conclusão da análise feita a partir dos dados obtidos no experimento bem como sugestões para trabalhos futuros acerca do tema deste.

6.1 Conclusão

Resumindo a análise feita a partir das respostas dos controles embarcado e em HIL, conclui-se que o controle embarcado é mais vantajoso na maioria dos casos pois, com este, consegue-se uma melhor resposta no quesito estabilidade e fluidez de movimento, principalmente nas juntas 1 e 2, as mais instáveis do sistema, e visualmente o braço manteve seus movimentos mais estáveis quando sob o controle embarcado.

Além disto, com o controle embarcado é possível trabalhar com períodos de amostragem menores. As vantagens de se trabalhar com *Sample Times* menores (ou frequências de amostragem maiores) implicam em se conseguir tratar mais dados num mesmo período de tempo. Com isso, consegue-se aproximar o dado discreto ao contínuo, perder menos estados em instantes do sistema e tratar sinais cujas frequências sejam naturalmente altas. Com valores de T_s menores, também permite-se que o mesmo controlador atinja melhor performance através de um refinamento da sintonia, que não poderia ser feito com T_s maiores.

O controle em HIL ainda demonstra a limitação do processamento dependente de um *Hardware* não-dedicado à tarefa em questão. Sendo assim, compartilha-se o processamento com outras tarefas em segundo plano no PC. Um controle bem sucedido em um computador pode não ser bem sucedido em outro, dependendo da máquina em questão. Ou pode, até mesmo, não ser bem sucedido no primeiro computador, se a situação de memória e processamento não for a mesma.

Os objetivos deste trabalho foram alcançados com êxito. Foi feita a modelagem do sistema em Espaço de Estados a partir das curvas colhidas dos sensores da planta, desenvolvido um controlador para este sistema com realimentação de estado da planta real e feita a comparação das respostas do sistema a este controlador nas duas abordagens de controle escolhidas.

Portanto, utilizando um *Hardware* dedicado, ainda que com processamento relativamente baixo comparado ao PC utilizado, melhora-se o controle do sistema; realiza-se uma mesma tarefa com melhor performance, com as vantagens de utilizar um *Hardware* mais modesto, com menor custo, tamanho, consumo de energia, etc.; e simplifica-se a arquitetura de controle por reduzi-la em um componente.

6.2 Sugestões para trabalhos futuros

- Projetar um controlador ótimo baseado no menor período de amostragem para o controle embarcado.
- Criar outros métodos de controle baseados em técnicas de controle avançado (como controle robusto, adaptativo, *fuzzy*, entre outros) e tentar embarcá-los na placa microcontroladora Arduino, ou semelhante.
- Simular o controle em HIL com o Matlab sendo rodado com prioridade pelo sistema do PC, a fim de tentar eliminar algumas das fraquezas do processamento dependente.

Referências

- ALBUQUERQUE, A. R. L. de. *Aplicações de hardware-in-the-loop no desenvolvimento de uma mão robótica*. Tese (Doutorado em Engenharia Mecânica, 2007. Acessado em 13/02/2018. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/18/18135/tde-13062008-093112/en.php>>. Citado na página 21.
- ANGELICO, B. A.; SCALASSARA, P. R.; VARGAS, A. N. Et35q - princípios de controle: Modelagem de sistemas de controle por espaço de estados. Acessado em 15/02/2018. 2015. Disponível em: <http://paginapessoal.utfpr.edu.br/avargas/courses-1/principios_de_controle/principios_de_controle/principiosCap13_v2.pdf>. Citado na página 18.
- ARDUINO. *What is Arduino?* 2018. Acessado em 13/02/2018. Disponível em: <<https://www.arduino.cc/en/Guide/Introduction>>. Citado na página 22.
- BARÃO, M. J. S. Sistemas dinâmicos em espaço de estados (teoria). Acessado em 15/02/2018. 2000. Disponível em: <<http://ltodi.est.ips.pt/rlargo/SD/sd-Teo.pdf>>. Citado na página 21.
- CABRAL, E. L. L. Controlabilidade e observabilidade: Observabilidade. Acessado em 15/02/2018. 2018. Disponível em: <<http://sites.poli.usp.br/d/pmr2400/10-%20Controlabilidade%20e%20Observabilidade.pdf>>. Citado na página 21.
- CAMPA, G. *Legacy MATLAB and Simulink Support for Arduino*. 2016. Disponível em: <https://www.mathworks.com/matlabcentral/fileexchange/32374-legacy-matlab-and-simulink-support-for-arduino?s_cid=srchtitle>. Citado na página 40.
- CARRARA, V. Introdução à robótica industrial. INPE, 2015. Acessado em 26/02/2018. Disponível em: <<http://urlib.net/8JMKD3MGP3W34P/3K5JPL8>>. Citado na página 24.
- GONÇALVES, C. da S.; SODRÉ, U. *Álgebra Linear: Mudança de base em um Espaço vetorial*. 2006. Acessado em 07/02/2018. Disponível em: <<http://www.uel.br/projetos/matessencial/superior/linear/mbase.htm>>. Citado 2 vezes nas páginas 19 e 20.
- ITALIA, D. *RD5 NT Didactic Robot with 5 axes - User's Manual*. [S.l.], 2008. Citado na página 24.
- LEMOS, J. M. Controlo em espaço de estados. Resumo da Teoria. 2015. Citado 2 vezes nas páginas 18 e 19.
- PALMA, R. A. Uma metodologia de suporte a hardware-in-the-loop simulation para modelagem de uma caldeira a gás. 2006. Citado na página 21.
- ROMANO, V. F. *Robotica Industrial, Aplicação na Industria de Manufatura e de Processos*. 1. ed. [S.l.]: Blucher, 2002. 280 p. Citado na página 15.

SEQUEIRA, J. L. T. C. *Sistema para a Verificação Lógica do Controlo: Análise de sistemas de controle por espaço de estados.* candthesis, 2003. Acessado em 15/02/18. Disponível em: <<https://web.fe.up.pt/~ee98031/cap04fin.pdf>>. Citado na página 18.

SILVEIRA, H. M. *Fundamentos de Controle - EEL7531:* Notas de aula - parte linear (versão 6). candthesis, 2010. Acessado em 15/02/2018. Disponível em: <<http://www.labspot.ufsc.br/~simoes/fc/Cap2.pdf>>. Citado na página 21.

SJÖBERG, J. et al. *Nonlinear black-box modeling in system identification: a unified overview.* [S.l.: s.n.], 1995. Citado na página 26.

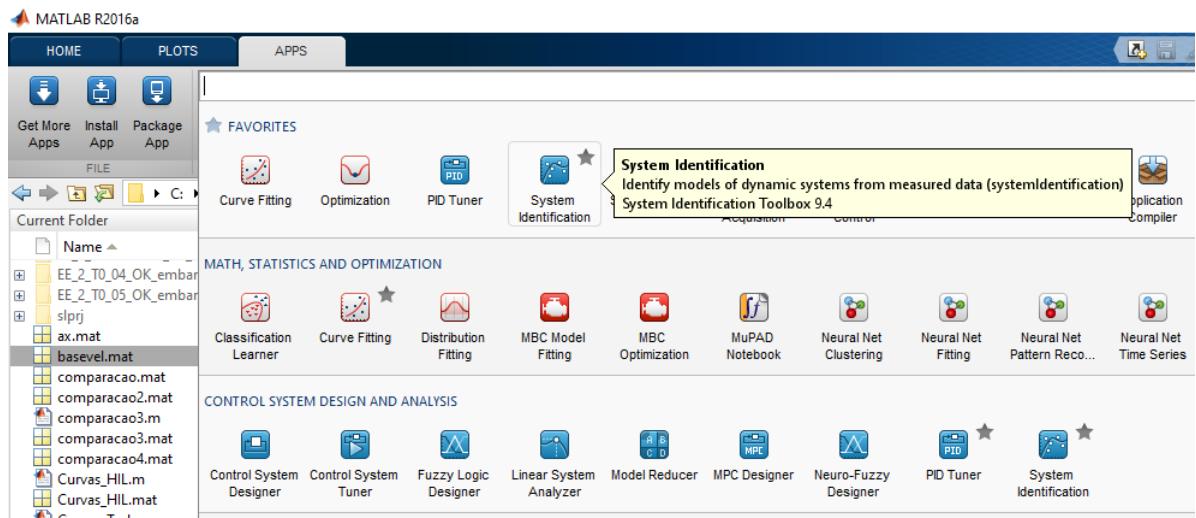
TUTUNJI, T. A.; SALEEM, A. A methodology for identification and control of electro-mechanical actuators. *MethodsX*, 2015. Acessado em 13/02/2018. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2215016115000242?via%3Dihub>>. Citado na página 22.

Apêndices

APÊNDICE A – *System Identification Tool*

Este apêndice apresenta uma breve explanação sobre a ferramenta *System Identification Tool* do Matlab, a qual foi utilizada para obter as funções de transferência dos atuadores das juntas do manipulador robótico. A Figura 101 mostra onde esta ferramenta está localizada no *software* Matlab.

Figura 101 – Localização do *System Identification Tool* no MATLAB



Fonte: Autores.

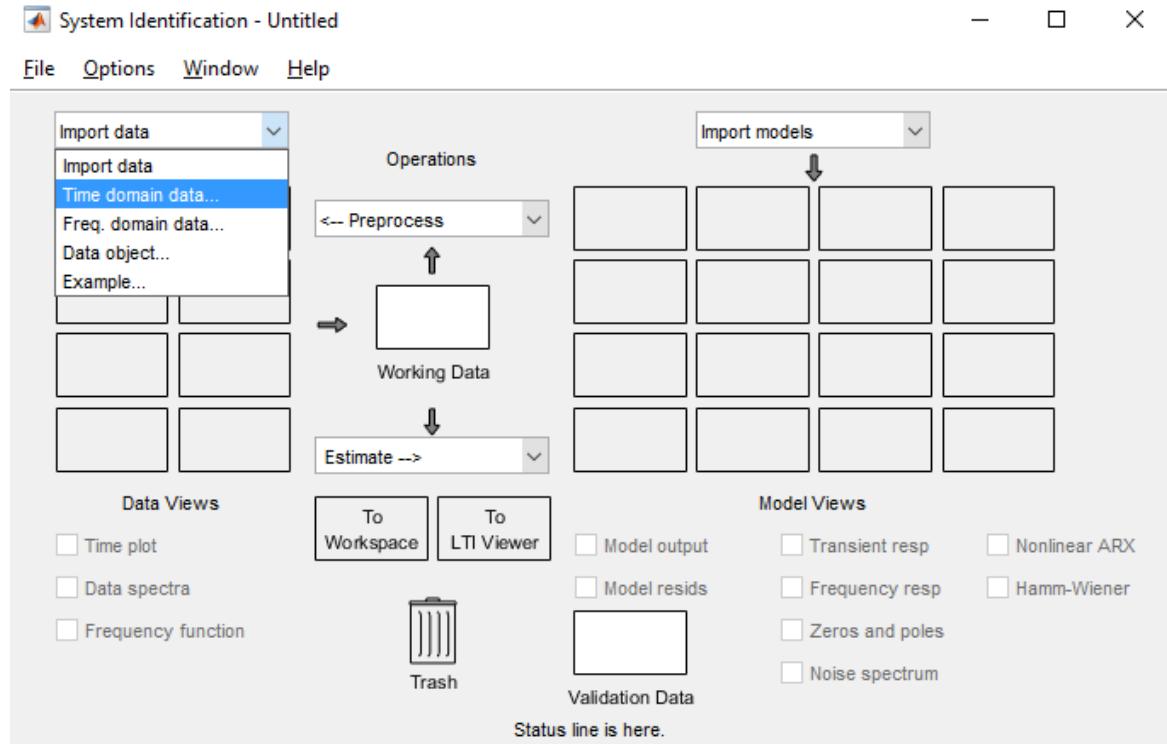
Esta ferramenta permite importar os dados lidos de um sensor no domínio do tempo ou da frequência juntamente com os dados da entrada (neste exemplo foi utilizado um degrau constante) no mesmo domínio, como mostrado na Figura 102 e na Figura 103.

Permite, ainda, tratar a curva dos dados com algumas ferramentas como, por exemplo, filtragem de ruídos, ilustrado na Figura 104.

Após a importação e tratamento dos dados pode-se fazer a estimativa da curva para vários tipos de modelos, como Função de Transferência, Espaço de Estados, entre outros, como mostrado na Figura 105.

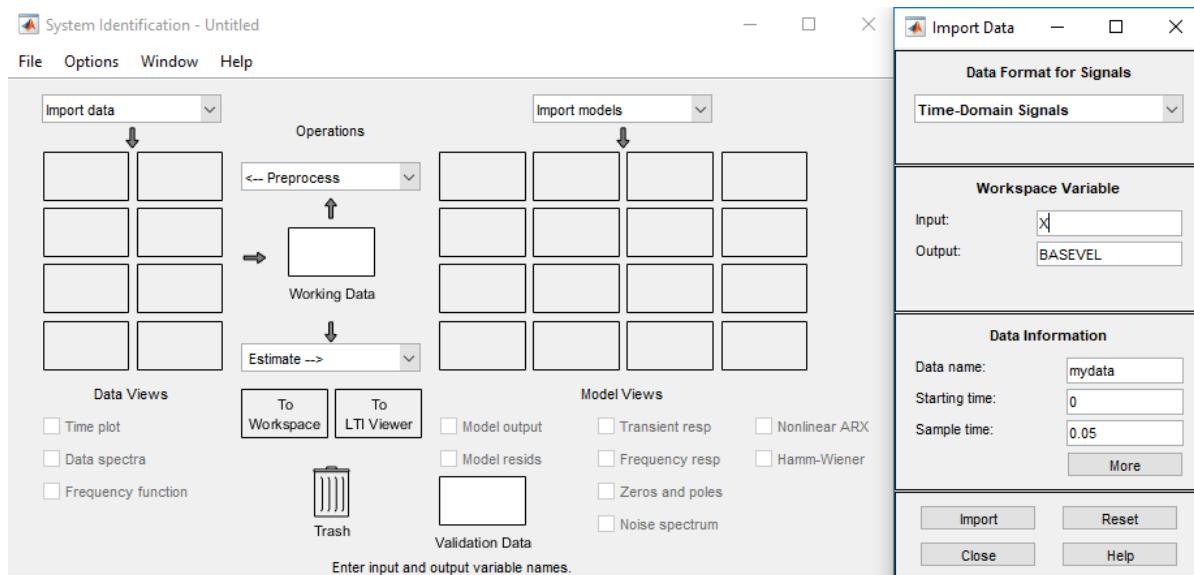
No exemplo a seguir estima-se uma função de transferência com 1 polo e nenhum zero, como apresentam a Figura 106 e a Figura 107.

Figura 102 – Importando dados no SIT



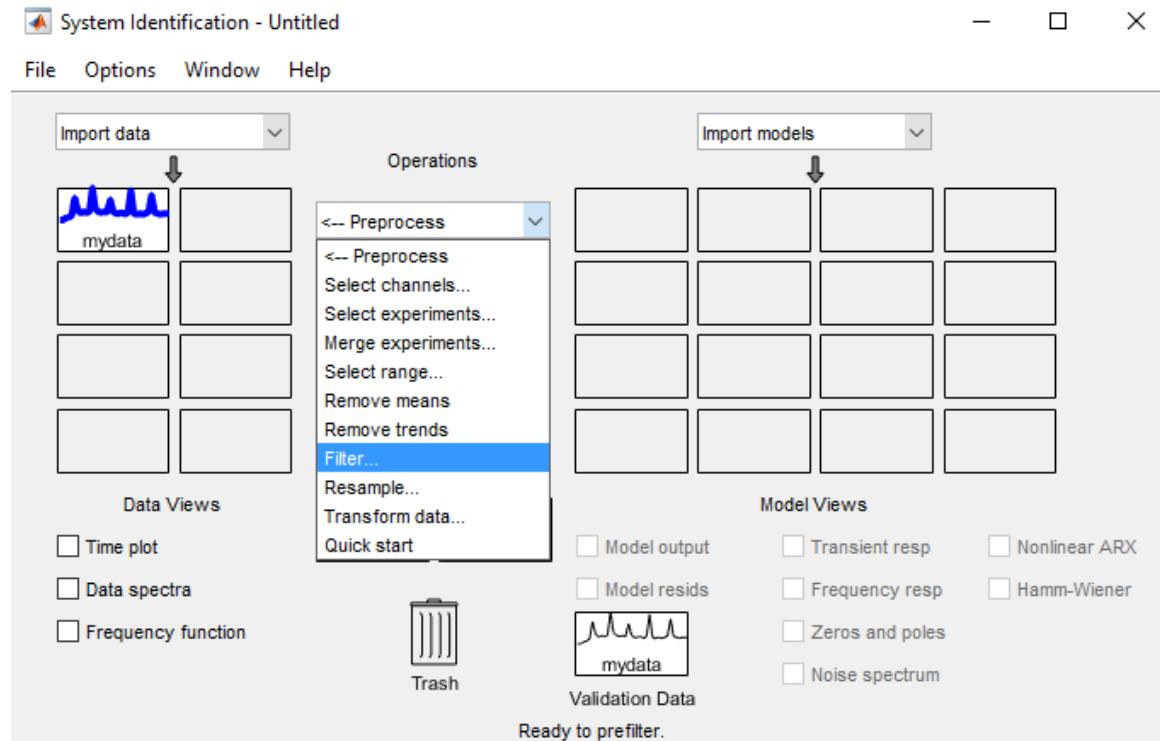
Fonte: Autores.

Figura 103 – Configurando dados no SIT



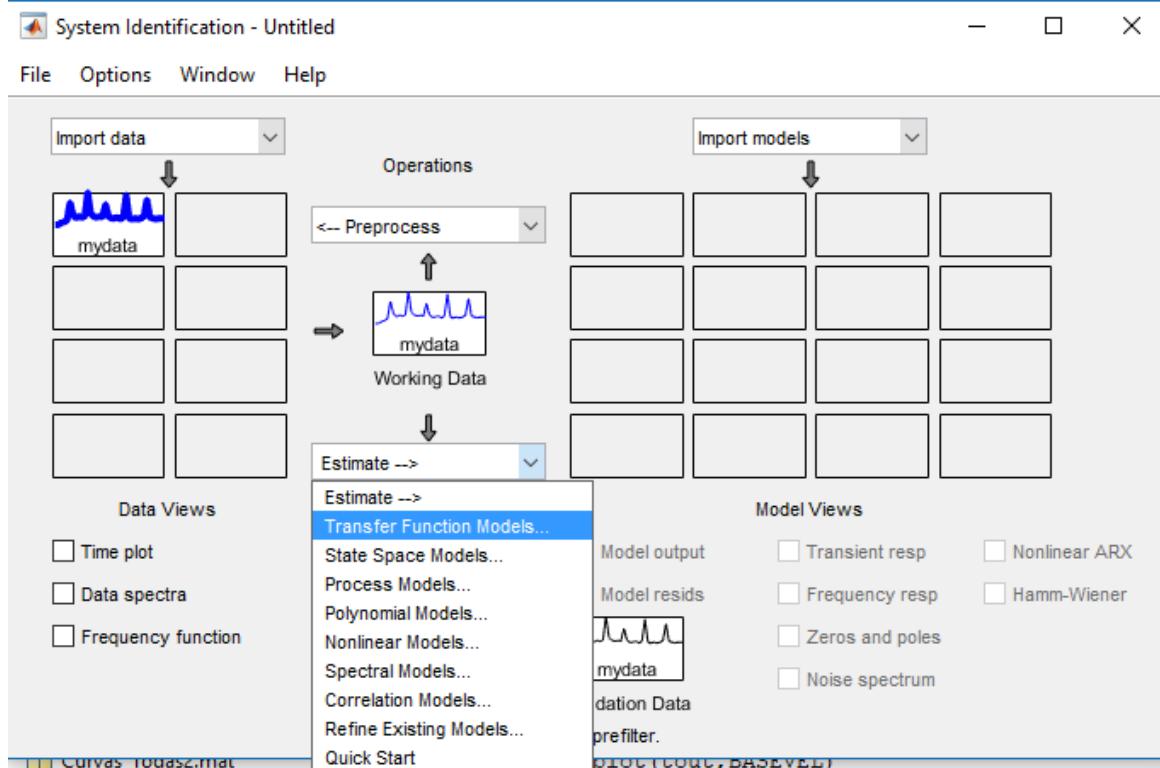
Fonte: Autores.

Figura 104 – Tratamento dos dados no SIT



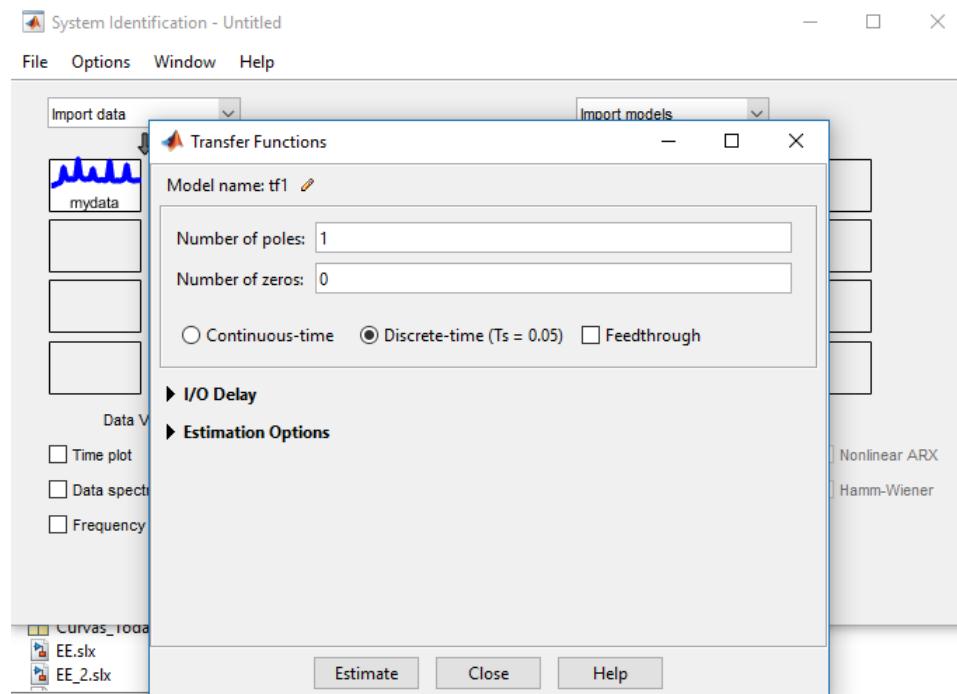
Fonte: Autores.

Figura 105 – Estimação de modelo do sistema no SIT



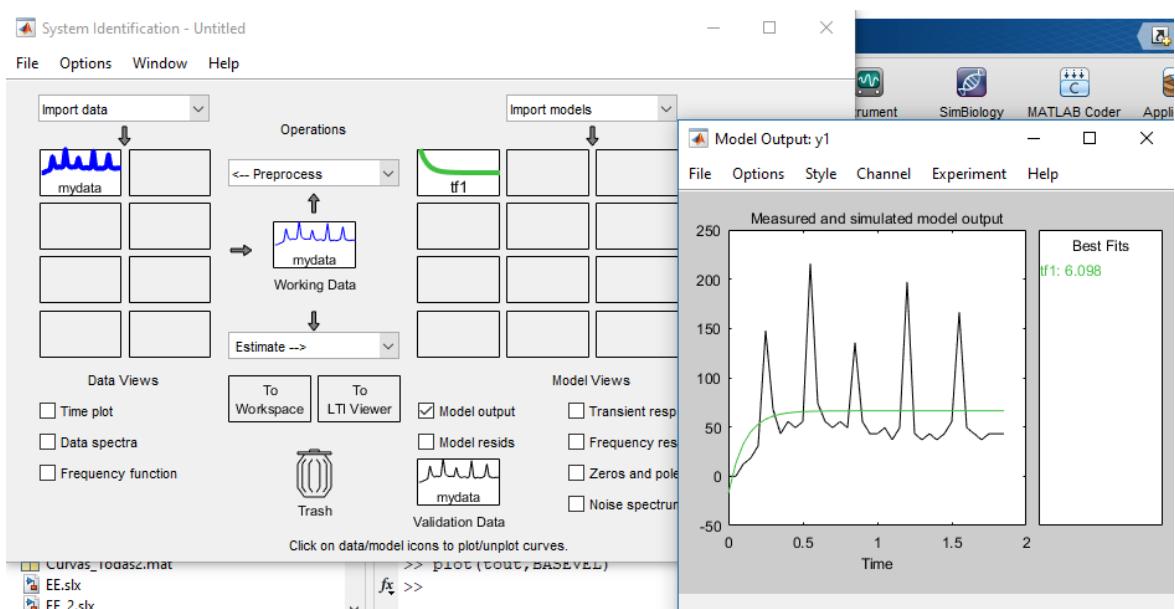
Fonte: Autores.

Figura 106 – Parâmetros de estimativa de Função de Transferência no SIT



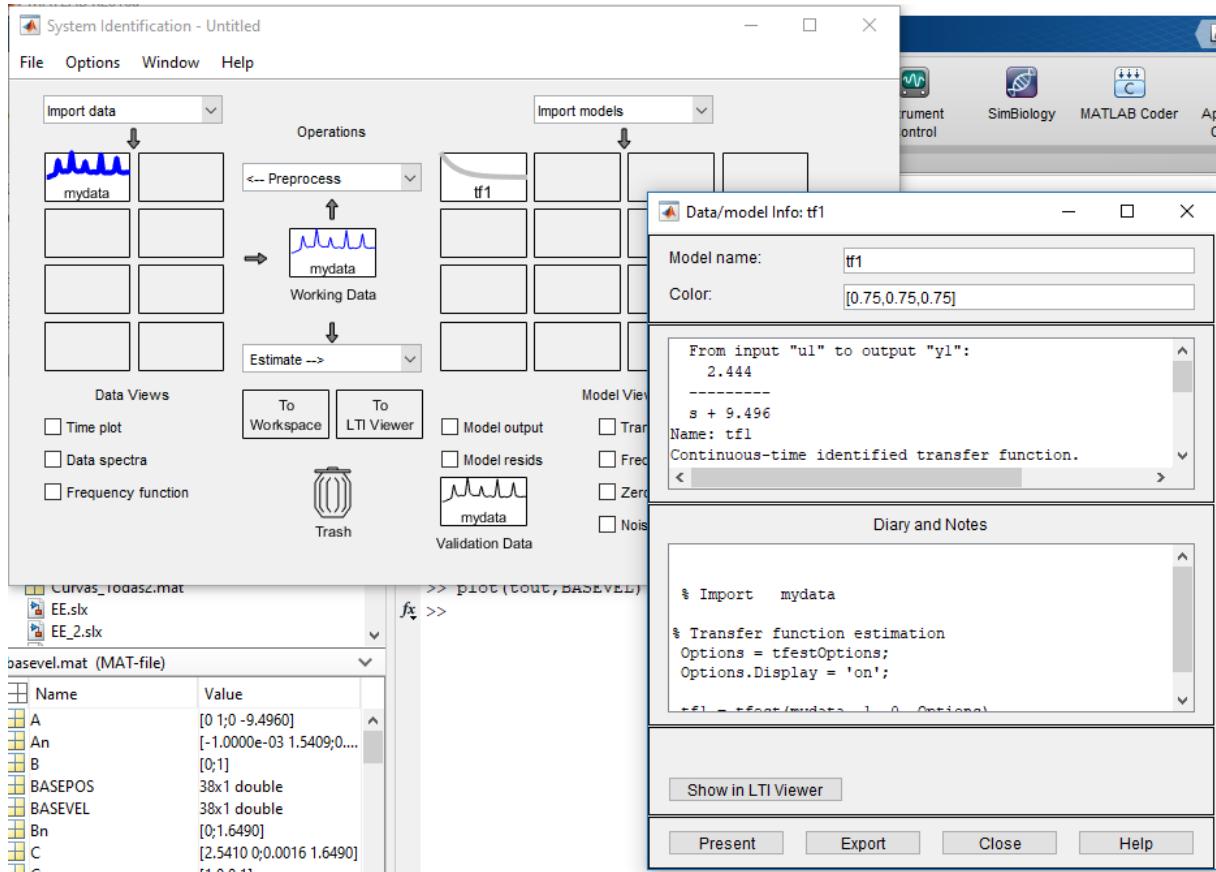
Fonte: Autores.

Figura 107 – Curva estimada pelo SIT



Fonte: Autores.

Figura 108 – Função de Transferência estimada pelo SIT



Fonte: Autores.

Clicando com o botão direito do *mouse* em cima da curva estimada são apresentadas suas informações, assim como o modelo estimado, como é mostrado na Figura 108.

No exemplo, a função de transferência estimada com um polo e nenhum zero foi:

$$G(s) = \frac{2,444}{s + 9,496}$$