

MURILO GOMES DE ANDRADE

PROJETO DE FIM DE CURSO II - FEMEC42100

**CONTROLE DE MESA CARTESIANA COM DOIS
GRAUS DE LIBERDADE**



UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA MECATRÔNICA

2020

MURILO GOMES DE ANDRADE

PROJETO DE FIM DE CURSO II - FEMEC42100

**CONTROLE DE MESA CARTESIANA COM DOIS GRAUS
DE LIBERDADE**

Trabalho de Conclusão de Curso apresentado
ao Curso de Graduação em Engenharia Mecatrônica da
Universidade Federal de Uberlândia, como parte dos
requisitos para a obtenção do título de **ENGENHEIRO
MECATRÔNICO**.

Orientador: Prof. Ms. Werley Rocherter Borges Ferreira

UBERLÂNDIA

2020

MURILO GOMES DE ANDRADE

PROJETO DE FIM DE CURSO II - FEMEC42100

**CONTROLE DE MESA CARTESIANA COM DOIS GRAUS
DE LIBERDADE**

Trabalho de Conclusão de Curso **APROVADO**
pelo Colegiado do Curso de Graduação em Engenharia
Mecatrônica da Faculdade de Engenharia Mecânica da
Universidade Federal de Uberlândia.

Banca Examinadora:

Prof. Ms. Werley Rocherter Borges Ferreira / (Orientador)

Prof. Dr. Edsonei Pereira Parreira

Prof. Dr. Valério Luiz Borges

UBERLÂNDIA

2020

*“Se fiz descobertas valiosas, foi
mais por ter paciência do que
qualquer outro talento.”*

(Isaac Newton)

AGRADECIMENTOS

Agradeço a Deus por me permitir alcançar meus objetivos, iluminando meu caminho, me dando força e coragem para sempre continuar.

A meus pais Clésio e Delma por terem me tornado quem sou e não medirem esforços para me acompanhar durante todos os passos da minha trajetória, sempre com dedicação, amor e orgulho.

Ao meu irmão Hiago, meu grande amigo, pelo companheirismo e apoio em tudo que precisei.

À minha namorada Camila por sempre me incentivar e estar comigo em todos os momentos, comemorando os bons e me amparando nos ruins.

Ao meu orientador, o professor Ms. Werley, pela paciência durante todo este trabalho e pela disposição em me auxiliar sempre.

A meus amigos e colegas de faculdade por terem feito parte desta luta, compartilhando esforços e dificuldades, assim como as horas de descontração.

À Universidade Federal de Uberlândia, todos os professores e profissionais que de alguma forma ajudaram na minha formação e me ajudaram a chegar até aqui, meu muito obrigado.

ANDRADE, M. G. **Controle de Mesa Cartesiana com Dois Graus de Liberdade**. 2020. 70p. Trabalho de Conclusão de Curso (Bacharel) – Curso de Graduação em Engenharia Mecatrônica, Universidade Federal de Uberlândia, Uberlândia, 2020.

RESUMO

Nos dias atuais, a indústria demanda constantemente por um aumento de produtividade, tanto em termos de velocidade de produção quanto de precisão e qualidade dos produtos. A produção manual vem se tornando cada vez mais incapaz de acompanhar tal necessidade, tornando necessária a aplicação de robôs industriais nas linhas de produção para garantir uma maior produtividade sem que se perca em qualidade. Visando suprir a demanda por tais equipamentos, diversas empresas passaram a produzi-los e disponibiliza-los no mercado, gerando uma ampla variedade de modelos disponíveis, cada um com sua aplicação, seus pontos fortes e pontos fracos. Muitas vezes, informações sobre tais características não se encontram muito acessíveis, ou estão até indisponíveis àqueles que necessitam das mesmas. Dito isto, este estudo constituiu-se de um levantamento, através de experimentação, das informações de atraso e de erro de um determinado equipamento presente no mercado, visando avaliar a fidelidade das trajetórias descritas pelo mesmo quando comparadas àquelas desejadas. Para isso, foram geradas no *software Matlab*[®] diversas trajetórias, utilizando os perfis mais comuns na robótica, e em seguida estas trajetórias foram fornecidas ao controlador, enquanto a resposta do mesmo era observada. Os resultados encontrados foram trajetórias descritas com deformações desprezíveis em relação ao esperado, divergindo apenas por conta de um tempo de atraso de 25 milissegundos entre ambas. Sendo assim, o controlador mostrou uma excelente qualidade, podendo ser aplicado em quase qualquer função na indústria pois irá descrever o percurso desejado com alta precisão.

Palavras-chave: Robótica, Trajetórias, Sistemas de Controle, Controle de Posição, Matlab[®], *Erro de Posicionamento, Tempo de Atraso.*

ANDRADE, M.G. **Two Degrees of Freedom Cartesian Robot Control**. 2020. 70p. Final Course Assignment (Bachelor) – Mechatronics Engineer Graduation, Universidade Federal de Uberlândia, Uberlândia, 2020.

ABSTRACT

Nowadays, industry constantly demands for increased productivity, both in terms of production speed and product accuracy and quality. Manual production is becoming increasingly unable to meet this need, making it necessary to apply industrial robots to production lines to ensure higher productivity without losing quality. In order to meet the demand for such equipment, several companies began to produce and make them available in the market, generating a wide variety of models available, each with its application, its strengths and weaknesses. Often information about such features is not very accessible, or even unavailable to those who need it. That said, this study consisted of a survey, through experimentation, of the delay and error of a particular equipment present in the market, aiming to evaluate the fidelity of the trajectories described by it when compared to those desired. For this, several paths were generated in the *Matlab*® software, using the most common profiles in robotics, and then these paths were provided to the controller, while the response was observed. The results were trajectories described with negligible deformations in relation to the expected, differing only because of a delay time of 25 milliseconds between them. Thus, the controller has shown excellent quality and can be applied to almost any function in the industry as it will describe the desired path with high accuracy.

Keywords: Robotics, Trajectories, Control Systems, Position Control, Matlab®, Positioning Error, Delay Time.

LISTA DE FIGURAS

Figura 2.1 – Esquema simplificado de um encoder translacional (Fonte: Site da Epilog Laser).	5
Figura 2.2 - Esquema de um encoder óptico do tipo tacômetro (ELLIOT, 2007).	6
Figura 2.3 – Esquema de um encoder óptico do tipo absoluto (ELLIOT, 2007).	7
Figura 2.4 – Padrão de código natural (DOEBELIN, 2004).	7
Figura 2.5 – Gray code (DOEBELIN, 2004).	7
Figura 2.6 – Servo-motor rotativo Yaskawa SigmaV (Fonte: Catálogo online da Yaskawa).	8
Figura 2.7 – Robô Cartesiano (Fonte: Catálogo online da Rexroth.)	9
Figura 3.1 – Diagrama de blocos do controlador de posição (Fonte: Manual de operação do driver.)	12
Figura 3.2 – Mesa cartesiana usada como objeto de estudo (Fonte: Elaborado pelo autor.)	13
Figura 3.3 – Gráfico com exemplo ilustrativo do tempo de atraso do sistema (Fonte: Elaborado pelo autor no software Matlab®.)	15
Figura 3.4 – Exemplo de trajetória com perfil de velocidade retangular (Fonte: Elaborado pelo autor no software Matlab®.)	16
Figura 3.5 – Exemplo de trajetória cúbica (Fonte: Elaborado pelo autor no software Matlab®.)	17
Figura 3.6 – Exemplo de trajetória de quinto grau (Fonte: Elaborado pelo autor no software Matlab®.)	18
Figura 3.7 – Exemplo de trajetória com perfil de velocidade trapezoidal (Fonte: Elaborado pelo autor no software Matlab®.)	20
Figura 4.1 – Leitura da velocidade na medição do tempo de atraso (Fonte: Elaborado pelo autor no software Matlab®.)	21
Figura 4.2 – Posição real vs posição desejada para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	23
Figura 4.3 – Erro de posicionamento no tempo para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	23
Figura 4.4 – Posição real vs posição desejada para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	24
Figura 4.5 – Erro de posicionamento no tempo para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	24

Figura 4.6 – Posição real vs posição desejada para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	25
Figura 4.7 – Erro de posicionamento no tempo para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	25
Figura 4.8 – Posição real vs posição desejada para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	27
Figura 4.9 – Erro de posicionamento no tempo para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	27
Figura 4.10 – Posição real vs posição desejada para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	28
Figura 4.11 – Erro de posicionamento no tempo para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	28
Figura 4.12 – Posição real vs posição desejada para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	29
Figura 4.13 – Erro de posicionamento no tempo para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	29
Figura 4.14 – Posição real vs posição desejada para trajetória cúbica a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	31
Figura 4.15 – Erro de posicionamento no tempo para trajetória cúbica a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	31
Figura 4.16 – Posição real vs posição desejada para trajetória cúbica a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	32
Figura 4.17 – Erro de posicionamento no tempo para trajetória cúbica a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	32
Figura 4.18 – Posição real vs posição desejada para trajetória cúbica a 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	33
Figura 4.19 – Erro de posicionamento no tempo para trajetória cúbica a 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	33
Figura 4.20 – Posição real vs posição desejada para trajetória de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	35
Figura 4.21 – Erro de posicionamento no tempo para trajetória de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	35
Figura 4.22 – Posição real vs posição desejada para trajetória de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	36
Figura 4.23 – Erro de posicionamento no tempo para trajetória de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	36

Figura 4.24 – Posição real vs posição desejada para trajetória de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	37
Figura 4.25 – Erro de posicionamento no tempo para trajetória de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	37
Figura 4.26 – Tempo de atraso para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	39
Figura 4.27 – Tempo de atraso para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	40
Figura 4.28 – Tempo de atraso para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	40
Figura 4.29 – Tempo de atraso para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	40
Figura 4.30 – Tempo de atraso para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	41
Figura 4.31 – Tempo de atraso para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	41
Figura 4.32 – Tempo de atraso para trajetória cúbica de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	42
Figura 4.33 – Tempo de atraso para trajetória cúbica de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	42
Figura 4.34 – Tempo de atraso para trajetória cúbica de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)	43
Figura 4.35 – Tempo de atraso para trajetória polinomial de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	43
Figura 4.36 – Tempo de atraso para trajetória polinomial de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	44
Figura 4.37 – Tempo de atraso para trajetória polinomial de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.).....	44

LISTA DE TABELAS

Tabela 4.1 – Parâmetros dos experimentos com perfil degrau (Fonte: Elaborado pelo autor.)	22
Tabela 4.2 – Erros observados para o perfil degrau (Fonte: Elaborado pelo autor.) ...	26
Tabela 4.3 – Parâmetros dos experimentos com perfil trapezoidal (Fonte: Elaborado pelo autor.)	26
Tabela 4.4 – Erros observados para o perfil trapezoidal (Fonte: Elaborado pelo autor.)	30
Tabela 4.5 – Parâmetros dos experimentos com trajetória cúbica (Fonte: Elaborado pelo autor.)	30
Tabela 4.6 – Erros observados para a trajetória cúbica (Fonte: Elaborado pelo autor.)	34
Tabela 4.7 – Parâmetros dos experimentos com trajetória de quinto grau (Fonte: Elaborado pelo autor.)	34
Tabela 4.8 – Erros observados para a trajetória de quinto grau (Fonte: Elaborado pelo autor.)	38
Tabela 4.9 – Médias dos tempos de atraso e constantes de tempo (Fonte: Elaborado pelo autor.)	45

SUMÁRIO

1. INTRODUÇÃO.....	1
1.1. Objetivos.....	2
1.2. Justificativa	2
1.3. Estrutura do Trabalho	3
2. FUNDAMENTAÇÃO TEÓRICA.....	4
2.1. Transdutores Digitais de Deslocamento (Encoders).....	4
2.2. Servo-motores	7
2.3. Mesas Cartesianas	8
3. CARACTERIZAÇÃO DOS EQUIPAMENTOS E METODOLOGIAS UTILIZADAS	10
3.1. Equipamentos Utilizados.....	10
3.1.1. Placa de Aquisição	10
3.1.2. Servo-motor	11
3.1.3. Encoder externo	11
3.1.4. Drivers de potência	11
3.1.5. Mesa Cartesiana.....	13
3.2. Metodologia	14
3.2.1. Parametrização dos drivers	14
3.2.2. Geração e aquisição de dados	14
3.2.3. Medição do atraso do sistema	14
3.2.4. Cálculo de Trajetórias	15
3.2.5. Medição do Erro de Posicionamento	20
4. RESULTADOS E DISCUSSÕES.....	21
4.1. Tempo de Atraso.....	21
4.2. Erros de Posicionamento	22
4.2.1. Perfil de Velocidade Degrau	22
4.2.2. Perfil de Velocidade Trapezoidal.....	26

4.2.3. Trajetória Cúbica	30
4.2.4. Trajetória Polinomial de Quinto Grau	34
4.3. Discussão dos Resultados	38
4.3.1. Tempo de Atraso	38
4.3.2. Erros de posicionamento	38
5. CONCLUSÕES	47
6. REFERÊNCIAS BIBLIOGRÁFICAS	48
APÊNDICE A – FUNÇÃO PARA CALCULAR TRAJETÓRIA DEGRAU DESENVOLVIDA NO <i>MATLAB</i>®	49
APÊNDICE B – FUNÇÃO PARA CALCULAR TRAJETÓRIA TRAPEZOIDAL DESENVOLVIDA NO <i>MATLAB</i>®	50
APÊNDICE C – FUNÇÃO PARA CALCULAR TRAJETÓRIA CÚBICA DESENVOLVIDA NO <i>MATLAB</i>®	51
APÊNDICE D – FUNÇÃO PARA CALCULAR TRAJETÓRIA DE QUINTO GRAU DESENVOLVIDA NO <i>MATLAB</i>®	52
APÊNDICE E – FUNÇÃO PARA CONVERTER TRAJETÓRIA CALCULADA EM SINAL DESENVOLVIDA NO <i>MATLAB</i>®.....	53
APÊNDICE F – <i>SCRIPT</i> PARA FORNECER SINAL E ADQUIRIR RESPOSTA PARA TRAJETÓRIA DEGRAU DESENVOLVIDA NO <i>MATLAB</i>®	54
APÊNDICE G – <i>SCRIPT</i> PARA FORNECER SINAL E ADQUIRIR RESPOSTA PARA TRAJETÓRIA TRAPEZOIDAL DESENVOLVIDA NO <i>MATLAB</i>®.....	55
APÊNDICE H – <i>SCRIPT</i> PARA FORNECER SINAL E ADQUIRIR RESPOSTA PARA TRAJETÓRIA CÚBICA DESENVOLVIDA NO <i>MATLAB</i>®	56
APÊNDICE I – <i>SCRIPT</i> PARA FORNECER SINAL E ADQUIRIR RESPOSTA PARA TRAJETÓRIA DE QUINTA ORDEM DESENVOLVIDA NO <i>MATLAB</i>®	57

CAPÍTULO I

INTRODUÇÃO

Atualmente, os robôs se encontram cada vez mais presentes na vida das pessoas. Seja para uso doméstico como em robôs aspiradores de pó, em carros autônomos, em uso militar como no caso de drones, na indústria, na medicina com os robôs cirurgiões, ou até mesmo na exploração espacial com as sondas em marte, por exemplo. Não há como negar que a robotização é uma tendência cada vez mais crescente no nosso dia-a-dia.

Sua maior utilização, no entanto, fica reservada à indústria. No cenário econômico contemporâneo, temos a quarta revolução industrial onde deseja-se implementar as maiores inovações tecnológicas dos campos de tecnologia da informação, automação e controle na indústria.

Neste novo modelo, temos a robotização dos meios de produção como peça chave. Os robôs industriais são capazes de trabalhar de forma praticamente ininterrupta, em menores espaços, com maior velocidade e precisão que os seres humanos em tarefas repetitivas e até mesmo atuar em áreas onde haveria riscos para a saúde de um trabalhador humano. Da perspectiva dos detentores de meios de produção, o emprego de robôs se apresenta bastante vantajoso, permitindo que os processos se tornem muito mais autônomos, eficientes e rapidamente adaptáveis.

Encontra-se hoje no mercado uma ampla gama de marcas e modelos para estes fins, cada um recomendado para uma determinada tarefa de acordo com suas características, sendo que para a realização destas tarefas é necessário um mínimo de confiabilidade em questões de posicionamento e velocidade ao longo de suas

trajetórias, confiabilidade esta que deve ser informada pelos fabricantes para cada modelo.

Diante destes fatos, neste trabalho será realizado um estudo de caso a fim de averiguar as margens de erro apresentadas por um certo modelo de controlador, uma vez que estas informações não são fornecidas pelo fabricante do equipamento.

1.1. Objetivos

O objetivo geral deste trabalho é mensurar os erros apresentados pela mesa cartesiana em questão, em função dos diferentes perfis de trajetória fornecidos a ela.

Os objetivos específicos são:

- Analisar os drivers e motores comerciais, isto é, aprender o funcionamento e suas limitações;
- Obter o tempo de resposta do sistema para uma determinada entrada;
- Verificar se os erros máximos de posicionamento dos motores estão de acordo com os informados pelo fabricante;
- Controle da mesa cartesiana para trajetórias simples.

1.2. Justificativa

Algumas atividades desempenhadas por robôs na indústria exigem que estes sigam as trajetórias designadas a eles com uma certa fidelidade, ou seja, apresentando uma quantidade de erro máximo tolerável. Alguns controladores de robôs comerciais são mais sofisticados que outros, chegando a apresentar erros percentuais extremamente pequenos seguindo, desta forma, as trajetórias designadas com altíssima precisão.

Tendo em vista que este nível de erro é uma informação crucial em tarefas delicadas, neste trabalho será averiguado através da experimentação com um modelo de controlador disponível no mercado, os erros percentuais máximos apresentados por este quando sujeito aos tipos de trajetória mais comuns na robótica e ainda, verificar se as informações fornecidas no manual deste equipamento são realmente confiáveis.

1.3. Estrutura do Trabalho

Este trabalho está estruturado em 5 capítulos, além deste. O **Capítulo 2** apresenta uma revisão bibliográfica sobre mesas cartesianas e os principais componentes que as caracterizam, assim como suas principais características.

O **Capítulo 3** mostra o as descrições e principais características dos equipamentos e ferramentas utilizados no projeto, juntamente com a metodologia aplicada no projeto.

O **Capítulo 4** os resultados obtidos pelo autor e suas respectivas discussões, no que diz respeito à análise de trajetórias e tempos de resposta apresentados pelo sistema.

O **Capítulo 5** apresenta as conclusões finais do trabalho.

As **Referências Bibliográficas** apresentam a relação de todas as obras referenciadas no trabalho.

CAPÍTULO II

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados os conceitos essenciais para o entendimento deste trabalho, sendo eles o tipo de robô em que o sistema a ser estudado se enquadra e os componentes que constituem o mesmo.

2.1. Transdutores Digitais de Deslocamento (Encoders)

No estudo e aplicação de sistemas de controle, a aquisição de dados tem papel fundamental. No caso de sistemas de controle de posição, os encoders são os dispositivos mais comumente utilizados para tal fim.

Encoders são transdutores capazes de transformar uma medida de posição, seja ela linear ou angular, em uma informação digital a ser fornecida ao sistema de controle. Podem ser classificados quanto a forma do movimento que eles leem, ao tipo de sensor que empregam e ao seu princípio de funcionamento.

Quanto a forma do movimento, eles podem ser divididos em rotacionais e translacionais. Encoders do tipo rotacional são os mais utilizados. Estes são capazes de medir posição, deslocamento relativo ou velocidade de dispositivos rotativos. São constituídos, geralmente, por um disco que apresenta um código de posição em sua superfície e gira em conjunto com o eixo de um motor, enquanto um ou mais sensores fixos à carcaça efetuam a leitura deste mesmo código.

Os do tipo translacional, assim como aqueles rotacionais, são capazes de medir posição, deslocamento e velocidade, mas apenas em uma direção linear. Apresentam princípio de funcionamento similar ao dos rotacionais, mas geralmente são constituídos de uma fita com o código e um sensor que se desloca ao longo desta, junto a um componente móvel.

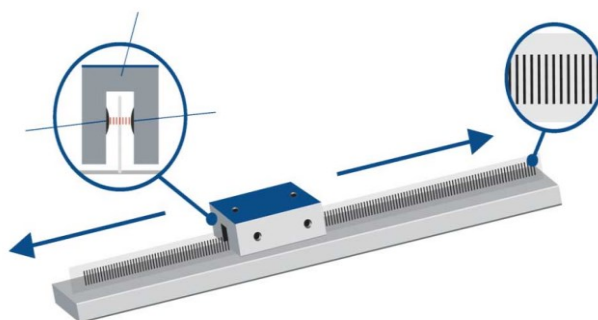


Figura 2.1 – Esquema simplificado de um encoder translacional (Fonte: Site da *Epilog Laser*¹).

Quanto ao sensor empregado, o encoder pode ser do tipo magnético, capacitivo, óptico, por contato, dentre outros. O funcionamento de um *encoder* por contato, por exemplo, consiste em contatos condutores que deslizam sobre uma superfície com partes condutoras e isolantes, formando um código. Através da configuração das leituras do conjunto de contatos, é possível gerar um sinal lógico, com o qual é determinada sua velocidade, posição ou deslocamento relativo (DOEBELIN, 2004).

O tipo de *encoder* mais comum no mercado é o óptico, este consiste em uma fonte de luz emitindo em direção a um sensor óptico, havendo entre os dois um corpo ranhurado. Este corpo ranhurado é um objeto de material opaco com certos pontos vazados ou refletivos que podem ser diferenciados pelo sensor, sendo que a disposição de tais pontos configura um determinado código que é então lido pelo sensor e interpretado determinando, assim, as grandezas desejadas.

Encoders ópticos são os que apresentam melhor resolução, sendo que existem instrumentos comerciais deste tipo capazes de distinguir arcos de fração de segundo. Utilizando-se uma fonte de luz de semiconductor, podem ser alcançados encoders de 81 mil pulsos por revolução em um diâmetro de apenas 36mm e, usando dispositivos eletrônicos externos para interpolar os valores de saída, a resolução pode atingir cerca de 10 milhões de pulsos por revolução (DOEBELIN, 2004).

Baseando-se no princípio de funcionamento, existem três tipos básicos de *encoder*, sendo eles: tacômetro, incremental e absoluto.

¹ Disponível em: <https://www.epiloglaser.com/laser-machines/legend-laser/legend-linear-encoders.htm>; acessado em maio de 2018.

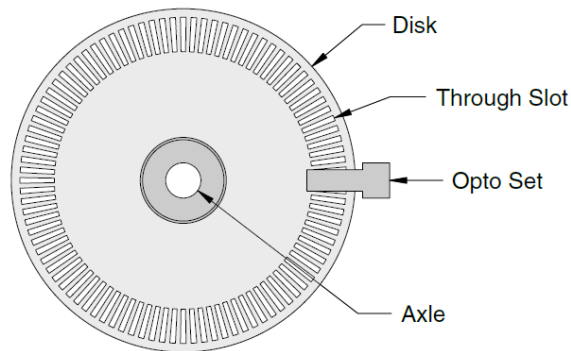


Figura 2.2 - Esquema de um encoder óptico do tipo tacômetro (ELLIOT, 2007).

O tacômetro apresenta somente um sinal de saída digital que consiste em um pulso para cada incremento detectado no deslocamento. Se o movimento for sempre unidirecional, um contador digital pode acumular estes pulsos para determinar o deslocamento a partir de um ponto inicial conhecido. Contudo, qualquer movimento contrário produziria os mesmos pulsos, causando erros na medição. Sendo assim, este tipo de *encoder* é geralmente utilizado na medição de velocidade ou de deslocamento angular unidirecional (DOEBELIN, 2004).

O *encoder* incremental resolve o problema do movimento inverso utilizando pelo menos dois elementos geradores de sinal. Defasando as duas faixas entre si em um quarto de ciclo, torna-se possível conhecer a direção do movimento com base na ordem de com que os pulsos se alteram, assim o contador pode subtrair pulsos quando houver movimento inverso. Uma terceira faixa possuindo apenas um pulso por revolução em determinado ponto pode ser acrescentada, servindo como uma referência de zero. Um *encoder* incremental possui a vantagem de ser capaz de rotacionar por quantas revoluções forem necessárias. Contudo, qualquer ruído apresentado no sistema pode ser contado como um pulso e persistirá mesmo quando o ruído terminar. A falta de energia também seria causa de problemas, pois com ela os dados de posição são perdidos, sem que seja possível recuperá-los quando a mesma for restaurada (DOEBELIN, 2004).

Encoders absolutos são geralmente limitados a uma única revolução e utilizam múltiplas faixas e saídas, que são lidas em paralelo para produzirem uma representação binária da posição angular do eixo. Como cada representação binária é única e específica para uma certa posição, os dados podem ser recuperados depois de uma interrupção na energia do sistema, assim como qualquer ruído elétrico causará erros de leitura apenas enquanto o ruído persistir. A Figura 2.3 mostra o esquema de um encoder absoluto (DOEBELIN, 2004).

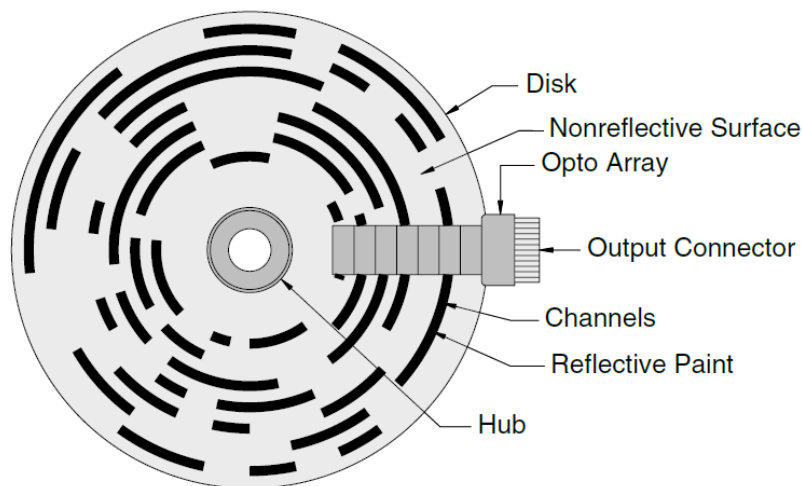


Figura 2.3 – Esquema de um encoder óptico do tipo absoluto (ELLIOT, 2007).

O padrão de código natural, como o padrão de 4 *bits* mostrado na Figura 2.4, é o mais conveniente para a visualização e descrição do movimento por associá-lo a um sistema binário de numeração, contudo, diversos *encoders* comerciais utilizam padrões diferentes, como o *Gray code* da Figura 2.5, onde apenas um *bit* muda a cada transição, evitando a possível ocorrência de erros devido a pequenos desalinhamentos dos trechos de código no sistema real.

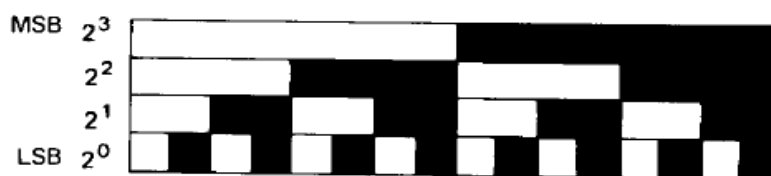


Figura 2.4 – Padrão de código natural (DOEBELIN, 2004).



Figura 2.5 – Gray code (DOEBELIN, 2004).

2.2. Servo-motores

Os servo-motores são caracterizados por um motor elétrico no qual já existe um sistema de controle em malha fechada acoplado a ele, podendo ser controlado quanto à sua posição, à velocidade ou ao torque apresentado, sendo a melhor opção para

posição (ELLIOTT, 2007). Geralmente, recebem como entrada a sua alimentação além de apresentarem alguma forma de saída de dados.



Figura 2.6 – Servo-motor rotativo *Yaskawa SigmaV* (Fonte: Catálogo online da *Yaskawa*²).

Devido à sua capacidade de lidar com a presença de cargas altas ou variáveis e de rápidas acelerações (para as quais são necessários um torque inicial alto e pequena inércia), assim como das limitações quanto ao espaço e o peso do sistema, geralmente os servo-motores são os mais utilizados em sistemas de controle (THALER e BROWN, 1960).

O sistema de controle é constituído por um motor elétrico que pode ser tanto de corrente alternada quanto contínua, um sensor que pode ser um *encoder* ou potenciômetro e um circuito que determinará a tensão e a corrente da alimentação do motor a partir da tensão de entrada e da leitura do sensor. Desta forma, é esperado um torque de saída proporcional à corrente do sinal de controle aplicado e que o sentido do torque seja determinado pela polaridade da mesma.

Servo-motores de corrente contínua são geralmente mais leves e apresentam maiores torques iniciais e inversos, enquanto aqueles de corrente alternada são mais simples, confiáveis, econômicos e apresentam resposta rápida, sendo também bastante utilizados em servomecanismos (KOSOW, 1985).

2.3. Mesas Cartesianas

Robôs manipuladores são dispositivos que apresentam atuadores em sua extremidade que, junto a sua capacidade de movimentação são capazes de executar

² Disponível em <https://catalogo.yaskawa.com.br/viewitems/servo-motores-rotativos-1/servomotor-rotativo-sgmav>; acessado em outubro de 2018.

tarefas pré-programadas e repetitivas de forma mais rápida, confiável e padronizada do que aquelas feitas por seres humanos. Esses robôs podem ser encontrados em diversas formas de estruturas, como de coordenadas cartesianas, cilíndricas, esféricas, estrutura paralela, entre outras.

Uma mesa cartesiana é um robô de coordenadas cartesianas e três graus de liberdade, ou seja, ele pode gerar movimentos pela composição de três translações, coincidentes com os eixos do sistema cartesiano de coordenadas, apresentando assim um espaço de trabalho em forma de paralelepípedo (ROMANO e DUTRA, 2002).

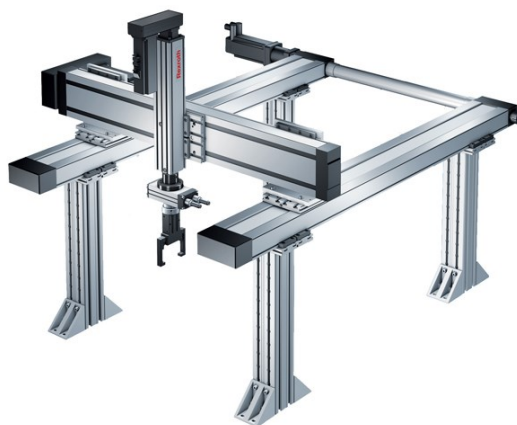


Figura 2.7 – Robô Cartesiano (Fonte: Catálogo *online* da *Rexroth*. ³)

Pode utilizar atuadores baseados princípios em atuadores eletropneumáticos, eletrohidráulicos ou eletromecânicos. No caso de atuadores eletropneumáticos ou eletrohidráulicos, o movimento é realizado pela abertura e fechamento de válvulas que acionam pistões ocasionando o deslocamento dos segmentos do robô, enquanto nos robôs com acionamento eletromecânico, os atuadores são motores elétricos ligados aos componentes móveis por correias ou por fusos conectados aos segmentos.

Este tipo de robô é amplamente utilizado em automação industrial, como em tarefas simples tais como movimentação, furação e pintura de componentes, desde que mudanças de orientação do elemento terminal não sejam necessárias.

³ Disponível em: <https://www.boschrexroth.com/en/us/products/product-groups/linear-motion-technology/linear-motion-systems/index> ; acessado em novembro de 2018.

CAPÍTULO III

CARACTERIZAÇÃO DOS EQUIPAMENTOS E METODOLOGIAS UTILIZADAS

Neste capítulo serão apresentados os equipamentos a que o trabalho se refere e as metodologias aplicadas no desenvolvimento do mesmo.

3.1. Equipamentos Utilizados

No desenvolvimento deste trabalho foram utilizados um computador pessoal com processador *Intel*[®] *Core*[™] *I7* de 1^a geração e 8GB de memória, com o *software MATLAB*[®] *instalado*, uma placa de geração e aquisição de dados e um conjunto de servo-motores e servo-drives de potência, os quais serão descritos mais detalhadamente a seguir.

3.1.1. Placa de Aquisição

Foi utilizada uma placa de aquisição de dados da marca *National Instruments*[®], modelo USB-6351. Tal placa conta com uma taxa de aquisição de até 1,25 milhões de amostras por segundo, apresentando 16 entradas e 2 saídas analógicas, 24 entradas e saídas digitais assim como 4 canais do tipo contador, capazes de realizar contagens de número de pulsos de *encoders*, assim como leituras de frequência e largura de pulso.

3.1.2. Servo-motor

Foram utilizados dois servo-motores industriais da marca *Yaskawa*[®], *modelo* SIGMAV-02A3A6C, disponíveis comercialmente.

Este modelo de motor apresenta potência de 200W com 0,637Nm de torque a até 3000rpm, ou até 6000rpm com redução de torque, contando com um *encoder* absoluto de 20 *bits* embutido.

Em conjunto com cada motor foi utilizado seu respectivo servo-driver. A estes, cabe um capítulo especial devido a todos os detalhes de sua configuração e operação.

3.1.3. Encoder externo

A fim de testar a confiabilidade da contagem de pulsos fornecida pelo driver, foi utilizado um encoder externo do tipo o incremental de 1024 pulsos e cujas leituras serão comparadas com as do driver.

3.1.4. Drivers de potência

Os drivers utilizados para operação dos servo-motores também são da linha SIGMAV da *Yaskawa*. Esses drivers apresentam parâmetros que devem ser configurados a fim de se obter um funcionamento correto do sistema. Dentre os parâmetros aceitos pelo driver, destacam-se os parâmetros básicos a seguir:

- Tipo de controle
- “Servo on”
- Sentido de movimento
- Fim de curso
- Relação de transmissão eletrônica

O tipo de controle permite selecionar qual grandeza deve ser controlada. Pode ser definido para controle de velocidade por tensão de referência, controle de posição por referência a trem de pulsos, controle do torque por tensão de referência, controle de velocidade por definição interna, ou até mesmo uma forma de controle que combine as quatro anteriores.

Para o funcionamento da mesa cartesiana, objetivo do trabalho, interessa apenas o modo de controle de posição, que utiliza como referência um sinal de trem de pulsos.

O principal parâmetro a ser definido neste modo é a forma do sinal de referência. Esta pode ser escolhida dentre cinco opções:

- Um pino que recebe um trem de pulsos enquanto outro pino recebe um sinal que determina a orientação do movimento (nível lógico alto para rotação direta e baixo para inversa).
- Um pino que recebe um trem de pulsos para rotação direta e outro para inversa. Neste modo, enquanto houver sinal para um dos sentidos, o do outro permanece em nível lógico baixo.
- Dois trens de pulsos com defasagem de 90° entre si, podendo ser interpretados com codificação x1, x2 ou x4 (referente ao número de incrementos ou decrementos por ciclo da quadratura).

Para as duas primeiras formas existem duas formas de trem de pulsos diferentes. Na forma positiva o sinal é normalmente em nível lógico baixo com pulso em nível lógico alto enquanto na negativa ocorre o contrário.

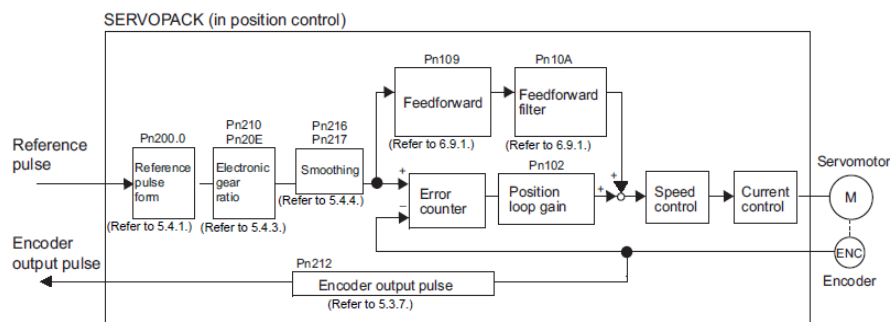


Figura 3.1 – Diagrama de blocos do controlador de posição (Fonte: Manual de operação do driver.)

O “servo on” determina se a alimentação do servo-motor está ligada ou desligada. Pode ser definido para receber um sinal externo, que pode ser configurado para algum sensor, botão ou outra forma de comando, ou definido como uma constante para que não dependa de nenhum comando extra do operador.

O sentido de rotação do motor pode ser definido através de um parâmetro para que este seja invertido sem alterar a polaridade do sinal de referência.

O parâmetro “fim de curso” permite que o sistema aguarde ou não pelo o sinal de um sensor do tipo fim de curso, sinal este que força as partes móveis a pararem no caso de atingirem o limite de espaço permitido. Pode bloquear tanto o movimento direto quanto o movimento inverso através de sinais distintos.

A fim de simplificar o cálculo do trem de pulsos de referência, o driver permite a parametrização de uma relação de transmissão de forma que possa ser fornecido ao

motor um sinal calculado em distância que será convertido em número de pulsos ao motor. A relação de transmissão se dá pela seguinte fórmula:

$$\frac{A}{B} = \frac{R}{d} * \frac{m}{n} \quad (1)$$

Onde,

A e B são os parâmetros Pn20E e Pn210 a serem configurados,

R é a resolução do encoder em número de pulsos,

d é o passo do eixo por revolução em unidades de referência (μm, neste caso),

$\frac{m}{n}$ é a relação de transmissão entre o motor e o eixo (1:1 neste caso).

Utilizando-se desta forma, é possível que o usuário forneça as trajetórias a serem seguidas com medidas reais, deixando a cargo do driver efetuar a conversão entre medidas reais e número de pulsos, tanto para a entrada quanto para a saída de dados, eliminando uma parte de suas responsabilidades e, com isso, facilitando a utilização do sistema.

3.1.5. Mesa Cartesiana

Utilizando os materiais descritos anteriormente em adição à alguns componentes mecânicos que incluem uma estrutura metálica que serve como base do robô, trilhos onde se apoiam os suportes dos motores e fusos de esferas que servem para transmitir o movimento dos motores, foi construída no laboratório a mesa cartesiana a ser utilizada no decorrer deste trabalho. O resultado final da construção deste robô pode ser observado na Figura 3.2.



Figura 3.2 – Mesa cartesiana usada como objeto de estudo (Fonte: Elaborado pelo autor.)

3.2. Metodologia

3.2.1. Parametrização dos drivers

Os drivers foram configurados para o modo de controle de posição, recebendo em um pino um trem de pulsos com lógica positiva e em outro, um sinal de orientação. Foi definido também um sinal de *servo on* conectado a uma chave de liga/desliga para ativar os motores e a relação de transmissão foi definida como 1:1, a fim de se ter um controle mais fino do posicionamento dos motores e manter o detalhamento dos dados obtidos e devendo ser convertidos no programa de geração e aquisição de dados.

3.2.2. Geração e aquisição de dados

Para a geração e aquisição de dados foi utilizado o *software Matlab®*. É utilizado um modelo de sessões onde os dados são calculados previamente, então quando a sessão é iniciada, são transmitidos à placa que fornece os sinais aos controladores dos motores enquanto lê os *encoders* pelos canais definidos.

Em todos os experimentos, foram utilizados programas com a mesma estrutura onde primeiramente a trajetória é calculada na unidade de medida real e com a forma desejada, em seguida é calculado o número de pulsos necessários para se obter os valores na unidade de medida escolhida para que então esse número de pulsos seja convertido num trem de pulsos na configuração esperada pelo driver do motor. Tais programas encontram-se nos Apêndices de A até I deste trabalho.

3.2.3. Medição do atraso do sistema

Como mencionado nos objetivos deste trabalho, deseja-se conhecer o tempo de atraso de transporte do nosso sistema. Este atraso é constituído pelo tempo que leva desde que o sinal é gerado, passando pelo processamento do mesmo por parte do controlador, até o momento em que é recebido pelo sistema e começa a causar um efeito que possa ser observado pelos sensores.

Para efetuar a medição deste intervalo de tempo deve ser enviado um sinal para o motor que está girando a uma certa velocidade constante, para que este aumente a sua velocidade a partir de um momento preestabelecido e então é observado o sinal do *encoder*. A diferença de tempo entre o momento em que o movimento deveria começar

e aquele em que foi observada a primeira variação na velocidade do motor é o tempo de atraso do sistema.

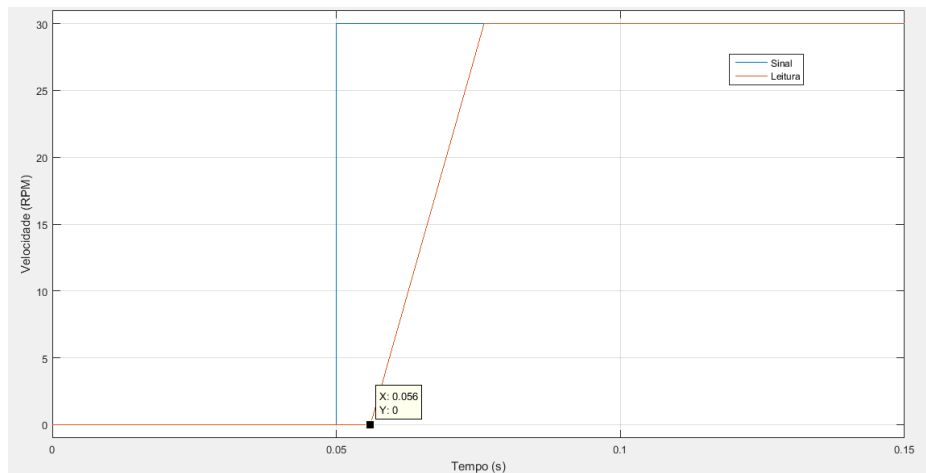


Figura 3.3 – Gráfico com exemplo ilustrativo do tempo de atraso do sistema (Fonte: Elaborado pelo autor no software *Matlab*®.)

No exemplo da Figura 3.2 o sinal fornecido apresenta variação inicial de velocidade em 0.05 segundos, contudo, só foi observada variação real na velocidade em 0.056 segundos, resultando em um tempo de atraso de 6 milissegundos.

3.2.4. Cálculo de Trajetórias

Com base em alguns perfis de posição, velocidade e aceleração comuns na robótica e na automação industrial, foram desenvolvidas algumas trajetórias a serem seguidas pelo robô cartesiano e analisados os erros de posicionamento apresentados no processo.

Algumas dessas trajetórias estão apresentadas a seguir:

- **Trajetória Linear (Perfil de Velocidade Degrau)**

Esta é a forma mais simples de trajetória polinomial, constituída por um movimento com velocidade máxima ao longo do tempo. Esta trajetória é definida pela equação a seguir:

$$q(t) = bt + a \quad (2)$$

Onde:

$$a = q_0 \quad (3)$$

$$b = \frac{q_f - q_0}{t_f} \quad (4)$$

Portanto,

$$q(t) = \frac{q_f - q_0}{t_f} t + q_0 = Vt + q_0 \quad (5)$$

Sendo que:

q_0 é a posição inicial,

q_f é a posição final,

t_f é o tempo final,

V é a velocidade constante do movimento

Desta forma, o perfil de posição desejado tem a forma mostrada na Figura 3.3.

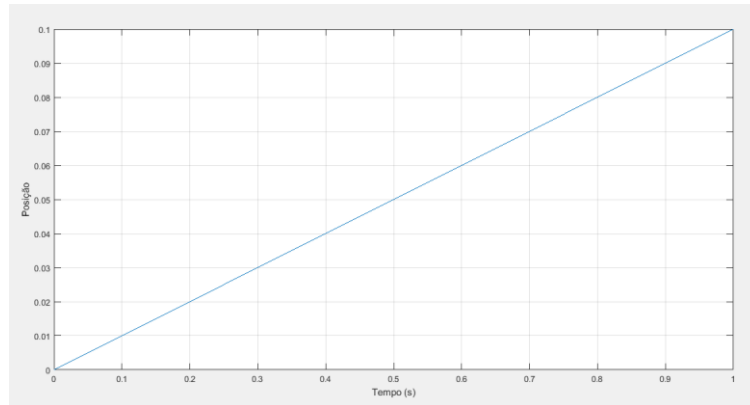


Figura 3.4 – Exemplo de trajetória com perfil de velocidade retangular (Fonte: Elaborado pelo autor no software *Matlab*®.)

- **Trajétória Cúbica (Perfil de Velocidade Parabólico)**

A fim de se evitar descontinuidades na aceleração, podem ser implementados polinômios de ordem superior. Trajetórias cúbica apresentam perfis de velocidade parabólicos e, portanto, acelerações lineares. Estas são descritas por equações da seguinte forma:

$$q(t) = dt^3 + ct^2 + bt + a \quad (6)$$

Onde, pelas restrições de posição e velocidade inicial e final nulas, temos que:

$$a = q_0 \quad (7)$$

$$b = \dot{q}_0 = 0 \quad (8)$$

$$c = \frac{3}{t_f^2}(q_f - q_0) - \frac{2}{t_f}\dot{q}_0 - \frac{1}{t_f}\dot{q}_f = \frac{3}{t_f^2}(q_f - q_0) \quad (9)$$

$$d = -\frac{2}{t_f^3}(q_f - q_0) + \frac{1}{t_f^2}(\dot{q}_f + \dot{q}_0) = -\frac{2}{t_f^3}(q_f - q_0) \quad (10)$$

Sendo que:

q_0 é a posição inicial,

q_f é a posição final,

$\dot{q}_0$ é a velocidade inicial,

$\dot{q}_f$ é a velocidade final,

t_f é o tempo final.

Desta forma, o perfil de posição desejado tem a forma da Figura 3.4.

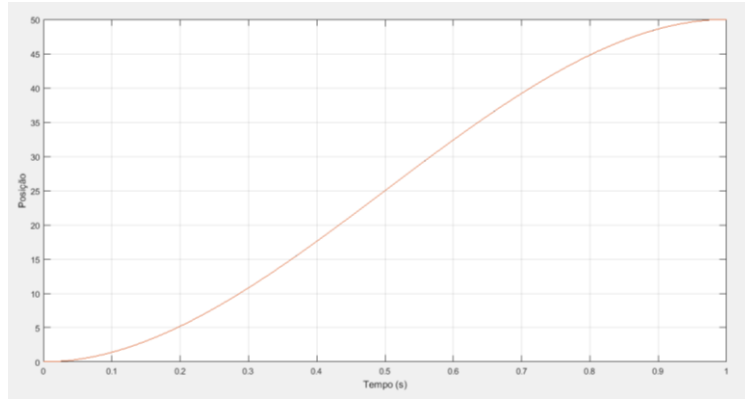


Figura 3.5 – Exemplo de trajetória cúbica (Fonte: Elaborado pelo autor no *software Matlab®*).

• Trajetória Polinomial de Quinto Grau

As trajetórias polinomiais de quinto grau são mais suaves que trajetórias cúbicas, além de apresentarem menor aceleração em momentos em que a velocidade é alta, não atingindo o torque máximo por muito tempo, possibilitando que haja uma sobra de torque para o controlador. Tais trajetórias são caracterizadas pela equação a seguir:

$$q(t) = ft^5 + et^4 + dt^3 + ct^2 + bt + a \quad (11)$$

Onde, pelas restrições de posição e velocidade inicial e final nulas, temos que:

$$a = q_0 \quad (12)$$

$$b = \dot{q}_0 = 0 \quad (13)$$

$$c = \frac{\ddot{q}_0}{2} = 0 \quad (14)$$

$$d = \frac{t_f \left((\ddot{q}_f - 3\ddot{q}_0)t_f - 8\dot{q}_f - 12\dot{q}_0 \right) + 12(q_f - q_0)}{2t_f^5} = \frac{12(q_f - q_0)}{2t_f^5} \quad (15)$$

$$e = \frac{t_f (16\dot{q}_0 + 14\dot{q}_f + (3\ddot{q}_0 - 2\ddot{q}_f)t_f) + 30(q_0 - q_f)}{2t_f^4} = \frac{30(q_0 - q_f)}{2t_f^4} \quad (16)$$

$$f = \frac{t_f \left((\ddot{q}_f - \ddot{q}_0)t_f - 6(\dot{q}_f + \dot{q}_0) \right) + 20(q_f - q_0)}{2t_f^3} = \frac{20(q_f - q_0)}{2t_f^3} \quad (17)$$

Sendo que:

q_0 é a posição inicial,

q_f é a posição final,

$\dot{q}_0$ é a velocidade inicial,

$\dot{q}_f$ é a velocidade final,

$\ddot{q}_0$ é a aceleração inicial,

$\ddot{q}_f$ é a aceleração final,

t_f é o tempo final.

Desta forma, o perfil de posição desejado tem a forma da Figura 3.5.

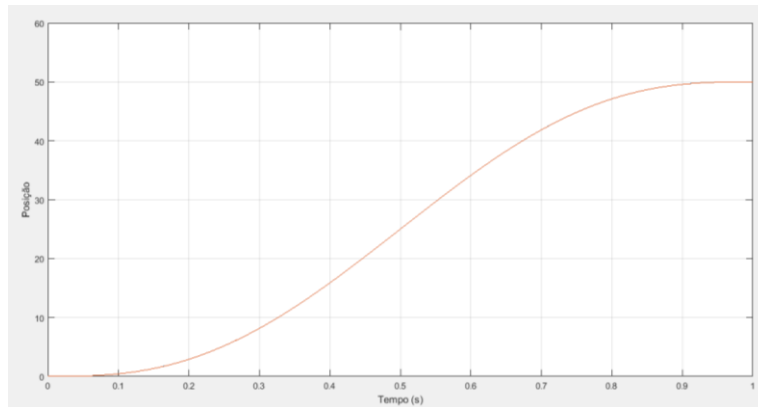


Figura 3.6 – Exemplo de trajetória de quinto grau (Fonte: Elaborado pelo autor no software Matlab®.)

- **Trajétória com Perfil de Velocidade Trapezoidal**

Outro tipo de trajetória que visa evitar as discontinuidades (velocidade e aceleração extremamente elevadas) no início e no fim do movimento, são as trajetórias com perfil de velocidade trapezoidal. Estas, apresentam um movimento com três etapas:

uma etapa de aceleração constante e velocidade que cresce linearmente, outra com aceleração nula e velocidade constante máxima e uma última de desaceleração constante e, portanto, velocidade com decrescimento linear. A equação a seguir representa uma trajetória deste tipo:

$$q(t) = ct^2 + bt + a, \quad 0 \leq t \leq t_1 \quad (18)$$

$$q(t) = e(t - t_1) + d, \quad t_1 \leq t \leq t_2 \quad (19)$$

$$q(t) = h(t - t_2)^2 + g(t - t_2) + f, \quad t_1 \leq t \leq t_2 \quad (20)$$

Desta forma, pelas condições de velocidade e aceleração no ponto inicial e final, e dada uma velocidade máxima v_{max} e uma aceleração máxima a_{max} , temos:

$$a = q_0 \quad (21)$$

$$b = \dot{q}_0 = 0 \quad (22)$$

$$c = \frac{a_{max}}{2} \quad (23)$$

$$t_1 = \frac{v_{max} - \dot{q}_0}{a_{max}} = \frac{v_{max}}{a_{max}} \quad (24)$$

$$d = \frac{v_{max}^2 - \dot{q}_0^2}{2a_{max}} + q_0 = \frac{v_{max}^2}{2a_{max}} + q_0 \quad (25)$$

$$e = v_{max} \quad (26)$$

$$f = q_f - \frac{\dot{q}_f^2 - v_{max}^2}{2a_{max}} = q_f + \frac{v_{max}^2}{2a_{max}} \quad (27)$$

$$g = v_{max} \quad (28)$$

$$h = -\frac{a_{max}}{2} \quad (29)$$

$$t_2 = \frac{f - d}{e} + t_1 = \frac{q_f - q_0}{v_{max}} + \frac{v_{max}}{a_{max}} \quad (30)$$

$$t_f = \frac{\dot{q}_f - g}{2h} + t_2 = \frac{q_f - q_0}{v_{max}} \quad (31)$$

Sendo que:

q_0 é a posição inicial,

q_f é a posição final,
 $\dot{q}_0$ é a velocidade inicial,
 $\dot{q}_f$ é a velocidade final,
 $\ddot{q}_0$ é a aceleração inicial,
 $\ddot{q}_f$ é a aceleração final,
 t_f é o tempo final.

Desta forma, o perfil de posição desejado tem a forma da Figura 3.6.

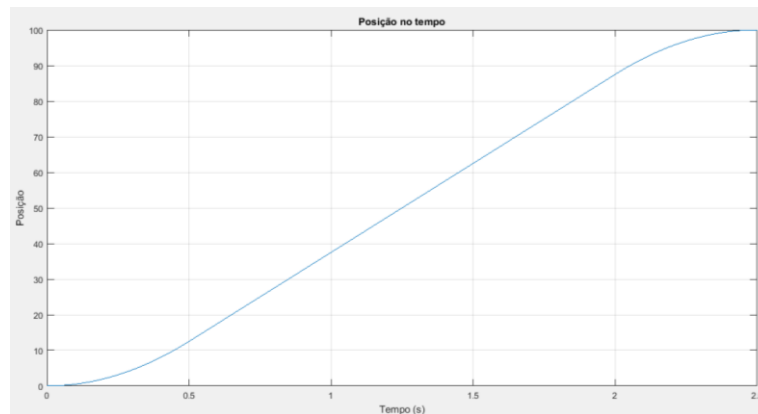


Figura 3.7 – Exemplo de trajetória com perfil de velocidade trapezoidal (Fonte: Elaborado pelo autor no software Matlab®.)

3.2.5. Medição do Erro de Posicionamento

Depois de serem calculados os perfis de trajetória, resta efetuar a medição do erro de posicionamento em questão. Para isto, os perfis são convertidos em sinais a serem fornecidos ao controlador, sendo estes constituídos por um trem de pulsos e um sinal digital que indica a orientação do movimento.

Com o motor vazio, envia-se ao controlador, então, o sinal calculado ao controlador enquanto é feita, simultaneamente, a aquisição dos dados referentes à posição real do motor.

Uma vez que se tem os dados referentes tanto à posição desejada quanto à posição observada ao longo do tempo, pode ser feita a comparação entre estes valores, sendo que o erro de posicionamento para cada momento é o valor da diferença entre ambas as posições naquele instante.

Os códigos utilizados nos procedimentos desta etapa encontram-se nos Apêndices de A até I do trabalho.

CAPÍTULO IV

RESULTADOS E DISCUSSÕES

Neste capítulo serão apresentados os resultados obtidos no decorrer do projeto e feitas discussões sobre o que estes valores representam, no que se refere ao controlador de que este trabalho diz respeito.

4.1. Tempo de Atraso

Como mencionado anteriormente, para se medir o tempo de atraso do sistema, foi enviado para o mesmo um sinal para que seguisse uma velocidade constante, velocidade a qual deveria mudar para uma mais alta, em certo momento.

Neste experimento, ficou definido que a velocidade deveria aumentar exatamente aos 5 segundos. Os resultados são apresentados na Figura 4.1.

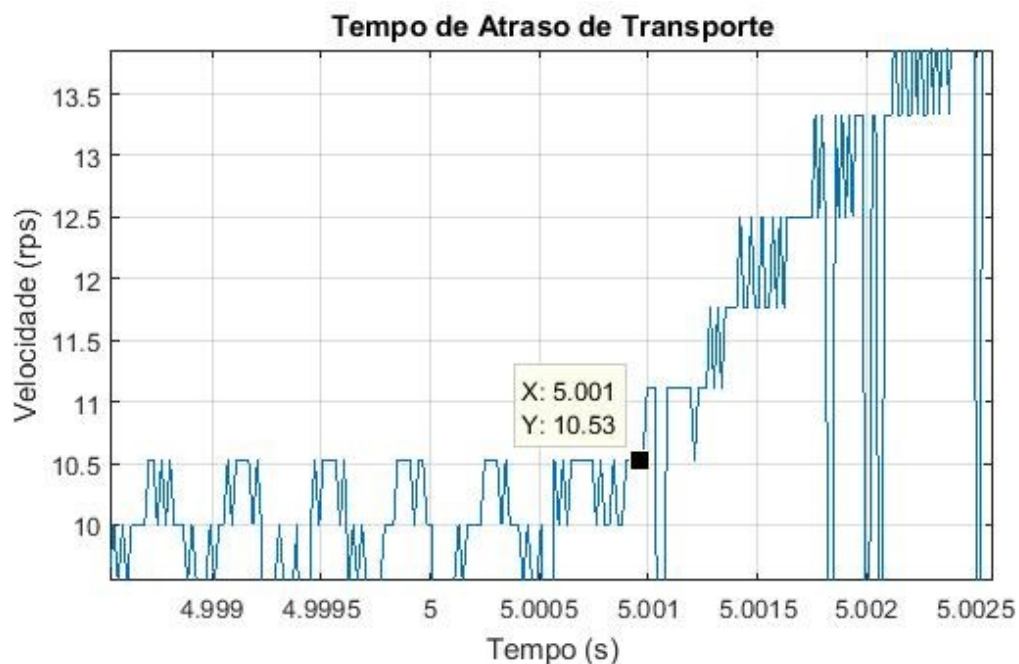


Figura 4.1 – Leitura da velocidade na medição do tempo de atraso (Fonte: Elaborado pelo autor no software Matlab®.)

Como pode ser observado na figura, a primeira variação observada na velocidade do motor ocorreu 1ms após o sinal ser enviado ao mesmo, caracterizando então um tempo de atraso do sistema de 1 milissegundo.

4.2. Erros de Posicionamento

Para mensurar os erros de posicionamento foram fornecidas trajetórias previamente calculadas ao controlador do motor e feito um comparativo entre a posição desejada e a posição real do motor ao longo do tempo. Para isso foram fornecidos os perfis de posição no tempo discutidos anteriormente, realizando três experimentos para cada com velocidades baixa, mediana e alta.

Em todos os resultados a seguir serão apresentados para cada nível de velocidade, um gráfico da posição real em comparação com a desejada e, em seguida, um gráfico mostrando o erro absoluto encontrado entre as duas, em milímetros.

4.2.1. Perfil de Velocidade Degrau

Para o perfil de velocidade degrau foram feitos três experimentos, utilizando os parâmetros da Tabela 4.1.

Tabela 4.1 – Parâmetros dos experimentos com perfil degrau (Fonte: Elaborado pelo autor.)

Velocidade (mm/s)	Distância Percorrida (mm)	Duração do Movimento (s)
10	10	1
100	100	1
300	300	1

Utilizando de tais parâmetros, foram encontrados os resultados das figuras a seguir.

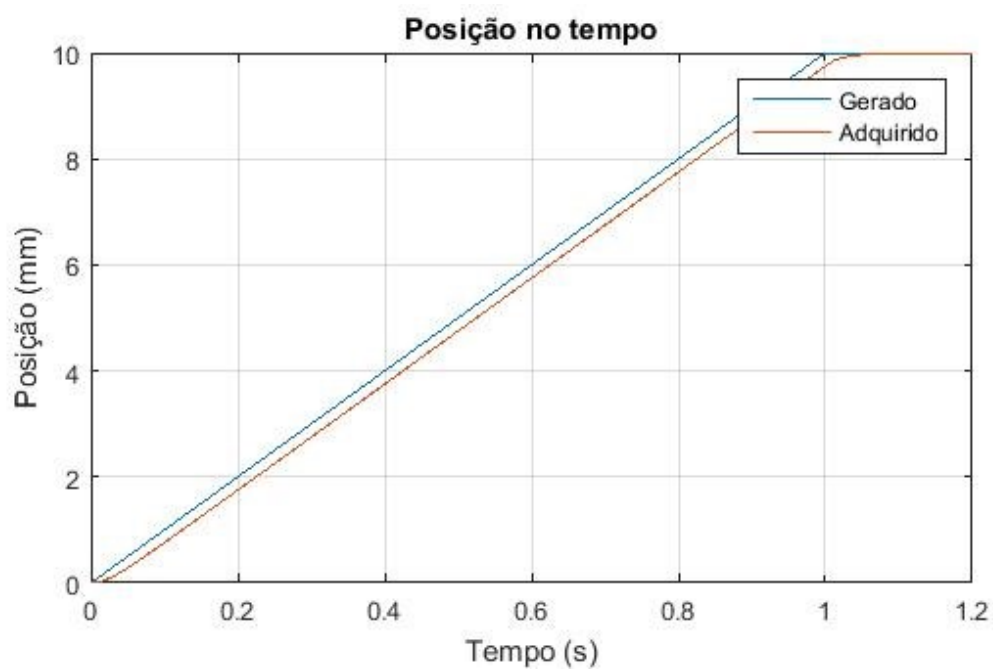


Figura 4.2 – Posição real vs posição desejada para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

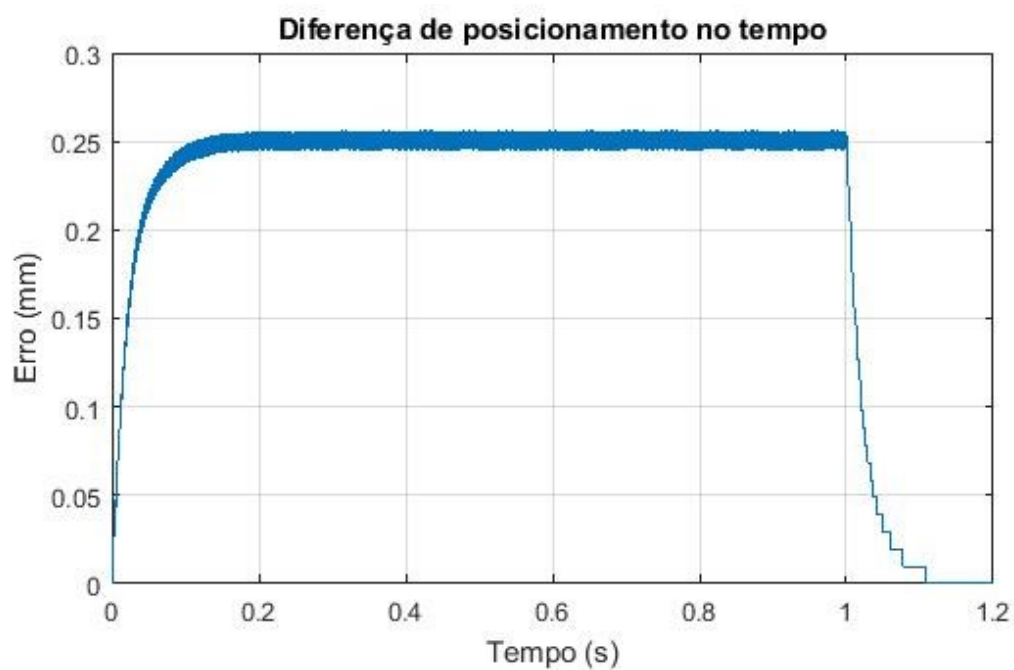


Figura 4.3 – Erro de posicionamento no tempo para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

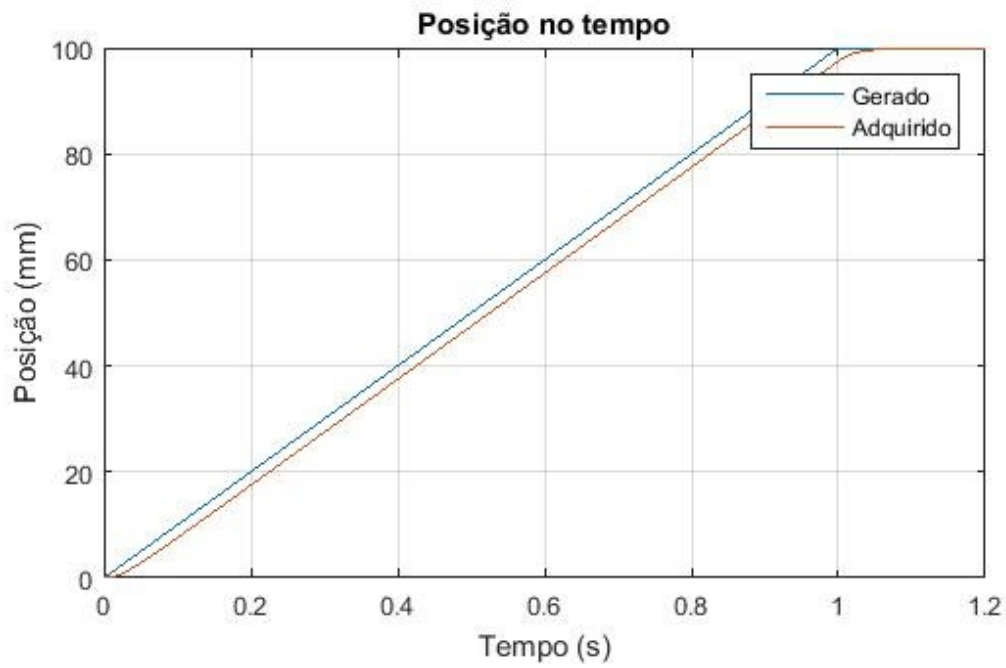


Figura 4.4 – Posição real vs posição desejada para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

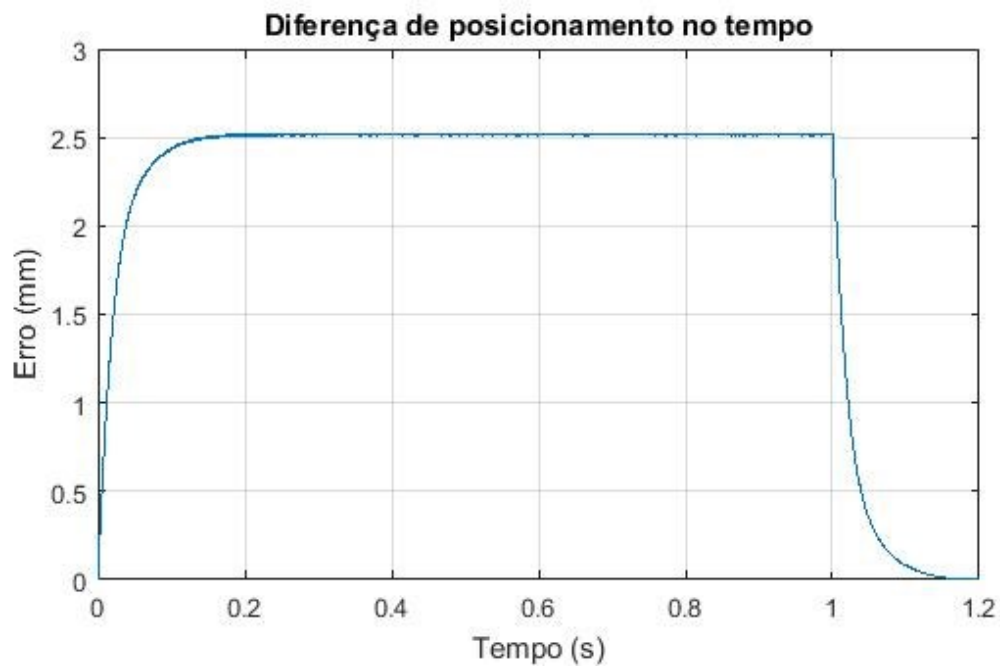


Figura 4.5 – Erro de posicionamento no tempo para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

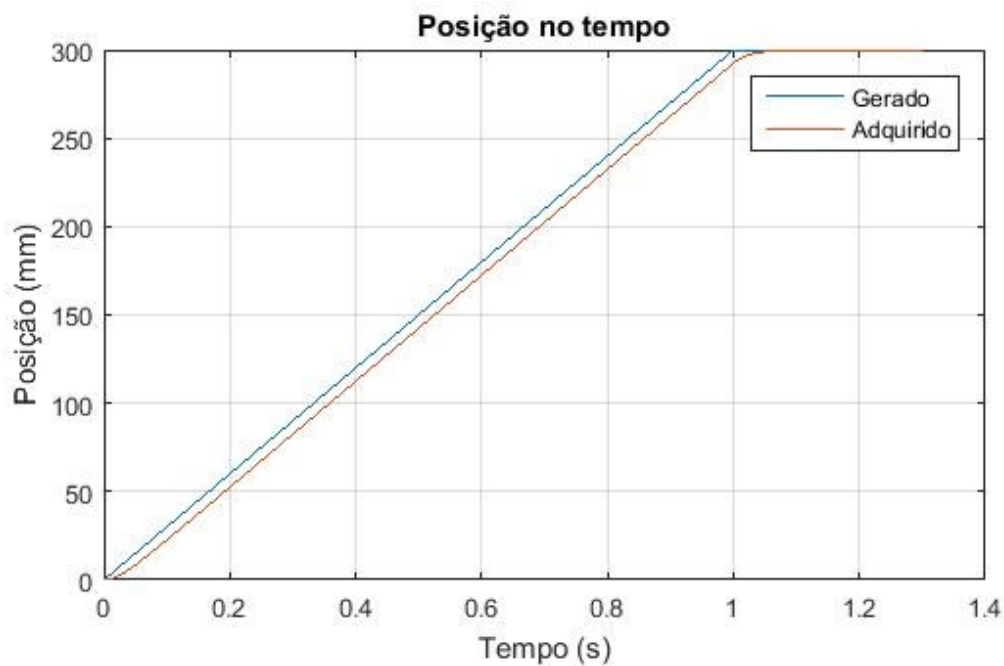


Figura 4.6 – Posição real vs posição desejada para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)



Figura 4.7 – Erro de posicionamento no tempo para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

Na Tabela 4.2 estão apresentados os erros que foram observados a partir destes resultados, sendo eles o erro de posicionamento máximo, o erro de posicionamento ao fim do movimento e o percentual destes erros sobre a distância total a ser percorrida.

Tabela 4.2 – Erros observados para o perfil degrau (Fonte: Elaborado pelo autor.)

Velocidade (mm/s)	Posição Final (mm)	Erro Máximo (mm)	Erro Final (mm)	Erro Percentual Máximo (%)	Erro Percentual Final (%)
10	10	0,256	0	2,56	0
100	100	2,524	0	2,52	0
300	300	7,553	0	2,52	0

4.2.2. Perfil de Velocidade Trapezoidal

Para o perfil de velocidade trapezoidal, foram feitos três experimentos utilizando os parâmetros da Tabela 4.3.

Tabela 4.3 – Parâmetros dos experimentos com perfil trapezoidal (Fonte: Elaborado pelo autor.)

Velocidade Máxima (mm/s)	Distância Percorrida (mm)	Tempo de Subida e Descida (s)	Duração do Movimento (s)
10	10	0,1	1,1
100	100	0,1	1,1
300	300	0,1	1,1

Utilizando de tais parâmetros, foram encontrados os resultados das figuras a seguir.

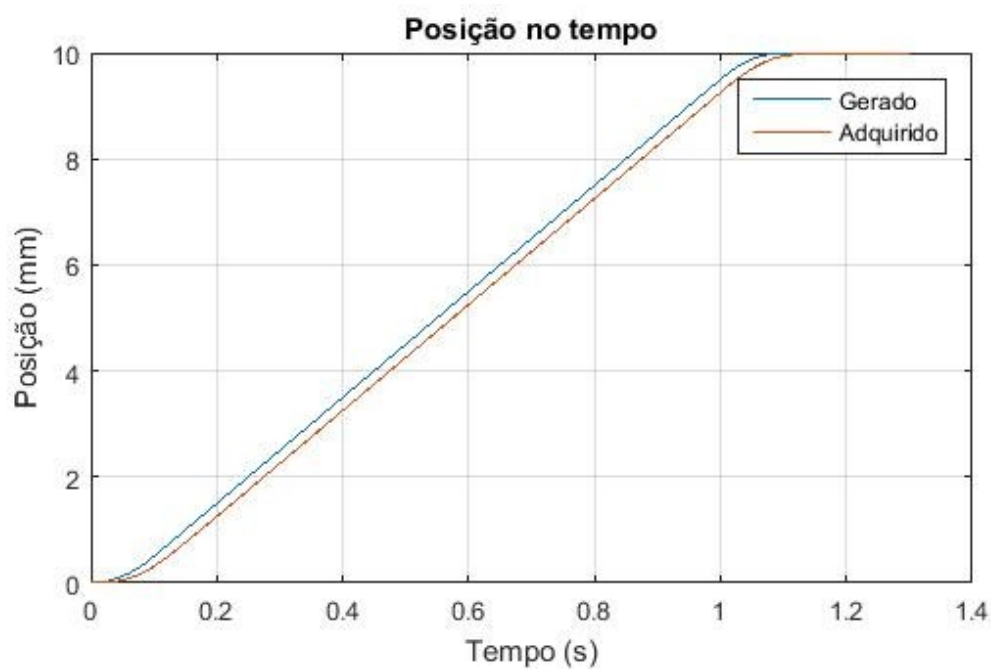


Figura 4.8 – Posição real vs posição desejada para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

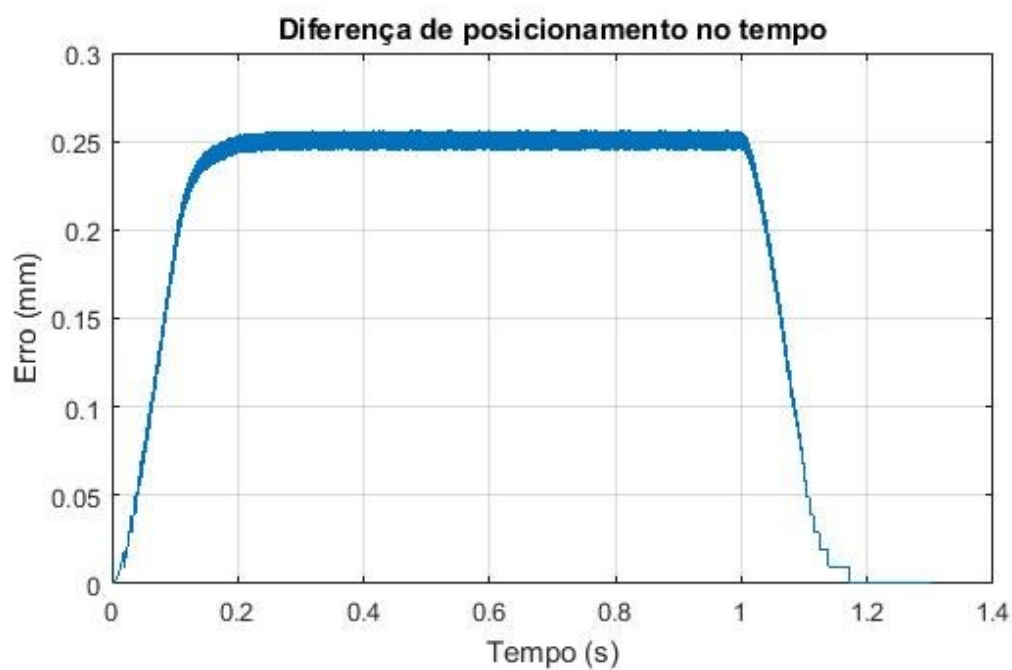


Figura 4.9 – Erro de posicionamento no tempo para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

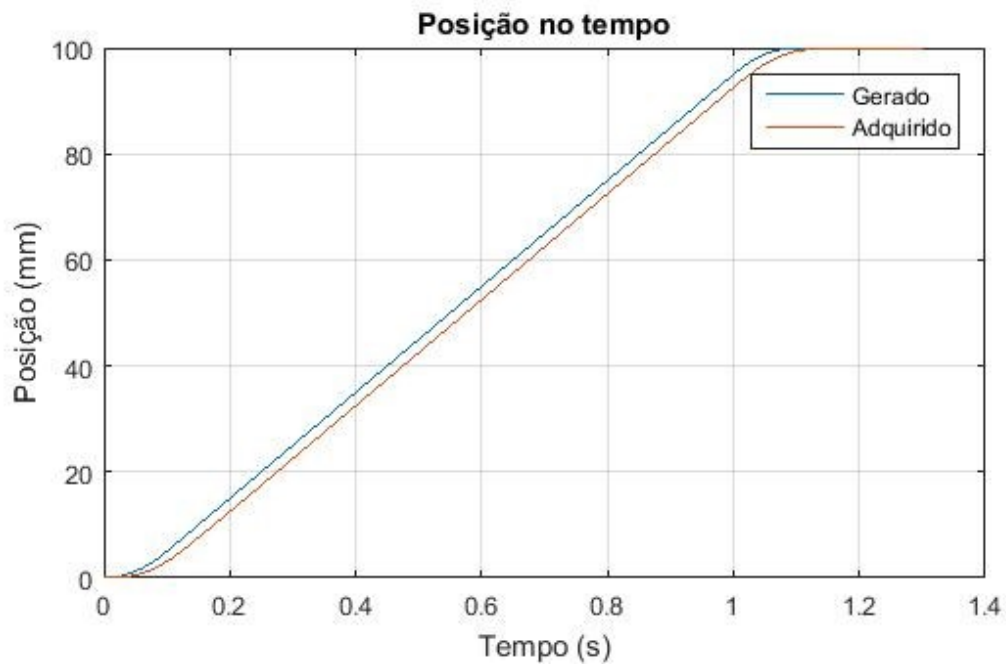


Figura 4.10 – Posição real vs posição desejada para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

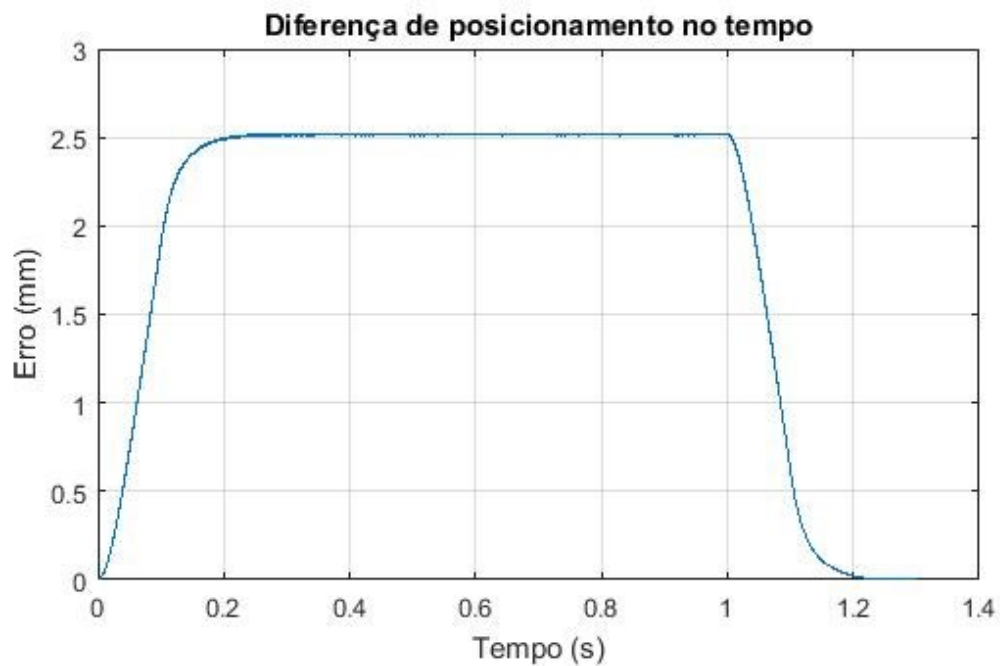


Figura 4.11 – Erro de posicionamento no tempo para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

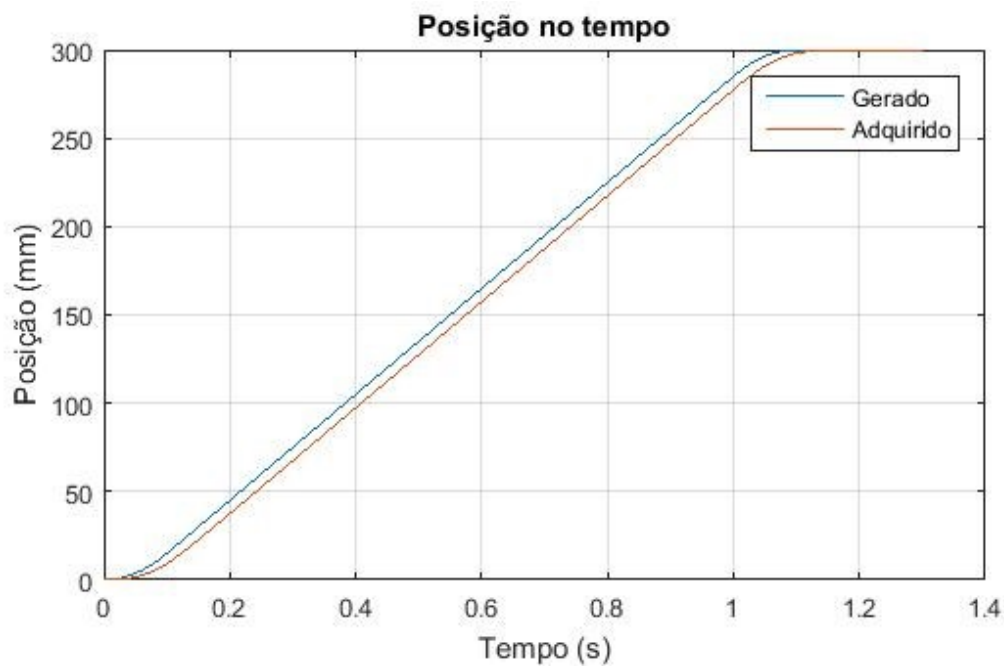


Figura 4.12 – Posição real vs posição desejada para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

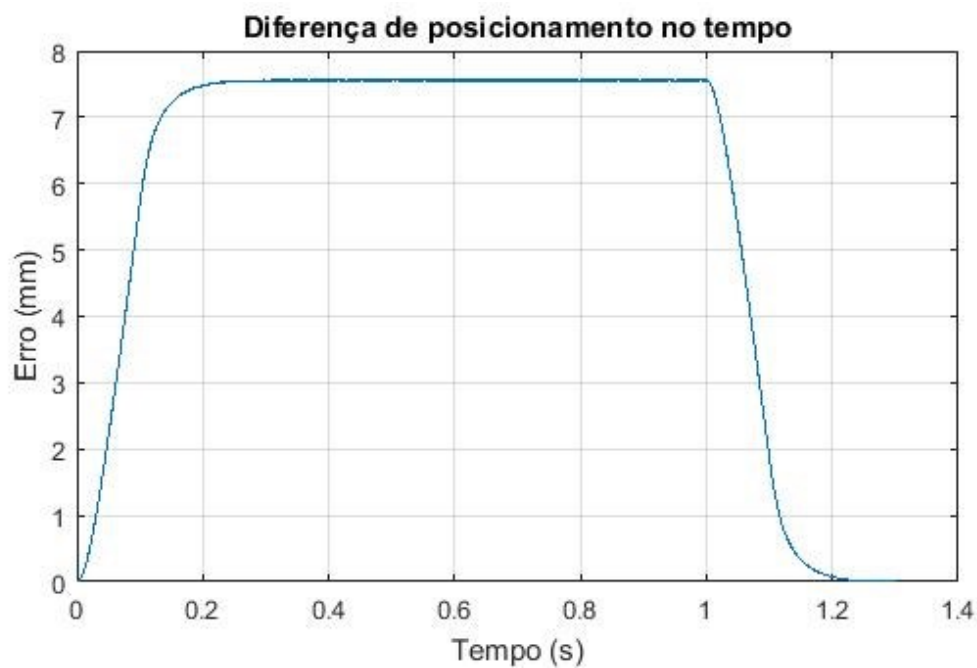


Figura 4.13 – Erro de posicionamento no tempo para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

Na Tabela 4.4 estão apresentados os erros que foram observados a partir destes resultados, sendo eles o erro de posicionamento máximo, o erro de posicionamento ao fim do movimento e o percentual destes erros sobre a distância total a ser percorrida.

Tabela 4.4 – Erros observados para o perfil trapezoidal (Fonte: Elaborado pelo autor.)

Velocidade Máxima (mm/s)	Posição Final (mm)	Erro Máximo (mm)	Erro Final (mm)	Erro Percentual Máximo (%)	Erro Percentual Final (%)
10	10	0,256	0	2,56	0
100	100	2,524	0	2,52	0
300	300	7,562	0	2,52	0

4.2.3. Trajetória Cúbica

Para a trajetória cúbica, foram feitos três experimentos utilizando os parâmetros da Tabela 4.5.

Tabela 4.5 – Parâmetros dos experimentos com trajetória cúbica (Fonte: Elaborado pelo autor.)

Velocidade Média (mm/s)	Velocidade Máxima (mm/s)	Distância Percorrida (mm)	Duração do Movimento (s)
10	15	10	1
100	150	100	1
300	450	300	1

Utilizando de tais parâmetros, foram encontrados os resultados das figuras a seguir.

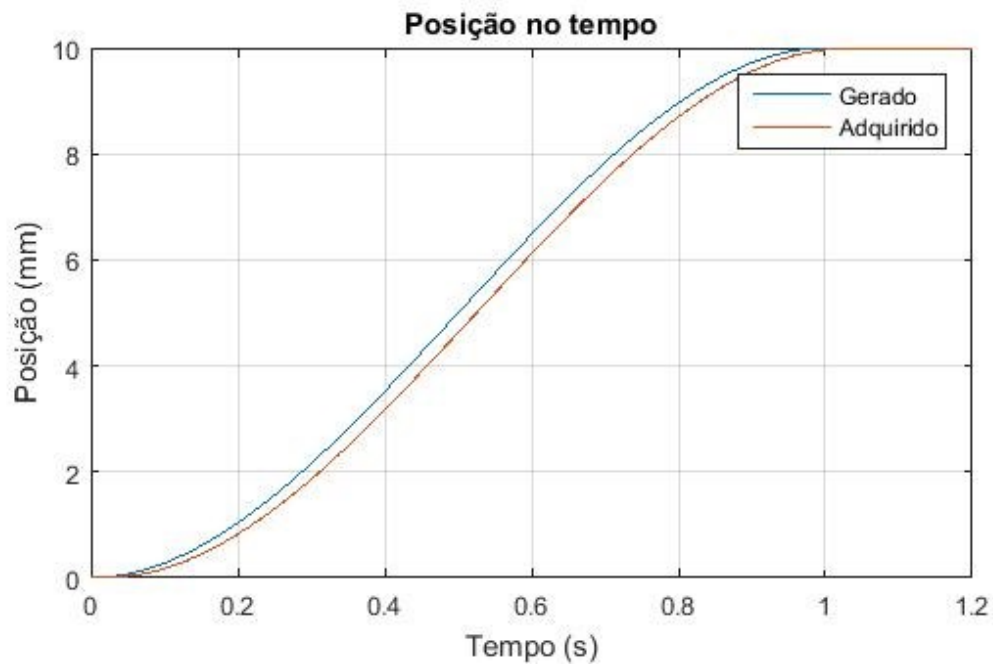


Figura 4.14 – Posição real vs posição desejada para trajetória cúbica a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

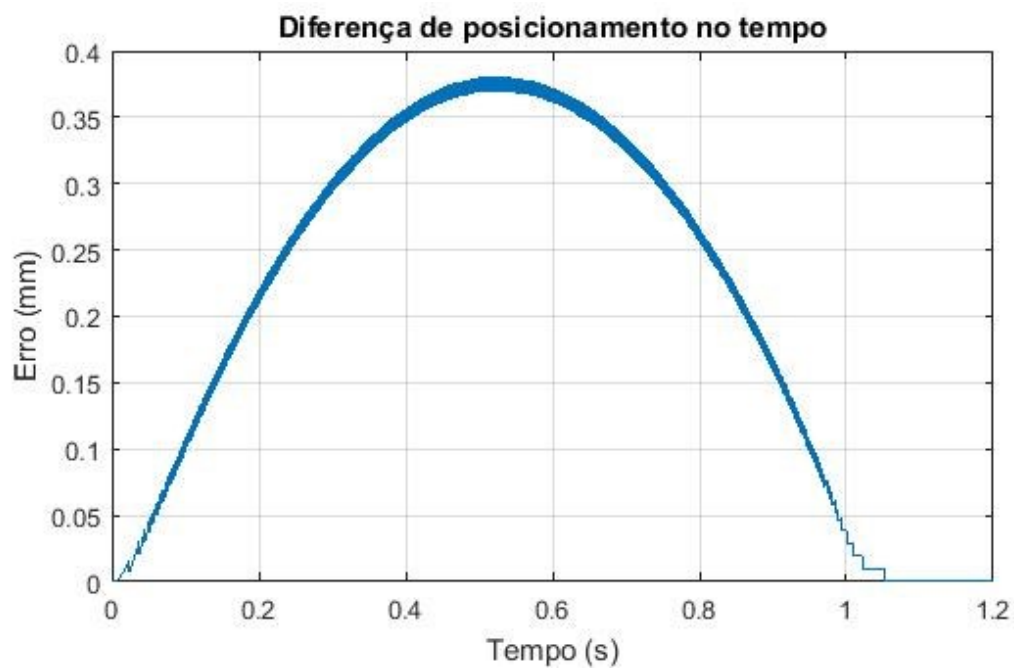


Figura 4.15 – Erro de posicionamento no tempo para trajetória cúbica a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

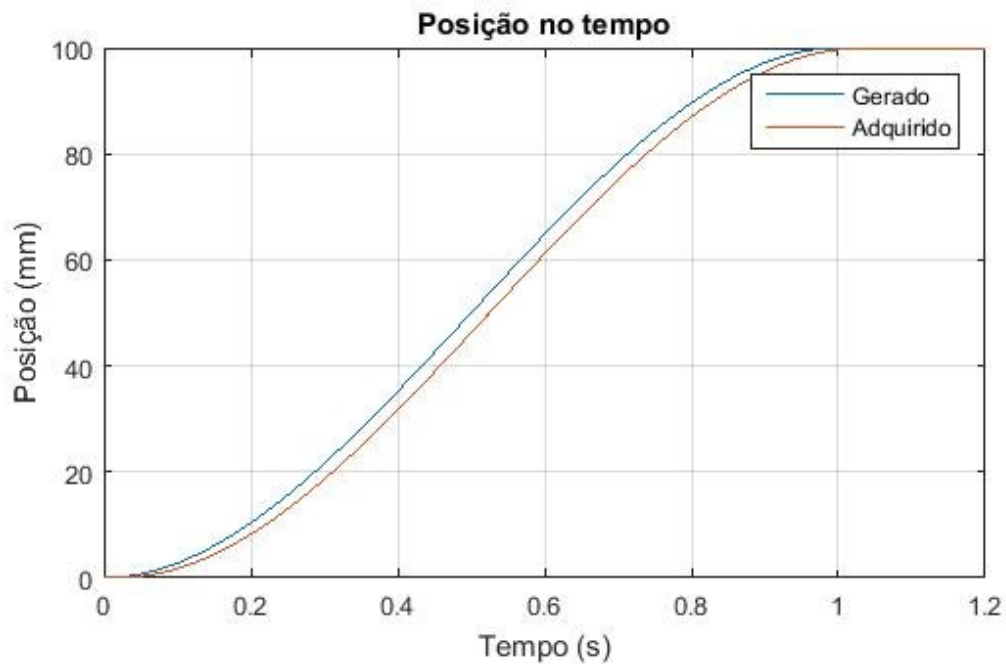


Figura 4.16 – Posição real vs posição desejada para trajetória cúbica a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

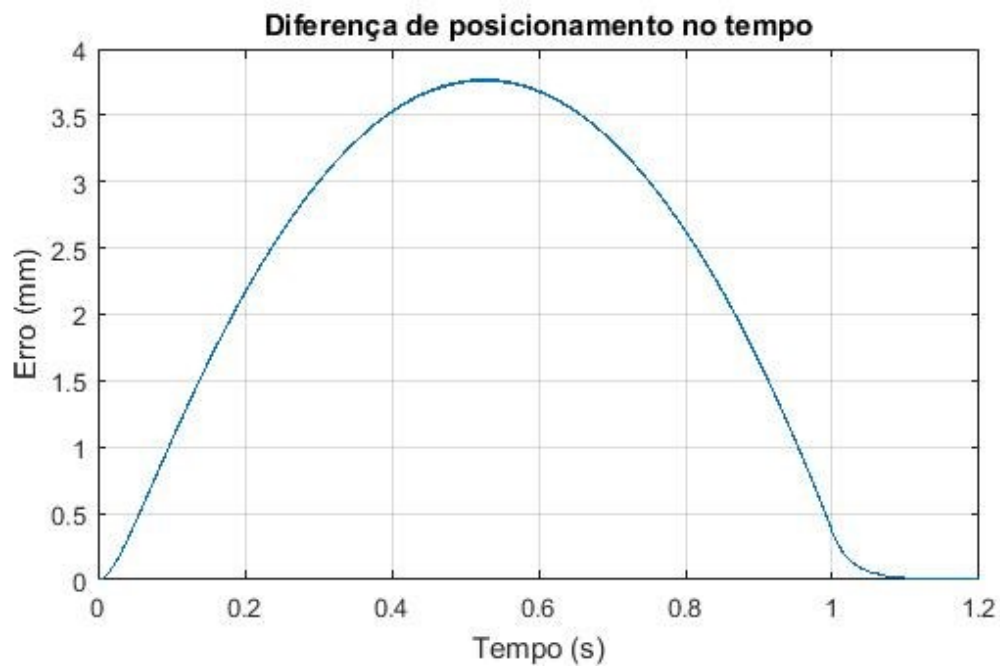


Figura 4.17 – Erro de posicionamento no tempo para trajetória cúbica a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

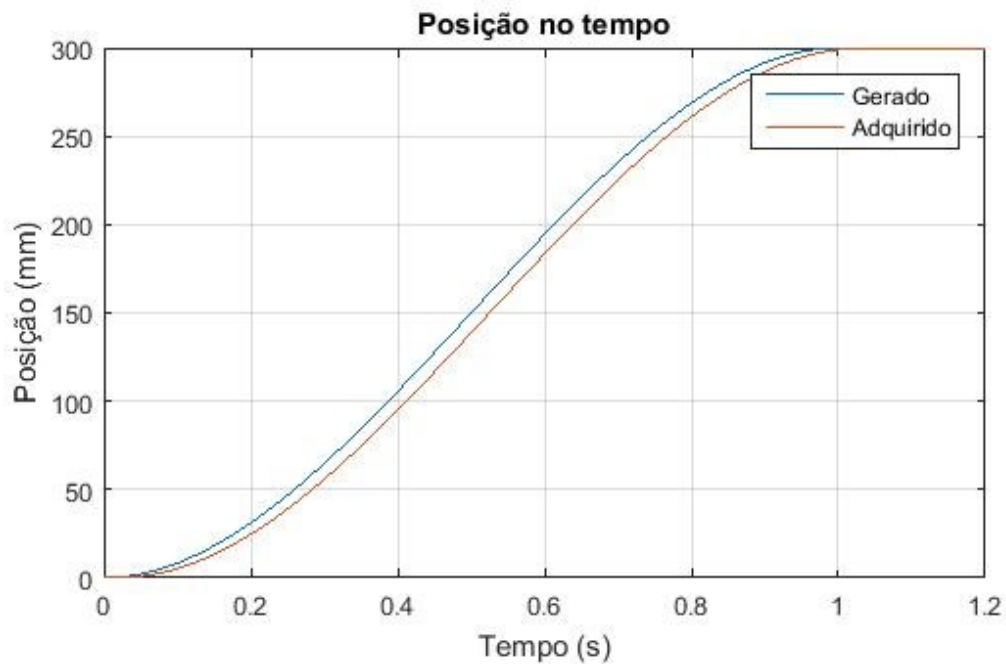


Figura 4.18 – Posição real vs posição desejada para trajetória cúbica a 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

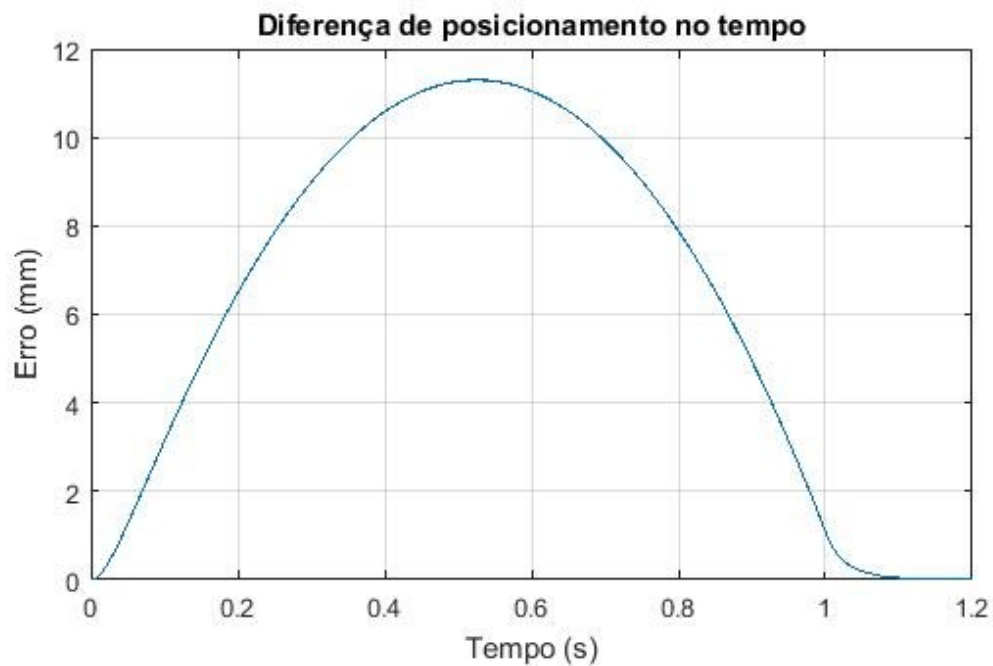


Figura 4.19 – Erro de posicionamento no tempo para trajetória cúbica a 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

Na Tabela 4.6 estão apresentados os erros que foram observados a partir destes resultados, sendo eles o erro de posicionamento máximo, o erro de posicionamento ao fim do movimento e o percentual destes erros sobre a distância total a ser percorrida.

Tabela 4.6 – Erros observados para a trajetória cúbica (Fonte: Elaborado pelo autor.)

Velocidade Média (mm/s)	Posição Final (mm)	Erro Máximo (mm)	Erro Final (mm)	Erro Percentual Máximo (%)	Erro Percentual Final (%)
10	10	0,381	0	3,81	0
100	100	3,770	0	3,77	0
300	300	11,303	0	3,77	0

4.2.4. Trajetória Polinomial de Quinto Grau

Para a trajetória de quinto grau, foram feitos três experimentos utilizando os parâmetros da Tabela 4.7.

Tabela 4.7 – Parâmetros dos experimentos com trajetória de quinto grau (Fonte: Elaborado pelo autor.)

Velocidade Média (mm/s)	Velocidade Máxima (mm/s)	Distância Percorrida (mm)	Duração do Movimento (s)
10	18,75	10	1
100	187,5	100	1
250	468,7	250	1

Utilizando de tais parâmetros, foram encontrados os resultados das figuras a seguir.

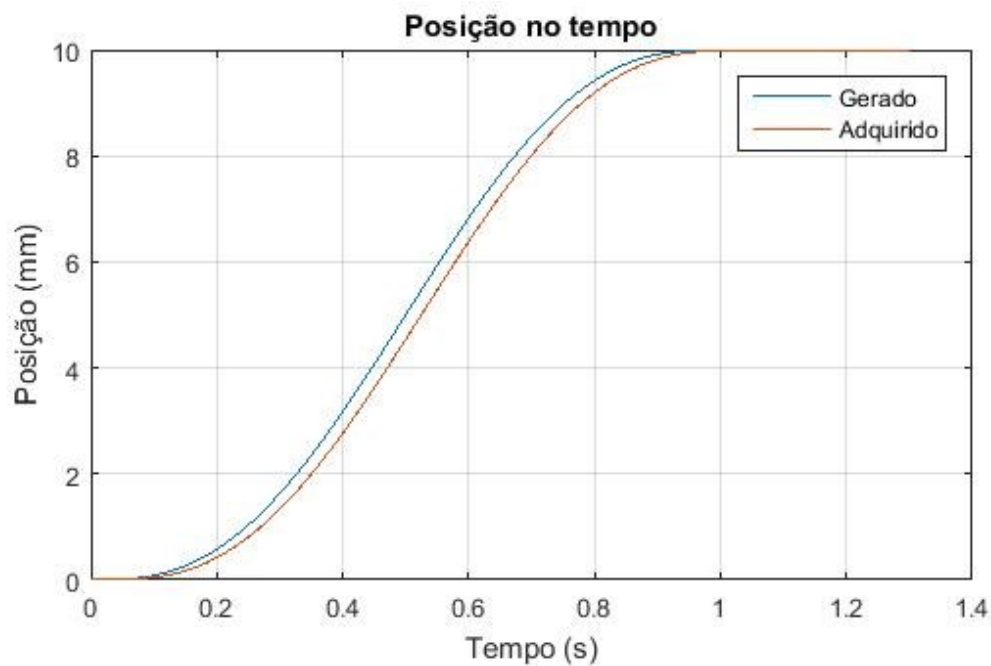


Figura 4.20 – Posição real vs posição desejada para trajetória de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

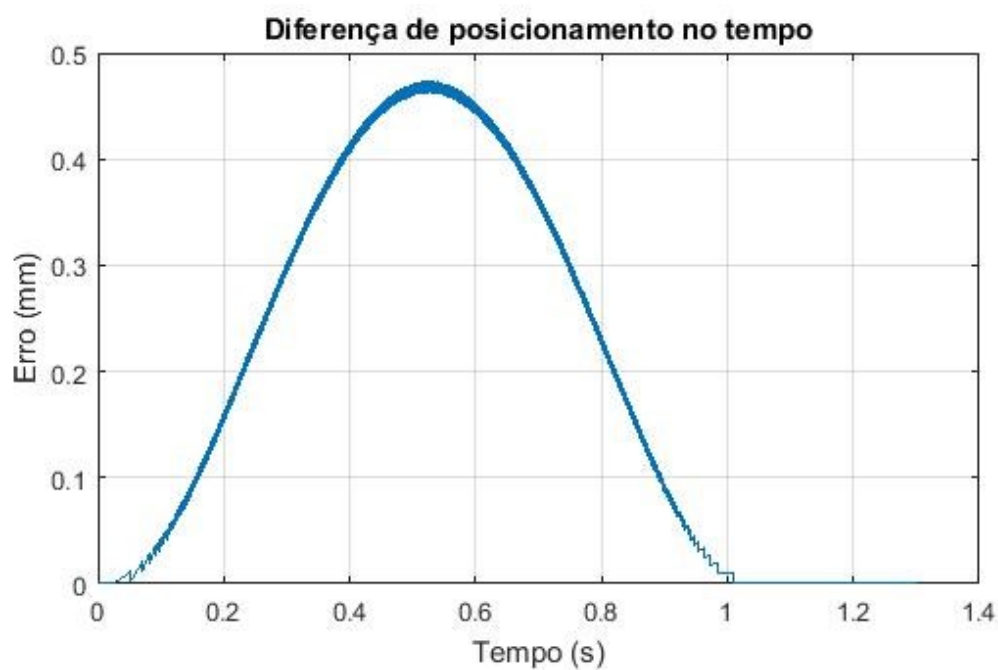


Figura 4.21 – Erro de posicionamento no tempo para trajetória de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

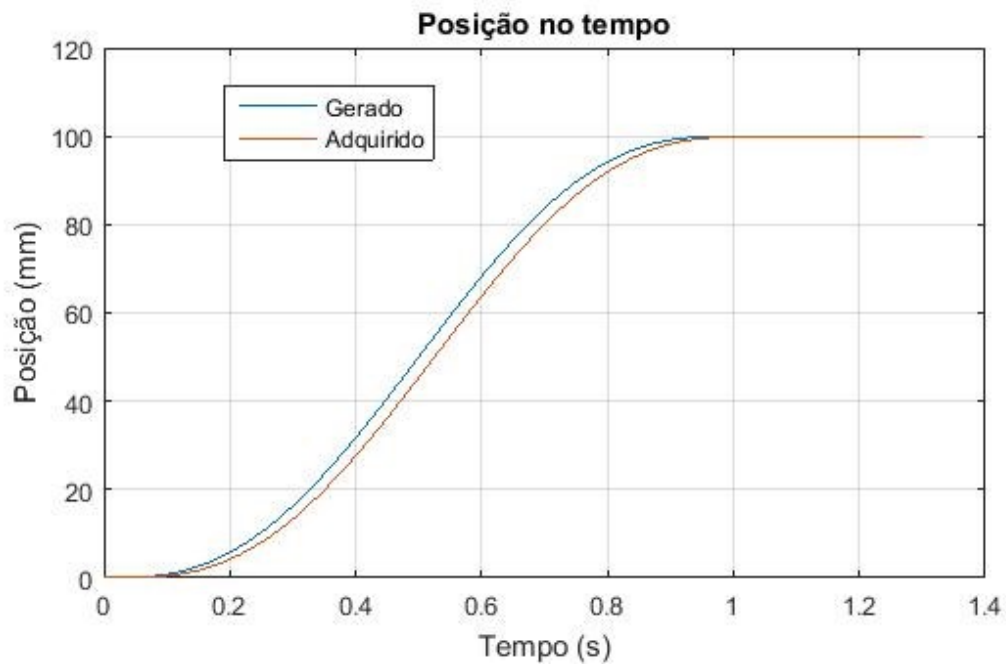


Figura 4.22 – Posição real vs posição desejada para trajetória de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

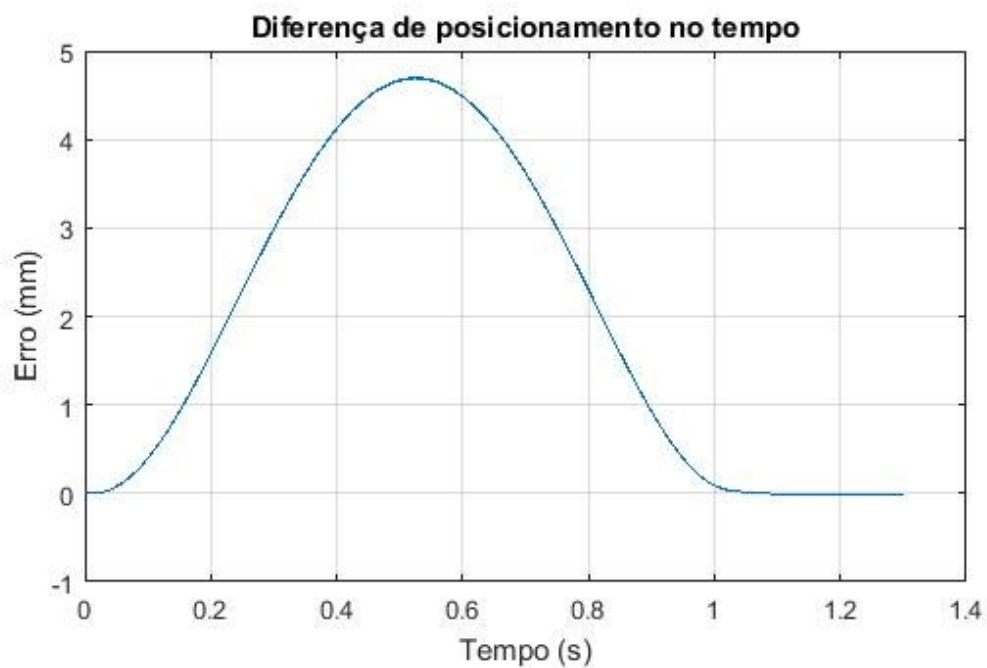


Figura 4.23 – Erro de posicionamento no tempo para trajetória de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

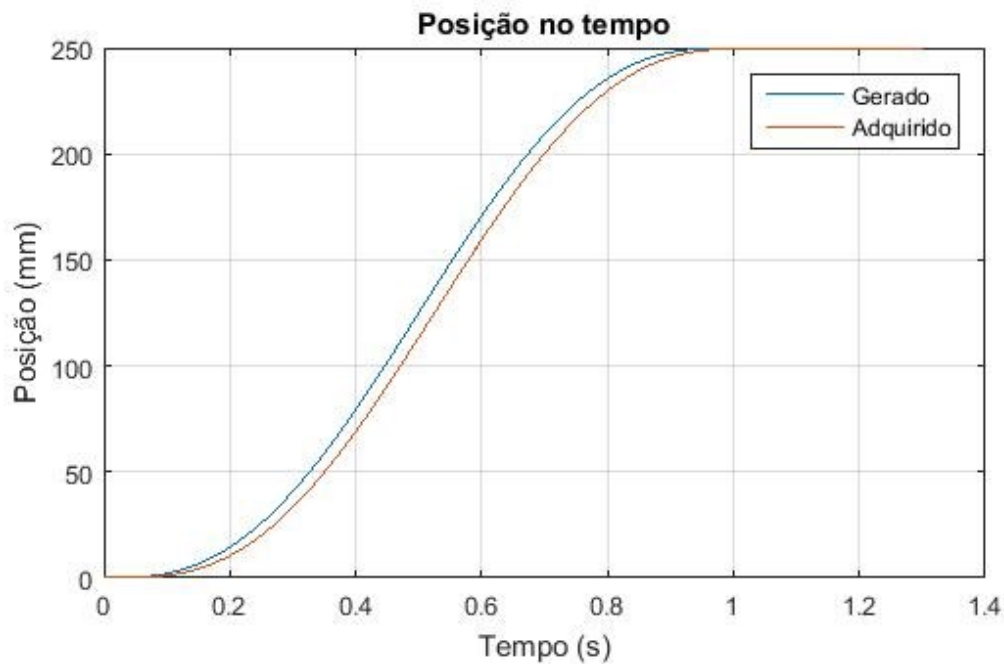


Figura 4.24 – Posição real vs posição desejada para trajetória de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

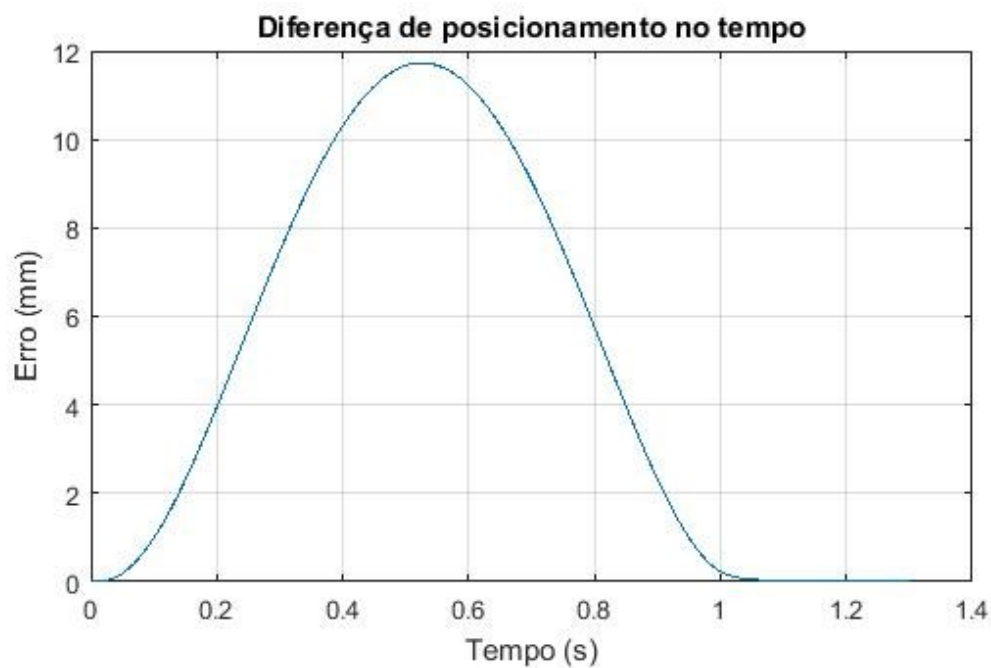


Figura 4.25 – Erro de posicionamento no tempo para trajetória de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

Na Tabela 4.8 estão apresentados os erros que foram observados a partir destes resultados, sendo eles o erro de posicionamento máximo, o erro de posicionamento ao fim do movimento e o percentual destes erros sobre a distância total a ser percorrida.

Tabela 4.8 – Erros observados para a trajetória de quinto grau (Fonte: Elaborado pelo autor.)

Velocidade Média (mm/s)	Posição Final (mm)	Erro Máximo (mm)	Erro Final (mm)	Erro Percentual Máximo (%)	Erro Percentual Final (%)
10	10	0,473	0	4,73	0
100	100	4,696	0	4,70	0
250	250	11,736	0	3,91	0

4.3. Discussão dos Resultados

4.3.1. Tempo de Atraso

Como foi apresentado anteriormente nos resultados, o tempo de atraso encontrado para o sistema foi de cerca de um milissegundo, valor encontrado com bastante confiabilidade, uma vez que o sistema de aquisição trabalha com um milhão de amostras por segundo sendo, então, capaz de perceber variações de até um microssegundo.

Devido à forma como o experimento foi realizado, partindo de uma velocidade constante para uma mais alta enquanto observa-se a variação da mesma e também pelo fato de todos os experimentos terem sido conduzidos no motor sem carga, podemos atribuir o tempo de atraso unicamente ao atraso de transporte do sistema, desprezando qualquer tipo de efeito dinâmico.

4.3.2. Erros de posicionamento

Os resultados encontrados para os erros de posicionamento se mostraram extremamente satisfatórios, uma vez que os formatos das trajetórias se assemelharam muito aos daquelas desejada, apresentando apenas um pequeno tempo de atraso em relação às mesmas. Além disso, em todos os experimentos, os erros de posicionamento finais encontrados foram iguais a zero.

Pôde ser observado também que os gráficos de erro no tempo se assemelharam muito às velocidades esperadas em cada trajetória, causando então a suspeita de que estes possam ser proporcionais.

Tendo-se a velocidade da trajetória e o erro de posicionamento no tempo, pode-se obter o tempo de atraso em cada ponto pela razão do erro pela velocidade. Os resultados desses cálculos para cada tipo de trajetória estão apresentados nas figuras a seguir.

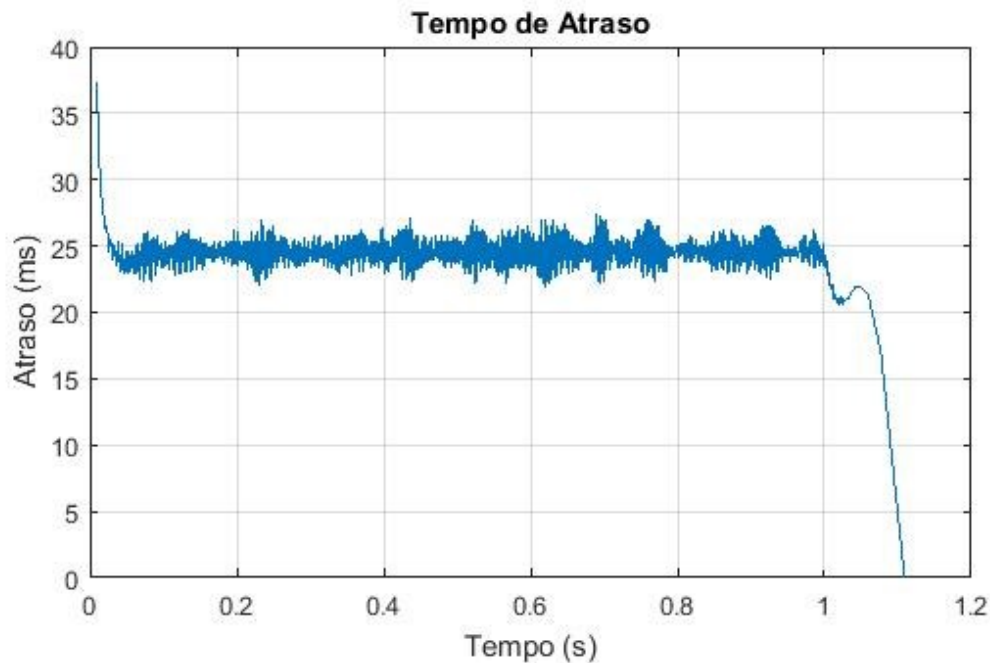


Figura 4.26 – Tempo de atraso para perfil degrau de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

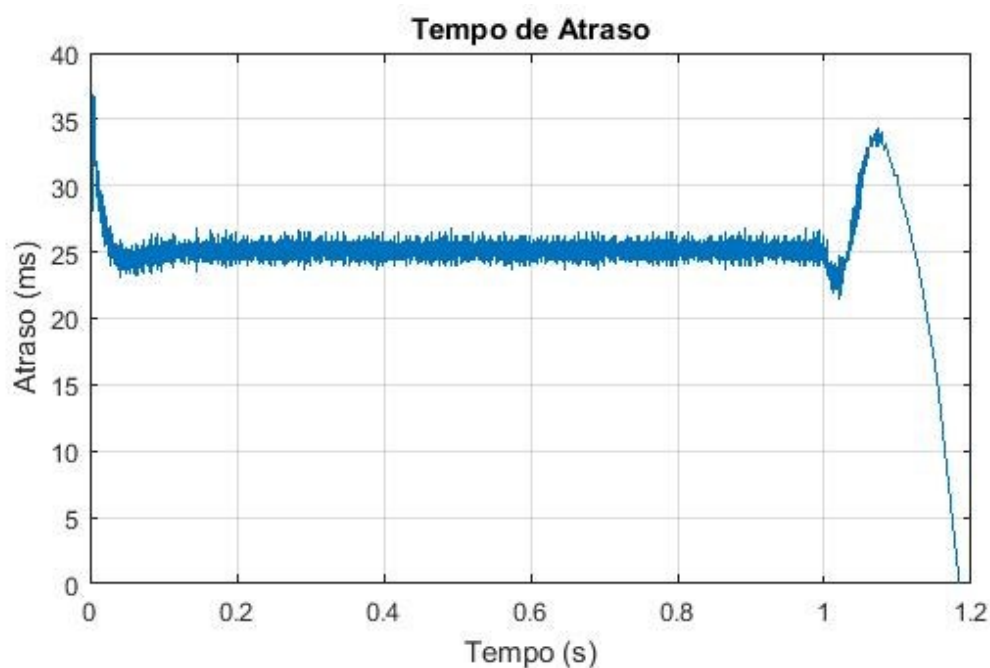


Figura 4.27 – Tempo de atraso para perfil degrau de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

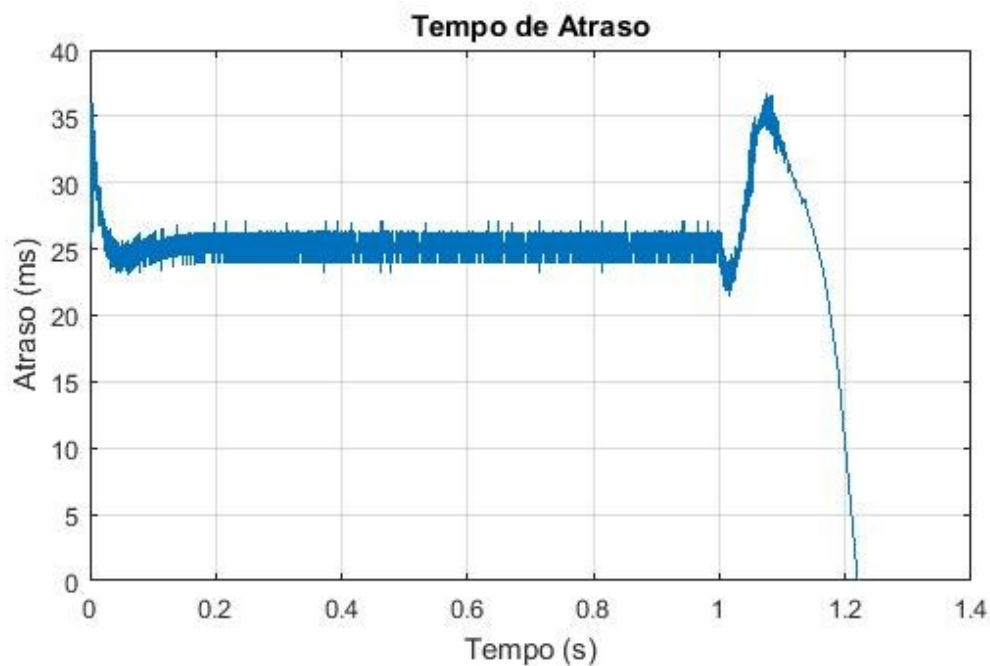


Figura 4.28 – Tempo de atraso para perfil degrau de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

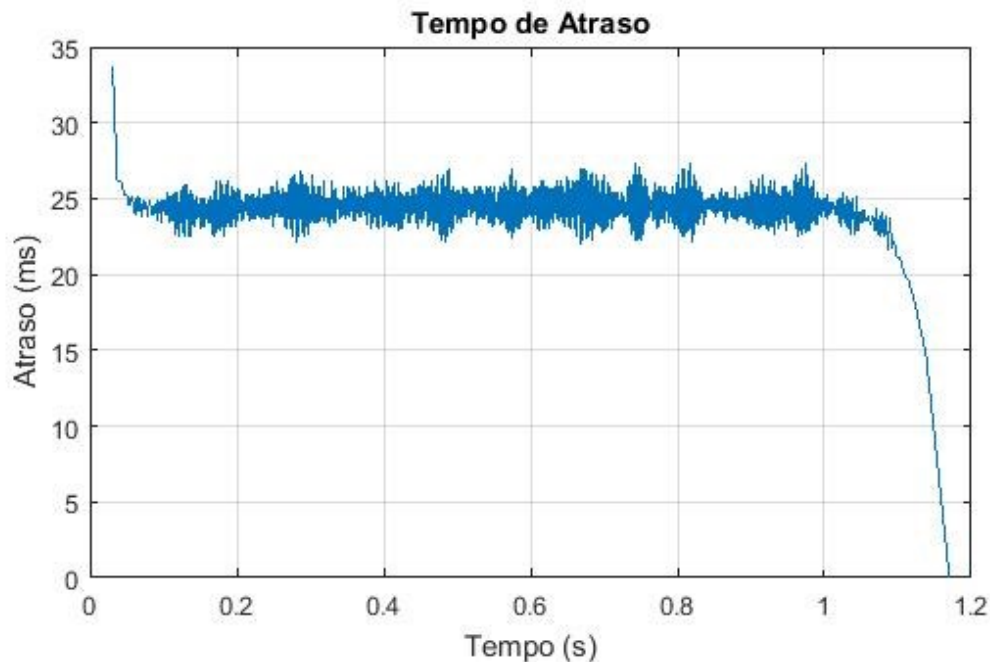


Figura 4.29 – Tempo de atraso para perfil trapezoidal de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

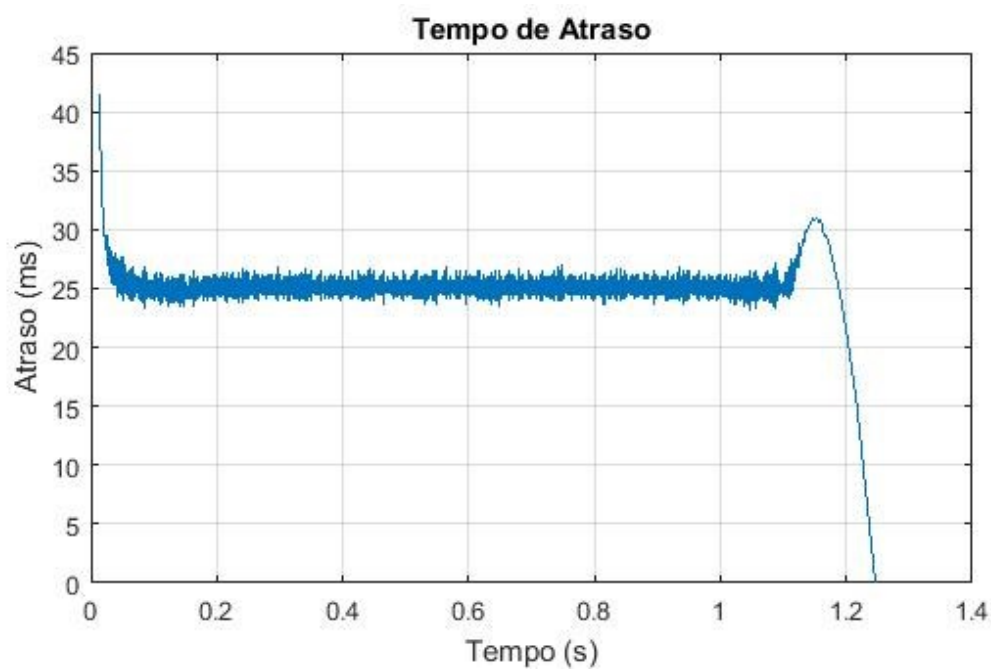


Figura 4.30 – Tempo de atraso para perfil trapezoidal de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

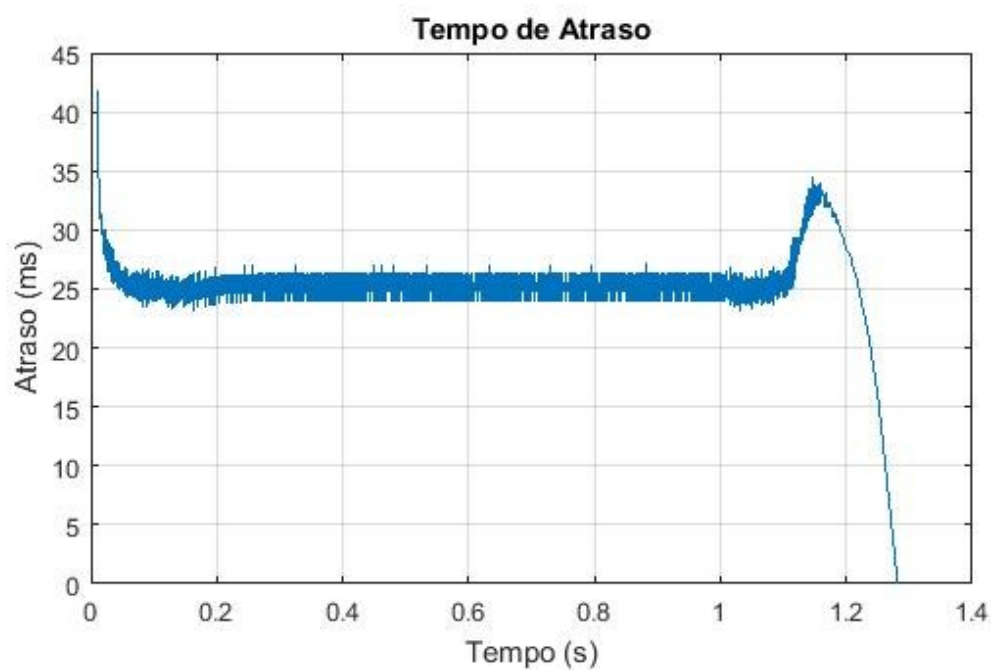


Figura 4.31 – Tempo de atraso para perfil trapezoidal de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

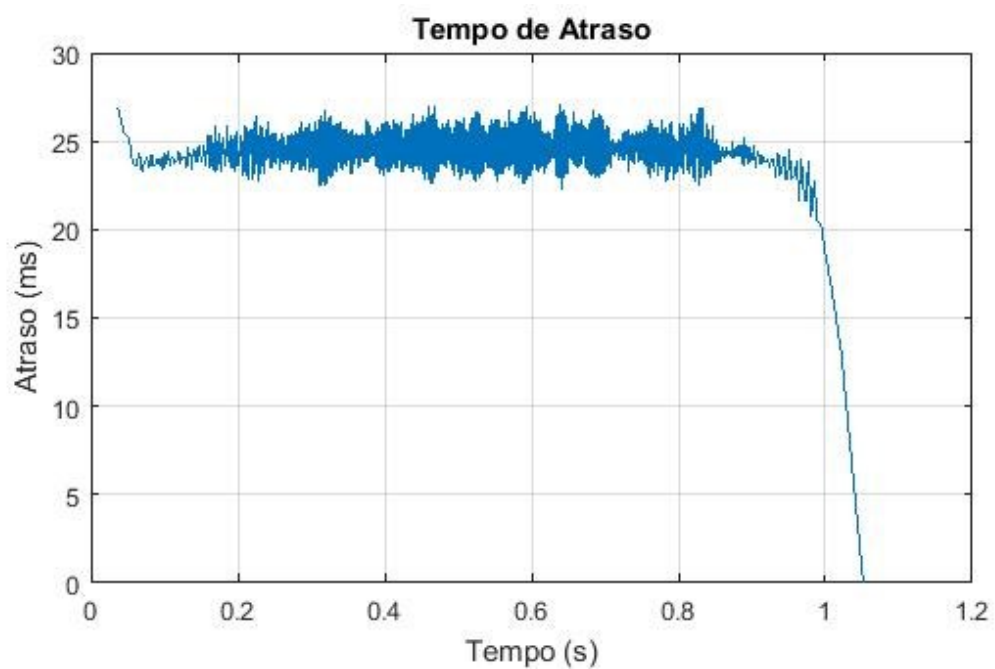


Figura 4.32 – Tempo de atraso para trajetória cúbica de 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

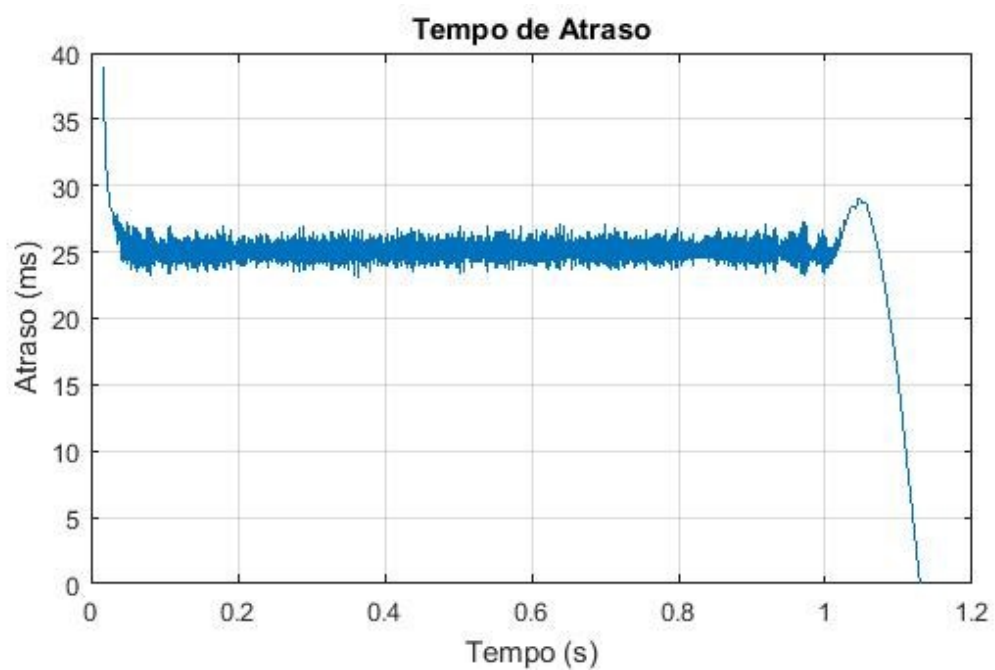


Figura 4.33 – Tempo de atraso para trajetória cúbica de 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

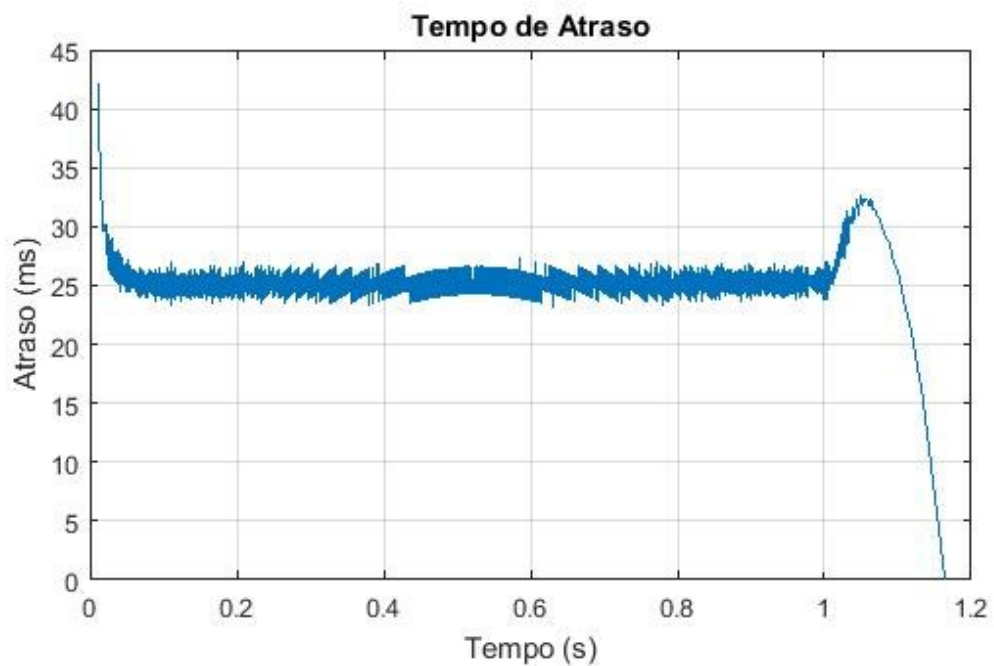


Figura 4.34 – Tempo de atraso para trajetória cúbica de 300mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

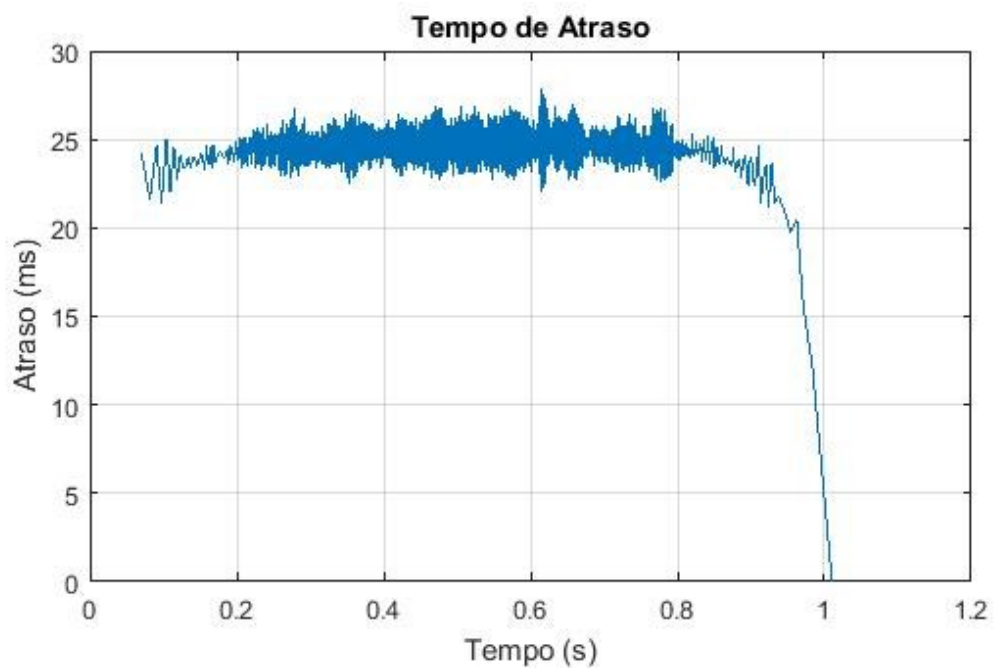


Figura 4.35 – Tempo de atraso para trajetória polinomial de quinto grau a 10mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

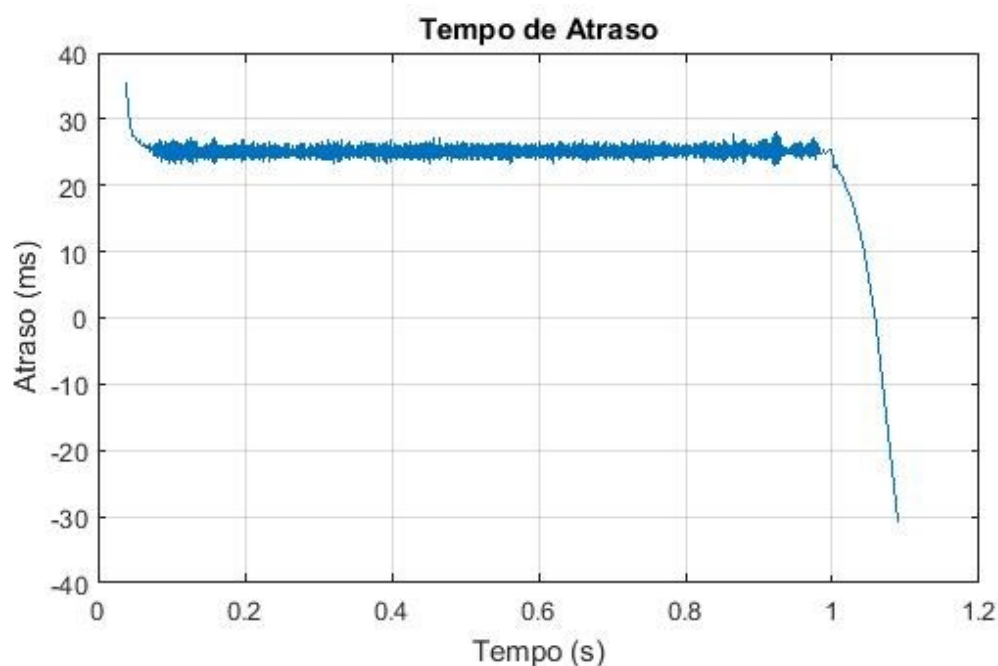


Figura 4.36 – Tempo de atraso para trajetória polinomial de quinto grau a 100mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

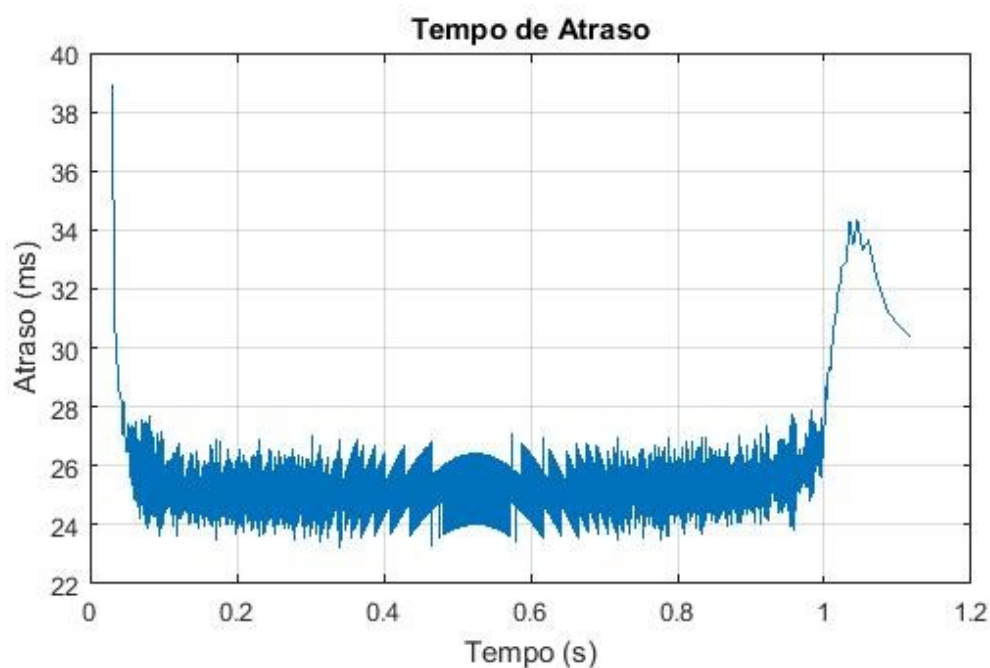


Figura 4.37 – Tempo de atraso para trajetória polinomial de quinto grau a 250mm/s (Fonte: Elaborado pelo autor no software Matlab®.)

A tabela 4.9 apresenta as médias e desvios padrões do tempo de atraso para todos os experimentos, assim como a constante de tempo encontrada, que é igual ao inverso do tempo de atraso.

Tabela 4.9 – Médias dos tempos de atraso e constantes de tempo (Fonte: Elaborado pelo autor.)

Trajetória	Velocidade (mm/s)	Média do Tempo de Atraso (ms)	Desvio Padrão (ms)	Constante de Tempo (s^{-1})
Degrau	10	24,489	1,199	40,835
	100	25,115	0,788	39,817
	300	25,166	0,821	39,736
Trapezoidal	10	24,475	1,148	40,858
	100	25,115	0,640	39,817
	300	25,166	0,639	39,736
Cúbica	10	24,553	1,153	40,728
	100	25,123	0,631	39,804
	300	25,169	0,600	39,731
Quinta Ordem	10	24,604	1,229	40,644
	100	25,121	0,678	39,807
	250	25,170	0,591	39,730

Pôde-se observar que, em geral, tempo de atraso oscilou em torno de uma constante em todos os casos. Esta constante é o tempo de atraso médio, decorrente da constante de tempo que é característica do sistema.

Como foi apresentado nos resultados anteriormente, o tempo de atraso encontrado foi de aproximadamente 25ms e, portanto, a constante de tempo encontrada foi de aproximadamente $40s^{-1}$. Tal tempo de atraso pode ser observado nos gráficos de trajetória real vs trajetória adquirida apresentados.

Além disso, pôde-se observar tempos de atraso mais discrepantes na partida e na parada. Isso acontece por efeito de um fator de suavização aplicado pelo controlador

em ambos os casos, evitando partidas e paradas mais bruscas e servindo como mais uma fonte de erros, por limitar a capacidade do motor reproduzir movimentos com maior fidelidade nesses momentos.

Estes dois fatores foram as principais fontes de erro observadas, contudo, elas não afetam tanto o resultado final da trajetória seguida pela mesa cartesiana pois por se tratarem apenas de tempos de atraso, não acumulam erros no fim da trajetória (resultando nos erros finais nulos que puderam ser observados) e, portanto, uma vez que o tempo de atraso passe, o elemento terminal alcançará a posição desejada sem discrepâncias.

CAPÍTULO V

CONCLUSÕES

Durante a execução deste trabalho, foi possível, através da realização de experimentos em laboratório, efetuar um levantamento de algumas características desconhecidas do sistema em questão, avaliando assim a qualidade dos resultados que ele é capaz de entregar.

A revisão bibliográfica realizada permitiu um melhor entendimento dos componentes que constituem o sistema em questão e seus respectivos funcionamentos, além de entender a aplicabilidade e a importância destes robôs em ambiente industrial.

Através do cálculo de trajetórias, da geração e aquisição de sinais, tornou-se possível obter os dados necessários para estabelecer um comparativo entre as trajetórias ideais calculadas e aquelas adquiridas experimentalmente. Tais conhecimentos se mostram essenciais para a robótica industrial, pois sem estes, nenhum sistema de controle funcionaria. O entendimento dos perfis de posição e velocidade mais comuns, permite, também, que estes sejam utilizados de forma a se obter os melhores resultados possíveis, dependendo da tarefa a ser cumprida.

Por meio do levantamento dos dados experimentais, foi observada a qualidade das trajetórias executadas pelo controlador ao receber um sinal de controle a ser seguido. A análise de tais dados mostrou que o sistema segue seus percursos de maneira bastante precisa, uma vez que apresenta como discrepância, apenas um atraso no tempo de aproximadamente **25 milissegundos**, e erro de posicionamento final nulo em todos os experimentos.

Com base nestes resultados, pode-se afirmar que tanto os objetivos gerais quanto os objetivos específicos foram concluídos e que os resultados superaram as melhores expectativas quanto à qualidade deste equipamento.

Por fim, vale destacar a importância deste tipo de trabalho na formação de um graduando em engenharia, aproximando este ainda mais da pesquisa científica e permitindo-o, durante este processo, desenvolver sua capacidade de aquisição e interpretação de dados, além de colocar em prática conhecimentos técnicos adquiridos ao longo de sua graduação.

REFERÊNCIAS BIBLIOGRÁFICAS

DOEBELIN, E. O. **Measurement Systems:** Application and Design. 5ª. ed. New York: McGraw-Hill, 2004.

ELLIOTT, B. S. **Electromechanical Devices & Components:** Illustrated Sourcebook. New York: McGraw-Hill, 2007.

KOSOW, I. L. **Máquinas Elétricas e Transformadores.** Tradução de Felipe Luis Daiello e Percy Antônio Soares. 5ª. ed. Rio de Janeiro: Globo, 1985.

ROMANO, V. F.; DUTRA, M. S. **Introdução à Robótica Industrial.** São Paulo: Edgard Blücher Ltda, 2002.

THALER, G. J.; BROWN, R. G. **Analysis and Design of Feedback Control Systems.** 2ª. ed. New York: McGraw-Hill, 1960.

APÊNDICE A – Função para Calcular Trajetória Degrau Desenvolvida no *Matlab*®

```
function [ t, q, sentido ] = trajetoriaDegrau( tf, ti, qi, qf, dt )

t=ti:dt:tf;

a1=(qf-qi)/(tf-ti);
a0=qi;
q=a1*t+a0;
v=a1

input('Deseja continuar?')

for i=1:1:length(t)-1
    if (q(i+1)-q(i))>0
        sentido(i)=0;
    else sentido(i)=1;
    end
end

extraTime=(tf+ dt):dt:tf+0.3;
t=[t extraTime];
q=[q (qf*ones(1,length(extraTime)))];
sentido=[sentido zeros(1,length(extraTime))];
sentido= [sentido 0];
end
```

APÊNDICE B – Função para Calcular Trajetória Trapezoidal Desenvolvida no *Matlab*®

```
function [ t, q, sentido ] = trajetoriaTrapezoidal( ti, qi, qf, dt)
vmax=300;
amax=10*vmax;

%Torricelli pra saber se vai atingir Vmax
V = sqrt(amax/2*qf);
if vmax > V
    vmax = V;
end
vmax
input('pressione para continuar')

a0=qi;
a1=0;
a2=amax/2;
t1=vmax/amax;

b0=(vmax^2)/(2*amax)+qi;
b1=vmax;

c0=qf-(vmax^2)/(2*amax);
t2=(qf-qi)/vmax;

c1=vmax;
c2=-amax/2;
tf=t2-c1/(2*c2);

t=ti:dt:tf;
for i=1:1:length(t)
    if (t(i)>=t2)
        q(i)=c2*(t(i)-t2)^2+c1*(t(i)-t2)+c0;
    else if (t(i)>=t1)
        q(i)=b1*(t(i)-t1)+b0;
    else
        q(i)=a2*(t(i)-ti)^2+a1*(t(i)-ti)+a0;
    end
end
end

for i=1:1:length(t)-1;
    if (q(i+1)-q(i))>0
        sentido(i)=0;
    else sentido(i)=1;
    end
end
sentido= [sentido 0];

extraTime=(tf+ dt):dt:tf+0.2;
t=[t extraTime];
q=[q (qf*ones(1,length(extraTime)))];
sentido=[sentido zeros(1,length(extraTime))];
end
```

APÊNDICE C – Função para Calcular Trajetória Cúbica Desenvolvida no *Matlab*®

```
function [ t, q, sentido ] = trajetoriaCubica(ti, tf, qi, qf, dt)
t=ti:dt:tf;

a3=-2*(qf-qi)/(tf-ti)^3;
a2=3*(qf-qi)/(tf-ti)^2;
a0=qi;

q=a3*(t-ti).^3+a2*(t-ti).^2+a0;

v=3*a3*(t-ti).^2+2*a2*(t-ti);

plot(t,v)
input('Deseja continuar?')

for i=1:1:length(t)-1
    if (q(i+1)-q(i))>0
        sentido(i)=0;
    else sentido(i)=1;
    end
end

extraTime=(tf+ dt):dt:tf+0.2;
t=[t extraTime];
q=[q (qf*ones(1,length(extraTime)))];
sentido=[sentido zeros(1,length(extraTime))];
sentido= [sentido 0];
end
```

APÊNDICE D – Função para Calcular Trajetória de Quinto Grau Desenvolvida no *Matlab*®

```
function [ t, q, sentido ] = trajetoriaQuintica( tf, ti, qi, qf, dt )
t=ti:dt:tf;

a5=12*(qf-qi)/(2*(tf-ti)^5);
a4=30*(qi-qf)/(2*(tf-ti)^4);
a3=20*(qf-qi)/(2*(tf-ti)^3);
a0=qi;

q=a5*(t-ti).^5+a4*(t-ti).^4+a3*(t-ti).^3+a0;

v=5*a5*(t-ti).^4+4*a4*(t-ti).^3+3*a3*(t-ti).^2;

plot(t,v)
input('Deseja continuar?')
for i=1:length(t)-1
    if (q(i+1)-q(i))>0
        sentido(i)=0;
    else sentido(i)=1;
    end
end

extraTime=(tf+ dt):dt:tf+0.3;
t=[t extraTime];
q=[q (qf*ones(1,length(extraTime)))];
sentido=[sentido zeros(1,length(extraTime))];
sentido= [sentido 0];
end
```

**APÊNDICE E – Função para Converter Trajetória Calculada em Sinal Desenvolvida no
*Matlab®***

```
function [ tempo, N, S] = trajetoriaParaSinal( tempo, posicao, ppr,  
passo )  
dt=tempo(2);  
pulsos=posicao*ppr/passo;  
N=floor(abs(pulsos));  
for i=1:1:length(tempo);  
    delta(i)=abs(pulsos(i)-N(i));  
    if delta(i)<0.5  
        S(i)=0;  
    else if delta(i)>=0.5 && delta(i)<1  
        S(i)=5;  
    else S(i)= 100;  
    end  
end  
  
end  
end
```

APÊNDICE F – Script para Fornecer Sinal e Adquirir Resposta para Trajetória Degrau
Desenvolvida no *Matlab*®

```
clear all; close all; clc;
ppr=1024;
pprA=1024;
passo= 10;
x0=0;
t0=0;
rate=10^6-1;
dt=1/rate;

T=input('Digite o tempo em segundos que essa trajetória deve ser
percorrida: ');
tf=t0+T;

xf= input('Digite a posição do ponto X: ');
[timeX, x, sentido]=trajetoriaDegrau(tf,t0,x0,xf,dt);
[timeN, N, S]=trajetoriaParaSinal(timeX,x,ppr,passo);

plot(timeX,x,timeN,N*passo/pprA)

s= daq.createSession('ni');
s.Rate=rate;

chA=addAnalogOutputChannel(s,'Dev1','ao1','Voltage');
chD=addDigitalChannel(s,'Dev1','port0/line1','OutputOnly');
chP=addCounterInputChannel(s,'Dev1','ctrl','Position');
chP.EncoderType='X1';
queueOutputData(s,[S' sentido']);
[data, timeA]=startForeground(s);
release(s);

counterNBits = 32;
signedThreshold = 2^(counterNBits-1);
signedData = (data(:,1))';
signedData(signedData > signedThreshold) = signedData(signedData >
signedThreshold) - 2^counterNBits;

position=passo*signedData/pprA;
figure(1)
plot(timeX,x,timeA',position)
title('Posição no tempo')
ylabel('Posição (mm)')
xlabel('Tempo (s)')
legend('Gerado','Adquirido')
grid on
figure(2)
plot(timeX,x-position)
erro=100*(1-abs(position(end)/(xf-x0)))
title('Diferença de posicionamento no tempo')
ylabel('Erro (mm)')
xlabel('Tempo (s)')
grid on
```

APÊNDICE G – Script para Fornecer Sinal e Adquirir Resposta para Trajetória Trapezoidal Desenvolvida no *Matlab*®

```
clear all; close all; clc;
ppr=1024;
pprA=1024;
passo= 10;
x0=0;
t0=0;
rate=10^6-1;
dt=1/rate;

xf= input('Digite a posição do ponto X: ');
[timeX, x, sentido]=trajetoriaTrapezoidal(t0,x0,xf,dt);
[timeN, N, S]=trajetoriaParaSinal(timeX,x,ppr,passo);

plot(timeX,x)
title('Posição no tempo')
ylabel('Posição')
xlabel('Tempo (s)')

s= daq.createSession('ni');
s.Rate=rate;
chA=addAnalogOutputChannel(s,'Dev1','ao1','Voltage');
chD=addDigitalChannel(s,'Dev1','port0/line1','OutputOnly');
chP=addCounterInputChannel(s,'Dev1','ctrl','Position');
chP.EncoderType='X1';
queueOutputData(s,[S' sentido']);
[data, timeA]=startForeground(s);
release(s);

counterNBits = 32;
signedThreshold = 2^(counterNBits-1);
signedData = (data(:,1))';
signedData(signedData > signedThreshold) = signedData(signedData >
signedThreshold) - 2^counterNBits;

position=passo*signedData/pprA;
figure(1)
plot(timeX,x,timeA',position)
title('Posição no tempo')
ylabel('Posição (mm)')
xlabel('Tempo (s)')
legend('Gerado','Adquirido')
grid on
figure(2)
plot(timeX,x-position)
erro=100*(1-abs(position(end)/(xf-x0)))
title('Diferença de posicionamento no tempo')
ylabel('Erro (mm)')
xlabel('Tempo (s)')
grid on
```

APÊNDICE H – Script para Fornecer Sinal e Adquirir Resposta para Trajetória Cúbica
Desenvolvida no *Matlab*®

```
clear all; close all; clc;
ppr=1024;
pprA=1024;
passo= 10;
x0=0;
t0=0;
rate=10^6-1;
dt=1/rate;

T=input('Digite o tempo em segundos que a trajetória deve ser
percorrida: ');
tf=t0+t0+T;
xf= input('Digite a posição final em X: ');
[timeX, x, sentido]=trajetoriaCubicaExtendida(t0,tf,x0,xf,dt);
[timeN, N, S]=trajetoriaParaSinal(timeX,x,ppr,passo);

s= daq.createSession('ni');
s.Rate=rate;
chA=addAnalogOutputChannel(s,'Dev1','ao1','Voltage');
chD=addDigitalChannel(s,'Dev1','port0/line1','OutputOnly');
chP=addCounterInputChannel(s,'Dev1','ctrl1','Position');
chP.EncoderType='X1';
queueOutputData(s,[S' sentido']);
[data, timeA]=startForeground(s);
release(s);

counterNBits = 32;
signedThreshold = 2^(counterNBits-1);
signedData = (data(:,1))';
signedData(signedData > signedThreshold) = signedData(signedData >
signedThreshold) - 2^counterNBits;

position=passo*signedData/pprA;
figure(1)
plot(timeX,x,timeA',position)
title('Posição no tempo')
ylabel('Posição (mm)')
xlabel('Tempo (s)')
legend('Gerado','Adquirido')
grid on
figure(2)
plot(timeX,x-position)
erro=100*(1-abs(position(end)/(xf-x0)))
title('Diferença de posicionamento no tempo')
ylabel('Erro (mm)')
xlabel('Tempo (s)')
grid on
```

**APÊNDICE I – Script para Fornecer Sinal e Adquirir Resposta para Trajetória de Quinta
Ordem Desenvolvida no *Matlab*®**

```
clear all; close all; clc;
ppr=1024;
pprA=1024;
passo= 10;
x0=0;
t0=0;
rate=10^6-1;
dt=1/rate;

T=input('Digite o tempo em segundos que essa trajetória deve ser
percorrida: ');
tf=t0+T;

xf= input('Digite a posição do ponto X: ');
[timeX, x, sentido]=trajetoriaQuintica(tf,t0,x0,xf,dt);
[timeN, N, S]=trajetoriaParaSinal(timeX,x,ppr,passo);

plot(timeX,x,timeN,N*passo/pprA)

s= daq.createSession('ni');
s.Rate=rate;

chA=addAnalogOutputChannel(s,'Dev1','ao1','Voltage');
chD=addDigitalChannel(s,'Dev1','port0/line1','OutputOnly');
chP=addCounterInputChannel(s,'Dev1','ctrl','Position');
chP.EncoderType='X1';
queueOutputData(s,[S' sentido']);
[data, timeA]=startForeground(s);
release(s);

counterNBits = 32;
signedThreshold = 2^(counterNBits-1);
signedData = (data(:,1))';
signedData(signedData > signedThreshold) = signedData(signedData >
signedThreshold) - 2^counterNBits;

position=passo*signedData/pprA;
figure(1)
plot(timeX,x,timeA',position)
title('Posição no tempo')
ylabel('Posição (mm)')
xlabel('Tempo (s)')
legend('Gerado','Adquirido')
grid on
figure(2)
plot(timeX,x-position)
erro=100*(1-abs(position(end)/(xf-x0)))
title('Diferença de posicionamento no tempo')
ylabel('Erro (mm)')
xlabel('Tempo (s)')
grid on
```