CONTROLO INTELIGENTE DE UM SISTEMA DE LEVITAÇÃO MAGNÉTICA

Miguel Ribeiro de Sousa Cardoso

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Automação e Sistemas
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2015

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha da Unidade Curricular de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Miguel Ribeiro de Sousa Cardoso, Nº 1100984, 1100984@isep.ipp.pt
Orientação científica: Ramiro de Sousa Barbosa, rsb@isep.ipp.pt

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Automação e Sistemas
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
23 de outubro de 2015

Resumo

A levitação magnética tem sido um tema bastante investigado sobretudo devido à sua utilização em sistemas ferroviários de transportes. É o método ideal quando existe a necessidade em aplicações de restringir do contacto físico, ou a conveniência, em termos energéticos, de eliminar o atrito. O princípio de funcionamento é simples, um eletroíman cria uma força sobre um objeto ferromagnético que contraria a gravidade. Contudo um sistema de levitação por atração é instável e não linear, o que significa a necessidade de implementar um controlador para satisfazer as características de estabilidade desejadas.

Ao longo deste projeto serão descritos os procedimentos teóricos e práticos que foram tomados na criação de um sistema de levitação eletromagnética. Desde a conceção física do sistema, como escolha do sensor, condicionamento de sinal ou construção do eletroíman, até aos procedimentos matemáticos que permitiram a modelação do sistema e criação de controladores. Os controladores clássicos, como o PID ou em avanço de fase, foram projetados através da técnica do Lugar Geométrico de Raízes. No projeto do controlador difuso, pelo contrário não se fez uso da modelação do sistema ou de qualquer relação matemática entre as variáveis. A utilização desta técnica de controlo destacou-se pela usa simplicidade e rapidez de implementação, fornecendo um bom desempenho ao sistema.

Na parte final do relatório os resultados obtidos pelos diferentes métodos de controlo são analisados e apresentadas as respetivas conclusões. Estes resultados revelam que para este sistema, relativamente aos outros métodos, o controlador difuso apresenta o melhor desempenho tanto ao nível da resposta transitória, como em regime permanente.

Palavras-Chave

Levitação Eletromagnética, Controlador Difuso, Controlador PID, MATLAB.

Abstract

The magnetic levitation has been a topic quite investigated mainly due to its use in rail transport systems. This method is ideal when there is a need in applications for restricting the physical contact, or the convenience, in terms of energy, to eliminate friction. The operating principle is simple, one coil creates a force on a ferromagnetic object that goes against the force of gravity. Yet levitation system by attraction is unstable and nonlinear, which means the need to implement a controller to meet the desired stability features.

Throughout this project it will be described the theoretical and practical procedures that have been taken to create an electromagnetic levitation system. The physical design of the system, such as the sensor selection, signal conditioning or the construction of the coil, to the mathematical procedures that enabled the system modeling and creation of the controllers are explained. The classic controllers, such as the PID or phase-lead controller, were designed using the root locus technique. In the fuzzy controller design, however, it was not used any type of system modeling or mathematical relations between variables. The use of this control technique stands out for its simplicity and fast implementation, providing a good performance to the levitation system.

The results obtained by the different control methods are then analyzed and the conclusions are presented. These results reveal that for this system, compared with other control methods, the fuzzy controller performs better both in terms of the transient response and steady state.

Keywords

Electromagnetic Levitation, Fuzzy Controller, PID Controller, MATLAB.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

AMPOP	–	Amplificador Operacional
A/D	–	*Analog-to-Digital Converter*
D/A	–	*Digital-to-Analog Converter*
EDL	–	*Electrodynamic Levitation*
EML	–	*Electromagnetic Levitation*
FEM	–	Força Eletromotriz
IDE	–	*Integrated Development Environment*
LGPL	–	GNU *Lesser General Public License*
LGR	–	Lugar Geométrico de Raízes
Maglev	–	*Magnetic Levitation*
MIMO	–	*Multiple Input Multiple Output*
MOSFET	–	*Metal Oxide Semiconductor Field Effect Transistor*
PD	–	Proporcional-Derivativo
PID	–	Proporcional-Integral-Derivativo
PWM	–	*Pulse Width Modulation*
SQL	–	*Superconductor Levitation*
SISO	–	*Single Input Single Output*

1. INTRODUÇÃO

A levitação magnética é uma tecnologia com grande potencial futuro, com aplicações práticas que cativam tanto pelo aspeto visual como pelas funcionalidades. Criar um sistema mecânico estável para levitar um objeto sem necessidade de contacto, onde a força gravítica é contrariada apenas por forças magnéticas tem sido um objetivo de engenheiros e cientistas desde à muitos anos a esta parte. No entanto, as características de instabilidade e não-linearidade inerentes a este tipo de sistemas torna-os difíceis de controlar, o que atrasou durante várias décadas a aplicação prática desta tecnologia.

Esta dificuldade em estabilizar a suspensão de um objeto quer por repulsão quer por atracão magnética foi estudada em 1842 por Samuel Earnshaw. No seu trabalho é demonstrado matematicamente a impossibilidade de um estado de equilíbrio mecânico estacionário entre duas cargas pontuais [1]. O trabalho de Earnshaw passou assim a ser um clássico para quem estuda o tema da levitação magnética, e a impossibilidade de equilíbrio estável quando uma lei do inverso do quadrado opera, passou a ser conhecido como "Teorema de Earnshaw".

No campo da suspensão magnética existem contudo meios de "contrariar" este teorema. Um desses meios é estabelecer uma variação do campo magnético ao longo do tempo, com o propósito de fornecer a intensidade necessária para ocorrer equilíbrio. Esta variação é

conseguida recorrendo ao uso de um eletroíman, que em conjunto com um sensor apropriado para determinar a posição do objeto permitem criar um controlo em malha fechada. Este será o método de levitação magnética abordado no trabalho.

A estrutura base para este projeto segue os sistemas tradicionais criados quando se pretende estudar o fenómeno da levitação magnética. Um eletroíman é colocado no topo de uma estrutura fixa, funcionando como único atuador responsável por gerar a força magnética. Como objeto a levitar é usada um objeto metálico cuja posição é determinada por um sensor. A informação do sensor é transmitida a um controlador que determina em cada instante qual a corrente a fornecer ao eletroíman, dependendo da proximidade do objeto com este. Se a estrutura física do projeto é comum à grande maioria dos trabalhos desenvolvidos nesta área, o processo de controlo efetuado é muito diversificado, podendo ser utilizado desde um controlo totalmente analógico, onde são implementados compensadores recorrendo a amplificadores operacionais (AMPOPs) até técnicas de controlo mais avançadas como lógica difusa ou redes neurais artificiais.

Para este trabalho optou-se por implementar no sistema de levitação magnética um controlador baseado em lógica difusa, assim como controladores clássicos de modo a poder comparar os modos de implementação de cada um, e no final concluir quanto à melhor opção.

1.1. CONTEXTUALIZAÇÃO

Este projeto surgiu do desejo de realizar o controlo de um sistema não-linear através de técnicas de controlo inteligente. Os sistemas de levitação magnética apresentam as características ideais para o estudo deste tipo de controladores, juntando à instabilidade inerente a estes sistemas, o entusiasmo de uma tecnologia recente e com rápido crescimento em aplicações práticas.

Apesar do elevado número de trabalhos já publicados nesta área, ainda são poucos os que se focam em técnicas de controlo inteligente para efetuar a levitação magnética, o que demonstra a complexidade deste tipo de projetos.

1.2. OBJETIVOS

O principal objetivo desta Tese de Mestrado é a criação de um sistema de levitação magnética com um grau de liberdade, onde seja possível controlar a posição de uma esfera metálica usando apenas a força magnética produzida por um eletroíman. No final do projeto o sistema deve ser capaz de estabilizar a localização do objeto através de um controlo por lógica difusa.

A implementação de diferentes tipos de controlo permite cumprir um segundo objetivo desta dissertação, o de comparar os métodos de controlo e concluir qual o mais apropriado para este tipo de sistemas.

1.3. CALENDARIZAÇÃO

A forma como foram organizadas as tarefas durante a execução do trabalho pode ser consultada na Tabela 1. Nesta podem ser examinadas as tarefas e o tempo que cada uma demorou durante o período de realização do projeto.

Tabela 1 Calendarização do projeto

ID	Nome da Etapa	Início	Fim	Duração
1	Definição de objetivos Estudo do estado da arte	15-12-2014	5-01-2015	3 Semanas
2	Estudo dos requisitos de *Hardware*	5-01-2015	12-01-2015	1 Semana
3	Desenvolvimento de *Hardware*	13-01-2015	2-02-2015	3 Semanas
5	Modelação e simulação do sistema	3-02-2015	3-03-2015	4 Semanas
7	Desenvolvimento dos controladores clássicos	3-03-2015	16-03-2015	2 Semanas
8	Experiências com controladores clássicos	16-03-2015	23-03-2015	1 Semana
9	Desenvolvimento do controlador difuso	23-03-2015	13-04-2015	3 Semanas

10	Experiências com controlador difuso	13-04-2015	27-04-2015	2 Semanas
11	Desenvolvimento de interface gráfico	27-04-2015	11-05-2105	2 Semanas
12	Experiências finais com interface gráfico	11-05-2015	25-05-2015	2 Semanas
13	Elaboração do relatório final	3-08-2015	5-10-2015	11 Semanas

1.4. ORGANIZAÇÃO DO RELATÓRIO

O relatório é constituído por 8 capítulos: Introdução, Estado da Arte, Revisão Bibliográfica, Construção do Sistema, Modelação do Sistema, Projeto e Simulação dos Controladores, Implementação Prática e Análise de Resultados e Conclusão.

No Capítulo 1 é contextualizado o trabalho no âmbito do tema "Levitação Magnética", definidos os objetivos a atingir e a calendarização para atingir esses objetivos.

No Capítulo 2, "Estado da Arte", apresenta-se os fundamentos teóricos que suportam o projeto, os tipos de levitação magnética, as aplicações práticas que já existem com esta tecnologia e por fim um estudo teórico dos controladores a implementar.

No Capítulo 3 é feita uma revisão bibliográfica de trabalhos já realizados neste tema.

No Capítulo 4 é feita uma descrição da componente física do sistema, bem como do programa de interface com o utilizador.

No Capítulo 5 é feita uma descrição detalhada do sistema incluindo uma justificação para a escolha dos componentes e a modelação matemática dos mesmos.

No Capítulo 6 são projetados os controladores clássicos e o controlador difuso e é feita uma descrição da sua implementação. São ainda realizadas simulações em ambiente MATLAB dos controladores obtidos.

No Capítulo 7, "Implementação Prática e Análise de Resultados" são implementados os controladores desenvolvidos na prática e analisados os resultados alcançados.

No Capítulo 8, “Conclusão”, apresenta-se as conclusões do trabalho e possíveis melhorias futuras.

2. ESTADO DA ARTE

As observações de fenómenos magnéticos naturais são muito antigas. Há mais de 2000 anos atrás os gregos conheciam uma pedra que atraía pedaços de ferro [2], e estas foram as primeiras e durante muitos anos, as únicas fontes de campo magnético conhecidas. O estudo moderno do magnetismo começou no início do século XIX, quando Jean Baptiste Biot e Felix Savart usaram um íman permanente para medir a força nas proximidades de um fio comprido e analisaram os resultados em termos do campo magnético por elementos de corrente ao longo do fio [2]. Pela primeira vez foi relacionada a presença de um campo magnético com a presença de corrente elétrica. Em 1826, André-Marie Ampère continuou o estudo desta relação, mostrando que um fio percorrido por corrente também experimenta uma força na presença de um campo magnético e que dois fios percorridos por uma corrente se atraem ou se repelem mutuamente, dependendo do sentido da corrente [2].

A lei de Biot-Savart, representada na equação (1), é usada para o cálculo de um campo magnético $d\vec{\boldsymbol{B}}$ produzido por um elemento de corrente $I.d\vec{\boldsymbol{L}}$:

$$d\vec{\boldsymbol{B}} = \frac{\mu_0}{4\pi} \frac{I.d\vec{\boldsymbol{L}} \times \hat{\boldsymbol{r}}}{r^2} \quad (1)$$

Esta equação refere que um campo magnético criado por um elemento de corrente I e de comprimento $d\vec{\boldsymbol{L}}$, cria num ponto p_1 situado na direção radial $\hat{\boldsymbol{r}}$, o campo magnético $d\vec{\boldsymbol{B}}$.

Para calcular o campo magnético total produzido por uma corrente, deve-se somar (integrar) o resultado de todos os elementos de corrente no circuito. Este cálculo é extremamente complexo, exceto quando são usadas geometrias de circuitos mais simples.

Uma dessas geometrias é o solenóide. Um solenóide consiste em enrolar fios em forma de hélice, formando um cilindro onde as espiras se encontram sobrepostas. Esta forma, é usada para criar um campo magnético uniforme no seu interior quando percorrido por corrente, semelhante a um campo magnético produzido por um íman em forma de barra, como mostra a Figura 1.

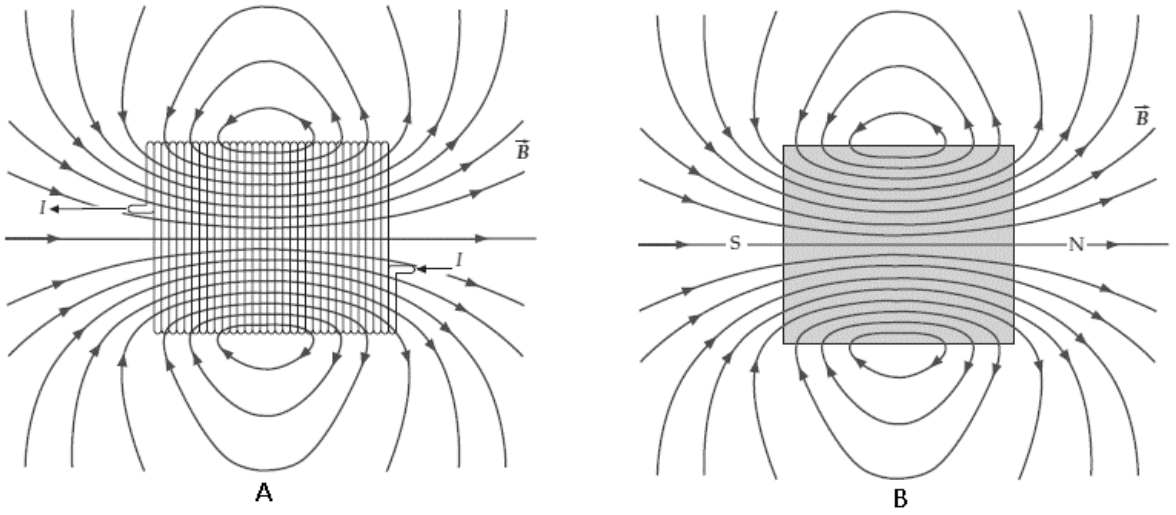

Figura 1 Linhas de campo magnético num solenóide (A) e íman permanente (B)

Na Figura 1 é visível uma representação das linhas de campo magnético tanto num íman (B), como num solenóide (A) quando percorrida por uma corrente. A forma paralela ao eixo das linhas do centro indicam um campo magnético intenso. De referir também que estas linhas ao contrário do campo elétrico são fechadas, e com o sentido Norte-Sul.

A lei de Biot-Savart pode ser usada no cálculo do campo magnético no interior do solenóide. Considere-se um solenóide de comprimento L, com N voltas de fio, percorrido por uma corrente I, como mostra a Figura 2, onde z_1 e z_2 representam as extremidades. Para um comprimento dz' à distância z' da origem, se $n = N/L$ é o número de voltas por unidade de

comprimento, existem $n.dz$ voltas de fio nesse elemento selecionado. O elemento é assim equivalente a uma única espira, percorrida por uma corrente $di = n.I.dz'$.

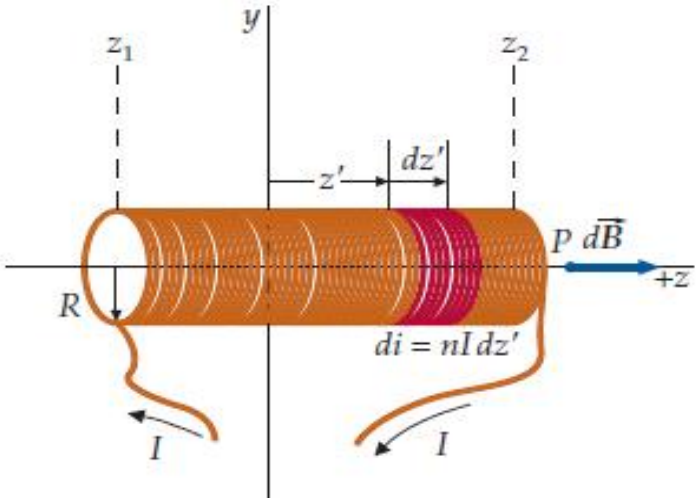

Figura 2 Representação geométrica de um solenóide [2]

O campo magnético num ponto sobre o eixo z devido à espira na origem, com corrente $n.I.dz$ é dado pela equação (2):

$$d\boldsymbol{B_z} = \frac{\mu_0}{4\pi}\frac{2\pi R^2 nIdz\prime}{(z\prime^2+R^2)^{3/2}} \tag{2}$$

Integrando de z_1 a z_2 a equação (2) é possível determinar o campo magnético na origem devido à totalidade do solenóide:

$$\boldsymbol{B_z} = \frac{\mu_0}{4\pi}\,2\pi R^2 nI\int_{z_1}^{z_2}\frac{dz}{(z\prime^2+R^2)^{3/2}}$$

$$\boldsymbol{B_z} = \frac{1}{2}\,\mu_0 nI\left(\frac{z_2}{\sqrt{z_2^2+R^2}} - \frac{z_1}{\sqrt{z_1^2+R^2}}\right) \tag{3}$$

Se for considerado que o solenóide tem um comprimento L muito superior ao raio, a expressão da campo magnético pode ser simplificada:

$$\boldsymbol{B_z} = \mu_0 nI \tag{4}$$

Outro conceito teórico pertinente para o estudo da levitação magnética é o da força magnética. Qualquer força exercida num objeto pode ser vista como a variação do trabalho exercido sobre esse mesmo objeto. Assim no caso da força magnética:

$$F_{mag} = \frac{dW}{dx} \tag{5}$$

Onde W é o trabalho magnético dado por:

$$W = \frac{1}{2} L(x).i^2 \tag{6}$$

O parâmetro $L(x)$ representa a soma de duas indutâncias: L_c, que é um valor fixo da indutância do solenóide; e L_b, uma parcela incremental criada pela presença de um material ferromagnético perto do solenóide [3]. No capítulo da modelação do sistema serão usadas as equações (5) e (6), para determinar a força magnética exercida no objeto a levitar.

2.1. MAGNETISMO NA MATÉRIA

Quanto às propriedades magnéticas os materiais podem ser divididos em três categorias: **paramagnéticos**, **ferromagnéticos** e **diamagnéticos.** Esta classificação está relacionada com propriedades atómicas dos materiais, isto é, do modo como os eletrões se comportam na presença de um campo magnético. Resumidamente o movimento orbital dos eletrões em torno do núcleo e o movimento de rotação sobre eles próprios geram um campo magnético. De um modo geral, como as órbitas dos eletrões não têm uma orientação pré-definida, os campos magnéticos que estes originam cancelam-se. O comportamento destas órbitas com o aproximar de um íman permanente categoriza estes materiais em termos magnéticos.

Um material diz-se não-magnético se as órbitas mantiverem a sua orientação, o que faz com que o objeto não seja atraído nem repelido pelo íman. Se os materiais reagirem, então são magnéticos. Deste tipo existe a classificação de paramagnéticos para os materiais que quando expostos a um campo exterior tendem a alinhar paralelamente o seu campo, mas que é contrabalançada pela agitação térmica, que tende a desalinhar os momentos [2]; os ferromagnéticos, que na presença de um pequeno campo magnético externo ocorre um alto grau de alinhamento dos momentos magnéticos, e em alguns casos este alinhamento persiste mesmos depois de o campo magnético ser removido; e os diamagnéticos que possuem um momento magnético total igual a zero, e sob um campo magnético exterior, são repelidos.

Esta classificação dos elementos é relevante, as noções deste tipo de propriedade permite aumentar o campo magnético produzido por um solenóide quando escolhido convenientemente o material a usar no núcleo.

Quando um material é submetido a um campo magnético, como o de um solenóide, o campo magnético tende a alinhar os momentos magnéticos que existem no interior do material [2], passando este a ter propriedades de um íman magnético. Este fenómeno é descrito por magnetização.

No caso do eletroíman produzido para este projeto, um cilindro do material ferromagnético foi introduzido no interior do solenóide. Quando o solenóide é percorrida por uma corrente, esta adquire uma magnetização $\vec{B}_{ap}$, que por sua vez magnetiza o material, fazendo com que este adquira uma magnetização $\vec{M}$. O campo magnético resultante num ponto interior do solenóide é dado pela soma destes dois campos magnéticos [2]:

$$\vec{B} = \vec{B}_{ap} + \mu_0 \vec{M} \tag{7}$$

No caso dos matérias paramagnéticos e ferromagnéticos, $\vec{M}$ tem o mesmo sentido que $\vec{B}_{ap}$, tornando o campo resultante mais forte. No caso dos materiais diamagnéticos têm sentidos opostos.

2.2. TIPOS DE LEVITAÇÃO MAGNÉTICA

De seguida serão apresentados os principais modos de levitação magnética bem como vantagens e desvantagens de cada método. O princípio base da levitação magnética é produzir forças capazes de sustentar um corpo sem nenhum contacto físico porém, o modo como este objetivo é alcançado, faz a separação desta levitação em três categorias:

1. Levitação Eletrodinâmica (EDL)
2. Levitação Eletromagnética (EML)
3. Levitação por Supercondução (SQL)

2.2.1. LEVITAÇÃO ELETRODINÂMICA

Para ocorrer este tipo de levitação é necessário existir um campo magnético em movimento relativamente a um material condutor. Este movimento faz com que o material fique submetido a uma variação de fluxo magnético, que de acordo com a lei da indução de Faraday induz uma tensão que criará correntes parasitas no material condutor. Estas correntes geram um campo magnético que contraria o campo magnético inicial. Da interação entre estes dois campos magnéticos surge uma força responsável pela levitação do corpo que possui o campo gerado.

Este tipo de levitação apresenta como principal desvantagem a necessidade de existir movimento da parte de um corpo. A principal vantagem é o reduzido custo dos materiais possíveis de serem usados em comparação com os outros tipos de levitação.

2.2.2. LEVITAÇÃO ELETROMAGNÉTICA

A levitação eletromagnética consiste na atração entre um eletroíman e um material ferromagnético. Neste tipo de levitação, que será utilizado neste trabalho, o campo magnético é gerado através do eletroíman, e a força produzida controlada por um sistema em malha fechada para manter a levitação estável.

Este método tem como vantagem a não necessidade de movimento por parte de nenhum corpo, e como desvantagem a maior complexidade de controlo requerida para o sistema.

2.2.3. LEVITAÇÃO POR SUPERCONDUÇÃO

Este tipo de levitação baseia-se no efeito Meisne de exclusão de campo magnético do interior dos supercondutores [12]. Para este tipo de levitação é necessária a presença de uma fonte de campo magnético como no caso da EDL. Um material específico é arrefecido até uma temperatura onde adquire propriedades supercondutoras. Estas propriedades fazem com que os materiais quando colocados na presença de um campo magnético exterior tendam a repelir a totalidade do campo magnético aplicado, o que leva a um efeito de levitação estável. A imagem de um tipo desta levitação está representada na Figura 3.

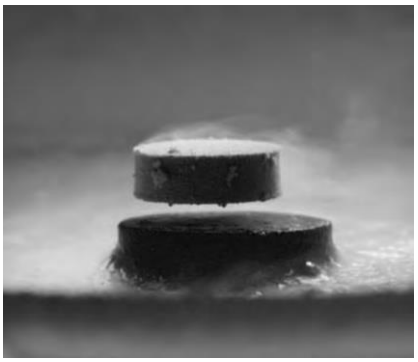

Figura 3 Levitação por supercondução[2]

A grande vantagem deste tipo de levitação é a estabilidade conseguida de forma natural, sem a necessidade de consumo de energia. Como desvantagem requer manter os materiais supercondutores a temperaturas muito baixas, para não perderem as suas propriedades. Ainda não existe qualquer tipo de aplicação prática do dia-a-dia com este tipo de levitação.

2.3. APLICAÇÕES DA LEVITAÇÃO MAGNÉTICA

A levitação magnética é uma tecnologia avançada, que apesar de recente já possui várias aplicações práticas. O ponto comum entre estas aplicações é a inexistência de contacto físico entre duas superfícies, o que elimina as forças de fricção. Este facto leva ao aumento da eficiência, redução de custos de manutenção, e aumento da vida útil dos sistemas. Já existem vários países que usam este tipo de tecnologia nas mais diversas áreas. De seguida são apresentados alguns exemplos dessa utilização.

Os princípios da levitação magnética são conhecidos há mais de 90 anos, quando em 1921 os cientistas americanos Robert Goddard e Emile Bachelet idealizaram o comboio de levitação magnética [21]. Contudo, apesar de os comboios terem um maior foco de interesse, esta tecnologia não se limita aos transportes. As aplicações da levitação magnética podem ser categorizadas do seguinte modo, dentro da área da engenharia:

1. Engenharia de transportes (comboio Maglev)
2. Engenhar do ambiente (turbinas de vento)
3. Engenharia aeroespacial (lançamento de foguetões)
4. Engenharia de armamento militar

5. Engenharia nuclear (centrifugador do reator nuclear)
6. Engenharia civil (elevadores, sistemas de ventilação)
7. Engenharia biomédica (bomba de coração)
8. Engenharia alimentar (avaliação de alimentos e bebidas)
9. Engenharia eletrotécnica (eletroíman)
10. Arquitetura (mobiliário)

COMBOIOS DE LEVITAÇÃO MAGNÉTICA

De entre as aplicações que usam tecnologia de levitação magnética, a mais importante é claramente os comboios Maglev. Estes comboios são hoje em dia os veículos mais avançados na indústria do transporte ferroviário.

Contrariamente aos veículos de carris tradicionais, não existe contacto físico entre os comboios e os carris. Movem-se ao longo de um campo magnético estabelecido entre o veículo e as guias que devem ser seguidas. Estas características de não contacto mecânico nem fricção proporciona que os comboios possam atingir grandes velocidades, sendo o recorde para este tipo de veículos de 581 Km/h [22]. Existe uma grande variedade de sistemas de transportes Maglev, e investigadores continuam a revelar novas ideias para novos sistemas.

Mas as vantagens deste tipo de transportes não se prende apenas com o facto de atingirem grandes velocidades ou o aspeto visual atrativo. Estes sistemas de transporte são uma resposta às necessidades do mundo atual. Os comboios Maglev são melhores em vários aspetos em relação aos comboios convencionais, incluindo a reduzida poluição, eliminação de ruído, baixos níveis de vibração, espaço de ocupação, velocidade, aceleração, custos de manutenção, segurança e tempo de viagem.

Os comboios de levitação magnética podem ser divididos em dois grupos, dependendo do tipo de levitação magnética que usam: Levitação por suspensão Eletromagnética ou Levitação por suspensão Eletrodinâmica. Existe uma grande variedade de veículos produzidos de acordo com estes dois princípios. Em ambos os sistemas existem três funções

principais que as forças magnéticas devem desempenhar: Levitação, Propulsão e Orientação. Em sistemas EDS o veículo levita entre 1 a 10 cm acima do carril usando forças repulsivas, em sistemas EMS o veículo levita entre 1 e 2 cm [23]. Os comboios Maglev Transrapid na Alemanha [10], ou o projeto suíço SwissRapid [11], são exemplos de EML. Levitação magnética EDS é utilizado no comboio japonês SCMaglev [9].

LANÇAMENTO DE FOGUETÕES

Uma aplicação da levitação magnética não tão conhecida como a anterior é o uso da levitação magnética para o lançamento de foguetões para o espaço (Figura 4). O centro espacial Marshall's Advanced Space Transportation da NASA, nos EUA, está a desenvolver tecnologias de levitação magnética que sajam capazes de lançar veículos com velocidade suficiente para ficar livre das forças gravíticas da Terra. O sistema irá usar campos magnéticos para levitar e acelerar o veículo ao longo de um traçado até atingir 600 Km/h [25]. Este sistema uma vez instalado poderá reduzir drasticamente o custo destas operações, bem como o peso destes veículos.

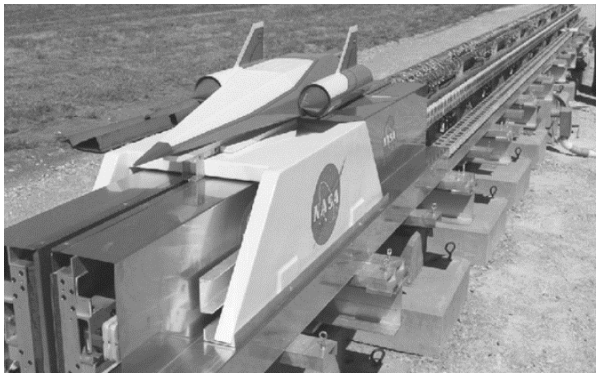

Figura 4 Pista magnética de lançamento de foguetões [25]

TURBINAS MAGLEV

As turbinas Maglev proporcionam um desempenho superior às tradicionais, com baixo nível de ruído, e maior tempo de vida. Usando forças magnéticas, estas turbinas não apresentam fricção entre as hélices e a parte fixa. As turbinas Maglev funcionam com os mesmos princípios de levitação apresentados até agora, mas estas não são apenas usadas para manter as turbinas a levitar, mas também para assegurar uma estabilidade durante os 360º do movimento.

Esta tecnologia pode ser aplicada a vários produtos, que requerem alto nível de transferência de calor, como computadores, projetores ou sistemas de áudio [29].

BOMBAS CARDÍACAS ARTIFICIAIS

Na área da medicina a tecnologia de levitação magnética tem maior expressão na criação de bombas de coração. Tradicionalmente, estas bombas de coração são efetuadas usando rolamentos, o que leva a um maior contacto entre o sangue e a bomba. Com o desenvolvimento da tecnologia de levitação magnética, as bombas cardíacas artificiais permitem evitar problemas como a fricção e necessidade de lubrificação, reduzindo os danos nas células de sangue, e aumentando o tempo de vida útil do aparelho [23].

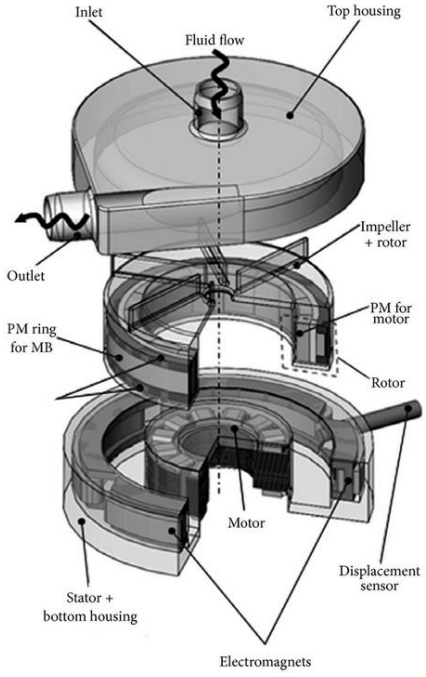

Figura 5 Configuração de um implante Maglev CBP [23]

A Figura 5 mostra o esquema de um implante ventricular desenvolvido com tecnologia Maglev.

ANÁLISE DE ALIMENTOS

A medição da densidade de uma certa substância é uma atividade importante na indústria alimentar por fornecer informação quanto à composição química da substância. Esta medição pode determinar por exemplo o conteúdo de açúcar em bebidas. As tecnologias

existentes para fazer estas medições estão longe de ser ideais, sendo necessária uma tecnologia mais simples e barata [24]. Uma ilustração desta tecnologia está representada na Figura 6.

Os cientistas estão a tentar desenvolver um sensor que use levitação magnética para ir de encontro a estas necessidades, suspendendo amostras de sólidos ou líquidos com a ajuda de eletroímanes para medir a densidade. Como mostra a Figura 6 estas amostras são colocadas dentro da estrutura, e a distância que se deslocam dentro do líquido fornece a medida da densidade.

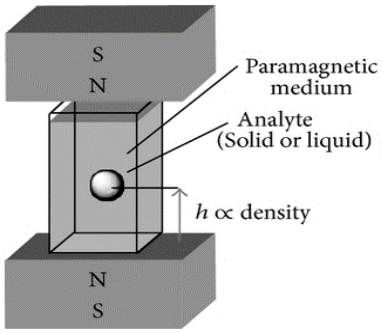

Figura 6 Sensor Maglev de medição de densidade [24]

Foram apresentados exemplos de uso de levitação magnética, alguns dos quais já largamente utilizados, existindo muitas outras aplicações não referidas. Os pontos comuns entre todas as aplicações incluem a eliminação de atrito por contacto, redução de custos de manutenção e duração de vida útil do produto. Os resultados desta pesquisa mostram claramente que a levitação magnética pode ser considerada uma solução para a melhoria de muitos produtos já existentes, bem como abrir possibilidade para a criação de novos.

2.4. METODOLOGIAS DE CONTROLO

Nos anos recentes o controlo de sistemas tem revelado uma importância cada vez maior no papel de desenvolvimento em civilizações modernas. Praticamente todos os aspetos da atividade do dia-a-dia é afetada por algum tipo de sistema de controlo. Mesmo o controlo de sistemas de levitação magnética são abrangidos pela teoria de controlo automático.

Os intervenientes básicos de um sistema de controlo podem ser descritos como [8]:

1. Objetos de controlo

2. Componentes do sistema de controlo

3. Resultados ou saídas

Os objetos de controlo podem ser identificados como as entradas ou sinais de atuação (no caso do sistema de levitação magnética, a corrente aplicada ao eletroíman). As saídas são as variáveis de controlo (posição do objeto). De um modo geral o objetivo de um sistema de controlo é controlar as saídas de um certo modo, dadas as entradas. Os componentes do sistema de controlo são variados e dependentes de cada aplicação, podendo ter condicionamento de sinais, controladores, conversores A/D, etc.

É comum num sistema de controlo existir uma relação entre a grandeza de saída e a grandeza de entrada, comparando-se e utilizando a diferença como meio de controlo. Tal acontece nos sistemas de levitação magnética, e este controlo é designado por controlo em malha fechada. Nestes sistemas o sinal de erro, que é a diferença entre o sinal de entrada e de realimentação, atua no controlador de modo a reduzir o erro e levar a saída para o valor desejado.

Também é possível a construção de sistemas onde o sinal de saída não afeta a ação do controlador, chamados de sistemas de controlo em malha aberta. A execução do sistema depende apenas da calibração, e uma presença de distúrbios leva a que o sistema de controlo não desempenhe a tarefa desejada. Na prática os sistemas de controlo em malha aberta são usados somente quando as relações entre entrada e saída do processo a controlar forem plenamente conhecidas e quando se tem a certeza de não existir distúrbios internos ou externos.

Outra definição importante no controlo de sistemas são as noções de sistemas lineares e sistemas não-lineares.

Na realidade não existem sistemas totalmente lineares, todos os sistemas físicos são não-lineares em alguma medida. Sistemas de controlo lineares são modelos idealizados e criados de forma analítica para simplificar a sua análise e conceção [8]. Assim quando a magnitude de sinais num sistema de controlo está limitada a um intervalo onde o sistema exibe características lineares, este pode ser considerado essencialmente linear.

Esta aproximação é efetuada, pois para sistemas lineares existe um grande número de técnicas de análise para a criação de controladores. Por outro lado, sistemas não lineares são

usualmente difíceis de tratar matematicamente e não existe um método geral disponível para resolver estes sistemas.

Todas as características descritas até agora levam ao principal objetivo de construir um sistema de controlo. Conhecendo o processo a controlar, o desenvolvimento de um sistema de controlo envolve três fases [8]:

1. Determinar o que o sistema deve fazer e como o deve fazer (especificações do sistema)
2. Determinar a configuração do controlador e como se liga ao restante sistema
3. Determinar os parâmetros do controlador para atingir os objetivos do ponto 1

As especificações do sistema são únicas para cada aplicação, e na maior parte das vezes incluem especificações sobre a estabilidade, erro em regime permanente, tempo de resposta e características de resposta em frequência. Em algumas aplicações é mesmo necessário especificações quanto à sensibilidade a variações de parâmetros ou rejeição de perturbações.

Existem vários modos de ligar o controlador ao restante sistema, como mostra a Figura 7.

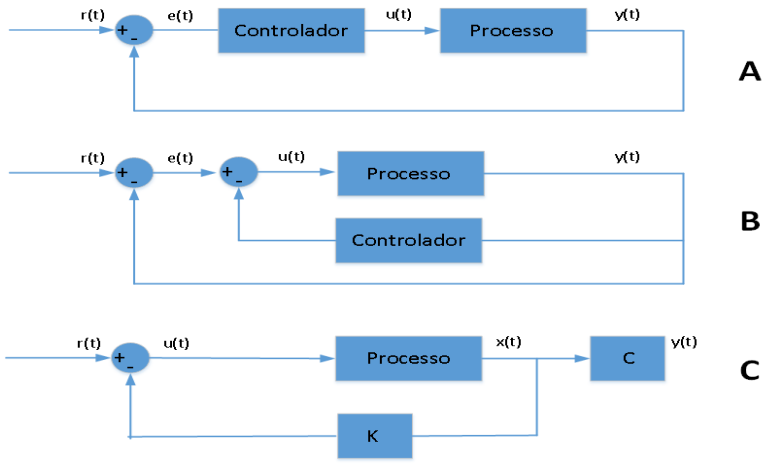

Figura 7 Diferentes configurações de controlo

A configuração em série (A) é a mais comum, com o controlador colocado em série com o processo. No controlo por realimentação (B), o controlador é colocado no menor caminho de retorno. No controlo por realimentação de entrada (C) o sistema gere um sinal de controlo

a partir de variáveis de estado. Na Figura 7 (C) apenas está marcado o controlo de uma variável.

Após a escolha da configuração do controlador ser feita devem ser encontrados os parâmetros de modo a atingir os objetivos definidos no ponto 1. Na prática a escolha do controlador deve ser o mais simples possível, desde que obedeça às características desejadas. Depois de escolhido o controlador a próxima etapa é determinar os valores dos parâmetros. Estes parâmetros são encontrados com técnicas pré-definidas como será demonstrado no capítulo 4, mas no entanto existem algumas regras que devem ser guias condutoras para propósito de projeto de controlador [3]:

1. Polos complexos conjugados da função de transferência em malha fechada leva a uma resposta subamortecida. Se todos os polos são reais, a resposta é sobreamortecida.

2. A resposta do sistema é dominada pelos polos mais próximos da origem do plano s.

3. Quanto mais à esquerda os polos dominantes no plano s estiverem, mais rápido o sistema irá responder.

4. Quando um polo e um zero da função de transferência quase se cancelam um ao outro, a porção da resposta do sistema associada a esse polo terá uma menor magnitude.

5. O domínio dos tempos e o domínio das frequências estão associados diretamente, por exemplo, o tempo de subida e a largura de banda são inversamente proporcionais.

2.4.1. CONTROLADORES PID

Hoje em dia a maioria dos controladores industriais em uso utiliza estratégias de controlo PID ou PID modificado [3]. Um controlador PID é um algoritmo de controlo de processo que tem uma componente de ação proporcional, integral e derivativa. Este controlador calcula o valor do erro entre uma variável medida e um valor desejado e tenta reduzir esse erro.

Como já referido o controlador PID engloba três ações: proporcional, integral e derivativo. De forma a simplificar estes termos podem ser interpretados em termos de tempo: P depende do presente erro, I da acumulação de erros passados, e D é a previsão de erros futuros.

$$U(t) = K_p.\left[e(t) + \frac{1}{T_i}.\int_0^t e(\tau)d\tau + T_d\frac{d}{dt}e(t)\right] \quad (8)$$

Quando se tem o modelo matemático do processo a controlar é possível através de diversas técnicas determinar os parâmetros do controlador, restringindo-o às especificações desejadas para regimes transitórios e estacionários do sistema.

Contudo nem sempre é possível obter o modelo matemático com facilidade, e nesse caso devem ser utilizadas técnicas experimentais de sintonia do PID. Um destes métodos mais conhecidos é o de Ziegler-Nichols, que pressupõem regras para determinar os valores de ganho K_p, T_d e T_i. Existem dois métodos designados como regras de Ziegler-Nichols. Em ambos pretende-se obter um decréscimo de amplitude de 25 % na oscilação a uma resposta em degrau [3].

No primeiro método obtém-se experimentalmente a resposta do processo a controlar a uma excitação em degrau unitário. Se não existir integrador ou pólos complexos conjugados a resposta assemelha-se a uma curva S. Esta curva pode ser caracterizada por duas constantes de tempo, o atraso L e a constante de tempo T. Estes parâmetros são então usados para a parametrização do controlador, usando as expressões da Tabela 2.

Tabela 2 Regras de sintonia Ziegler-Nichols (método 1)

Tipo de Controlador	$\boldsymbol{K_p}$	$\boldsymbol{T_i}$	$\boldsymbol{T_d}$
P	$\frac{T}{L}$	∞	0
PI	$0{,}9.\frac{T}{L}$	$\frac{L}{0{,}3}$	0
PID	$1{,}2.\frac{T}{L}$	$2L$	$0{,}5L$

No segundo método ajusta-se os parâmetros $T_i = \infty$ e $T_d = 0$. Utiliza-se somente a ação proporcional aumentando K_p até uma valor crítico (K_{cr}) para o qual o nível de saída apresenta oscilações (Figura 8). Os parâmetros são depois determinados experimentalmente utilizando os valores já conhecidos de ganho e período crítico, tal como mostra a Tabela 3.

Tabela 3 Regras de sintonia Ziegler-Nichols (método 2)

Tipo de Controlador	K_p	T_i	T_d
P	$0{,}5.K_{cr}$	∞	0
PI	$0{,}45.K_{cr}$	$\frac{1}{1{,}2}.P_{cr}$	0
PID	$0{,}6.K_{cr}$	$0{,}5.P_{cr}$	$0{,}125.P_{cr}$

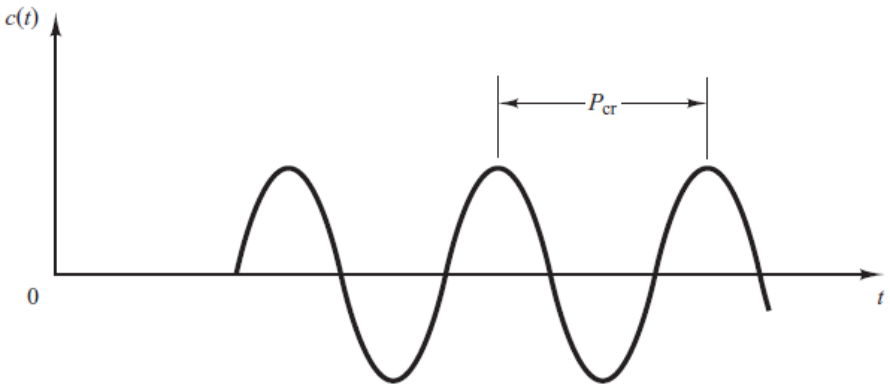

Figura 8 Oscilação do sistema para K_{cr}

Quando os controladores PID são aplicados a problemas de controlo, na maioria dos casos o resultado é satisfatório, no entanto em algumas aplicações podem ter um mau desempenho e em geral nunca apresentam um resultado ótimo. O próximo método de controlo apresentado diferencia-se pelo uso de uma técnica que suporta os modos de raciocínio aproximados em oposição aos exatos, os controladores baseados em lógica difusa.

2.4.2. CONTROLADORES DE LÓGICA DIFUSA

O termo "lógica difusa*"* foi introduzido em 1965 por Lofti A. Zadeh. O termo "lógica" refere-se ao estudo dos métodos e princípios inerentes à razão humana. A lógica clássica, usada normalmente, trabalha com preposições que são ou verdadeiras ou falsas, criando assim conjuntos e regras que levam também a resultados que podem assumir apenas dois valores. Hoje em dia, no entanto, sabe-se que muitas proposições são tanto parcialmente verdadeiras, como parcialmente falsas [14]. Para descrever estes valores parciais é necessário utilizar novas regras de maneira a que se possa estender ou generalizar a lógica clássica de dois valores. Neste contexto insere-se a lógica difusa, que tem por objetivo fornecer fundamentos para aproximações de variáveis subjetivas. Um exemplo da diferença entre estas duas lógicas pode ser observado na Figura 9. Enquanto no caso A, qualquer ponto da figura pode ser definido como amarelo ou azul (lógica clássica), no caso B isso não é possível. Devem-se descrever os pontos como tendo uma porção de amarelo e uma porção de azul (lógica difusa).

Figura 9 Lógica clássica (A) vs Lógica difusa (B) [14]

Hoje em dia, um controlo baseado neste tipo de lógica é bastante usado em aparelhos eletrónicos. Por exemplo, a empresa LG comercializa máquinas de lavar que através de um controlador difuso determinam a quantidade de água e detergente a utilizar.

Um controlador difuso é composto por 4 grandes blocos como mostra a Figura 10. O *fuzzificador*, que realiza o mapeamento entre os valores numéricos das variáveis de entrada do controlador para os graus de compatibilidade com os valores linguísticos; a máquina de inferência, onde se efetua o processamento difuso; a base de regras que consiste no conjunto de regras linguísticas; e o *defuzzificador* que transforma o conceito linguístico obtido pelo

processamento de inferência num valor numérico bem definido e que irá ser a saída efetiva do controlador difuso.

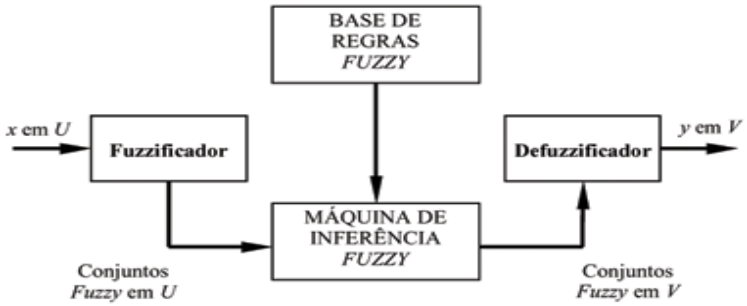

Figura 10 Diagrama de um sistema difuso[15]

A construção de um controlador difuso começa pela definição das variáveis de entrada, por exemplo a temperatura de uma sala num controlador de temperatura. De seguida é necessário limitar os valores máximos e mínimos que as variáveis podem ter, criando o denominado universo de discurso. Depois escolhe-se o número e forma das funções pertença. Não existe uma escolha ótima apenas algumas regras específicas que se podem definir.

No exemplo apresentado na Figura 11 existem quatro funções de pertença que combinam para a criação do gráfico. Cada função de pertença define um valor linguístico.

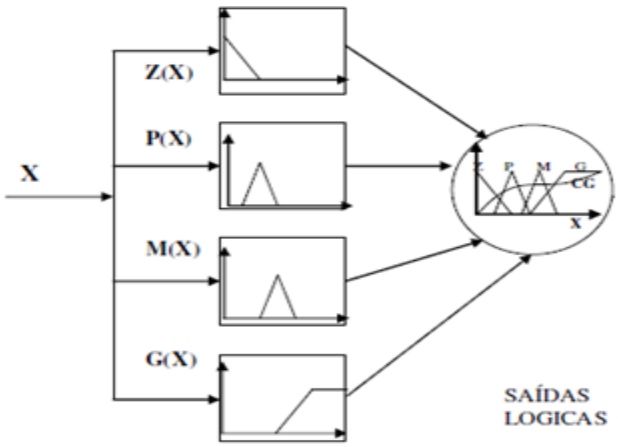

Figura 11 Funções pertença para a variável difusa X [15]

O procedimento que se realiza para as variáveis de entrada, realiza-se também para as variáveis de saída. Normalmente os valores de entrada e saída têm o mesmo número de funções pertença.

Uma etapa importante para o sucesso de um controlador difuso é a definição da base de regras. Estas regras, tipicamente da forma "*Se…Então"*, definem as relações de entrada e saída como mostra a Figura 12. Nesta figura é possível ver um sistema composto por duas variáveis de entrada, E e dE, e uma variável de saída não representada. Para cada combinação das duas variáveis de entrada, é definido um valor linguístico à variável de saída.

E \ dE	NL	NM	NS	Z	PS	PM	PL
NL	NL	NL	NL	NL	NM	NS	Z
NM	NL	NL	NM	NM	NS	Z	PS
NS	NL	NM	NS	NS	Z	PS	PM
Z	NL	NM	NS	Z	PS	PM	PL
PS	NM	NS	Z	PS	PS	PM	PL
PM	NS	Z	PS	PM	PM	PL	PL
PL	Z	PS	PM	PL	PL	PL	PL

Figura 12 Exemplo de uma base de regras de um sistema difuso [15]

A saída do controlador requer um sinal não-difuso, devendo existir uma interpretação entre o conjunto difuso e o universo de saída determinado. Este processo chama-se *defuzzificação*, e pode ser feito através de diferentes métodos, os mais conhecidos são a Média dos Máximos e Centro de Gravidade. No primeiro a saída é obtida calculando-se a média entre os dois elementos extremos no universo que correspondem aos maiores valores da função de pertença. Com o método Centro de Gravidade a saída é o valor do universo que divide a área sobre a curva da função pertença em duas partes iguais.

O comportamento e desempenho de controladores difusos depende de vários aspetos relacionados com a sua estrutura e implementação tais como: número de conjuntos difusos pertencentes a cada variável; correta escolha da forma das funções pertença; base de regras e escolha do processo adequado de *defuzzificação*.

3. REVISÃO BIBLIOGRÁFICA

O efeito visual atrativo de um sistema de levitação magnética faz com que estes sistemas sejam um tema bastante comum quando se pretende estudar os mais diversos tipos de controlo. Não é por isso de admirar a existência de um grande número de trabalhos nesta área. Neste capítulo serão apresentadas as referências aos métodos de controlo que foram deparados aquando da pesquisa para a realização deste trabalho, bem como vantagens e desvantagens de cada método e principais resultados.

3.1. LÓGICA DIFUSA

A lógica difusa é a lógica que suporta os modos de raciocínio aproximados em oposição ao exatos. Modelação e controlo de sistemas de lógica difusa utiliza técnicas para o tratamento de informações qualitativas de forma rigorosa [6]. Este tipo de lógica admite valores intermédios entre 0 (falso) e 1 (verdadeiro), assim sendo um valor difuso é um valor qualquer no intervalo entre 0 e 1.

Um exemplo de uma aplicação deste controlo em levitação magnética foi realizado por Kashif Ishaque , et al. no trabalho “Modeling and Control of Magnetic Levitation System via Fuzzy Logic Controller”. Neste controlador difuso foram definidas duas variáveis de

entrada: o erro entre a posição da esfera e a posição desejada e a variação desse erro. Como variável de saída tem uma tensão que pode variar entre os valores [-2;2].

Em termos de processo de *fuzzificação* o autor definiu sete funções de pertença todas triangulares. Como se pode ver na Figura 13, estas funções são todas iguais com exceção das funções de extremos NL e PL, que são trapezóides.

A base de regras para a saída obedece a uma simetria típica nestes casos, onde por exemplo um erro PL e uma variação de erro NL corresponde a uma saída Z.

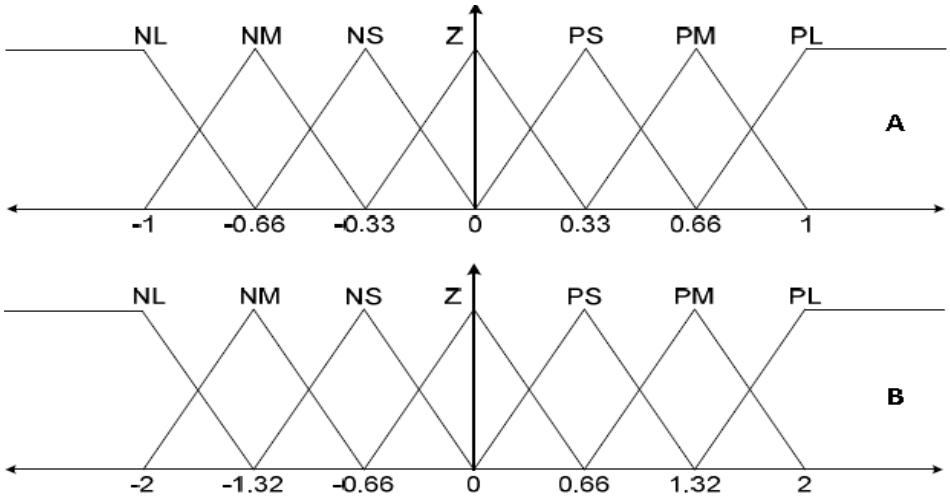

Figura 13 Funções de pertença das variáveis de entrada (A) e saída (B) [6]

Como resultados obtidos neste trabalho, o controlador difuso foi comparado a um controlador PID, e os resultados apresentados na Figura 14.

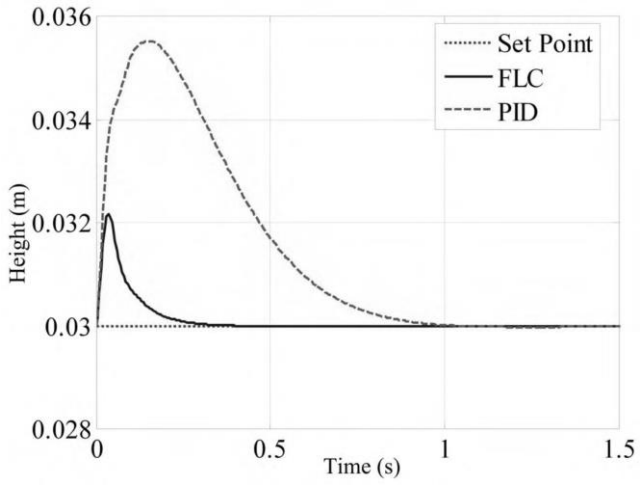

Figura 14 Resposta de ambos os controladores ao valor de referência [6]

O controlador difuso tem um *overshoot* de 6 % comparado com 18 % do controlador PID. Assim como o tempo de subida é de 0,3 segundos comparado com o resultado de 1 segundo do PID.

Neste trabalho são comparados os controladores difuso e PID. No entanto não é referido como foi criado o algoritmo PID, e se era possível sintonizar melhor para obter resultados mais próximos do controlador difuso. Também não é feita referência sobre o modo de deteção da posição do objeto a levitar, ou que circuito de potência é utilizado.

Num outro trabalho publicado em 2014 por Arjan C. Unni e A. Junghare denominado "Fuzzy Logic Controller and LQR for Magnetic Levitation System", são desenvolvidos para um sistema de levitação magnética dois controladores, um baseado em lógica difusa e um controlador linear quadrático.

O método de implementação do controlador difuso é algo semelhante ao descrito no trabalho anterior, são usadas as mesmas variáveis de entrada "erro" e "variação de erro" mas ao contrário do trabalho anterior estas variáveis não apresentam simetria entre as funções pertença, e são apenas utilizados cinco valores linguísticos para cada variável, podendo as variáveis de entrada e saída ter valores entre -100 e 100 (Figura 15).

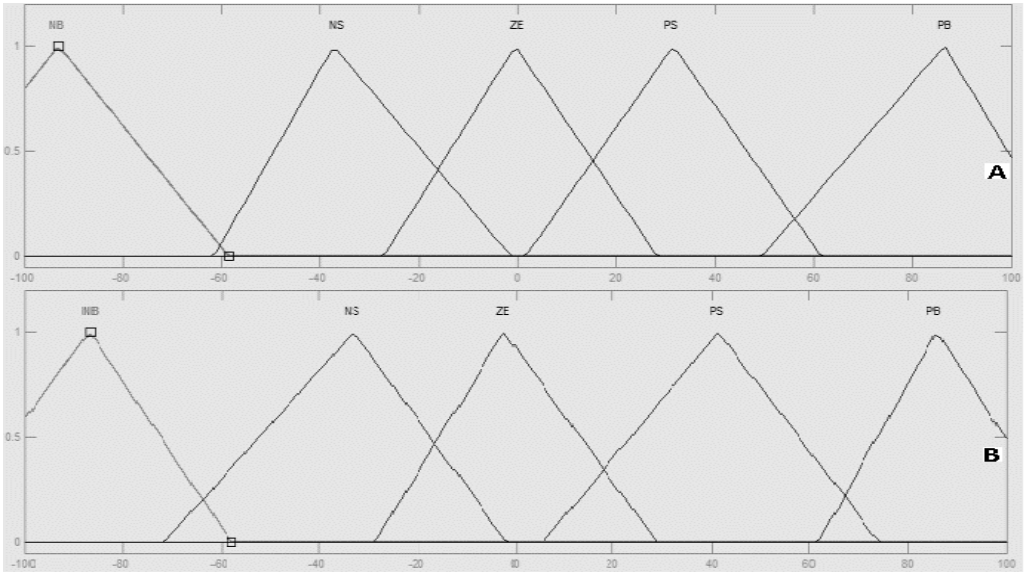

Figura 15 Funções de pertença das variáveis de entrada (A) e saída (B) [7]

Este trabalho não faz referência de como é tratada a variável de erro, ou seja, de como é obtido a diferença entre a posição da esfera e o valor de referência, nem que condicionamento é feito para estar entre os valores de -100 e 100. O mesmo se passa com a variável de saída,

não se sabendo como é feito o acionamento do eletroíman e como a variável de saída faz a interação com esse acionamento.

Como conclusão o trabalho refere que ambos os controladores foram implementados com sucesso, com o controlador difuso a ter um *overshoot* menor mas um tempo de estabelecimento maior. É também referido que a implementação do controlador difuso foi mais rápida, pois não foi necessário modelar o sistema e foi facilmente sintonizado pois lida com variáveis linguísticas. Na Tabela 4 podem ser consultados os resultados obtidos neste trabalho.

Tabela 4 Comparação dos resultados dos controladores [7]

Controlador	Tempo de subida (s)	*Overshoot* (%)	Tempo de estabelecimento (s)
LQR	0,116	3,1	0,116
Difuso	0,116	0,005	1,16

3.2. ALGORITMO PID

O controlo por algoritmo PID é um método de controlo em malha fechada largamente usado na indústria. Este controlador recebe como entrada a diferença entre um valor de saída do processo e um valor de referência. Existem vários trabalhos desenvolvidos no controlo de um sistema de levitação por controlo PID, são apresentados de seguida alguns deles.

Um exemplo de controlo de um sistema de levitação magnética foi proposto por Hypiusová no trabalho "PID Controller design for magnetic levitation model" [16]. A implementação do modelo consiste na levitação de uma esfera por um eletroíman. A posição da esfera é fornecida por um sensor linear indutivo ligado a um conversor A/D. O eletroíman é alimentado por um amplificador de potência ligado a um conversor D/A.

O diagrama de blocos do controlo desejado é apresentado na Figura 16, onde $G_R(s)$ é o controlador PID, $G_P(s)$ a função de transferência do modelo, w, e, u e y são a referência, o erro, a variável de saída do PID e a posição da esfera, respetivamente.

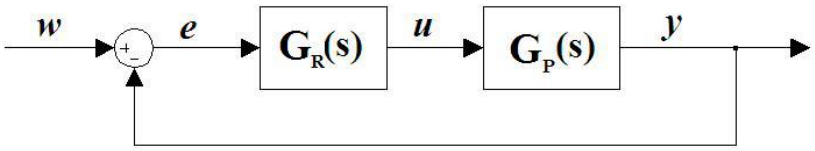

Figura 16 Diagrama de blocos do sistema [16]

Com estes elementos o autor pretende controlar o sistema usando um PID sintonizando-o através do método Neimark D-Partition, alcançando não só a estabilidade mas também a margem de fase desejada.

Este método, apesar de pouco utilizado, apresenta resultados satisfatórios quando se pretende impor requisitos de margem de fase ao sistema. O autor começa por encontrar a função de transferência em malha fechada, fazendo a divisão em parte real e parte imaginária:

$$RE: k = -\frac{A(j\omega)}{B(j\omega)} \qquad IM: -\frac{k_i}{\omega} j + k_d . j . \omega = -\frac{A(\omega)}{B(\omega)} \tag{9}$$

Sendo no domínio das frequências, os blocos do diagrama G_P e G_R representados por:

$$G_P(j\omega) = \frac{B(j\omega)}{A(j\omega)} \qquad G_R(j\omega) = k + \frac{k_i}{j\omega} + k_d . j . \omega . \tag{10}$$

Variando ω para o intervalo de $\omega \in (0, \infty)$ na parte real foi calculado o vetor de números complexos dependentes da frequência, que cria a curva-D no plano complexo para o parâmetro k, sendo o mesmo feito para a parte imaginária.

Após este procedimento a criação do controlador PID consiste em dois passos: primeiro criar um controlador PD, e depois um controlador PI é aplicado em conjunto com o primeiro. Desta forma o controlador PD é usado para estabilizar e o controlador PI assegura a margem de fase desejada, e elimina o erro em regime permanente.

Neste trabalho foi definido como margem de fase desejada de 65^0 e o controlador final obtido foi:

$$G_R(s) = 4{,}476 + \frac{7{,}513}{s} + 0{,}0222s \tag{11}$$

Nem sempre são necessários microcontroladores ou placas de aquisição de dados para implementar um controlador para sistemas de levitação magnética. De seguida serão

apresentados dois trabalhos onde são implementados controladores PID totalmente analógicos para o controlo de sistemas de levitação magnética.

No primeiro destes trabalhos [17] foi criado um sistema de levitação magnética de uma esfera através de um eletroíman. Como sensor de posição são usados um par de fototransmissores óticos e 2 *leds* de infravermelhos, colocados na zona de equilíbrio.

Uma particularidade deste trabalho é a decomposição da corrente em corrente de *off-set* e corrente dinâmica. Sendo a corrente de *off-set* a corrente que o eletroíman necessita para manter o objeto na posição de equilíbrio, e a corrente dinâmica a variação da corrente para compensar deslocações da esfera.

Os parâmetros do controlador foram encontrados utilizando a ferramenta "*rltool"* do programa MATLAB, tendo a função de transferência:

$$T(s) = 50(1 + 0{,}02s) \tag{12}$$

Como é possível ver o controlador é do tipo PD, por isso é de esperar um erro em regime permanente. No trabalho não é feita qualquer menção a este erro. Quanto aos parâmetros encontrados não é apresentada justificação para estes valores, apenas que este controlador estabiliza o sistema, parecendo no entanto que os parâmetros foram escolhidos para uma mais fácil implementação do controlador analógico.

Este controlador é implementado usando um AMPOP LF365, no circuito elétrico apresentado na Figura 17.

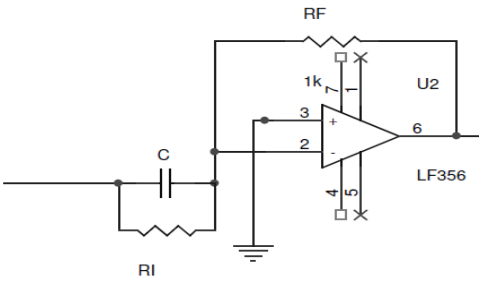

Figura 17 Circuito elétrico de um controlador PD

Sendo os parâmetros P e D dados pelas expressões:

$$P = -\frac{R_F}{R_1}, \qquad D = -R_F.C \tag{13}$$

Este trabalho conclui que é possível levitar uma esfera por um controlador PD analógico, assim como foi satisfatória a linearização em torno do ponto de equilíbrio para este sistema não linear.

O segundo exemplo de um controlo analisado de um sistema Maglev distingue-se dos restantes em termos construtivos pelo uso de um concatenador de linhas de campo magnético (Figura 18), que faz aumentar a força magnética a que está sujeita a esfera.

Figura 18 Estrutura do concatenador [18]

A estrutura do concatenador é formada por um esqueleto de ferro, que possibilita a concatenação das linhas de campo magnético [18].

Em termos de controlo não é implementado um controlador com base no algoritmo PID, mas em controlador em avanço de fase. Os parâmetros deste controlador foram escolhidos conforme a análise de estabilidade do sistema utilizando o método do lugar geométrico de raízes. Na Figura 19 são apresentadas as raízes do sistema $G(s)$ sem controlador (A) e com controlador (B).

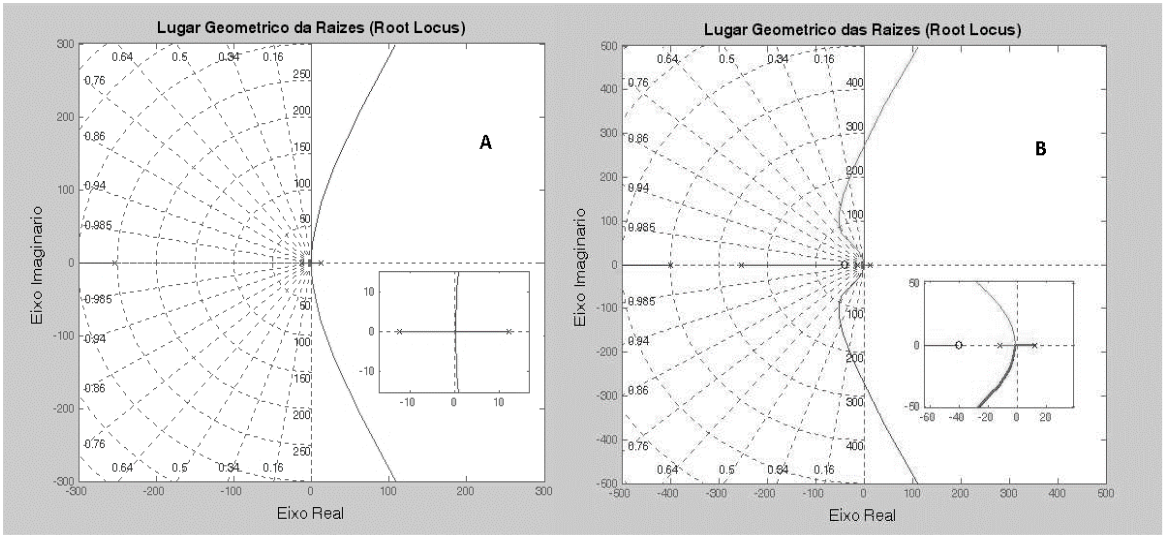

Figura 19 LGR sem controlador (A) e com controlador (B) [18]

Tal como o trabalho apresentado anteriormente os valores do controlador foram escolhidos tendo em conta os valores dos componentes disponíveis. Como resultados obtidos o sistema tem um tempo de estabelecimento de 0,1 s e um *overshoot* de 46%. Como conclusão é referido que o sistema pode operar a uma distância de 18 mm, também graças às características do concatenador. É ainda referido que toda a operação de estabilização do sistema depende do compromisso entre os valores de condensadores e resistências e o controlador teórico ideal. Este ajuste foi feito recorrendo a ferramentas do MATLAB.

3.3. OUTROS MÉTODOS DE MODELAÇÃO E CONTROLO

Muitos outros métodos de controlo foram desenvolvidos em sistemas de levitação magnética. De seguida serão apresentados trabalhos onde são aplicados diferentes métodos de controlo e modelação.

O primeiro trabalho apresenta uma modelação por gráficos de Bond [19]. Gráficos de Bond é um modo de representação da dinâmica de sistemas, tal como o diagrama de blocos. O modelo físico do sistema é idêntico aos dos restantes trabalhos, apenas diferindo na forma como é obtida a função de transferência.

A justificação apresentada pelo autor para o uso de gráficos de Bond na modelação do sistema é a possibilidade de nos gráficos de Bond, em cada ligação transferir a informação de duas variáveis, ao contrário do que acontece nos diagramas de blocos. Esta modelação pode ser vista na Figura 20.

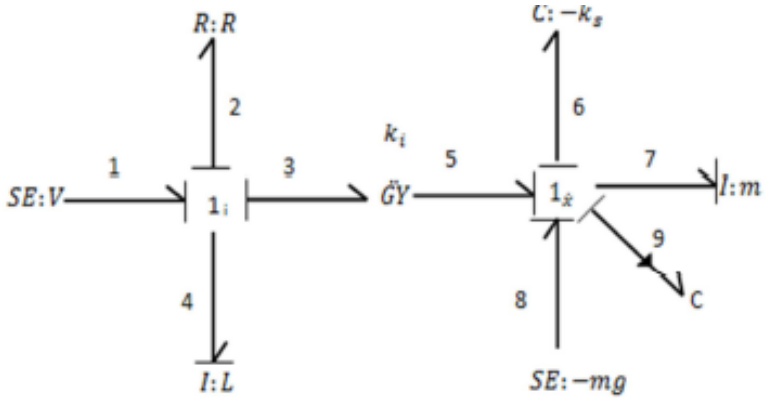

Figura 20 Modelo em gráfico de Bond de um sistema de levitação magnética [19]

O controlo deste trabalho é feito com o algoritmo PID, similar aos de trabalhos descritos anteriormente, por isso não será analisado. Como conclusão é dito que a esfera estabiliza na posição desejada. Quanto aos gráficos de Bond na modelação, é mais trabalhoso que o diagrama de blocos, mas adiciona mais informação e flexibilidade à modelação.

Até agora os sistemas de levitação magnética analisados foram tratados como sistemas SISO. No entanto em sistemas modernos complexos podem existir múltiplas entradas e saídas (MIMO), o que aumenta a dificuldade da análise. Ao tratar com sistema MIMO, através de espaço de estados, o enfoque passa para as variáveis de estado do sistema. Um exemplo do uso de espaço de estados no controlo de levitação magnética é o trabalho "Modelagem e simulação de um sistema de levitação magnética através de programação em linguagem gráfica" [20]. Como variáveis de estado foram escolhidas a posição, velocidade da esfera, e corrente consumida.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} d \\ \dot{d} \\ i \end{bmatrix}$$

Como variáveis de entrada e saída foram selecionadas a tensão aplicada e posição da esfera, respetivamente. A representação no espaço de estados é descrita da seguinte forma:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}u$$

$$y = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}u$$

A principal vantagem da modelação no espaço de estados é que com a realimentação de todas as variáveis de estado é possível reposicionar os polos do sistema em qualquer região

do plano complexo com a utilização dos ganhos adequados. Após a escolha da localização dos polos, foi utilizada a fórmula de Ackerman para a obtenção dos ganhos.

Este trabalho apenas ficou pela simulação do sistema em LABVIEW, não tendo sido implementado num sistema real. No entanto fica a referência de um outro modo de modelação do sistema, e como se determina um controlador para estes casos.

3.4. *TOOLBOX* DE IMPLEMENTAÇÃO DE UM SISTEMA DIFUSO

Em muitos dos trabalhos apresentados são usadas ferramentas que permitem implementar algoritmos de controlo de um modo fácil e intuitivo para o utilizador, possibilitando a abstração de aspetos não relevantes para o projeto. Estas ferramentas denominadas de *toolbox* estão normalmente associadas a outros *softwares* de controlo, como por exemplo a *fuzzy toolbox* do MATLAB, ou o PID VI do LABVIEW.

Relativamente à lógica difusa o uso de uma *toolbox* deste tipo é fundamental para a implementação rápida e eficaz de um controlador. A *toolbox* eFLL (*Embedded Fuzzy Logic Library*) é uma boa escolha quando se pretende utilizar lógica difusa em sistemas onde são usados microcontroladores [32]. Esta *toolbox* foi desenvolvida pelo *Robotic Research Group* da Universidade Estadual de Piauí no Brasil. Escrita totalmente em C/C++, utiliza o *standard* da biblioteca *stdlib.h*, e pode por isso ser utilizada em todas as aplicações que tem por base este tipo linguagem. Em seguida serão apresentadas as funções e objetos que compõem esta *toolbox*, bem como um exemplo de algumas partes da programação de um controlador bastante simples, com uma variável de entrada (distância) e uma variável de saída (velocidade).

O objeto **Fuzzy** é definido no início do código do controlador, e engloba todo o sistema difuso. Através dele é possível manipular os conjuntos, definir as entradas e saídas do sistema, e criar as regras linguísticas. No exemplo de código em baixo é definido um objeto deste tipo.

```
Fuzzy* fuzzy = new Fuzzy();
```

O objeto **FuzzyInput** define as variáveis difusas de entrada, agrupando todos os conjuntos difusos pertencentes ao mesmo domínio. Este objeto recebe como parâmetro o número da

variável que se pretende criar. No exemplo apresentado é definida a variável de entrada difusa *distance*.

```
FuzzyInput* distance = new FuzzyInput(1);
```

O objeto **FuzzyOutput** é semelhante ao anterior, mas usado para as variáveis de saída.

```
FuzzyOutput* velocity = new FuzzyOutput(1);
```

O objeto **FuzzySet** é um dos principais elementos desta *toolbox*, pois é através dele que é possível modelar cada conjunto difuso do sistema que se está a desenvolver. São suportadas funções pertença do tipo triangular, trapezóidal e *singleton*, que são definidos através dos parâmetros de entrada deste objeto.

```
FuzzySet* average = new FuzzySet(10, 20, 30, 40);
```

No exemplo de código em cima é criado um objeto com o nome *average*, correspondente a um valor linguístico de uma variável difusa. Os parâmetros que este objeto recebe vão definir a localização e a forma que a função de pertença vai ter. O exemplo de código mostrado produz a função de pertença trapezoidal apresentada na Figura 21.

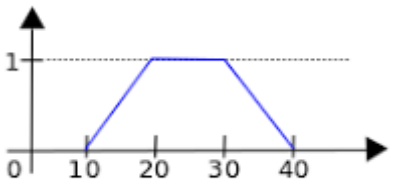

Figura 21 Função de pertença da variável *avarege*

Caso se pretenda uma função triangular os valores do meio a passar para o objeto **FuzzySet** devem ser os mesmos, por exemplo, (10,20,20,30) produziria uma função de pertença triangular com o vértice superior em 20.

O objeto **FuzzyRule** é usado para montar uma base de regras para o sistema. Associados à criação de regras estão ainda os objetos **FuzzyRuleAntecedent** responsável por montar a expressão condicionada da antecedente, ou seja, a parte "se*"* da regra, e o objeto **FuzzyRuleConsequence,** responsável pela expressão consequente. Para tornar mais claro

este processo o seguinte exemplo de código define a regra: "se a distância é pequena a velocidade deve ser lenta".

```
//FuzzyRule "SE distancia = pequena ENTAO velocidade = lenta"
FuzzyRuleAntecedent* ifDistanceSmall = new FuzzyRuleAntecedent();
ifDistanceSmall->joinSingle(small);
FuzzyRuleConsequent* thenVelocitySlow = new FuzzyRuleConsequent();
thenVelocitySlow->addOutput(slow);
FuzzyRule* fuzzyRule01 = new FuzzyRule(1, ifDistanceSmall,
thenVelocitySlow);
```

Pode-se ver a criação de um objeto do tipo **FuzzyRuleAntecedent** denominado "ifDistanceSmall". De seguida o método **joinSingle** é usado para associar ao antecedente da regra um valor linguístico criado anteriormente, neste caso o "*small*" associado a uma distância pequena. Após isto é utilizado o objeto **FuzzyRuleConsequent** *"thenVelocitySlow*" para a segunda parte da regra. Para associar este objeto a um valor linguístico da variável de saída é utilizado o método **addOutput**. O último passo para a definição de uma regra é utilizar o objeto **FuzzyRule**, que recebe simplesmente como parâmetros o número da regra, o antecedente e o consequente.

Estão apresentados todos os objetos desta biblioteca, nos passos seguintes são utilizados somente métodos aplicados diretamente ao objeto **Fuzzy**.

O primeiro método é o **setInput**, utilizado para passar os valores de entrada numéricos para o sistema. Como se pode ver no exemplo de código em baixo este método recebe dois parâmetros: o primeiro indica o número da variável difusa de entrada, o segundo o valor que essa variável assume.

```
fuzzy->setInput(1, dist);
```

O método **fuzzify** é utilizado para iniciar o processo de fuzzificação das variáveis e o mecanismo de inferência.

```
fuzzy->fuzzify();
```

Por fim o método **defuzzify** utiliza-se para finalizar o processo e obter um valor numérico para uma determinada saída difusa. No seguinte exemplo o valor de saída do controlador para a variável de saída difusa 1, é guardado na variável *output* do tipo *float*.

```
float output = fuzzy->defuzzify(1);
```

De referir que ao contrário dos objetos desta biblioteca que são apenas definidos uma vez, e geralmente no início do programa, os últimos três métodos apresentados são utilizados ciclicamente, com um período definido por quem projeta o sistema.

Quanto às limitações desta *toolbox*, estas são relativas às ausências de escolhas de alguns parâmetros do controlador difuso. Assim o mecanismo de inferência é sempre do tipo *Mamdani*, não é possível implementar um método de implicação, e o único método de defuzzificação disponível é o método do centro de gravidade. Quanto à quantidade de variáveis de entrada e saída, e número de regras, estas estão apenas limitadas à capacidade de processamento e armazenamento do microcontrolador.

4. CONSTRUÇÃO DO SISTEMA

O sistema de levitação magnética tem como objetivo fazer levitar um pequeno objeto metálico numa posição estável definida pelo utilizador. Este objetivo é conseguido usando um eletroíman que produz uma força magnética quando percorrido por uma corrente, cujo valor é dimensionado por um controlador. Neste projeto, para esse efeito é utilizado o microcontrolador Atmega 2560, embutido na plataforma de prototipagem Arduíno Mega. Este controlo é efetuado tendo como base a localização do objeto fornecida por um sensor de posição.

O sistema é essencialmente uma plataforma de ensaio, constituída por um circuito de potência e a placa de prototipagem. É ainda adicionado um PC com um interface gráfico para permitir ao utilizador escolher qual o controlador a usar e qual a posição do íman. Por fim o último bloco do sistema é constituído pelo conjunto eletroíman e sensor de posição. A Figura 22 mostra as ligações básicas entre os blocos: o sensor fornece um valor de tensão ao Arduíno consoante a posição do íman; o Arduíno é responsável por definir o valor de PWM, e enviar a posição do íman para o programa no PC associado; no programa do PC deve-se

escolher qual o algoritmo de controlo a utilizar; e por fim o circuito de potência converte o PWM que recebe em corrente para o eletroíman.

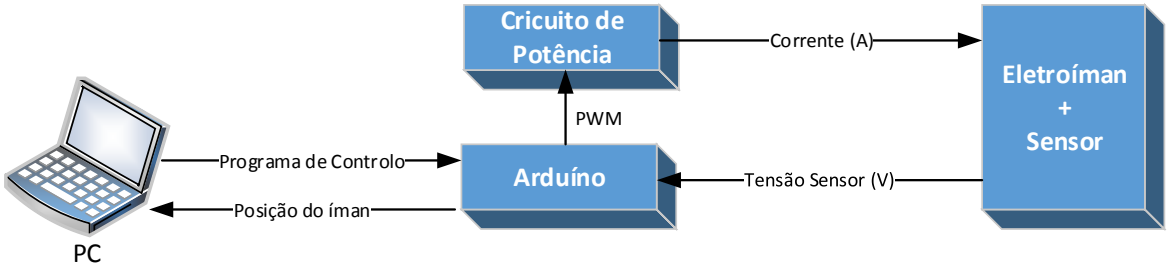

Figura 22 Diagrama do sistema de levitação magnética

4.1. ELETROÍMAN E SENSOR DE POSIÇÃO

O eletroíman e sensor de posição utilizado no projeto foram adquiridos na empresa Zeltom [30], que comercializa produtos educativos para o teste de sistemas de controlo a baixo preço. Optou-se por comprar um eletroíman, pois conceber um de raiz seria mais caro, e com o inconveniente de no fim não ser corretamente dimensionado.

As dimensões do eletroíman são apresentadas na Figura 23. Este tem uma resistência de 2,4 Ω, e uma indutância de 15 mH. A corrente máxima é de 1,5 A, uma corrente superior pode danificar o eletroíman. Este aspeto limitador influencia a escolha da posição de levitação e peso do objeto. Uma posição demasiado baixa, ou um objeto demasiado pesado podem requerer uma corrente superior a este limite.

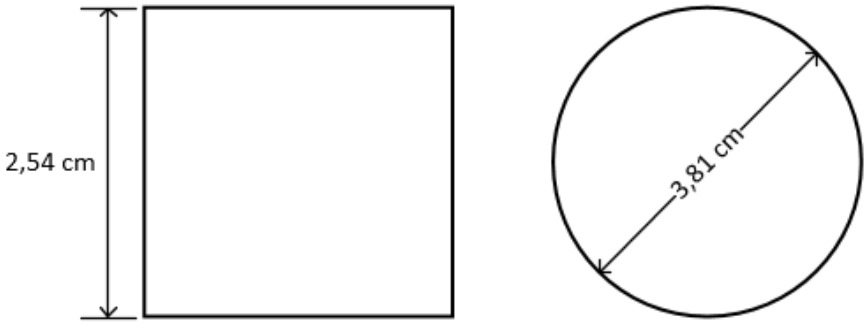

Figura 23 Dimensões eletroíman

Para determinar a posição do objeto utiliza-se o sensor de efeito de Hall A1324 do fabricante Allegro [31]. O sensor é colocado na base do eletroíman como mostra a Figura 24. Neste tipo de sensor o valor da tensão de saída varia conforme a intensidade do campo magnética detetada. Uma análise mais aprofundada deste sensor será efetuada no capítulo 5. Esta escolha para determinar a posição influenciou o tipo de objeto a levitar. Este necessita de ser um íman permanente, que altere o campo magnético na proximidade do sensor.

O íman permanente usado é um íman de neodímio, em forma de disco, com 1 cm de diâmetro, 3 milímetros de altura e peso de 2 gramas.

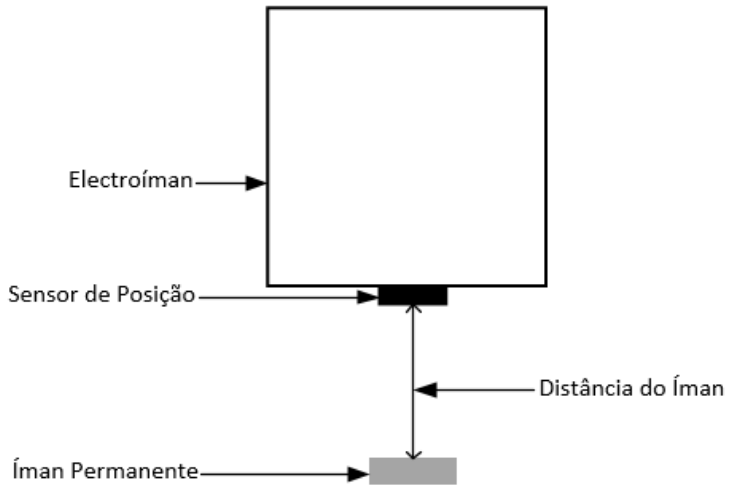

Figura 24 Esquema de deteção do íman

Seria possível utilizar outros tipos de sensor, como sensores indutivos, sensores infravermelho, ou LDR, como realizado em muitos dos trabalhos já referenciados. No entanto o sensor de efeito de Hall apresenta várias vantagens em relação a estas opções: tem um preço mais baixo, não necessita de *hardware* adicional, e elevada fiabilidade. O principal inconveniente da utilização deste sensor é o facto de detetar não só o campo magnético criado pelo íman enquanto levita, mas também detetar o campo magnético gerado pelo eletroíman, podendo com isto ocorrer leituras erradas por parte do sensor. No capítulo 5 este aspecto será abordado em mais detalhe, bem como possíveis formas de o corrigir.

4.2. MICROCONTROLADOR

Como já referido para a aplicação do algoritmo de controlo é usada a plataforma de prototipagem Arduíno Mega 2560, onde é possível utilizar uma linguagem de programação baseada em C/C++. A opção por esta placa deve-se ao facto do seu baixo custo, flexibilidade e existência de um grande número de bibliotecas de funções que facilitam o desenvolvimento de programas. Este Arduíno utiliza o Atmega 2560, um microcontrolador de 8 bits de arquitetura RISC avançada, com 256 KB de memória Flash, 8 KB de RAM e 4 KB de EEPROM [26].

Referir ainda a existência de 54 pinos I/O, dos quais 15 podem ser configurados como saídas em PWM, 16 entradas analógicas de 10 bits de resolução e 4 canais de comunicação série.

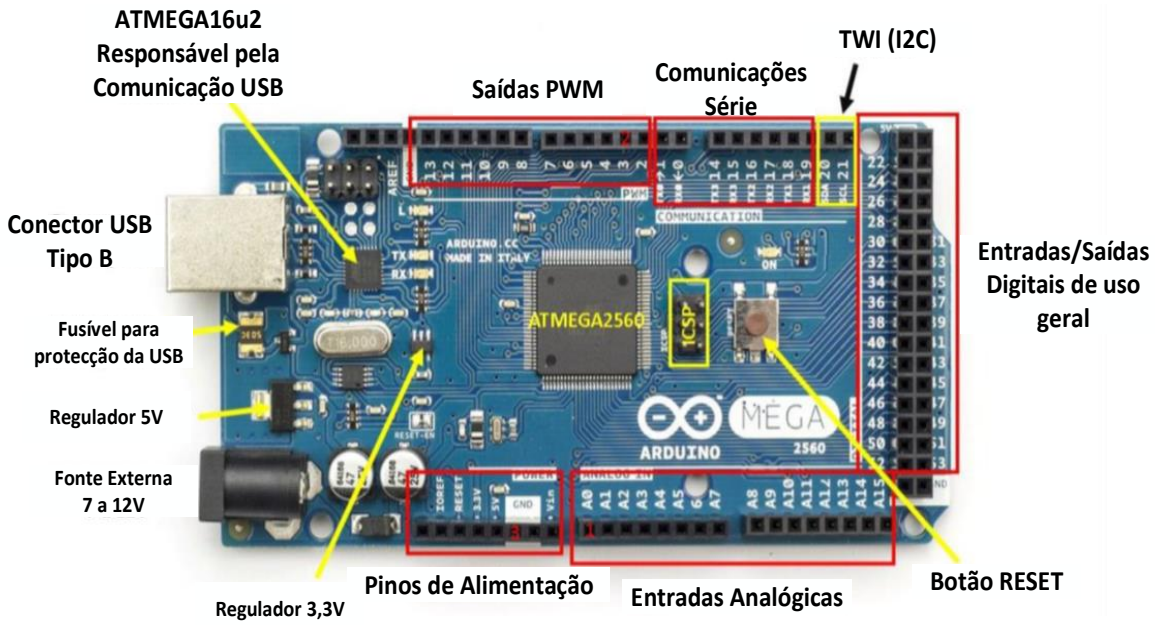

Figura 25 Resumo da placa Arduíno Mega 2560

Um resumo dos componentes do Arduíno Mega 2560 pode ser visto na Figura 25, onde estão ainda assinalados os três pinos mais importantes na utilização do projeto. A entrada analógica (1) que irá ler o valor de tensão de saída do sensor, a saída PWM (2) para o circuito de potência e pino GND (3).

4.3. CIRCUITO DE POTÊNCIA

A elevada corrente requerida pelo eletroíman leva a que seja necessário implementar um circuito de potência. A Figura 26 mostra a configuração deste circuito, que tem como entrada um sinal PWM, fornecido pelo microcontrolador, e como saída um valor de corrente, que alimentará o eletroíman. A relação entre a saída e a entrada é analisada no capítulo 5 onde se realiza a modelação deste circuito de potência.

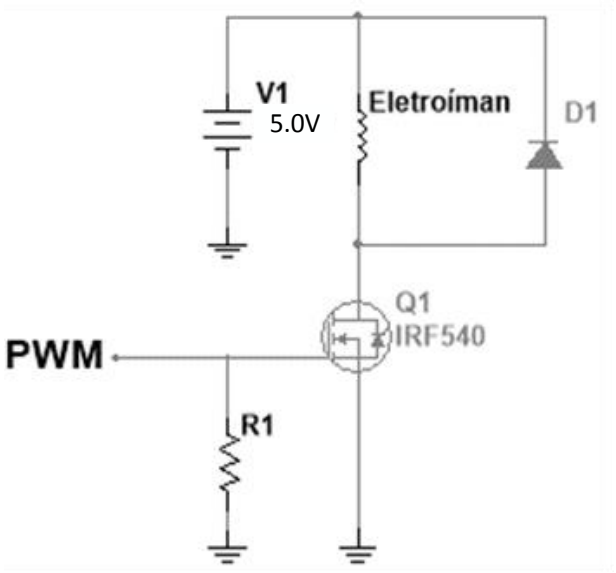

Figura 26 Circuito de potência

O funcionamento do circuito é bastante simples, e utilizado quando se necessita fornecer uma corrente elevada a uma carga. É usado um MOSFET controlado por um sinal em PWM ligado à *gate* que determina o estado deste. Uma resistência de *pull-down* (R1) é utilizada para manter a *gate* com o valor de 0 volts caso o pino de saída do PWM esteja mal configurado ou quando estiver a ocorrer o processo de *booting* e ainda não existir a definição dos pinos.

O díodo D1 é utilizado como díodo *bypass* para prevenir a força contra eletromotriz, quando o MOSFET não estiver a conduzir. Este componente é especialmente importante por se tratar de uma carga altamente indutiva. A Tabela 5 mostra todos componentes usados para o circuito de potência e a sua descrição.

Tabela 5 Componentes do circuito de potência

Componente	Descrição	Especificações
R1	Resistência	10 kΩ Precisão-1% 250 mW
Q1	MOSFET de Potência	100 V 17 A
V1	Fonte de Tensão	5 V
D1	Díodo	500 mW 12 V

4.4. INTERFACE COM PC

Para o utilizador ter a possibilidade de configurar o controlo e sinais de referência do sistema foi desenvolvida uma interface gráfica em ambiente de programação *Visual Basic*, através do IDE *Visual Studio* 2015. Esta aplicação permite ao utilizador visualizar dois gráficos com a informação da posição do íman e do erro deste em relação à referência definida. Existe ainda uma imagem real do sistema, fornecida por uma Webcam. A interface gráfica está ilustrada na Figura 27. Uma descrição completa do modo de funcionamento desta aplicação é apresentada no anexo B.

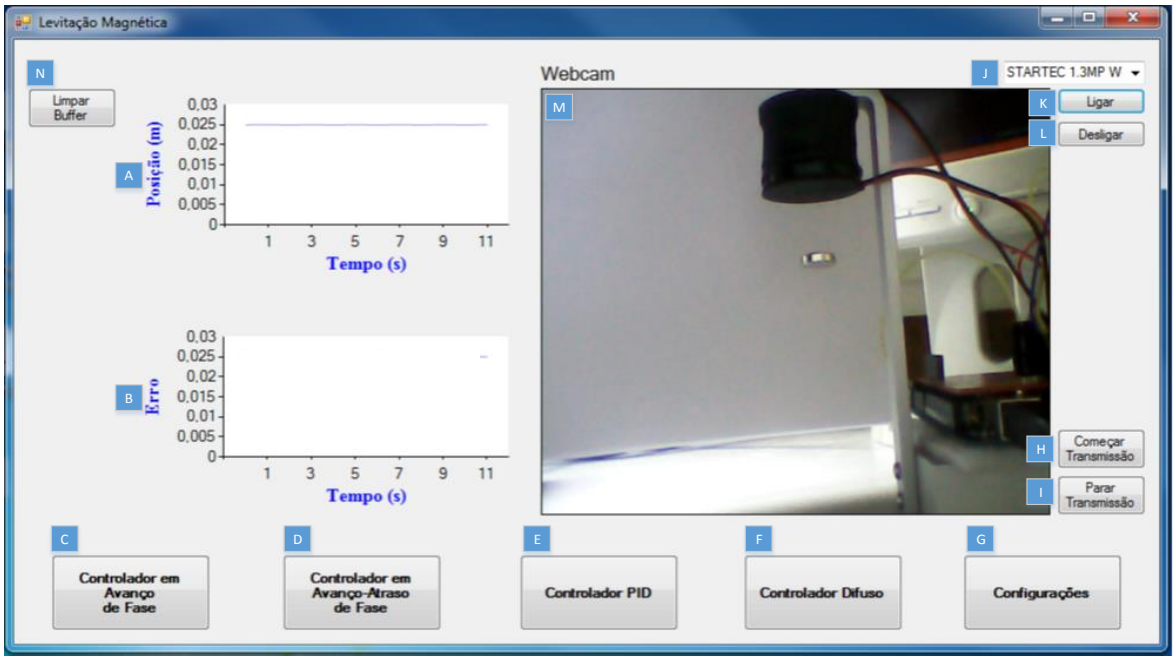

Figura 27 Interface gráfica para sistema de levitação magnética

Onde:

A) Gráfico de posição do íman
B) Gráfico do erro da posição do íman em relação à referência
C) Botão de seleção do controlador em avanço de fase
D) Botão de seleção do controlador em avanço-atraso de fase
E) Botão de seleção do controlador PID
F) Botão de seleção do controlador Difuso
G) Botão para abrir a janela de configurações
H) Botão para começar a transmissão série entre PC e microcontrolador
I) Botão para terminar a transmissão série entre PC e microcontrolador
J) *Combobox* para seleção da fonte da Webcam
K) Ligar Webcam
L) Desligar Webcam
M) Imagem da Webcam
N) Botão para limpar o *Buffer* de receção de dados

A comunicação entre o programa e o microcontrolador é de uma comunicação série, bidirecional feita através do protocolo RS232 a uma taxa de 115200 bps. Este tipo de comunicação é utilizado neste projeto devido à facilidade de implementação, quer da parte do programa do PC, quer da parte do Arduíno.

O envio de dados do Arduíno para o PC é feito de forma contínua, a uma taxa definida através de uma interrupção interna do microcontrolador. É enviado o valor de cada leitura efetuado pelo A/D, devendo o programa do PC converter esse valor para posição, calcular o erro, e criar os gráficos da Figura 27. Para evitar bloqueios por parte do programa do PC devido a tempo de espera de receção de dados, definiu-se um intervalo de transmissão de dados no Arduíno, mais pequeno que no programa de PC. Assim existem sempre valores no *buffer* de transmissão para o programa. O inconveniente deste método é o atraso que este desfasamento origina nos gráficos, que se vai tornando mais notório com o passar do tempo.

Para evitar este inconveniente criou-se um botão que elimina os valores guardados em *buffer*, permitindo assim a receção de valores mais atualizados.

O envio de dados do PC para o Arduíno é feito apenas quando o utilizador do programa altera o tipo de controlador ou o sinal de referência, através da janela de configuração representado na Figura 28. Da parte do programa do Arduíno é verificado a cada ciclo se existe algum valor no *buffer* de receção.

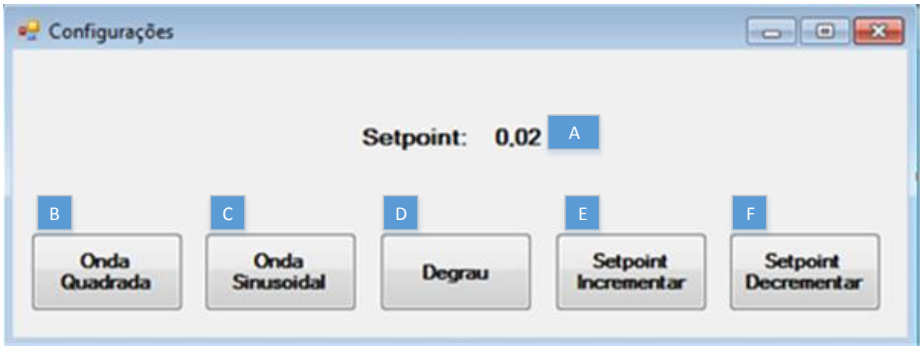

Figura 28 Janela de configurações do interface gráfico

Onde:

A) Indicação do *Setpoint* (Referência)

B) Botão para escolha de onda quadrada como referência

C) Botão para escolha de onda sinusoidal como referência

D) Botão para escolha de sinal constante como referência

E) Botão para incrementar o valor de *Setpoint*

F) Botão para decrementar o valor de *Setpoint*

O código referente a esta aplicação encontra-se disponibilizado no Anexo C.

4.5. CONSTRUÇÃO DE *SHIELD* DE LEVITAÇÃO MAGNÉTICA

A última parte da construção do sistema de levitação magnética é o projeto de uma *shield* a colocar no topo do Arduíno. Esta *shield* recebe os componentes do circuito de potência, fornece ligações para o sensor e eletroíman, e apresenta ainda 4 botões a ser ligados a 4 entradas digitais do Arduíno. Esta placa foi desenvolvida no *software* de CAD Eagle. O circuito e o *layout* final estão representados na Figura 29 e Figura 30, respetivamente.

A placa foi projetada para ter pistas nas duas faces, estando as vias da face de cima representadas a vermelho, e as vias da face de baixo a azul.

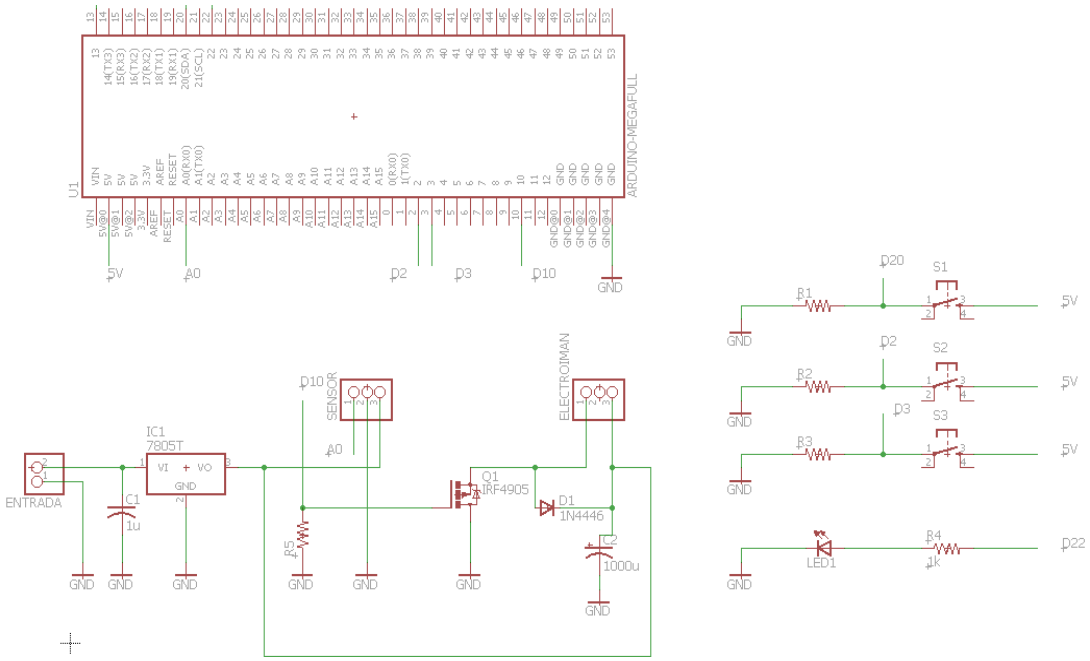

Figura 29 Circuito da *Shield* para Arduíno

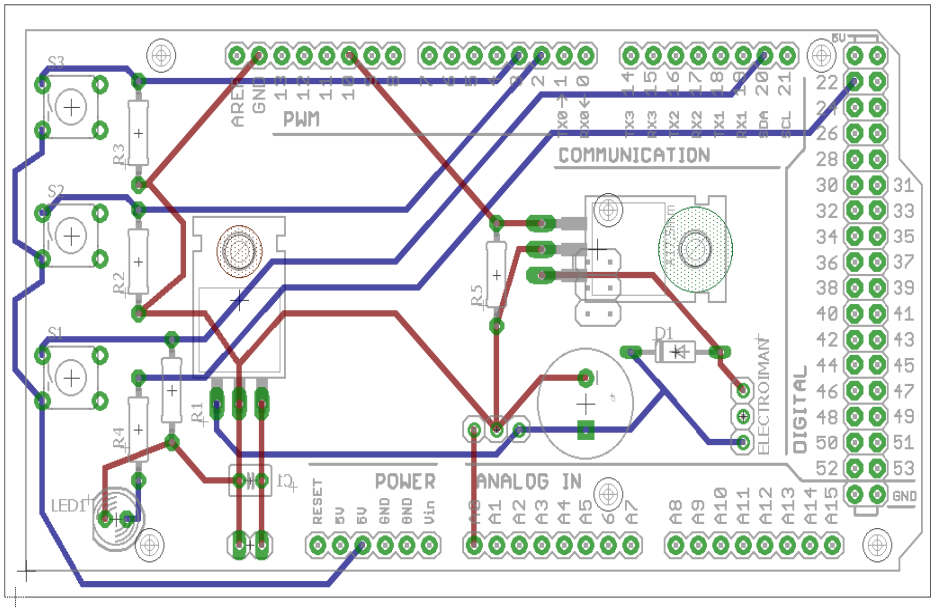

Figura 30 *Layout* da *Shield* para Arduíno

O resultado final da construção do sistema de levitação magnética é apresentado na Figura 31. A caixa estanque é usada para proteção do Arduíno e da *Shield*.

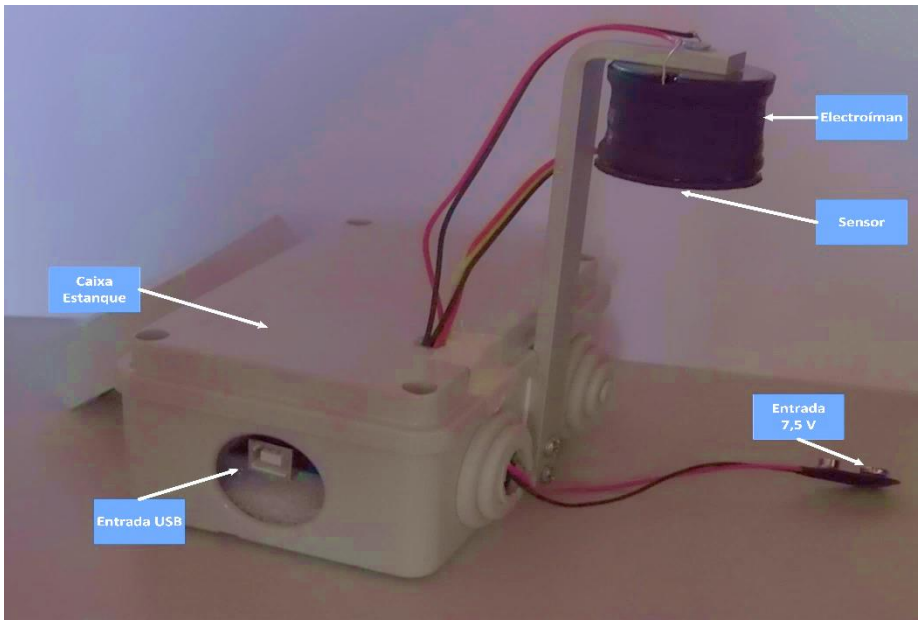

Figura 31 Sistema de levitação magnética

5. MODELAÇÃO DO SISTEMA

A modelação de um sistema dinâmico é definido pelo conjunto de equações que representa a dinâmica de seu comportamento, obtidas com base nas leis físicas pertinentes ao mesmo [3].

Em engenharia a utilização de modelos matemáticos que descrevam o sistema que se pretende controlar permite não só prever o seu comportamento, como também criar facilmente um processo de controlo de modo a obter os resultados desejados.

5.1. DESCRIÇÃO DO SISTEMA

O sistema de levitação magnética concebido baseia-se na criação de um campo magnético através da passagem da corrente por um solenóide. Este campo magnético tem como consequência o surgimento de uma força magnética num objeto de massa **m**, cuja posição é fornecida por um sensor de efeito Hall. A modelação deste sistema permite criar as equações matemáticas que regem toda a dinâmica do sistema, e com isso facilitar a criação de um controlador apropriado para este sistema.

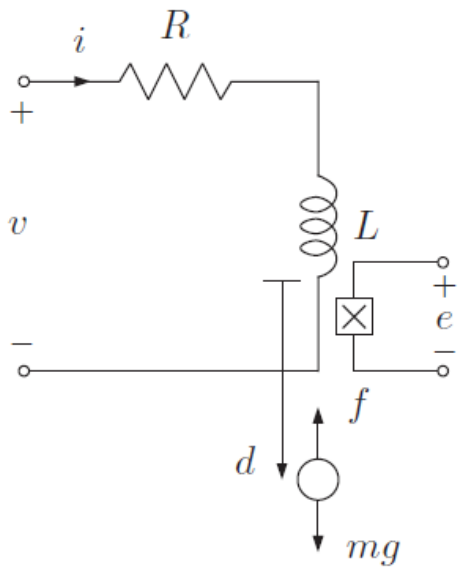

Figura 32 Representação das forças atuantes do sistema

Assumindo que $\boldsymbol{x_0}$ representa a posição de equilíbrio (2 cm), o somatório das duas forças que atuam no objeto, representadas na Figura 32, é igual a zero. Para este equilíbrio ocorrer é necessário uma corrente $\boldsymbol{I_0}$ de modo a manter esta posição de equilíbrio. Aplicando a 2ª lei de Newton, obtém-se:

$$m\ddot{x} = mg - F_{mag} \tag{14}$$

Onde: m – Massa da esfera.

x – Posição da esfera.

g – Aceleração da gravidade.

F_{mag} – Força magnética.

A equação (14) irá ser utilizada para a modelação do sistema. Através dela é obtida uma relação entre a posição da esfera e a força magnética produzida pelo eletroíman. Como esta força está relacionada com a corrente fornecida, primeiramente é necessário fazer o modelo matemático do eletroíman.

5.2. ELETROÍMAN

A força magnética gerada pela bobina percorrida por uma corrente $\boldsymbol{i}$, é dada por [4]:

$$F_{mag} = -\frac{i^2}{2} \times \frac{dL}{dx} \tag{15}$$

Onde: i – Corrente que percorre a bobina.

L – Indutância total do sistema.

F_{mag} – Força magnética.

O valor da indutância da bobina não é fixa, variando conforme a posição da esfera. Quando esta se aproxima da bobina, a indutância total aumenta, e quando a esfera se afasta, a indutância atinge um valor mínimo denominado indutância total natural da bobina [5].

A indutância total é dada por:

$$L = L_1 + \frac{L_0 \times x_0}{x} \tag{16}$$

onde: L_1 – Indutância natural da bobina.

L_0 – Indutância adicionada pela presença da esfera na posição x_0.

Reescrevendo a equação (15), substituindo a indutância pela equação (16), obtém-se:

$$F_{mag} = -\frac{i^2}{2} \times \frac{d}{dx}\left(L_1 + \frac{L_0 \times x_0}{x}\right)$$

$$= -\frac{i^2}{2} \times \left(\frac{-L_0 \times x_0}{x^2}\right)$$

$$= \frac{L_0 \times x_0}{2} \times \left(\frac{i}{x}\right)^2 \tag{17}$$

Atribuindo o valor de $k = \frac{L_0 \times x_0}{2}$, pode-se reescrever a equação (17) como:

$$F_{mag} = \ k \times \left(\frac{i}{x}\right)^2 \tag{18}$$

Definido um ponto de equilíbrio, a linearização em torno desse ponto pode ser realizada através de uma aproximação linear pela expansão da equação (17) em série de Taylor:

$$F_{mag}(x,i) \cong f(x_0,i_0) + \frac{df}{dx_{(x_0,i_0)}}(x-x_0) + \frac{df}{di_{(x_0,i_0)}}(i-i_0)$$

$$= k\left(\frac{i_0}{x_0}\right)^2 - \ 2k\left(\frac{{i_0}^2}{{x_0}^3}\right)x(t) + \ 2k\left(\frac{i_0}{{x_0}^2}\right)i(t) \tag{19}$$

onde: i_0 – Corrente de equilíbrio.

x_0 – Posição de equilíbrio.

i – Corrente incremental.

x – Posição de incremental.

Sabendo que na posição de equilíbrio a esfera se encontra parada, a força resultante nesta é igual a zero, o que leva a que força gravítica tenha o mesmo valor absoluto da força magnética.

$$mg = k\left(\frac{i_0}{x_0}\right)^2$$

$$k = mg\left(\frac{x_0}{i_0}\right)^2 \tag{20}$$

Através da equação (20) é possível calcular a constante k. Para os parâmetros:

m = 0,002 Kg

g = 9,81 m/s^2

x_0 = 0,02 m

i_0 = 0,13 A

$$k = 0.002 \times 9{,}81 \left(\frac{0{,}02}{0{,}13}\right)^2$$

$$k = 4{,}64 \times 10^{-3}\ N.m^2/A^2$$

Sabendo que é necessário controlar as variáveis incrementais corrente e posição para manter a esfera em equilíbrio, para pequenos deslocamentos é possível reescrever a equação (19), substituindo a força magnética pela sua série de Taylor:

$$m\ddot{x} = mg - k\left(\frac{i_0}{x_0}\right)^2 - 2k\left(\frac{{i_0}^2}{{x_0}^3}\right)x(t) + 2k\left(\frac{i_0}{{x_0}^2}\right)i(t)$$

$$m\ddot{x} = -2k\left(\frac{{i_0}^2}{{x_0}^3}\right)x(t) + 2k\left(\frac{i_0}{{x_0}^2}\right)i(t)$$

Substituindo $-2k\left(\frac{{i_0}^2}{{x_0}^3}\right)$ pela constante K_1, e $2k\left(\frac{i_0}{{x_0}^2}\right)$ pela constante K_2:

$$m\ddot{x} = K_1.x(t) + K_2.i(t) \tag{21}$$

Aplicando a transformada de Laplace à equação (10):

$$\frac{X(s)}{I(s)} = \frac{K_2}{ms^2 - K_1} \tag{22}$$

Para as condições de massa e posição e corrente de equilíbrio:

$$\frac{X(s)}{I(s)} = \frac{150{,}77}{s^2 - 981} \tag{23}$$

$$\frac{X(s)}{I(s)} = \frac{150{,}77}{(s - 31{,}320).(s + 31{,}320)} \tag{24}$$

Na equação (24) é apresentada a função de transferência do eletroíman, que será usada na modelação completa da planta. Existe um polo positivo, como é possível ver na Figura 33, logo o sistema é instável, tal como o esperado. O projeto do controlador irá permitir estabilizar o sistema.

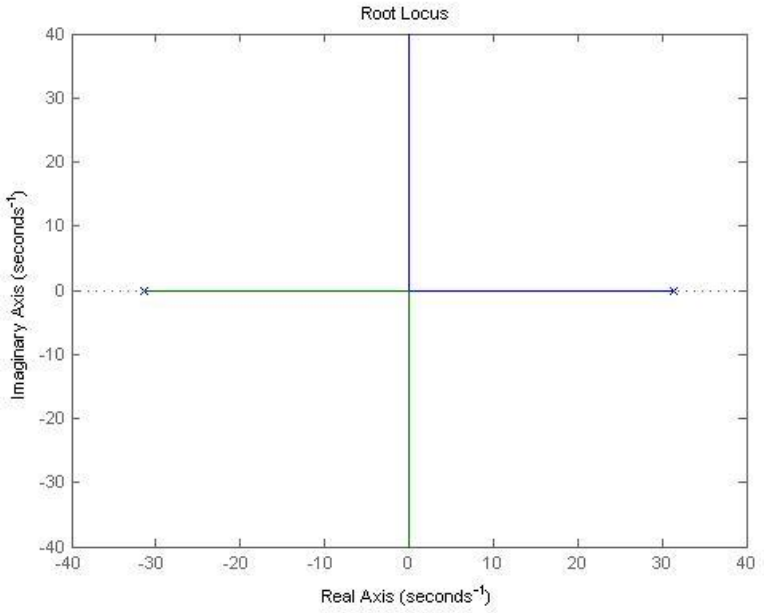

Figura 33 Lugar de raízes contínuo

5.3. SENSOR DE POSIÇÃO

Para determinar a posição do íman optou-se no projeto pela utilização de um sensor de efeito de Hall modelo A1324 do fabricante Allegro. Este integrado, representado na Figura 34, apresenta uma sensibilidade de 5 mV/G, representando G o valor do campo magnético em unidades CGS. Este sensor é colocado na base do eletroíman, o que coloca o problema de tanto detetar o campo magnético produzido pelo íman que levita, como pelo eletroíman quando percorrido por corrente.

Figura 34 Sensor de efeito de Hall A1324

Assim a tensão de saída do sensor tem três componentes como apresentado na equação (25). Uma componente constante α, uma componente dependente da distância do íman β, e uma componente γ relacionada com a corrente que percorre o eletroíman.

$$e = \alpha + \beta.\frac{1}{d^2} + \gamma.\,\mathrm{i} \qquad (25)$$

Apesar destes três componentes de saída, como as variações de corrente na zona de levitação são reduzidas, é possível ver este parâmetro como constante. Também o parâmetro da distância pode ser linearizado na zona de estabilidade, o que simplifica o modelo.

As medições nas imediações da zona de equilíbrio e linearização da saída do sensor pode ser consultado na Tabela 6 e Figura 35.

Tabela 6 Tensão de saída do sensor em função da distância

Distância (mm)	Tensão de saída (V)
15	3,13
20	3,00
25	2,86
30	2,72

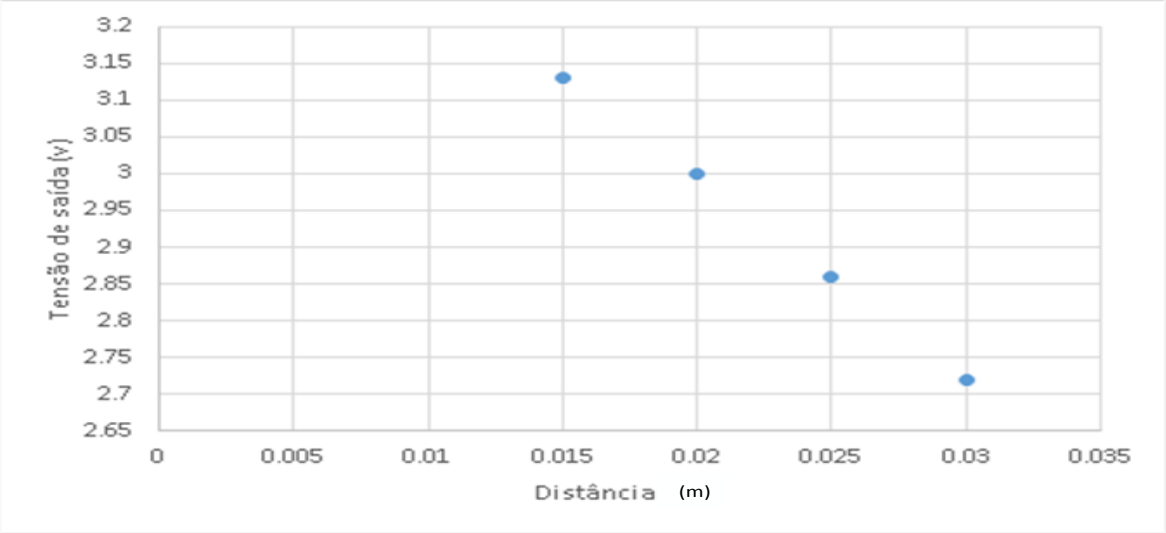

Figura 35 Gráfico da tensão de saída do sensor

A partir dos dados experimentais da Tabela 6 foi obtida a Figura 35, e a saída linearizada para a seguinte equação:

$$v = -27{,}4.\, d + 3{,}544 \tag{26}$$

5.4. CIRCUITO DE POTÊNCIA

Como já referido, devido à elevada corrente que o eletroíman necessita para originar a força magnética, não é possível fornecer essa corrente diretamente do Arduíno. Na posição de equilíbrio, por exemplo, são necessários 0,13 A, sendo que a corrente máxima de saída de 40 mA.

O circuito de potência utilizado é controlado por um MOSFET de potência IRF540N. A entrada da *gate* é ligada ao sinal PWM fornecido pelo Arduíno, que controla a corrente fornecida ao eletroíman. Para linearizar este circuito variou-se a saída do PWM entre 0 e 255 (8 bits) e mediu-se a corrente que percorre o eletroíman. Os resultados podem ser observados na Tabela 7 e Figura 36. Devido à já referida limitação de corrente por parte do eletroíman optou-se por utilizar a saída PWM até aproximadamente metade do valor possível, limitação que também será levada em conta na implementação digital do controlador.

Tabela 7 Corrente de saída do circuito de potência em função do PWM

Saída PWM (8 bits)	Corrente Eletroíman (A)
0	0
10	0
20	0
30	0
40	0,04
50	0,10
60	0,17
70	0,25
80	0,32

90	0,40
100	0,48
110	0,54
120	0,63
130	0,71

A linearização resulta na seguinte equação:

$$i = 0{,}0073.PWM - 0{,}2509 \quad (27)$$

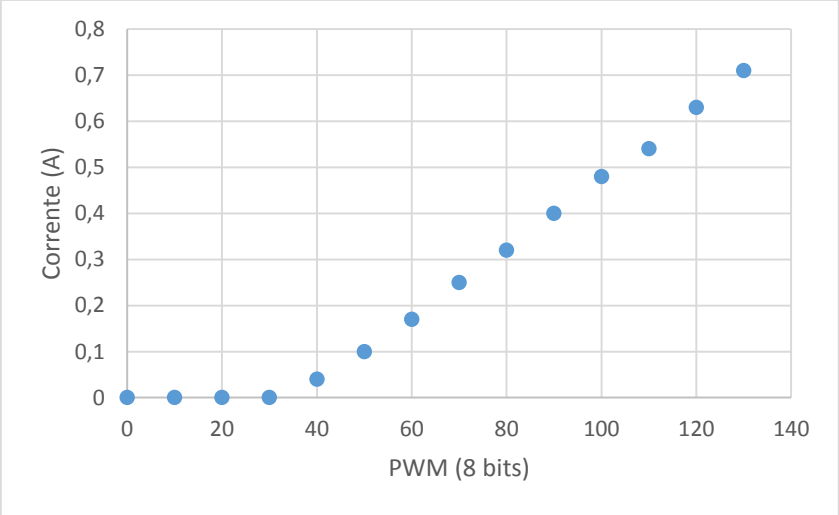

Figura 36 Gráfico da corrente de saída do circuito de potência

6. PROJETO E SIMULAÇÃO DOS CONTROLADORES

Neste capítulo serão descritos os controladores projetados para o sistema de levitação magnética, bem como todos os aspetos que envolveram esse projeto, como características, ajustes de ganhos e simulação.

Usando os modelos matemáticos determinados no capítulo anterior é possível criar um diagrama de blocos para fins de simulação através da aplicação Simulink. Os controladores têm de satisfazer duas especificações temporais: tempo de subida e *overshoot*, como indicados na Tabela 8.

Tabela 8 Requisitos de sistema para uma entrada em degrau

Percentagem de *Overshoot* (M_p)	$\leq 70\%$
Tempo de Subida (T_r)	$\leq 0{,}016\ s$

Na escolha destes parâmetros teve-se o cuidado de não definir valores que exigissem do controlador características que este não pudesse fornecer, como por exemplo uma corrente demasiado elevada. Assim, os valores de 70 % de *overshoot* e 0,016 s de tempo de subida

são escolhas conservadoras mas aceitáveis para esta aplicação, e permitem uma maior margem no projeto do controlador.

6.1. ANÁLISE EM MALHA ABERTA

A análise em malha aberta consiste no eletroíman e no amplificador de potência. O modelo de linearização está apresentado na Figura 37. A análise do lugar de raízes mostra que o sistema é instável, e como tal não estabiliza numa resposta ao degrau como mostra a Figura 38. Para possuirmos um sistema estável e que respeite as características desejadas é necessário efetuar um controlo em malha fechada.

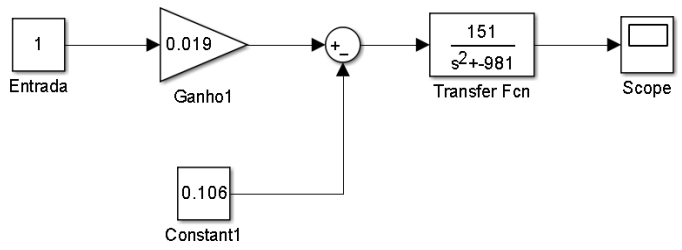

Figura 37 Linearização do modelo em malha aberta

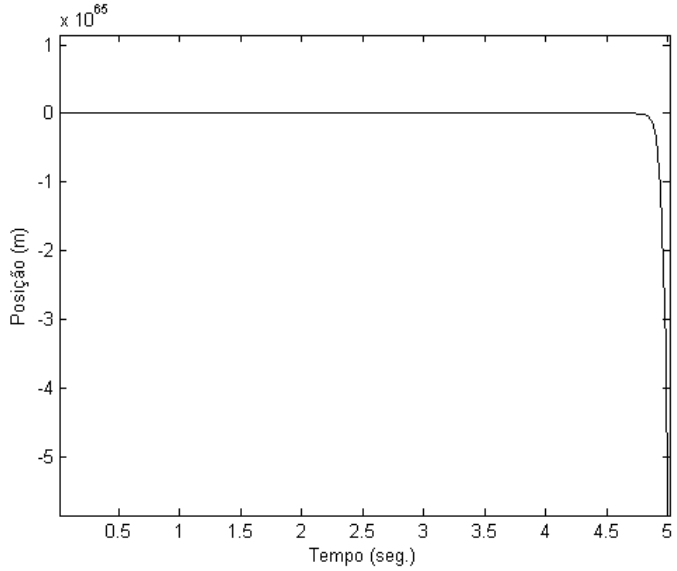

Figura 38 Resposta do sistema em malha aberta a entrada em degrau

6.2. ANÁLISE EM MALHA FECHADA

Para atingir a estabilidade, o sistema de levitação magnética deve ser colocado em malha realimentada com um controlador, sendo a posição do íman fornecida pelo sensor mencionado no capítulo 4.3. O diagrama de blocos completo pode ser visto na Figura 39.

Como o controlador vai ser implementado digitalmente é necessário ainda um conversor D/A (emulado por uma saída PWM) e um conversor A/D, para fazer interface com o controlador representado por $D(z)$ na Figura 39. Ambos os conversores estão presentes no microcontrolador Atmega 2560.

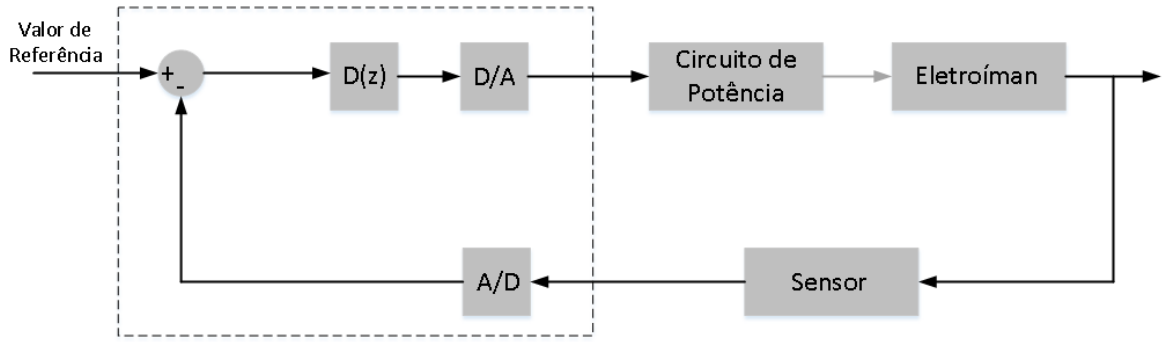

Figura 39 Diagrama de blocos do sistema

Apesar de implementados digitalmente, alguns controladores serão criados no domínio contínuo, outros no domínio digital de modo a experimentar diversos métodos de projeto. Quando projetados no domínio contínuo, de seguida os controladores são convertidos para discretos, verificando se o desempenho dos mesmos é alterado com esta conversão.

6.2.1. CONTROLADOR EM AVANÇO DE FASE

A introdução de um controlador em avanço de fase permite garantir ao sistema estabilidade e aumento de velocidade de resposta. Um controlador em avanço de fase é dado pela equação (28).

$$D(s) = K.\frac{s+z}{s+p}\ ,\ |z| < |p| \tag{28}$$

A correta posição do polo e do zero do controlador permite produzir um LGR com as características descritas na Tabela 8.

Este primeiro controlador será projetado no domínio discreto, logo é necessário começar por discretizar a função de transferência do eletroíman. O período de amostragem selecionado foi de T=0,003 s.

$$G(z) = (1 - z^{-1})\mathbb{Z}\left\{\frac{G(s)}{s}\right\}$$

$$= (1 - z^{-1})\mathbb{Z}\left\{\frac{150{,}77}{s(s-31{,}32).(s+31{,}23)}\right\}$$

$$= 6{,}8 \times 10^{-4} \times \frac{(z+1)}{(z-1{,}0994)(z-0{,}9095)} \qquad (29)$$

Novamente é possível ver que o sistema sem controlador é instável, pois tem um polo fora do círculo unitário, e não existe um ganho para o qual o sistema estabilize, tal como apresentado no lugar de raízes na Figura 40.

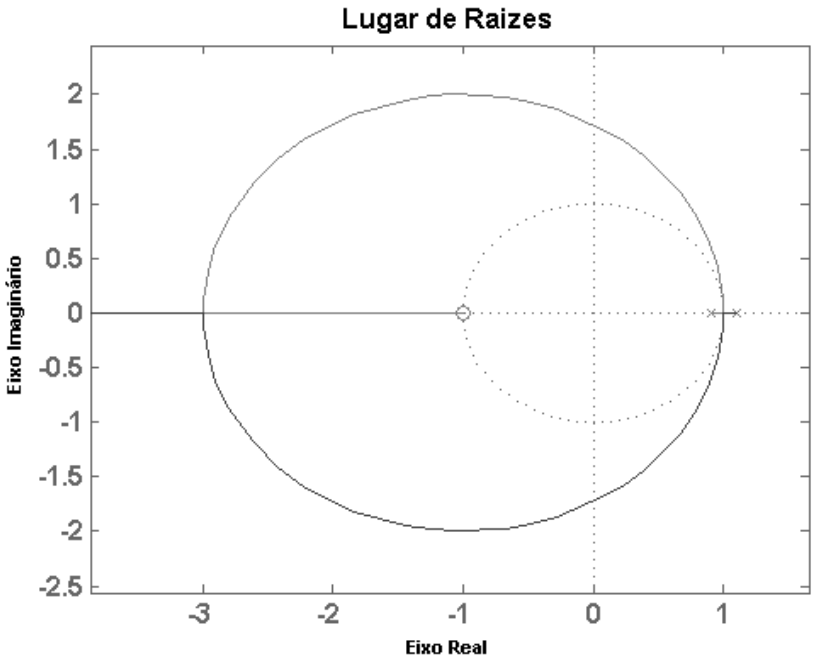

Figura 40 Lugar de raízes discreto do sistema sem controlador

De seguida é encontrada a função de transferência do sistema em malha aberta, denominada $G1(z)$, onde K_a e K_s representam os ganhos do circuito de potência e sensor, respetivamente.

$$G1(z) = K_a.K_s.G(z)$$

$$= 1{,}36 \times 10^{-4} \times \frac{(z+1)}{(z-1{,}0994)(z-0{,}9095)} \quad (30)$$

As especificações do controlador definidas na Tabela 8 permitem calcular o coeficiente de amortecimento (ζ) e a frequência natural (ω_n) que se deseja para o sistema.

Para $M_P \leq 70\ \%, T_r \leq 0{,}016\ s$:

$$\zeta = 0{,}6.(1 - M_P) = 0{,}18 \quad (31)$$

$$\omega_n = \frac{1{,}8}{T_r} = 112\ \text{rad/s} \quad (32)$$

Os polos desejados são encontrados através da expressão:

$$z_{1,2} = r.e^{\pm j\theta} \quad (33)$$

Sendo que os parâmetros r e θ são calculados do seguinte modo:

$$\begin{cases} r = e^{-\zeta\omega_n T} = 0{,}941 \\ \theta = \omega_n.\ \text{T}.\sqrt{1-\zeta^2} = 0{,}330 \end{cases} \quad (34)$$

Os polos desejados para o sistema controlado são:

$$z_{1,2} = 0{,}941.e^{\pm j0{,}330}$$

$$= 0{,}890 \pm j.\,0{,}305 \quad (35)$$

Sendo o controlador digital de primeira ordem, da forma $D(z) = K\frac{z-\alpha}{z-\beta}$, neste método o zero do controlador deve cancelar um polo do processo (z=0,9095).

De seguida para encontrar o ganho e o polo do controlador, são aplicadas as condições de fase e módulo:

$$\angle D(z).G1(z)|_{z=z_1} = -180^0 \quad (36)$$

$$|D(z).G1(z)|\Big|_{z=z_1} = 1 \quad (37)$$

Através da equação (36) obtém-se o valor do polo do controlador em avanço β =0,7408 e da equação (37) o valor do ganho K=96,234, obtendo-se o seguinte controlador:

$$D(z) = 96{,}234\frac{z-0{,}9095}{z-0{,}7408} \quad (38)$$

Após encontrar um controlador é pertinente fazer uma análise da função de transferência do sistema em malha fechada, onde se pode observar se as condições especificadas para o sistema são ou não respeitadas.

Na Figura 41 está representado o lugar de raízes do sistema. Pode-se verificar que o sistema é estável, pois os polos encontram-se dentro do círculo unitário, e na posição desejada.

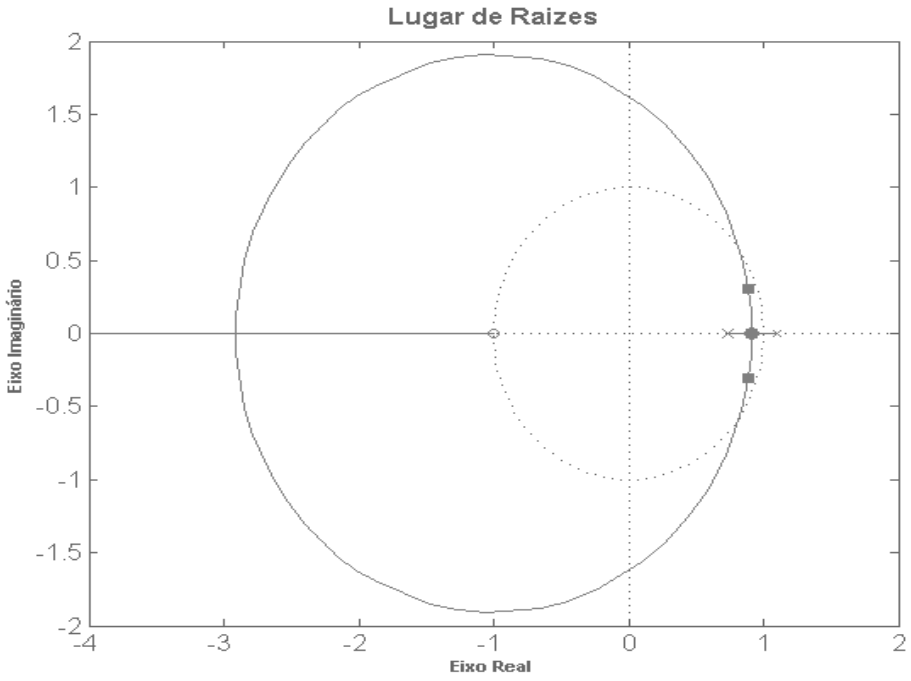

Figura 41 Lugar de raízes discreto com controlador

Com a resposta ao degrau do sistema visível na Figura 42 confirma-se que o controlador respeita as condições iniciais pretendidas. O sistema apresenta um tempo de subida de $T_r = 0{,}0108$ s e um *overshoot* de $M_p = 54{,}1\%$. Notar que este sistema apresenta um erro em regime permanente.

Figura 42 Resposta ao degrau unitário do sistema

6.2.2. CONTROLADOR EM AVANÇO-ATRASO DE FASE

Com o controlador em avanço de fase foi possível estabilizar o sistema, obter o tempo de subida e *overshoot* desejados. No entanto este tipo de controladores podem apresentam um erro em regime permanente elevado que não é desejado para a aplicação. Uma das formas de reduzir esse erro é introduzir uma componente de atraso de fase ao controlador projetado na secção anterior, combinando as vantagens da compensação específica por atraso e avanço de fase, tal como ilustra a Figura 43.

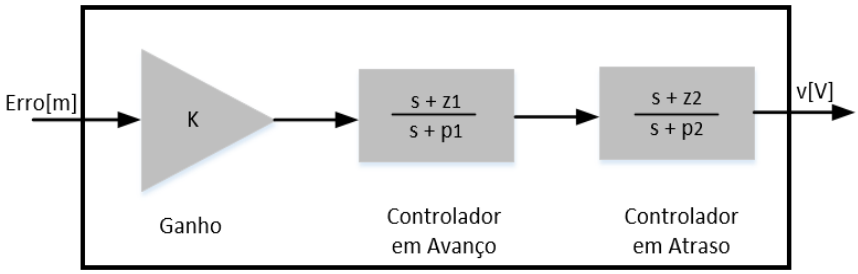

Figura 43 Bloco do controlador Avanço-Atraso de Fase

Ao contrário do que aconteceu na secção anterior este novo controlador será projetado no domínio contínuo, de forma a apresentar um outro método de projeto de sistemas de controlo. Por isso, como será utilizado o controlador em avanço, é necessário convertê-lo do domínio discreto para contínuo. Essa conversão foi executada através do comando "d2c" do Matlab, apresentado na Figura 44.

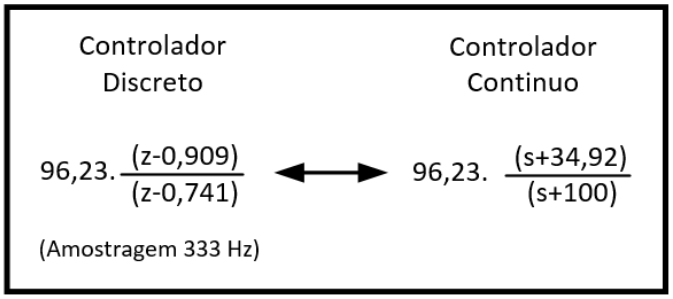

Figura 44 Controlador em Avanço de Fase contínuo e discreto

O próximo passo é calcular o erro em regime permanente do controlado em avanço e estipular qual o erro aceitável para o controlador em avanço-atraso de fase.

O erro em regime permanente do controlador em avanço é dado pela expressão:

$$E_{ss} = \lim_{s\to 0} s.\frac{u(s)}{1+\frac{G_n(s)}{G_d(s)}} \tag{39}$$

Onde $u(s)$ é a entrada em degrau:

$$u(s) = \frac{1}{s} \tag{40}$$

$G_n(s)$ e $G_d(s)$ são o numerador e denominador resultante da multiplicação do controlador em avanço e da função de transferência do sistema. Reescrevendo a equação (31):

$$E_{ss} = \lim_{s\to 0} \frac{G_d(s)}{G_d(s)+G_n(s)}$$

$$= \frac{-98100}{-98100+506438} = -0{,}24 \tag{41}$$

Como é possível ver na equação (41) o erro em regime permanente é de -0,24. De referir que o erro é a posição de referência menos a posição real. Para o caso de a entrada em degrau ser 0,02 (posição de equilíbrio do íman) o erro é de -0,0048, ou seja, prevê-se que a posição final do íman seja de 0,0248 m. Ao implementar a parcela de atraso de fase ao controlador deseja-se reduzir este erro para ¼ do seu valor, ou seja, terá uma posição final de 0,0212 m.

Ao adicionar uma componente em atraso de fase ao controlador adiciona-se um polo (***P***) e um zero (***Z***), assim o cálculo do novo erro permanente passa a ser:

$$E_{ss} = \lim_{s\to 0} s.\frac{u(s)}{1+\frac{G_n(s).(s-\boldsymbol{Z})}{G_d(s).(s-\boldsymbol{P})}}$$

$$= \lim_{s\to 0} \frac{G_d(s).\boldsymbol{P}}{G_d(s).\boldsymbol{P}+G_n(s).\boldsymbol{Z}} \quad (42)$$

Reescrevendo a equação (42):

$$\frac{\boldsymbol{Z}}{\boldsymbol{P}} = \frac{G_d(0)-G_d(0).E_{ss}}{G_n(0).E_{ss}}$$

Para o valor de erro (E_{ss}) desejado de 0,0012 m:

$$\frac{\boldsymbol{Z}}{\boldsymbol{P}} = \frac{-98100-5886}{-30386} = 3{,}51 \quad (43)$$

A relação entre polo e o zero do controlador em atraso de fase tem de ser de 3,51 para o erro pretendido, no entanto nem todos os valores podem ser usados. Como não se pretende alterar o lugar de raízes do controlador original, o ângulo do polo menos o ângulo do zero deve ser o mais próximo possível de zero. No entanto existe um limite para a proximidade do polo e do zero ao eixo imaginário, relacionados com as limitações do controlador real. Para evitar alterações da resposta do sistema em regime transitório, uma regra empírica utilizada é colocar o zero do compensador em avanço à direita da projeção do eixo real dos polos dominantes, numa razão entre 50 a 100 [27].

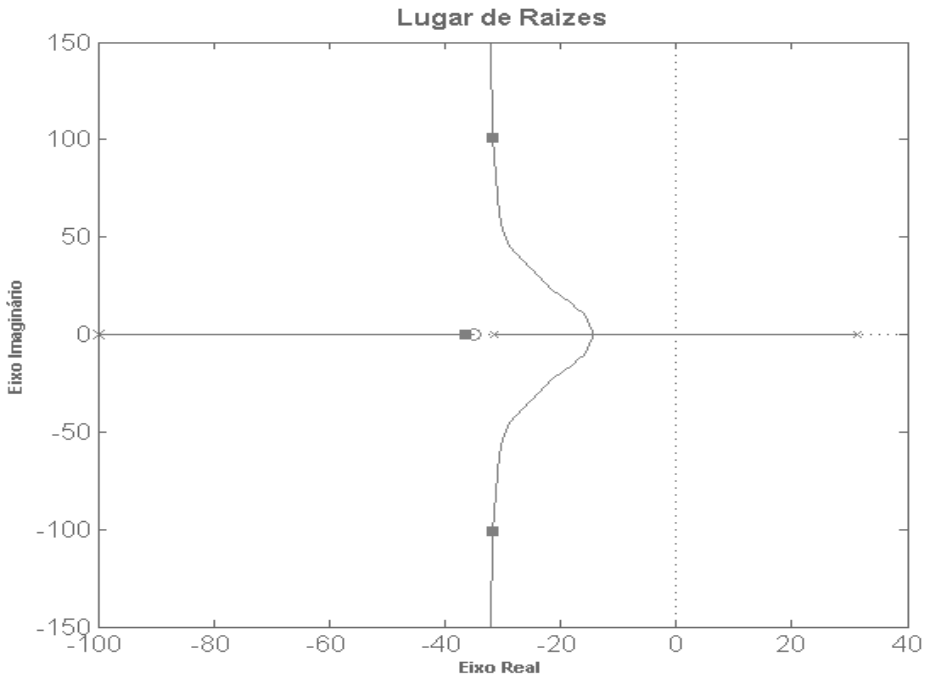

Figura 45 Lugar de raízes contínuo com controlador em Avanço de Fase

Pode-se observar na Figura 45 que no eixo real os polos dominantes do sistema em malha fechada encontra-se em -30. Assim o novo polo e zero do controlador serão colocados em:

$$\mathbf{Z} = \frac{30}{50} = 0{,}6$$

$$\mathbf{P} = \frac{0{,}6}{3{,}51} = 0{,}17$$

O controlador final é apresentado na Figura 46.

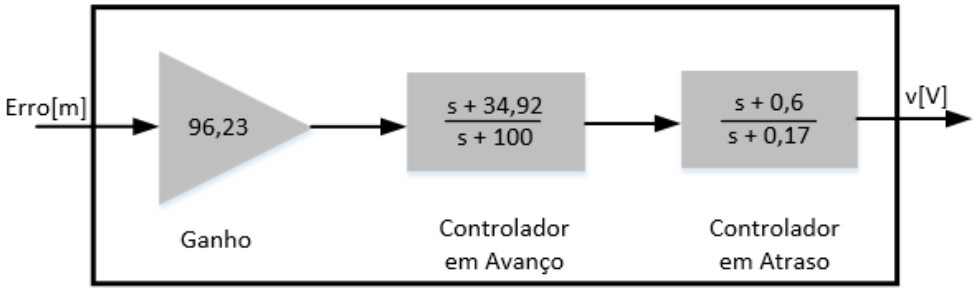

Figura 46 Controlador contínuo Avanço-Atraso de Fase

O novo controlador permite que o sistema apresente um tempo de subida de $T_r = 0{,}0107$ s, não alterando o controlador em avanço, mas o valor de *overshoot* passou para 64,8 %. O tempo de estabelecimento é de 3,08 s, o que é bastante elevado, no entanto aceitável para esta aplicação. Uma preocupação que se deve ter é no caso de para uma referência em onda quadrada, dar tempo suficiente para o sistema estabilizar. O erro passou para o valor desejado de -0,0012 m. Na Figura 47 está representada a resposta do sistema ao degrau.

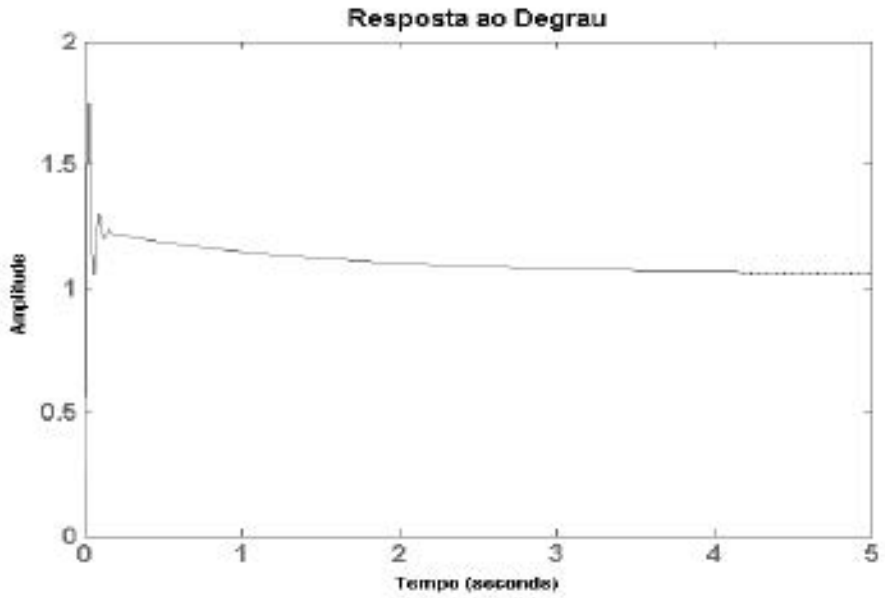

Figura 47 Resposta ao degrau unitário do sistema com controlador Avanço-Atraso contínuo

De seguida é necessário passar o controlador para o domínio discreto para ser implementado no microcontrolador e verificar se as características do contínuo se mantêm. A conversão foi executada através do comando "c2d" do Matlab, apresentado na Figura 48.

Controlador Continuo		Controlador Discreto
$96{,}23.\frac{(s+34{,}92).(s+0{,}6)}{(s+100).(s+0{,}17)}$	⟷	$96{,}23.\frac{(z-0{,}9982).(z-0{,}9096)}{(z-0{,}9995).(z-0{,}7408)}$
		(Amostragem 333 Hz)

Figura 48 Controlador em Avanço-Atraso de Fase contínuo e discreto

A passagem para discreto alterou algumas características do sistema. O tempo de subida diminui para 0,00957 s, o *overshoot* aumentou para 81,2 %, e o erro em regime permanente permaneceu em -0,0012 m. A resposta pode ser observada na Figura 49.

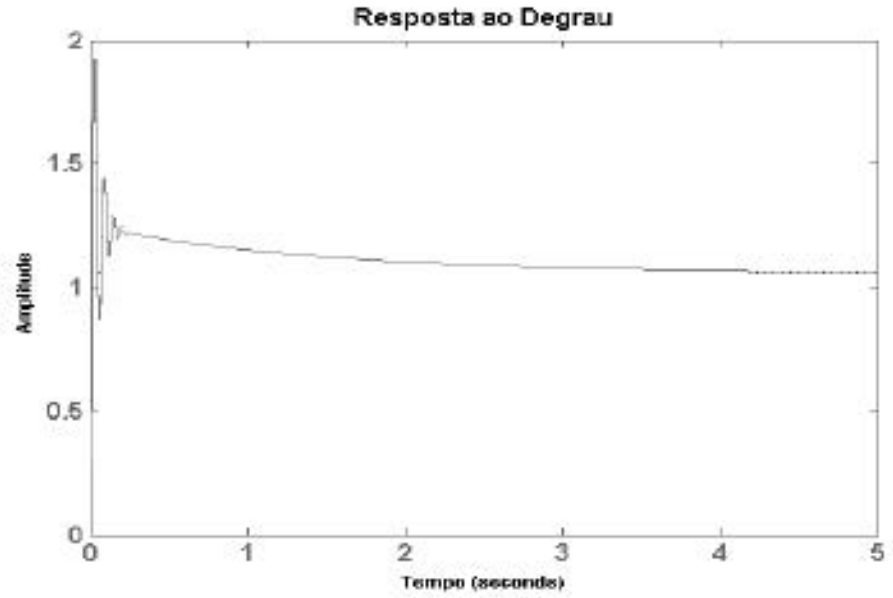

Figura 49 Resposta ao degrau unitário do sistema com controlador Avanço-Atraso discreto

6.2.3. CONTROLADOR PID

O controlador PID foi projetado no lugar de raízes, não sendo necessário sintonizar o controlador com um dos métodos apresentados no capítulo 2.4.1., visto ser conhecido o modelo matemático do processo. Após encontrados os parâmetros do PID e a respetiva equação, é apresentado o equivalentes discretos para este controlador.

No domínio das frequências o controlador PID pode ser representado da seguinte forma:

$$C(s) = K_p(1 + \frac{1}{T_i.s} + T_d.\, s)$$

$$= \frac{K_p.T_d}{T_i} . \frac{\left(s+\frac{1}{T_i}\right).\left(s+\frac{1}{T_d}\right)}{s} \qquad (44)$$

$$= K.\frac{(s+\alpha).(s+\beta)}{s} \quad (45)$$

Onde:

$$\begin{cases} K = \frac{K_p.T_d}{T_i} \\ \alpha = \frac{1}{T_i} \\ \beta = \frac{1}{T_d} \end{cases}$$

Como se pode ver na equação (45), o controlador apresenta um polo em s=0, e dois zeros em s=$-\alpha$ e s=$-\beta$. Os requisitos da resposta ao degrau são os mesmos dos controladores anteriores, logo os polos do sistema possuem a mesma posição ($p_{1,2} = -20{,}25 \pm j.110{,}66$).

Para determinar a posição dos dois zeros e ganho do controlador utiliza-se o mesmo método usado para projetar o controlador em avanço. Um zero é colocada em -31,32, anulando um polo do sistema. O outro zero e o ganho são encontrados usando as equações (36) e (37). O controlador é apresentada na equação (46).

$$D(s) = 0{,}475\frac{(s+31{,}32).(s+175{,}4)}{s} \quad (46)$$

Sendo os valores de α e β, e K:

$$\begin{cases} \alpha = 31{,}32 \\ \beta = 175{,}4 \\ K = 0{,}475 \end{cases}$$

Para estes valores, através da equação (44) é possível obter os valores de K_p, T_i e T_d:

$$\begin{cases} K_p = 2{,}658 \\ T_i = 0{,}0319 \text{ s} \\ T_d = 0{,}0057 \text{ s} \end{cases}$$

Os resultados por simulação do controlador PID podem ser observados nas figuras seguintes. O *overshoot* do sistema é de 68,4%, o tempo de subida é de 0,008 segundos e o erro permanente como esperado é eliminado. O LGR e a resposta ao degrau estão apresentados na Figura 50 e Figura 51, respetivamente.

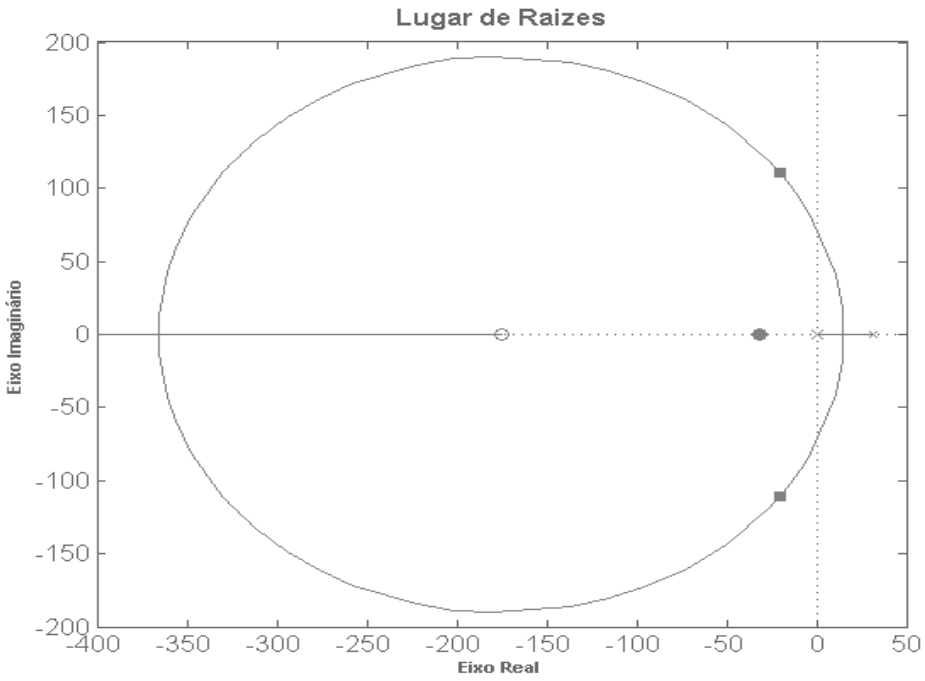

Figura 50 Lugar de raízes contínuo com controlador PID

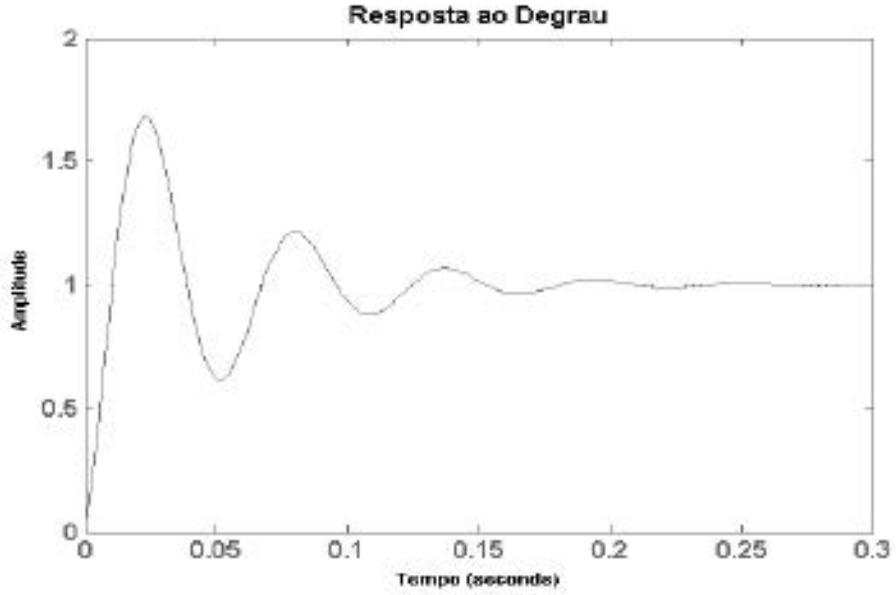

Figura 51 Resposta ao degrau unitário do sistema com controlador PID

A discretização do controlador PID é feita recorrendo-se a uma aproximação dos termos integral e derivativo. No domínio dos tempos o controlador PID apresenta a seguinte forma:

$$U(t) = K.\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau \ + T_d\frac{de(t)}{dt}\right) \tag{47}$$

Fazendo a aproximação dos termos integral e derivativo a:

$$\int_0^t e(\tau)d\tau \ \approx T\sum_{i=0}^{k} e(k) \tag{48}$$

$$\frac{de(t)}{dt} \approx \frac{e(k)-e(k-1)}{T} \tag{49}$$

Esta aproximação permite escrever a equação (50) no domínio discreto:

$$U(k) = K.\left(e(k) + \frac{T}{T_i}\sum_{i=0}^{k} e(k) + \frac{T_d}{T}\big(e(k) - e(k-1)\big)\right) \tag{50}$$

Fazendo as substituições:

$$\begin{cases} K_p = K \\ K_I = \frac{K.T}{T_i} \\ K_D = \frac{K.T_D}{T} \\ S(k) = \sum_{i=0}^{k} e(i) = S(k-1) + e(k) \end{cases}$$

Obtém-se a equação do controlador PID a implementar no controlador:

$$U(k) = K_p.e(k) + \ K_I S(k) + \ K_D\big(e(k) - e(k-1)\big) \tag{51}$$

Para os valores K_p= 2,658, T_i = 0,0319 s, T_d = 0,0057 s e T=0,003 s:

$$U(k) = 2{,}658.e(k) + \ 0{,}250.S(k) + \ 5{,}050.\big(e(k) - e(k-1)\big) \tag{52}$$

6.2.4. CONTROLADOR DIFUSO

O último controlador projetado para o sistema de levitação magnética foi um controlador difuso. Procura-se que este controlador tenho como principal característica a robustez, ou seja, deve ser capaz de suportar o melhor possível eventuais perturbações no sistema. Deseja-se também constatar se é possível obter melhores especificações temporais que no caso dos controladores convencionais.

Para este sistema difuso o mecanismo de inferência escolhido é do tipo *Mamdani*, descrito no capítulo 2.3.2. Com este mecanismo a saída das funções de pertença são conjuntos difusos, sendo por isso necessário cada variável de saída passar por um processo de defuzzificação. A alternativa a este método é o mecanismo de inferência do tipo *Sugeno*, onde não é necessário defuzzificação. Na Figura 52, é apresentado o diagrama de blocos de como o controlador difuso será integrado no sistema.

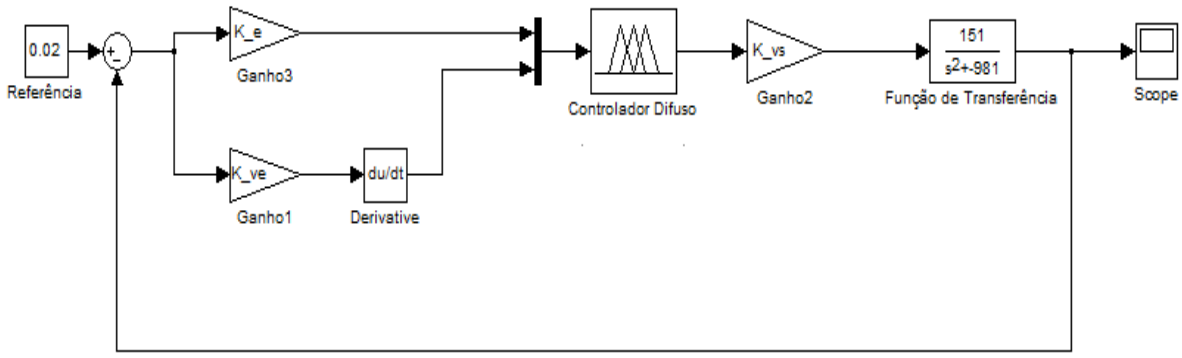

Figura 52 Diagrama de blocos do sistema com controlador Difuso

Para aplicar o mecanismo de inferência difuso escolhido são necessárias 5 etapas [28]:

1. Definição e fuzzificação das variáveis de entrada.
2. Aplicação dos operadores difusos (E, OU) nas variáveis para a criação da base de regas.
3. Criação das saídas difusas.
4. Agregação de todas as saídas.

5. Defuzzificação.

O primeiro passo foi a definição das variáveis difusas. O controlador apresenta duas variáveis de entrada, o erro ($\boldsymbol{e}$) e a variação do erro ($\boldsymbol{ve}$), e uma variável de saída, a tensão de saída ($\boldsymbol{vs}$).

- $\boldsymbol{e} \in [-\mathbf{1}, \mathbf{1}]$
- $\Delta\boldsymbol{e} \in [-\mathbf{1}, \mathbf{1}]$
- $\boldsymbol{vs} \in [-\mathbf{2}, \mathbf{2}]$

A cada uma destas variáveis será associado um ganho, por isso é possível alterar a amplitude de valores aceites, de modo a enquadrar os valores de entrada no intervalo escolhido.

A fuzzificação das variáveis tem como objetivo determinar, para cada entrada, o grau de pertença ao valor linguístico apropriado. Deste modo cada variável é qualificada segundo um valor linguístico que será requerido posteriormente para a base de regras. Assim o próximo passo na implementação do controlador é a definição desses valores linguísticos.

Tanto às variáveis de entrada como à de saída foram atribuídas 7 valores linguísticos: **NB** (negativo baixo), **NM** (negativo médio), **NA** (negativo alto), **Z** (zero)**, PB** (positivo baixo)**, PM** (positivo médio) e **PA** (positivo alto). A todos estes valores linguísticos foram associados numa função pertença, como mostra a Figura 53 e a Figura 54. Cada valor linguístico tem uma função triangular, com exceção dos valores linguísticos da ponta **NB** e **PA**, que são de forma trapezoidal.

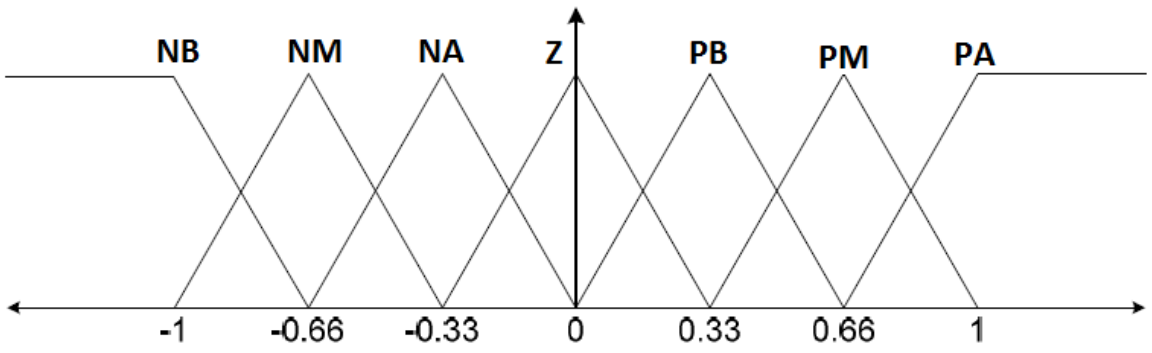

Figura 53 Função pertença das variáveis difusas de entrada

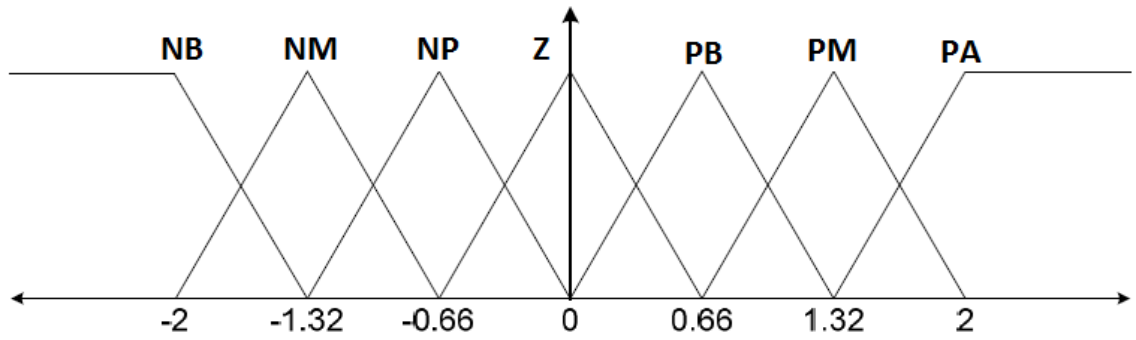

Figura 54 Função pertença da variável difusa de saída

A cada uma destas variáveis difusas é associado um fator de escala para sintonizar o controlador, como mostra a Tabela 9. Estes ganhos são ajustados por tentativa e erro durante a simulação até se atingir um controlo aceitável para o sistema.

Tabela 9 Fatores de escala

Fator de Escala	Valor Usado
e	k_e
ve	k_ve
vs	k_vs

O seguinte passo na implementação é definição de uma base de regras. Usar 7 valores linguísticos em cada uma das variáveis de entrada permite que a base de regras (Tabela 10) tenha 49 entradas. A criação das regras segue uma simetria, muitas vezes usadas nos controladores difusos com duas variáveis de entrada e uma de saída.

Tabela 10 Base de regras do controlador Difuso

e / Δe	**NB**	**NM**	**NA**	**Z**	**PB**	**PM**	**PA**
NB	NB	NB	NB	NB	NM	NA	Z
NM	NB	NB	NB	NM	NA	Z	PB
NA	NB	NB	NM	NA	Z	PB	PM
Z	NB	NM	NA	Z	PB	PM	PA
PB	NM	NA	Z	PB	PM	PA	PA
PM	NA	Z	PB	PM	PA	PA	PA
PA	Z	PB	PB	PA	PA	PA	PA

Cada uma das regras permite a atribuição de um *peso* associado, de forma a tornar algumas regras mais importantes que outras. Para este controlador foi escolhido atribuir o *peso* de 1 para cada regra, não existindo assim diferenciação entre elas.

O resultado do mapeamento feito é apresentado na Figura 55. Pode-se observar que existe simetria na superfície, que resulta da simetria da base de regras.

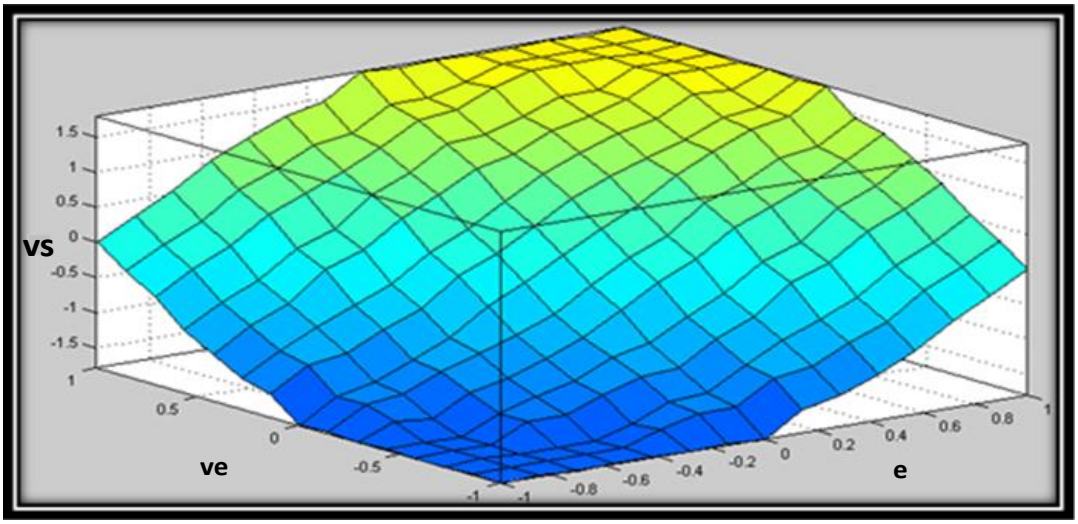

Figura 55 Superfície do controlador Difuso

Estando definidas as entradas e as regras é implementado o método de implicação, onde é definido como o controlador escala as funções de pertença de cada variável difusa. Os dois métodos mais comuns são o método de implicação mínimo e o método de implicação produto. No primeiro método a função pertença de saída é *cortada* no valor correspondente ao valor do respetivo peso. Um exemplo deste método é apresentado na Figura 56, onde uma variável de saída com três funções de pertença associadas, tem como peso de regra de cada uma de 0,8 , 0,3 e 0,5.

Figura 56 Exemplo do método de implicação mínimo numa variável difusa de saída

No método por produto existe uma atenuação das funções pertença, também influenciada pelo peso de cada regra. Na Figura 57, à mesma variável do exemplo anterior é aplicado o método de implicação por produto. No controlador projetado é indiferente qual dos métodos é usado, pois a cada regra é atribuída a mesma importância, logo o mesmo peso.

Figura 57 Exemplo do método de implicação produto numa variável difusa de saída

O próximo passo é escolher qual o método de agregação a usar. A agregação é o processo através do qual os gráficos que representam a saída de cada regra são combinados num único gráfico. Este processo ocorre para cada variável de saída e antecede o processo de defuzzificação. Existem três métodos principais para o processo de agregação [28]: *Max* (máximo) que será utilizado neste controlador; *Probor* (probabilístico OU); e *Sum* (soma) onde é simplesmente somado a saída gráfica de cada regra.

Por fim deve ser feita a definição de como será realizada a defuzzificação. Este processo tem como entrada todo o conjunto difuso criado no processo de agregação, e como saída um único valor. Os métodos de defuzzificação mais comuns são:

- Método do centro de massa ou Centróide – O valor numérico obtido representa o centro de gravidade da distribuição de probabilidade de saída do sistema difuso.

- Método da média dos máximos – O valor numérico obtido representa o valor médio de todos os valores centrais ativados pelas entradas.

- Método da média ponderada dos máximos - O valor numérico é obtido considerando a média ponderada dos valores centrais ativos, levando em conta o *peso* das regras.

- Método de critério máximo (mínimo) – O valor numérico é igual ao máximo (mínimo) valor ativado pelas entradas.

Neste controlador como método de defuzzificação foi escolhido o método de centro de massa.

Para efeitos de simulação foi criado um controlador difuso usando a GUI *fuzzy toolbox* do MATLAB, e o sistema testado na aplicação Simulink. Através do método de tentativa e erro, foi encontrada uma combinação de ganhos, que durante a simulação permitiram um bom resultado. O diagrama de blocos resultante pode visto na Figura 58, onde:

- $\boldsymbol{k\_e = 1}$
- $\boldsymbol{k\_ve = 0,05}$
- $\boldsymbol{k\_vs = 25}$

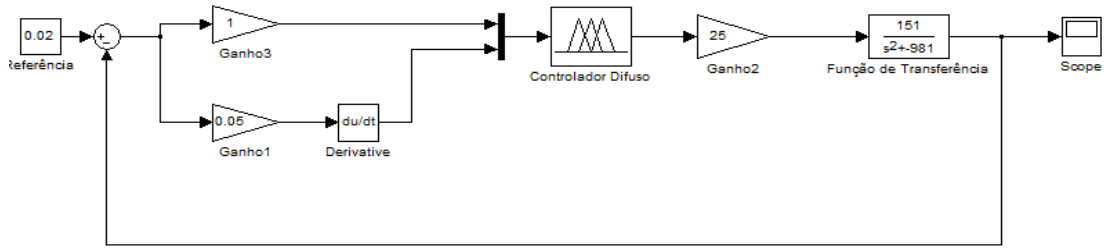

Figura 58 Diagrama final de blocos do sistema com controlador Difuso

A resposta do sistema ao degrau pode ser vista na Figura 59. Como se pode observar a resposta é bastante diferente da dos controladores convencionais projetados anteriormente. Não existe *overshoot* e o tempo de subida é bastante curto.

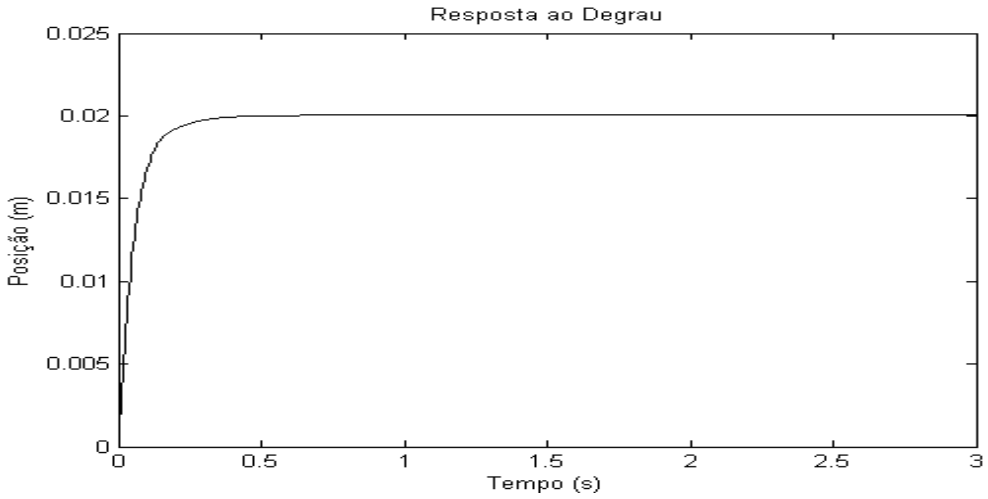

Figura 59 Resposta ao degrau do sistema com controlador Difuso

Apresentado que está o projeto e simulação dos quatro controladores, no próximo capítulo serão comparados os resultados obtidos em simulação com os resultados práticos, que levará à validação do modelo e dos controladores criados.

7. IMPLEMENTAÇÃO PRÁTICA E ANÁLISE DE RESULTADOS

A implementação do sistema de controlo foi realizada com sucesso, produzindo em todos os controladores um estado estável de levitação do íman, como mostra a Figura 60. De seguida serão apresentados aspetos comuns a todos os controladores, como configuração do PWM e interrupções, assim como análise de resultados.

Figura 60 Estado estável de levitação do íman

A existência de quatro controladores levou a que fosse necessário criar uma forma de a qualquer momento escolher qual dos controladores usar. O diagrama da Figura 61 mostra que esta escolha é feita através do valor da variável `cmd`, por exemplo, se tem o valor 2, executa o código correspondente ao controlador PID.

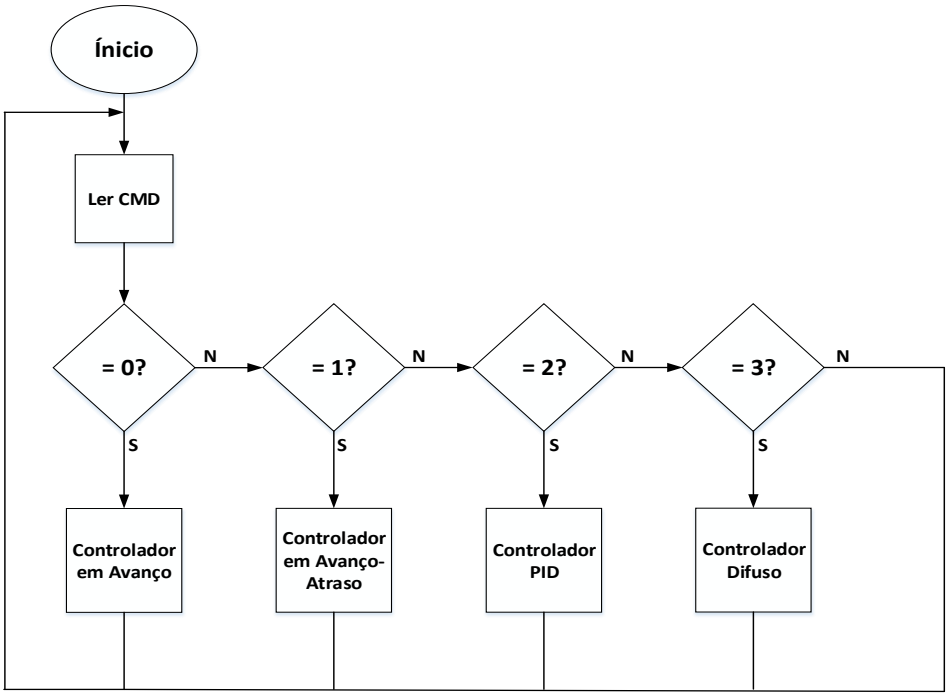

Figura 61 Fluxograma de escolha do controlador

A alteração do valor de `cmd` é feita por computador, enviando o comando correspondente ao controlador desejado. Do lado do Arduíno é verificado se existe alguma alteração no fim de cada controlador, através do código:

```
processCommand(Serial.read());
```

A função `processCommand` é assim responsável por fazer as alterações do código consequentes do envio de comando por parte do programa do computador. Estas alterações não são apenas quanto ao controlador a escolher, mas também é possível incrementar ou decrementar o valor de *setpoint* do controlador, ou mesmo ter como referência uma onda quadrada ou sinusoidal. O código completo da função `processCommand` pode ser consultado no anexo A.

Antes de descrever o modo de implementação de cada controlador e apresentar os respetivos resultados práticos, convém referir mais alguns aspetos de código gerais a todos os

controladores. Começando pela conversão A/D, por omissão esta conversão é realizada com um fator de divisão de 128, ou seja, reduz a frequência de trabalho durante a conversão em 128. Este fator de divisão é alterado para 16, de modo às conversões A/D se realizarem mais rapidamente.

```
ADCSRA &= ~PS_128;
ADCSRA |= PS 16;
```

Ainda relativamente à leitura analógica, a cada pedido de leitura por parte dos controladores é executada a função `ler_analog`, que faz a média de 20 leituras analógicas e retorna um valor em tensão. Este tipo de procedimento emula a aplicação de um filtro passa-baixo à entrada analógica.

```
float ler_analog()
{
  total=0;
  for (i=0; i<numReadings; i++){
    total=total + float(analogRead(A0));
  }
  return total/aux_analog;
}
```

Outro especto importante de referir é a saída PWM do controlador. Por omissão, todas as saídas PWM do Arduíno têm uma frequência de 500 Hz, que é uma frequência demasiado baixa para esta aplicação. É necessário aumentar este valor, e para isso altera-se o registo TCCR2B associado ao Timer2 que controla o PWM do pino de saída utilizado.

O seguinte código é usado para alterar a frequência para 4 KHz, colocando o bit CS21 com o valor de 1, o que permite um *prescaler* de 8:

```
void setupPWM()
{
   pinMode(10, OUTPUT);

   int myEraser = 7;
   int myPrescaler = 2;

   TCCR2B &= ~myEraser;
   TCCR2B |= myPrescaler;
}
```

Por fim referir que são usadas três tipos de interrupções: uma externa e duas internas. Uma interrupção interna está associada a Timer3, e executa a função `enviar` a cada 0,5 segundos.

```
Timer3.initialize(500000);
Timer3.attachInterrupt(enviar);
```

A outra interrupção está associada ao Timer1 e é utilizada para gerar as funções quadradas e sinusoidais, possíveis de usar como referência.

```
Timer1.initialize(4000000);
Timer3.attachInterrupt(callback_Quadrado);
```

Apesar de não estar previsto usar, criou-se no código uma interrupção externa, associada ao pino 3. O número de interrupção associada a este pino é a interrupção 1, e a função é ativada no flanco ascendente da alteração do valor lógico do pino de 0 para 1. Esta interrupção faz com que ative o controlador difuso. Para esta função funcionar corretamente foi ainda necessário implementar um simples código para *debounce*.

```
attachInterrupt(1, blink, RISING);

void blink()
{
static unsigned long last_interrupt_time = 0;
unsigned long interrupt_time = millis();

if (interrupt_time - last_interrupt_time > 200)
 {
   programa=3;
 }
   last_interrupt_time = interrupt_time;
}
```

Como já referido não foi implementado em *hardware* um botão para interrupção externo, pela razão de que todas as interfaces do programa podem ser feitas através da comunicação RS232 com o computador.

Apresentados que estão os aspetos de código mais relevantes e gerais a todos os controladores, de seguida serão analisados os controladores individualmente, e apresentados os resultados práticos e respetiva comparação com os obtidos por simulação.

7.1. CONTROLADOR EM AVANÇO DE FASE

O primeiro controlador a ser implementado foi o controlador em avanço de fase. Na Figura 62 pode-se observar o fluxograma que serviu como base para a implementação deste controlador. De referir que entre a simulação e o controlador implementado o ganho do

controlador teve de ser diminuído de 96,23 para 21,00. Esta alteração permitiu estabilizar o sistema, e impedir a saturação do controlador. A equação discreta para controlador foi obtida a partir da equação (38) do capítulo 6, sendo:

$$U[k] = 21{,}00.\,E[k] - 19{,}00.\,E[k-1] + 0{,}7408.\,U[k-1]$$

Para efeitos de implementação foi substituída a parcela $19{,}00.\,E[k-1] - 0{,}7408.\,U[k-1]$ por $X[k]$ obtendo-se:

$$U[k] = 21{,}00.\,E[k] + X[k]$$

Como se pode ver no fluxograma da Figura 62 existe a atualização do valor de $X[k]$ em cada ciclo antes do tempo de espera T.

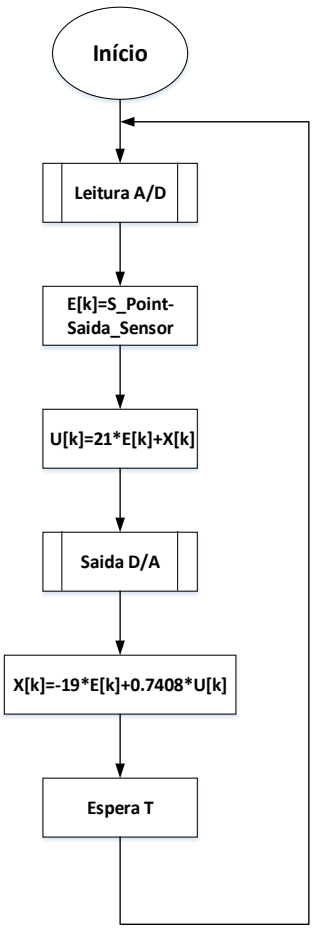

Figura 62 Fluxograma do controlador em Avanço de Fase

No código do controlador implementado pode-se observar que existe uma conversão de corrente para PWM, cuja relação pode ser vista na Figura 36. Existe ainda uma limitação do PWM de 128 para proteger o eletroíman do aquecimento.

```
processCommand(Serial.read());
saida_sensor=ler_analog();
err=s_point-saida_sensor;
gPWMCommand=21.0*err+x;

gPWMCommand_1=gPWMCommand*139.75 +35.355;

if (gPWMCommand_1>128  )
   {
    gPWMCommand_1=128;}
else if (gPWMCommand_1<0)
    {
gPWMCommand_1=0;}

analogWrite(10, gPWMCommand_1);
x=-19*err + 0.7408*gPWMCommand;
delay(1);
```

Aplicando os controladores é útil saber se o erro de posição em regime estacionário é semelhante ao erro obtido em simulação. A comparação entre a leitura do sensor e a simulação podem ser observadas na Figura 63 e Figura 64.

Na Figura 63, gráfico ilustrativo do sistema no primeiro segundo após estabilizar, pode-se observar que a posição do íman coincide sensivelmente com o obtido na simulação e nos cálculos teóricos. Do gráfico obtido conclui-se ainda que existem pequenas oscilações irregulares, que no entanto não são observáveis na prática.

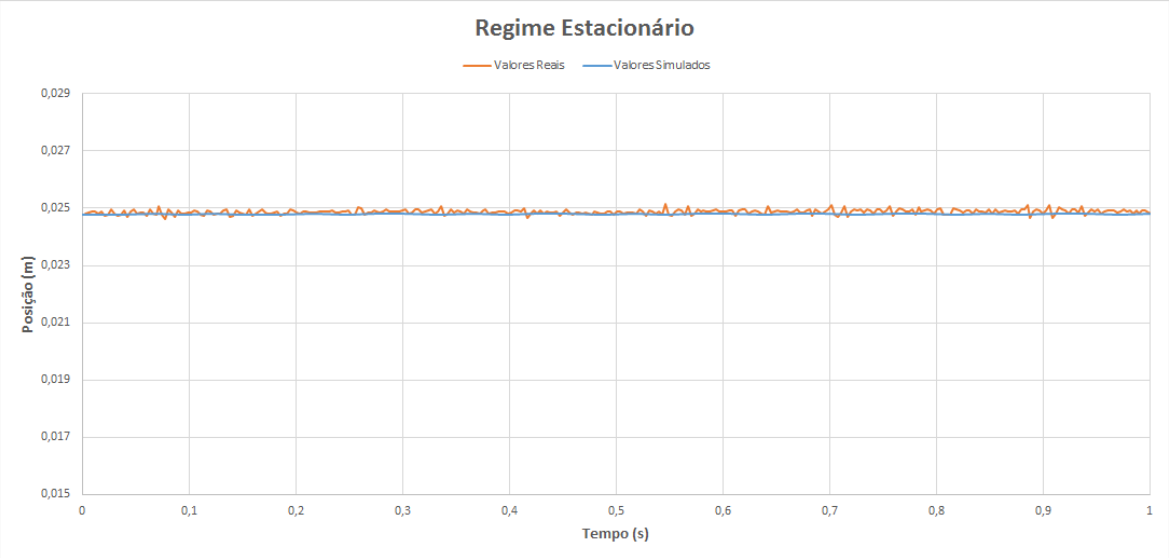

Figura 63 Posição em função do tempo – controlador em Avanço de Fase

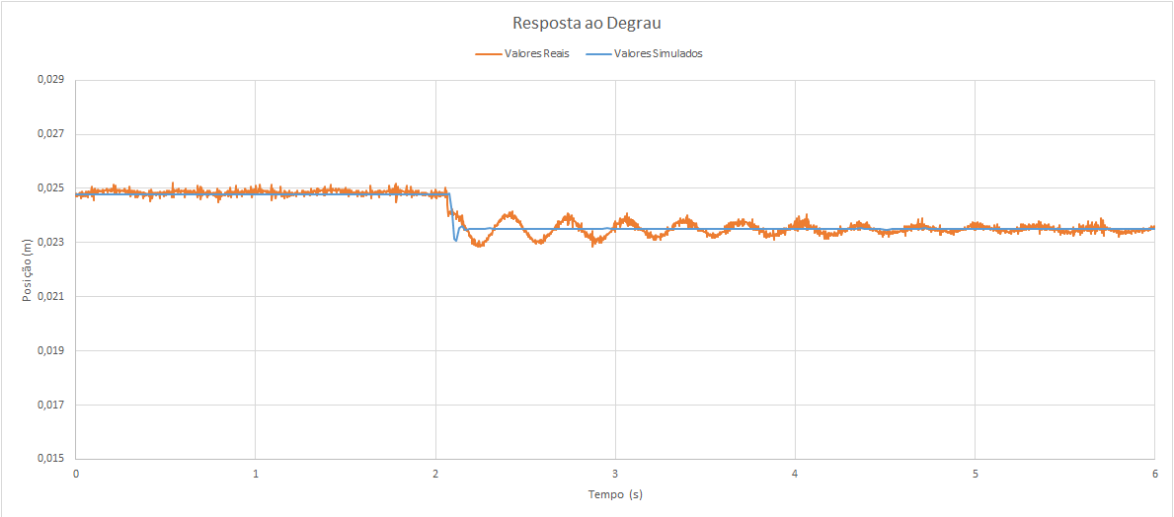

Figura 64 Resposta ao degrau – controlador em Avanço de Fase

Na Figura 64 vê-se que o controlador apresentou um *overshoot* superior ao da simulação para uma entrada em degrau que originou uma variação de 1,3 mm na posição do íman. Relativamente ao simulado os resultados práticos mostram também um tempo de estabelecimento maior.

7.2. CONTROLADOR EM AVANÇO-ATRASO DE FASE

O segundo controlador analisado é o controlador Avanço-Atraso de fase. Como se pode observar no fluxograma da Figura 65, a implementação no microcontrolador é bastante semelhante à do controlador anterior.

Também como no controlador em avanço de fase é necessário encontrar a equação discreta para implementar no microcontrolador. Neste caso a equação é:

$$U[k] = 31{,}50.E[k] - 60{,}00.E[k-1] + 1{,}7408.U[k-1] + 28{,}50.E[k-2] - 0{,}7408.U[k-2]$$

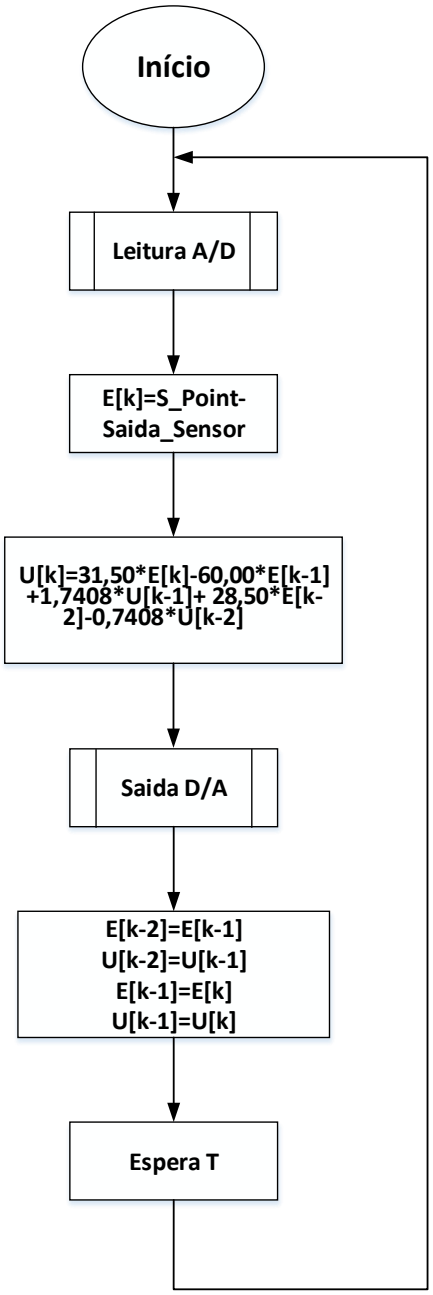

Figura 65 Fluxograma do controlador em Avanço-Atraso de Fase

Tal como no controlador em avanço também o ganho do controlador em Avanço-Atraso de Fase teve de ser reduzido, neste caso de 92,23 para 31,5.

Nos ensaios com o controlador em Avanço-Atraso de Fase, apresentados na Figura 66, constata-se que existe pequenas deslocações do íman, que se encontra apesar disso à distância esperada.

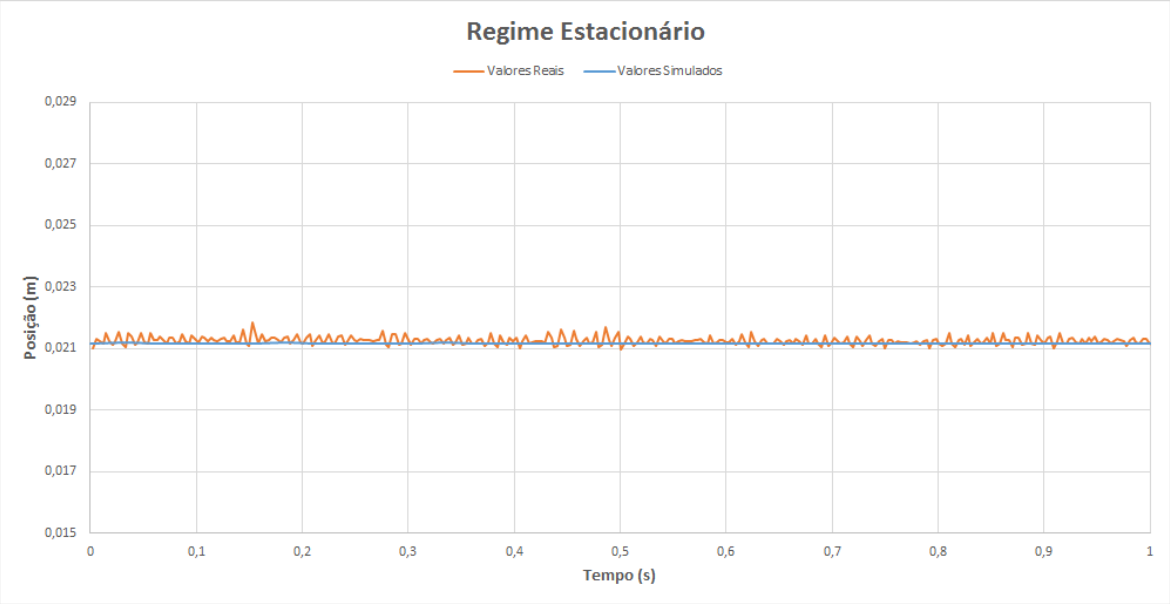

Figura 66 Posição em função do tempo – controlador em Avanço-Atraso de Fase

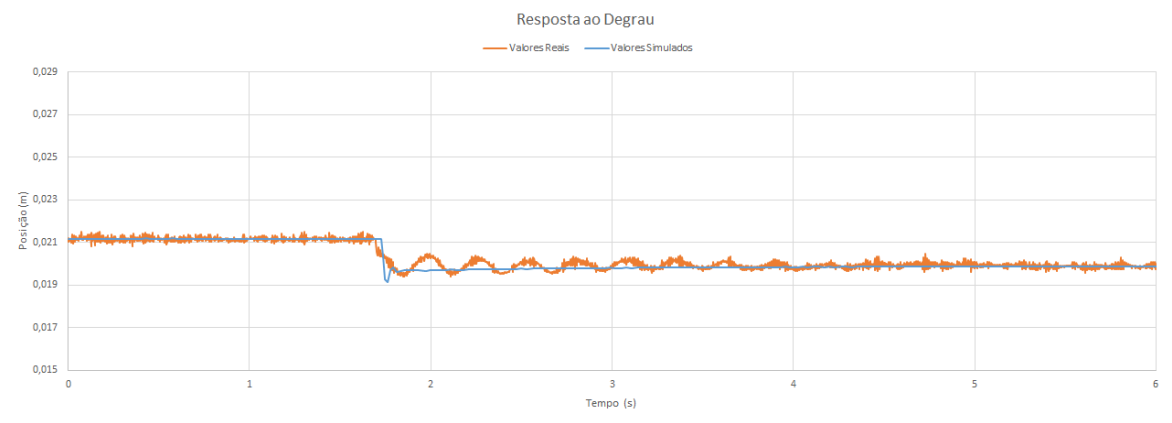

Figura 67 Resposta ao degrau – controlador em Avanço-Atraso de Fase

O degrau aplicado ao controlador em Avanço-Atraso de fase desloca o íman acima da posição de equilíbrio (2 cm), havendo uma alteração também de 1,3 mm para baixo (Figura 67). O *overshoot* foi também mais elevado que a simulação, com um maior tempo de estabelecimento e maior oscilação na altura da transição.

7.3. CONTROLADOR PID

No código do controlador PID, começa-se por definir os ganhos proporcional, derivativo e integral. Durante a fase de testes do controlador estes valores foram alterados, aumentando o ganho de 2,658 para 10,632. Os valores de T_d e T_i mantiveram-se os mesmos da simulação.

Neste caso a equação discreta do controlador já foi determinada, e é apresentada no capítulo 6, sendo a equação (52).

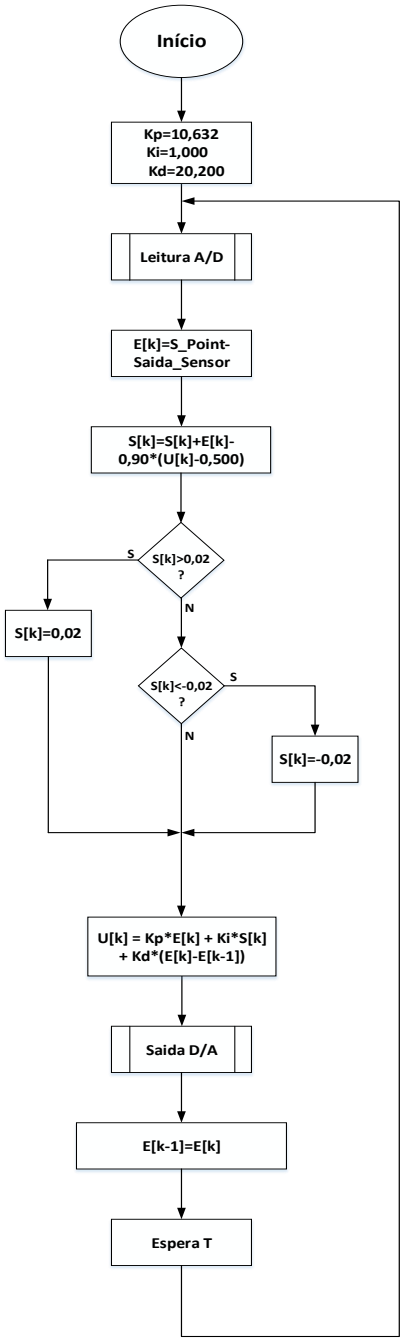

Figura 68 Fluxograma do controlador PID

No fluxograma do controlador PID da Figura 68, é apresentado um limitador do ganho do valor integral. Para limitar este valor é subtraído um valor proporcional da diferença entre o valor de saída $U[k]$, e o seu valor máximo admissível. Para além desta limitação são ainda estabelecidos valores fixos de máximo e mínimo de respetivamente 0,02 e -0,02. Estes valores de limitações foram escolhidos por tentativa e erro.

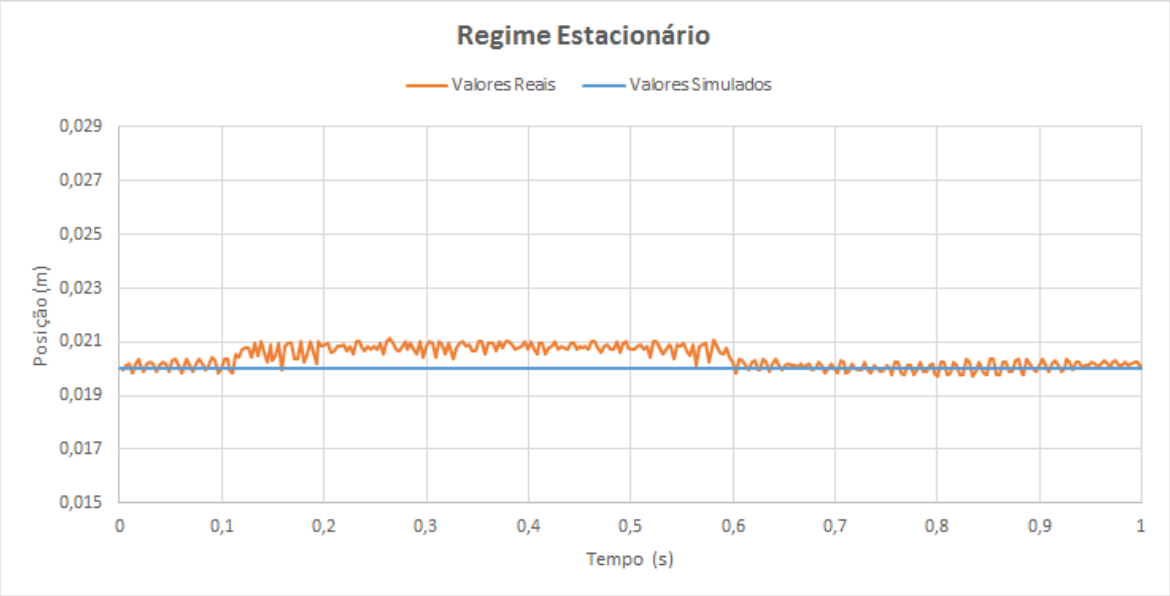

Figura 69 Posição em função do tempo – controlador PID

Os resultados práticos obtidos por este controlador não são satisfatórios. Como se pode ver na Figura 69, existem pequenas deslocações do íman ao longo da posição de equilíbrio, não sendo possível estabilizá-lo por completo nesta posição. Não foi possível também fazer uma análise em regime transitório para este sistema, pois aplicar um degrau à semelhança do que foi realizado nos dois controladores anteriores, coloca o sistema na instabilidade.

7.4. CONTROLADOR DIFUSO

O último controlador implementado foi o controlador difuso. Foi necessário a utilização da biblioteca externa eFLL, já apresentada, para implementar este controlador. Esta biblioteca está disponível para *download* em https://github.com/zerokol/eFLL/, e tem licença de utilização LGPL.

Para implementar o controlador começa-se por definir um objeto do tipo *fuzzy*. Após isso são criadas as funções de pertença de cada variável e os respetivos valores linguísticos. Estes

valores são definidos através do comando `FuzzySet`, que recebe, como já referido no capítulo 3, quatro valores como parâmetros. O primeiro e último valores representam os vértices da base de trapezóide da função de pertença, e os valores do meio os vértices do topo. Quando os valores do meio são idênticos, a função pertença passa a ser triangular, o que é o caso das funções deste controlador.

```
Fuzzy* fuzzy = new Fuzzy();

//Funções Pertença Erro

FuzzySet* NL_E = new FuzzySet(-1.33, -1.00, -1.00, -0.67);
FuzzySet* NM_E = new FuzzySet(-1, -0.67, -0.67, -0.33);
FuzzySet* NS_E = new FuzzySet(-0.67,-0.33, -0.33, 0.00);
FuzzySet* Z_E = new FuzzySet(-0.33, 0.00, 0.00, 0.33);
FuzzySet* PS_E = new FuzzySet(0.00, 0.33, 0.33, 0.67);
FuzzySet* PM_E = new FuzzySet(0.33, 0.67, 0.67, 1.00);
FuzzySet* PL_E = new FuzzySet(0.67, 1.00, 1.00, 1.33);

//Funções Pertença Variação Erro

FuzzySet* NL_V = new FuzzySet(-1.33, -1.00, -1.00, -0.67);
FuzzySet* NM_V = new FuzzySet(-1, -0.67, -0.67, -0.33);
FuzzySet* NS_V = new FuzzySet(-0.67,-0.33, -0.33, 0.00);
FuzzySet* Z_V = new FuzzySet(-0.33, 0.00, 0.00, 0.33);
FuzzySet* PS_V = new FuzzySet(0.00, 0.33, 0.33, 0.67);
FuzzySet* PM_V = new FuzzySet(0.33, 0.67, 0.67, 1.00);
FuzzySet* PL_V = new FuzzySet(0.67, 1.00, 1.00, 1.33);

//Funções Pertença Saida

FuzzySet* NL_S = new FuzzySet(-2.67, -2.00, -2.00, -1.33);
FuzzySet* NM_S = new FuzzySet(-2.00, -1.33, -1.33, -0.67);
FuzzySet* NS_S = new FuzzySet(-1.33,-0.67, -0.67, 0.00);
FuzzySet* Z_S = new FuzzySet(-0.67,0.00, 0.00, 0.67);
FuzzySet* PS_S = new FuzzySet(0.00, 0.67, 0.67, 1.33);
FuzzySet* PM_S = new FuzzySet(0.67, 1.33, 1.33, 2.00);
FuzzySet* PL_S = new FuzzySet(1.33, 2.00, 2.00, 2.66);
```

A definição das regras é feita criando para cada regra um objeto `FuzzyRuleAntecedent` que representa a condição *if* da regra. Neste objeto são definidos duas possibilidades do valor linguístico das entradas, ligadas com o conector *and*. A segunda fase da criação das regras é feita com o objeto `addOutput` que recebe como parâmetro um dos seis valores linguísticos das variáveis de saída. Por fim cria-se a regra com o comando `FuzzyRule` que recebe três parâmetros: o número da regra; a condição *if* criada anteriormente; e a parte *else* constituído por um valor linguístico definido anteriormente. De seguida são apresentados três exemplos da criação de regras.

```
//1-If erro NL_E e verro PL_V then saida Z_S
FuzzyRuleAntecedent* iferroNL_EAndVerroPL_V = new
FuzzyRuleAntecedent();

iferroNL_EAndVerroPL_V->joinWithAND(NL_E, PL_V);

FuzzyRuleConsequent* thensaidaZ_S = new FuzzyRuleConsequent();

thensaidaZ_S->addOutput(Z_S);

FuzzyRule* fuzzyRule1 = new FuzzyRule(1,  iferroNL_EAndVerroPL_V,
thensaidaZ_S);

fuzzy->addFuzzyRule(fuzzyRule1);

//2-If erro NL_E e verro PM_V then saida NS_S
FuzzyRuleAntecedent* iferroNL_EAndVerroPM_V = new
FuzzyRuleAntecedent();

iferroNL_EAndVerroPM_V->joinWithAND(NL_E, PM_V);

FuzzyRuleConsequent* thensaidaNS_S = new FuzzyRuleConsequent();

thensaidaNS_S->addOutput(NS_S);

FuzzyRule* fuzzyRule2 = new FuzzyRule(2, iferroNL_EAndVerroPM_V,
thensaidaNS_S);

fuzzy->addFuzzyRule(fuzzyRule2);
```

O funcionamento do controlador difuso, após as definições mostradas no código anterior, é executado definindo os valores de entrada do erro e da variação do erro, representados pelas variáveis `erro` e `v_erro` no código. De seguida é feito o processo de fuzificação e defuzificação, atribuindo à variável `output1` o valor do controlador difuso.

```
fuzzy->setInput(1,erro);

fuzzy->setInput(2,v_erro);

fuzzy->fuzzify();

output1 = (fuzzy->defuzzify(1));
```

Tal como os outros controladores foi feito uma análise em regime estacionário e regime transitório. Em cada uma das análises o controlador difuso apresenta os melhores resultados, tal como mostra a Figura 70 e Figura 71. Em regime estacionário o íman estabiliza na posição

desejada, existindo um ruído normal do sensor, como presente em todos os controladores anteriores.

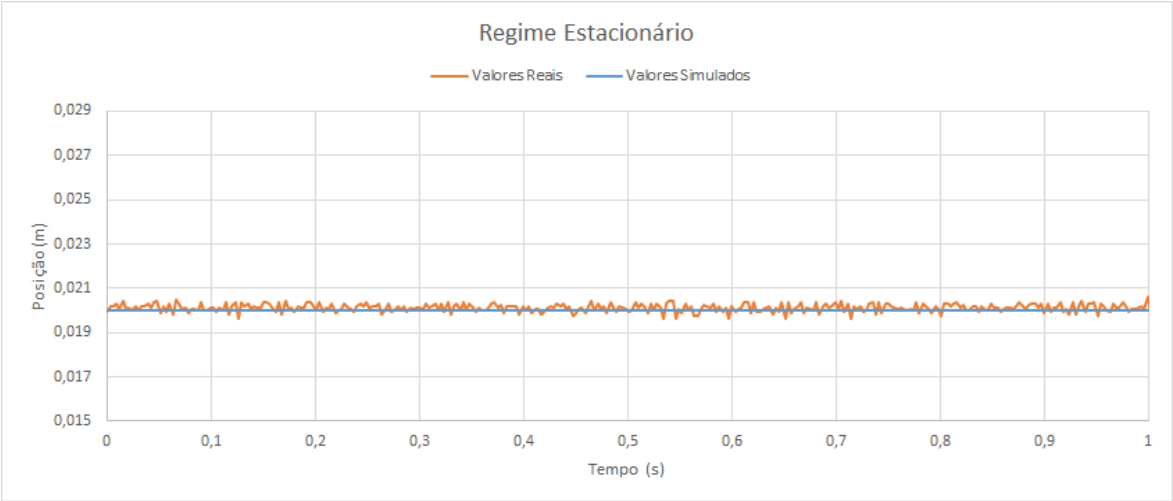

Figura 70 Posição em função do tempo – controlador Difuso

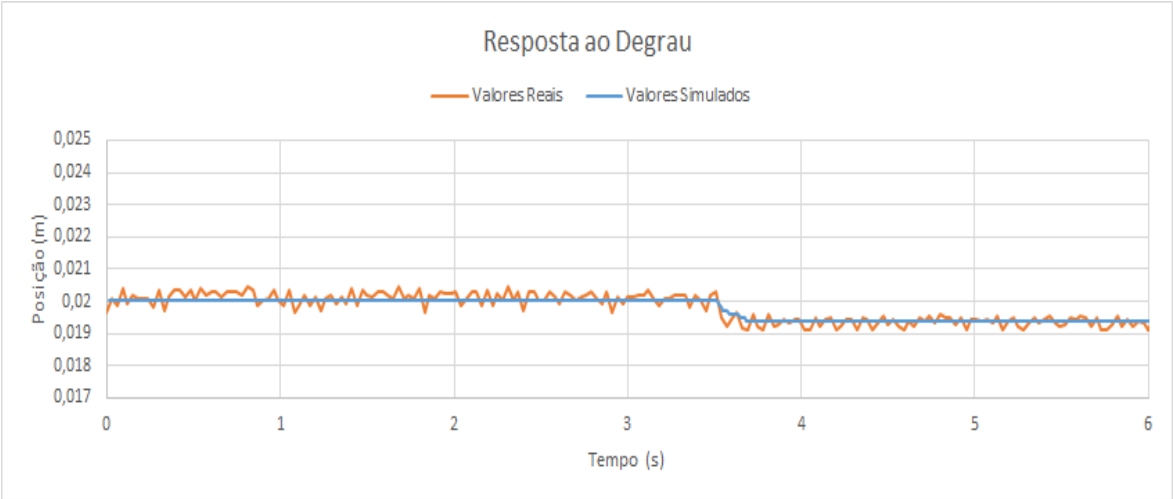

Figura 71 Resposta ao degrau – controlador Difuso

Na análise do regime transitório foi introduzido um degrau de 0,75 mm (Figura 71). Este é o controlador que apresenta o maior intervalo de funcionamento. Uma análise sobre a zona de funcionamento de cada controlador será elaborada na secção seguinte.

7.5. LIMITAÇÕES DOS CONTROLADORES

Uma preocupação que deve existir quando se projeta um controlador tendo em conta a linearização de um modelo, são as limitações de zona de operação. Para este sistema o íman é capaz de levitar de forma estável em diferentes intervalos para cada controlador. Um dos aspetos importantes para perceber o que origina as limitações de operação de cada controlador, é o facto de o campo magnético criado pelo íman permanente detetado pelo sensor é o inverso do quadrado da distância. Em termos práticos isto significa que uma deslocação da zona de posição de equilíbrio para mais perto do íman significa uma maior variação da leitura do sensor que uma deslocação para mais longe. Este facto leva a que o sistema suporte melhor deslocações a baixo da zona de equilíbrio que a cima.

A Tabela 11 compara as diferentes zonas de funcionamento para todos os controladores aplicados.

Tabela 11 Zonas de funcionamento dos controladores

	Distância Mínima (m)	**Distância Equilíbrio (m)**	**Distância Máxima (m)**	**Variação (m)**
Controlador em Avanço de fase	0,0243	0,0248	0,0264	0,0021
Controlador em Avanço-Atraso de fase	0,0200	0,0212	0,0226	0,0026
Controlador PID	0,020	0,020	0,020	0
Controlador Difuso	0,0195	0,020	0,0230	0,0035

Comparando os resultados constata-se que o controlador que apresenta maior amplitude de zona de funcionamento é o controlador difuso, revelando-se também o controlador mais robusto. No controlador PID não é possível fazer qualquer alteração significativa na posição de equilíbrio. Estes resultados foram determinados variando o valor de referência em 0,0001 até o íman sair da posição de equilíbrio. Posteriormente foi utilizada a linearização do sensor no capítulo 5 para determinar a posição.

7.6. COMPARAÇÃO ENTRE CONTROLADORES

Foram utilizados quatro controladores diferentes para este sistema, e validado que está o modelo matemático criado para o mesmo através da aplicação prática de cada um, é possível fazer uma comparação entre todos os controladores. A comparação teórica dos controladores é apresentada na Figura 72. Pode-se ver a similaridade das respostas dos controladores em avanço de fase e em avanço-atraso de fase no primeiro décimo de segundo, com *overshoot* e tempo de subida similares. Na figura não aparece o valor final do controlador em avanço-atraso de fase devido ao seu elevado tempo de estabelecimento. O controlador difuso apresenta os melhores resultados sem *overshoot* e um tempo de estabelecimento bastante rápido. No controlador PID, vê-se por comparação com os controladores em avanço e avanço-atraso, o maior número de oscilações, apesar do *overshoot* menor em relação aos controladores convencionais projetados.

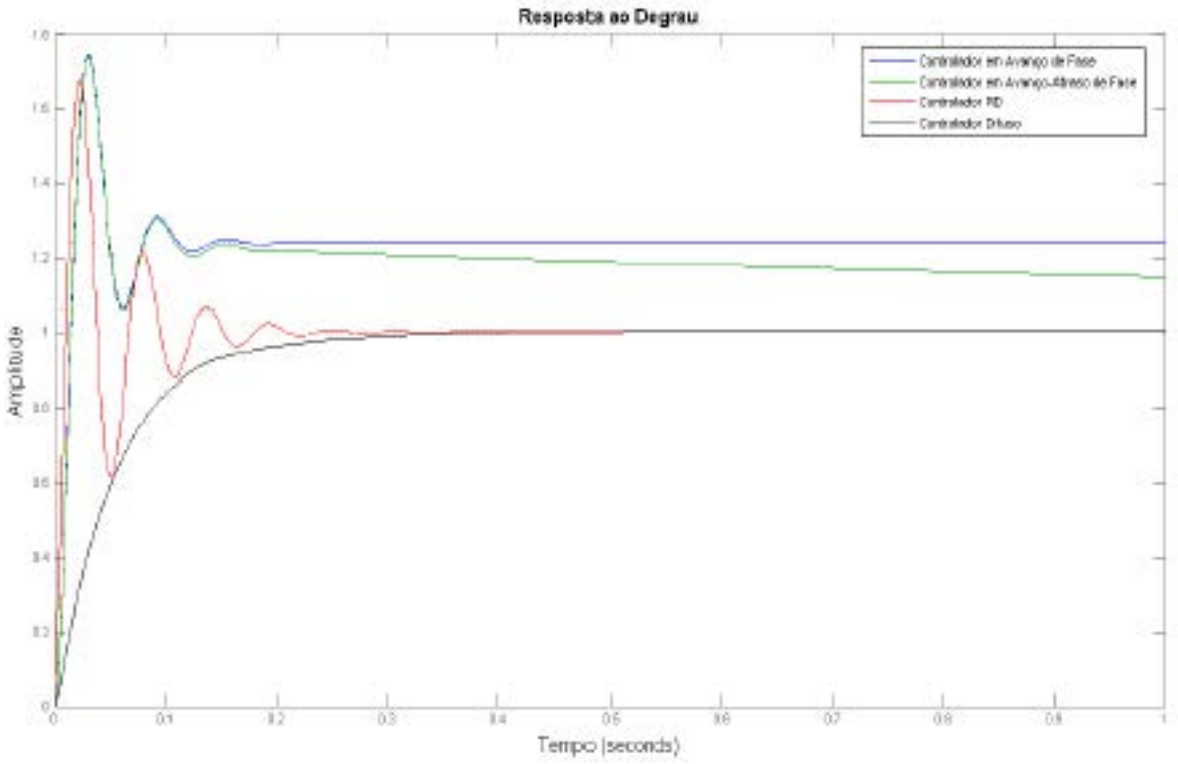

Figura 72 Comparação da resposta ao degrau dos controladores

Quanto à comparação dos controladores na aplicação prática, os controladores em avanço de fase e em avanço-atraso de fase, apresentam as mesmas características que na simulação. O elevado tempo de estabelecimento não prejudica na prática, pois são aplicadas variações

da referência em degrau muito reduzidas. Quanto ao controlador PID a sua aplicação não é a melhor pelas razões que serão apresentadas na conclusão, e assim quando se pretende que este sistema não apresente erro permanente, deve-se optar pelo controlador difuso.

8. CONCLUSÃO

O principal objetivo desta tese foi a construção de um sistema de levitação magnética onde fosse possível um estudo sobre o controlo inteligente baseada em lógica difusa. Como complemento, também foram estudados diferentes métodos de controlo tradicionais, de modo a comparar não só a precisão e estabilidade dos métodos, como os passos necessários para o correto dimensionamento de cada controlador.

No projeto começou-se pela idealização e construção de um sistema portátil e de baixo custo que permitisse cumprir esse objetivo. Esse sistema, composto por um eletroíman, sensor de posição, circuito de potência, e um microcontrolador, mostrou-se adequado para o projeto pela sua simplicidade e possibilidade de transporte. Depois de o sistema construído foram projetados os controladores, que permitiram uma levitação do íman de forma estável. As experiências foram analisadas, e os resultados comparados com os obtidos teoricamente e por simulação.

O primeiro dos controladores utilizados foi o controlador em avanço de fase. Este controlador permitiu deslocar os polos do sistema para um local que o tornasse estável, conferindo as características desejadas em tempo de subida e *overshoot*. Como resultados práticos este controlador apresentou um erro permanente de cerca de 5 mm, e mostrou ser

bastante estável. O projeto deste controlador no domínio discreto mostrou ser o método ideal para de uma forma rápida obter as características desejadas. Para sistemas onde seja aceitável um erro em regime permanente este controlador apresenta-se como uma boa solução.

O segundo controlador projetado foi um controlador em Avanço-Atraso de fase. A componente em avanço de fase foi aproveitada do controlador anterior, pretendendo-se apenas acrescentar uma componente de atraso de fase, que diminui-se o erro sem alterar as restantes características. Ao contrário do primeiro controlador, este foi projetado no domínio contínuo, sendo necessário efetuar conversões entre os dois domínios. Esta conversão fez alterar algumas características definidas, por isso o projeto do controlador diretamente no domínio discreto teria sido melhor opção. Ao nível prático o erro apresentado foi o esperado, no entanto referir que em termos de estabilidade este controlador apresenta piores resultados que o anterior. A conservação das características de *overshoot* e tempo de subida do controlador anterior fizeram com que o tempo de estabelecimento fosse bastante elevado. Para um controlador mais eficiente na resposta ao degrau teria de ser necessário projetar um controlador de raiz, não aproveitando o controlador anterior.

Relativamente ao controlador PID, este foi o que apresentou piores resultados. Nos ensaios em regime estacionário apresenta pequenas oscilações da posição de equilíbrio, o que não acontece com os outros controladores. Também não foi possível uma análise ao regime transitório, pois uma alteração do valor de referência faz com que o sistema entre em instabilidade. As razões para os resultados piores deste controlador prende-se com o facto de o conversor A/D de 10 bits poder não ter resolução suficiente para uma implementação da componente derivativa que depende da variação de duas leituras analógicas consecutivas. Sem uma componente derivativa no controlador não é possível estabilizar convenientemente este sistema. Por fim referir que apenas foi possível utilizar este controlador devido à subtração do cálculo da soma integral de um valor proporcional da diferença entre o valor de saída e o seu valor máximo admissível. Sem esta parcela, utilizando apenas uma limitação simples do valor integral não era possível estabilizar o sistema. Ao contrário do que era de esperar pela sua adaptabilidade o controlador PID não é o indicado para o sistema criado.

Um dos aspetos interessantes nos sistemas de levitação magnética é a possibilidade de utilização de técnicas de controlo não convencional, como os controladores baseados em lógica difusa. Uma grande vantagem deste tipo de controladores é a não necessidade de modelação do sistema. São escolhidas as variáveis de entrada e de saída e definidas regras

linguísticas de como estas variáveis interagem entre si, mais ou menos de forma a modelar a decisão humana. Como o projeto destes controladores tem um fator subjetivo, a experiência do projetista é um dado importante. Como resultados práticos foi possível levitar o objeto na posição pretendida, e também é o controlador que possibilita maior amplitude de deslocação. Como especto negativo para este controlador, existe a necessidade da definição dos ganhos para as variáveis de entrada, e ajuste de ganho de saída do controlador. Estes ganhos foram determinados por tentativa e erro durante a simulação, pois a função de transferência do sistema era conhecida, no entanto, no caso do projeto de um controlador onde não fosse conhecida esta função, os ganhos teriam de ser sintonizados através de testes práticos.

Como melhorias futuras para este trabalho seria desejável melhorar o interface entre os controladores e o PC. Estas melhorias passariam por estabelecer uma comunicação mais eficiente e fiável, e também a possibilidade de definir todos os parâmetros dos controladores clássicos pelo programa, como a localização dos polos e do ganho. Seria igualmente conveniente o programa apresentar mais informação sobre o controlador difuso. Também um microcontrolador com maior resolução de A/D poderia trazer melhorias de desempenho em alguns controladores. Melhorias relativamente ao controlador difuso passariam pelo projeto de mais controladores difusos, onde fossem alterados paramentos como o número de funções pertença, mecanismo de inferência, número de variáveis de entrada, etc. Deste modo seria possível uma análise de como cada um destes parâmetros influencia o controlo do sistema.

Referências Documentais

[1] JAYAWANT, B. V.—Electromagnetic Suspension and Levitation. University of Sussex. UK, 1981.

[2] TIPLER, Paul A.; MOSCA, Gene—Física para cientistas e engenheiros. 6ª ed., 2009.

[3] OGATA, K. — Modern Control Engineering, 5º ed., 2010.

[4] SHAHIAN, B.; HASSUL, M. — Control System Design using Matlab, 1993.

[5] BROWN, R.; THALER, G. — Analysis & Design of Feedback Control Systems, 1966.

[6] KASHIF, I. — Modeling and Control of Magnetic Levitation System via Fuzzy Logic Controller. University of Engineering & Technology Lahore. 2011.

[7] UNNI, A. C.; JUNGHAR, A. — Fuzzy Logic Controller and LQR for Magnetic and Levitation System, International Journal of Recent Technology and Engineering, 2014.

[8] GOLWARHGHI, F.; KUO, B.C. — Automatic Control Systems, 9ª ed., 2010.

[9] NIMRAD, A. R.; DHANGAR, N. Y. — Maglev Monorail, 22nd IRF International Conference, 2015.

[10] SANDS, B. D. — The Transrapid magnetic levitation system, the university of California Transportation Center, 1992.

[11] "SwissRapide Maglev Railway System…The Train that Flies!" — Visitado em 20/06/2015, disponível em: http://www.swissrapide.com/htm/e_home.htm.

[12] MOON, F.C. — Superconducting Levitaiton, John Wiley & Sons, USA, 1994.

[13] MOISES, L.A — Aplicações de Supercondutividade, Brasil, 1992.

[14] BENEVIDES, M. — Introdução, Motivação e Conjuntos Fuzzy, Visitado em 20/06/2015, disponível em: http://www.cos.ufrj.br/~mario/logica/logicaFuzzy.pdf

[15] CAVALVANTI, J.H; MELO, H. — Lógica *Fuzzy* aplicada às engenharias, Brasil, 2012.

[16] HYPIUSOVÁ, M.; OSUSKÝ, J. — PID Controller design for magnetic levitation model, International Conference Cybernetics and informatics, 2010.

[17] GOMES, R. R.; STEPHAN, R. M. — Um experimento para ilustrar o Sistema de levitação electromagnética utilizado em trens MAGLEV. Brasil, 2013.

[18] COSTA, F. D; BARBOSA, L. F — Controlo analógico de um levitador magnético de simples construção. Brasil, 2012.

[19] NISHIRA, A.K.; RAINA, R. — Modeling and Simulation of Levitating Ball by Electromagnetic using Bond Graph. 16th National Conference on Machines and Mechanisms. India 2013.

[20] RECH, E.; CAMPO, A. B. — Modelagem e simulação de um sistema de levitação magnética através de programação em linguagem gráfica. 10º Conferência Brasileira de Dinâmica, Controlo e Aplicações. Brasil, 2011.

[21] YAGHOUBI, H.; BARAZI, N.; AOLIAEI, M.R — Infrastructure Design, Signalling and Security in Railway, Maglev, ch 6. Croacia, 2012.

[22] YAGHOUBI, H. — Magnetically Levitated Trains, Maglev, vol. 1. Iran, 2008.

[23] WU, H. ; WANG, Z. — Design and simulation of axial flow maglev blood pump, International Journal of Information Engineering and Electronic Business. 2011.

[24] MIRICA, K.A; PHILLIPS, C.R — Magnetic levitation in the analysis of foods and water. Journal of Agricultural and Food Chemistry. 2010.

[25] HULL, J. R. ; FISKE, K. — Analysis of levitational systems for a superconducting launch ring. Proceedings of the Applied Superconductivity Conference. 2006.

[26] Datasheet Atmega 2560 — Atmel ATmega640/V–1280/V -1281/V-2560/V-2561V, disponível em http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf.

[27] BEALE, G. — Compensator Design to Improve Steady-State Error Using Root Locus. Electrical and Computer Engineering Department George Mason University. Disponível em http:// http://ece.gmu.edu/~gbeale/ece_421/comp_root_ess.pdf.

[28] Fuzzy Logic Tollbox, User´s Guide, Matlab R2015b — Disponível em http://www.mathworks.com/help/pdf_doc/fuzzy/fuzzy.pdf.

[29] BASANTANI, M. — The Super-powered Magnetic Wind Turbine — Disponível em http://inhabitat.com/super-powered-magnetic-wind-turbine-maglev.

[30] Zeltom Educational and Industrial Control Systems — Disponível em http://www.zeltom.com/product/magneticlevitation/emlscomponents.

[31] Datasheet Sensor Efeito de hall A1324 Allegro — Disponível em http://www.allegromicro.com/en/Products/Magnetic-Linear-And-Angular-Position-Sensor-ICs/Linear-Position-Sensor-ICs/A1324-5-6.aspx.

[32] eFLL - Uma Biblioteca Fuzzy para Arduino e Sistemas Embarcados — Disponível em http://softwarelivre.org/psl-pi/blog/aj-alves-efll-%E2%80%93-uma-biblioteca-fuzzy-para-arduino-e-sistemas-embarcados.

Anexo A. Código Arduíno

```
#include <Fuzzy.h>
#include <FuzzyComposition.h>
#include <FuzzyInput.h>
#include <FuzzyIO.h>
#include <FuzzyOutput.h>
#include <FuzzyRule.h>
#include <FuzzyRuleAntecedent.h>
#include <FuzzyRuleConsequent.h>
#include <FuzzySet.h>
#include "TimerOne.h"
#include <TimerThree.h>


//Referencia
float s_point=3.000;
float err=0.0000;
float err_p=0.000;
float gPWMCommand=0;
float gPWMCommand_1=0;
float gPWMCommand_p=0;
float x=0.000;
float y=0.000;
float saida_sensor;
float v_erro = 0.00;
float output1 = 0.00;
int aux_a;
int aux_b;
float erro = 0.00;


unsigned long time;
unsigned long start, finished, elapsed;


int numReadings = 10;
float readings[100];
float total;
int i;
float aux_sin=0;
int aux_serial=0;


float Kp=10.632;
float Ki=1.000;
float Kd=20.020;
```

```
float int_aux =3.00;
char programa;  //0-Avanço;1-Atraso-Avanço;2-PID;3-Fuzzy;4-Nenhum
float aux_analog;
Fuzzy* fuzzy = new Fuzzy();
char input;
float s;

// Define various ADC prescaler
const unsigned char PS_4 = (1 << ADPS1);
const unsigned char PS_8 = (1 << ADPS1) | (1 << ADPS0);
const unsigned char PS_16 = (1 << ADPS2);
const unsigned char PS_32 = (1 << ADPS2) | (1 << ADPS0);
const unsigned char PS_64 = (1 << ADPS2) | (1 << ADPS1);
const unsigned char PS_128 = (1 << ADPS2) | (1 << ADPS1) | (1 <<
ADPS0);

void Init_Fuzzy()
{
//************************************************************//
//**********************____FUZZY___**********************//

//Funções Pertença Erro
FuzzySet* NL_E = new FuzzySet(-1.33, -1.00, -1.00, -0.67);
FuzzySet* NM_E = new FuzzySet(-1, -0.67, -0.67, -0.33);
FuzzySet* NS_E = new FuzzySet(-0.67,-0.33, -0.33, 0.00);
FuzzySet* Z_E = new FuzzySet(-0.33, 0.00, 0.00, 0.33);


FuzzySet* PS_E = new FuzzySet(0.00, 0.33, 0.33, 0.67);
FuzzySet* PM_E = new FuzzySet(0.33, 0.67, 0.67, 1.00);
FuzzySet* PL_E = new FuzzySet(0.67, 1.00, 1.00, 1.33);


//Funções Pertença Variação Erro
FuzzySet* NL_V = new FuzzySet(-1.33, -1.00, -1.00, -0.67);
FuzzySet* NM_V = new FuzzySet(-1, -0.67, -0.67, -0.33);
FuzzySet* NS_V = new FuzzySet(-0.67,-0.33, -0.33, 0.00);
FuzzySet* Z_V = new FuzzySet(-0.33, 0.00, 0.00, 0.33);


FuzzySet* PS_V = new FuzzySet(0.00, 0.33, 0.33, 0.67);
FuzzySet* PM_V = new FuzzySet(0.33, 0.67, 0.67, 1.00);
FuzzySet* PL_V = new FuzzySet(0.67, 1.00, 1.00, 1.33);


//Funções Pertença Saida
FuzzySet* NL_S = new FuzzySet(-2.67, -2.00, -2.00, -1.33);
FuzzySet* NM_S = new FuzzySet(-2.00, -1.33, -1.33, -0.67);
FuzzySet* NS_S = new FuzzySet(-1.33,-0.67, -0.67, 0.00);
FuzzySet* Z_S = new FuzzySet(-0.67,0.00, 0.00, 0.67);
FuzzySet* PS_S = new FuzzySet(0.00, 0.67, 0.67, 1.33);
FuzzySet* PM_S = new FuzzySet(0.67, 1.33, 1.33, 2.00);
FuzzySet* PL_S = new FuzzySet(1.33, 2.00, 2.00, 2.66);
```

```
// FuzzyInput Erro
  FuzzyInput* er = new FuzzyInput(1);
  er->addFuzzySet(NL_E);
  er->addFuzzySet(NM_E);
  er->addFuzzySet(NS_E);
  er->addFuzzySet(Z_E);
  er->addFuzzySet(PS_E);
  er->addFuzzySet(PM_E);
  er->addFuzzySet(PL_E);

  fuzzy->addFuzzyInput(er);

  // FuzzyInput Variação Erro
  FuzzyInput* verro = new FuzzyInput(2);
  verro->addFuzzySet(NL_V);
  verro->addFuzzySet(NM_V);
  verro->addFuzzySet(NS_V);
  verro->addFuzzySet(Z_V);
  verro->addFuzzySet(PS_V);
  verro->addFuzzySet(PM_V);
  verro->addFuzzySet(PL_V);


  fuzzy->addFuzzyInput(verro);


  // FuzzyOutput Saida
  FuzzyOutput* saida = new FuzzyOutput(1);
  saida->addFuzzySet(NL_S);
  saida->addFuzzySet(NM_S);
  saida->addFuzzySet(NS_S);
  saida->addFuzzySet(Z_S);
  saida->addFuzzySet(PS_S);
  saida->addFuzzySet(PM_S);
  saida->addFuzzySet(PL_S);

  fuzzy->addFuzzyOutput(saida);


  //Construção das Regras 49 REGRAS
  //1-If erro NL_E e verro PL_V then saida Z_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroPL_V = new


FuzzyRuleAntecedent();

  iferroNL_EAndVerroPL_V->joinWithAND(NL_E, PL_V);


  FuzzyRuleConsequent* thensaidaZ_S = new FuzzyRuleConsequent();
  thensaidaZ_S->addOutput(Z_S);


  FuzzyRule* fuzzyRule1 = new FuzzyRule(1,
iferroNL_EAndVerroPL_V, thensaidaZ_S);

  fuzzy->addFuzzyRule(fuzzyRule1);
```

```
//2-If erro NL_E e verro PM_V then saida NS_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroPM_V->joinWithAND(NL_E, PM_V);

  FuzzyRuleConsequent* thensaidaNS_S = new
FuzzyRuleConsequent();
  thensaidaNS_S->addOutput(NS_S);

  FuzzyRule* fuzzyRule2 = new FuzzyRule(2,
iferroNL_EAndVerroPM_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule2);

  //3-If erro NL_E e verro PS_V then saida NM_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroPS_V->joinWithAND(NL_E, PS_V);

  FuzzyRuleConsequent* thensaidaNM_S = new
FuzzyRuleConsequent();
  thensaidaNM_S->addOutput(NM_S);

  FuzzyRule* fuzzyRule3 = new FuzzyRule(3,
iferroNL_EAndVerroPS_V, thensaidaNM_S);
  fuzzy->addFuzzyRule(fuzzyRule3);

  //4-If erro NL_E e verro Z_V then saida NL_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroZ_V->joinWithAND(NL_E, Z_V);

  FuzzyRuleConsequent* thensaidaNL_S = new
FuzzyRuleConsequent();
  thensaidaNL_S->addOutput(NL_S);

  FuzzyRule* fuzzyRule4 = new FuzzyRule(4,
iferroNL_EAndVerroZ_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule4);

//5-If erro NL_E e verro NS_V then saida NL_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroNS_V->joinWithAND(NL_E, NS_V);

  FuzzyRule* fuzzyRule5 = new FuzzyRule(5,
iferroNL_EAndVerroNS_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule5);

  //6-If erro NL_E e verro NM_V then saida NL_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroNM_V->joinWithAND(NL_E, NM_V);

  FuzzyRule* fuzzyRule6 = new FuzzyRule(6,
iferroNL_EAndVerroNM_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule6);
```

```
  //7-If erro NL_E e verro NL_V then saida NL_S
  FuzzyRuleAntecedent* iferroNL_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroNL_EAndVerroNL_V->joinWithAND(NL_E, NL_V);

  FuzzyRule* fuzzyRule7 = new FuzzyRule(7,
iferroNL_EAndVerroNL_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule7);



//************************************************************//



  //8-If erro NM_E e verro PL_V then saida Z_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroPL_V->joinWithAND(NM_E, PL_V);

  FuzzyRuleConsequent* thensaidaPS_S = new
FuzzyRuleConsequent();
  thensaidaPS_S->addOutput(PS_S);

  FuzzyRule* fuzzyRule8 = new FuzzyRule(8,
iferroNM_EAndVerroPL_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule8);

  //9-If erro NM_E e verro PM_V then saida NS_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroPM_V->joinWithAND(NM_E, PM_V);

  FuzzyRule* fuzzyRule9 = new FuzzyRule(9,
iferroNM_EAndVerroPM_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule9);

  //10-If erro NM_E e verro PS_V then saida NM_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroPS_V->joinWithAND(NM_E, PS_V);

  FuzzyRule* fuzzyRule10 = new FuzzyRule(10,
iferroNM_EAndVerroPS_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule10);

  //11-If erro NM_E e verro Z_V then saida NM_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroZ_V->joinWithAND(NM_E, Z_V);

  FuzzyRule* fuzzyRule11 = new FuzzyRule(11,
iferroNM_EAndVerroZ_V, thensaidaNM_S);
  fuzzy->addFuzzyRule(fuzzyRule11);
```

```
//13-If erro NM_E e verro NM_V then saida NL_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroNM_V->joinWithAND(NM_E, NM_V);

  FuzzyRule* fuzzyRule13 = new FuzzyRule(13,
iferroNM_EAndVerroNM_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule13);

  //14-If erro NM_E e verro NL_V then saida NL_S
  FuzzyRuleAntecedent* iferroNM_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroNM_EAndVerroNL_V->joinWithAND(NM_E, NL_V);

  FuzzyRule* fuzzyRule14 = new FuzzyRule(14,
iferroNM_EAndVerroNL_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule14);



//*********************************************************//



  //15-If erro NS_E e verro PL_V then saida Z_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroPL_V->joinWithAND(NS_E, PL_V);

  FuzzyRuleConsequent* thensaidaPM_S = new
FuzzyRuleConsequent();
  thensaidaPM_S->addOutput(PM_S);

  FuzzyRule* fuzzyRule15 = new FuzzyRule(15,
iferroNS_EAndVerroPL_V, thensaidaPM_S);
  fuzzy->addFuzzyRule(fuzzyRule15);

  //16-If erro NS_E e verro PM_V then saida NS_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroPM_V->joinWithAND(NS_E, PM_V);

  FuzzyRule* fuzzyRule16 = new FuzzyRule(16,
iferroNS_EAndVerroPM_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule16);

  //17-If erro NS_E e verro PS_V then saida Z_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroPS_V->joinWithAND(NS_E, PS_V);

  FuzzyRule* fuzzyRule17 = new FuzzyRule(17,
iferroNS_EAndVerroPS_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule17);
```

```
//18-If erro NS_E e verro Z_V then saida NS_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroZ_V->joinWithAND(NS_E, Z_V);

  FuzzyRule* fuzzyRule18 = new FuzzyRule(18,
iferroNS_EAndVerroZ_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule18);

  //19-If erro NS_E e verro NS_V then saida NM_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroNS_V->joinWithAND(NS_E, NS_V);

  FuzzyRule* fuzzyRule19 = new FuzzyRule(19,
iferroNS_EAndVerroNS_V, thensaidaNM_S);
  fuzzy->addFuzzyRule(fuzzyRule19);

  //20-If erro NS_E e verro NM_V then saida NL_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroNM_V->joinWithAND(NS_E, NM_V);

  FuzzyRule* fuzzyRule20 = new FuzzyRule(20,
iferroNS_EAndVerroNM_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule20);

  //21-If erro NS_E e verro NL_V then saida NL_S
  FuzzyRuleAntecedent* iferroNS_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroNS_EAndVerroNL_V->joinWithAND(NS_E, NL_V);

  FuzzyRule* fuzzyRule21 = new FuzzyRule(21,
iferroNS_EAndVerroNL_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule21);


//*********************************************************//



//22-If erro Z_E e verro PL_V then saida PL_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroPL_V->joinWithAND(Z_E, PL_V);

  FuzzyRuleConsequent* thensaidaPL_S = new
FuzzyRuleConsequent();
  thensaidaPL_S->addOutput(PL_S);

  FuzzyRule* fuzzyRule22 = new FuzzyRule(22,
iferroZ_EAndVerroPL_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule22);
```

```
//23-If erro Z_E e verro PM_V then saida PM_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroPM_V->joinWithAND(Z_E, PM_V);

  FuzzyRule* fuzzyRule23 = new FuzzyRule(23,
iferroZ_EAndVerroPM_V, thensaidaPM_S);
  fuzzy->addFuzzyRule(fuzzyRule23);

  //24-If erro Z_E e verro PS_V then saida PS_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroPS_V->joinWithAND(Z_E, PS_V);

  FuzzyRule* fuzzyRule24 = new FuzzyRule(24,
iferroZ_EAndVerroPS_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule24);

  //25-If erro Z_E e verro Z_V then saida Z_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroZ_V->joinWithAND(Z_E, Z_V);

  FuzzyRule* fuzzyRule25 = new FuzzyRule(25,
iferroZ_EAndVerroZ_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule25);

  //26-If erro Z_E e verro NS_V then saida NS_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroNS_V->joinWithAND(Z_E, NS_V);

  FuzzyRule* fuzzyRule26 = new FuzzyRule(26,
iferroZ_EAndVerroNS_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule26);

  //27-If erro Z_E e verro NM_V then saida NM_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroNM_V->joinWithAND(Z_E, NM_V);

  FuzzyRule* fuzzyRule27 = new FuzzyRule(27,
iferroZ_EAndVerroNM_V, thensaidaNM_S);
  fuzzy->addFuzzyRule(fuzzyRule27);

//28-If erro Z_E e verro NL_V then saida NL_S
  FuzzyRuleAntecedent* iferroZ_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroZ_EAndVerroNL_V->joinWithAND(Z_E, NL_V);

  FuzzyRule* fuzzyRule28 = new FuzzyRule(28,
iferroZ_EAndVerroNL_V, thensaidaNL_S);
  fuzzy->addFuzzyRule(fuzzyRule28);


//*********************************************************//
```

```
  //29-If erro PS_E e verro PL_V then saida PL_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroPL_V->joinWithAND(PS_E, PL_V);

  FuzzyRule* fuzzyRule29 = new FuzzyRule(29,
iferroPS_EAndVerroPL_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule29);

  //30-If erro PS_E e verro PM_V then saida PL_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroPM_V->joinWithAND(PS_E, PM_V);

  FuzzyRule* fuzzyRule30 = new FuzzyRule(30,
iferroPS_EAndVerroPM_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule30);

  //31-If erro PS_E e verro PS_V then saida NM_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroPS_V->joinWithAND(PS_E, PS_V);

  FuzzyRule* fuzzyRule31 = new FuzzyRule(31,
iferroPS_EAndVerroPS_V, thensaidaPM_S);
  fuzzy->addFuzzyRule(fuzzyRule31);

  //32-If erro PS_E e verro Z_V then saida PS_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroZ_V->joinWithAND(PS_E, Z_V);

  FuzzyRule* fuzzyRule32 = new FuzzyRule(32,
iferroPS_EAndVerroZ_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule32);

  //33-If erro PS_E e verro NS_V then saida Z_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroNS_V->joinWithAND(PS_E, NS_V);

  FuzzyRule* fuzzyRule33 = new FuzzyRule(33,
iferroPS_EAndVerroNS_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule33);


  //34-If erro PS_E e verro NM_V then saida NS_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroNM_V->joinWithAND(PS_E, NM_V);

  FuzzyRule* fuzzyRule34 = new FuzzyRule(34,
iferroPS_EAndVerroNM_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule34);
```

```
//35-If erro PS_E e verro NL_V then saida NM_S
  FuzzyRuleAntecedent* iferroPS_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroPS_EAndVerroNL_V->joinWithAND(PS_E, NL_V);

  FuzzyRule* fuzzyRule35 = new FuzzyRule(35,
iferroPS_EAndVerroNL_V, thensaidaNM_S);
  fuzzy->addFuzzyRule(fuzzyRule35);


//*********************************************************//

  //36-If erro PM_E e verro PL_V then saida PL_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroPL_V->joinWithAND(PM_E, PL_V);

  FuzzyRule* fuzzyRule36 = new FuzzyRule(36,
iferroPM_EAndVerroPL_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule36);

  //37-If erro PM_E e verro PM_V then saida PL_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroPM_V->joinWithAND(PM_E, PM_V);

  FuzzyRule* fuzzyRule37 = new FuzzyRule(37,
iferroPM_EAndVerroPM_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule37);

  //38-If erro PM_E e verro PS_V then saida PL_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroPS_V->joinWithAND(PM_E, PS_V);

  FuzzyRule* fuzzyRule38 = new FuzzyRule(38,
iferroPM_EAndVerroPS_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule38);

  //39-If erro PM_E e verro Z_V then saida PM_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroZ_V->joinWithAND(PM_E, Z_V);

  FuzzyRule* fuzzyRule39 = new FuzzyRule(39,
iferroPM_EAndVerroZ_V, thensaidaPM_S);
  fuzzy->addFuzzyRule(fuzzyRule39);

  //40-If erro PM_E e verro NS_V then saida PS_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroNS_V->joinWithAND(PM_E, NS_V);

  FuzzyRule* fuzzyRule40 = new FuzzyRule(40,
iferroPM_EAndVerroNS_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule40);
```

```
//41-If erro PM_E e verro NM_V then saida Z_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroNM_V->joinWithAND(PM_E, NM_V);

  FuzzyRule* fuzzyRule41 = new FuzzyRule(41,
iferroPM_EAndVerroNM_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule41);

  //42-If erro PM_E e verro NL_V then saida NS_S
  FuzzyRuleAntecedent* iferroPM_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroPM_EAndVerroNL_V->joinWithAND(PM_E, NL_V);

  FuzzyRule* fuzzyRule42 = new FuzzyRule(42,
iferroPM_EAndVerroNL_V, thensaidaNS_S);
  fuzzy->addFuzzyRule(fuzzyRule42);


//*********************************************************//

  //43-If erro PL_E e verro PL_V then saida PL_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroPL_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroPL_V->joinWithAND(PL_E, PL_V);

  FuzzyRule* fuzzyRule43 = new FuzzyRule(43,
iferroPL_EAndVerroPL_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule43);

  //44-If erro PL_E e verro PM_V then saida PL_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroPM_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroPM_V->joinWithAND(PL_E, PM_V);

  FuzzyRule* fuzzyRule44 = new FuzzyRule(44,
iferroPL_EAndVerroPM_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule44);

  //45-If erro PL_E e verro PS_V then saida PL_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroPS_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroPS_V->joinWithAND(PL_E, PS_V);

  FuzzyRule* fuzzyRule45 = new FuzzyRule(45,
iferroPL_EAndVerroPS_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule45);

  //46-If erro PL_E e verro Z_V then saida PL_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroZ_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroZ_V->joinWithAND(PL_E, Z_V);

  FuzzyRule* fuzzyRule46 = new FuzzyRule(46,
iferroPL_EAndVerroZ_V, thensaidaPL_S);
  fuzzy->addFuzzyRule(fuzzyRule46);
```

```
//47-If erro PL_E e verro NS_V then saida PM_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroNS_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroNS_V->joinWithAND(PL_E, NS_V);

  FuzzyRule* fuzzyRule47 = new FuzzyRule(47,
iferroPL_EAndVerroNS_V, thensaidaPM_S);
  fuzzy->addFuzzyRule(fuzzyRule47);

  //48-If erro PL_E e verro NM_V then saida PS_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroNM_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroNM_V->joinWithAND(PL_E, NM_V);

  FuzzyRule* fuzzyRule48 = new FuzzyRule(48,
iferroPL_EAndVerroNM_V, thensaidaPS_S);
  fuzzy->addFuzzyRule(fuzzyRule48);

  //49-If erro PL_E e verro NL_V then saida Z_S
  FuzzyRuleAntecedent* iferroPL_EAndVerroNL_V = new
FuzzyRuleAntecedent();
  iferroPL_EAndVerroNL_V->joinWithAND(PL_E, NL_V);

  FuzzyRule* fuzzyRule49 = new FuzzyRule(49,
iferroPL_EAndVerroNL_V, thensaidaZ_S);
  fuzzy->addFuzzyRule(fuzzyRule49);

//*********************************************************//
//*********************************************************//

}
```

```
void processCommand( int cmd ) {

   switch(cmd)
   {
      case 'a':
          Timer1.initialize(4000000);
          Timer1.attachInterrupt(callback_Quadrado);
         break;
      case 's':
         s_point=3.000;
         Timer1.detachInterrupt();
         break;
      case 'd':
          Timer1.initialize(200000);
          Timer1.attachInterrupt(callback_Sin);
         break;
      case '+':
          s_point=s_point+0.01;
         break;
      case '-':
          s_point=s_point-0.01;
         break;
      case 'q':
          programa=0; //-Avanço
          break;
      case 'w':
          programa=1; //-Avanço/Atraso
         break;
      case 'e':
          programa=2; //-PID
         break;
      case 'r':
          programa=3; //-Fuzzy
         break;
      case 'm':
          programa=4; //-Nenhum
         break;
      default:
         break;
         }
}

//Onda Quadrada
void callback_Quadrado()
{
    if (s_point==3.00){
        s_point=2.985;
    }

   else if (s_point==2.985){
        s_point=3.01;
    }
```

```
  else if (s_point==3.01){
        s_point=2.985;
    }
}

//Onda Sinusoidal
void callback_Sin()
{
   aux_sin=aux_sin+0.20;
   s_point=3.00+0.02*sin(6.28*aux_sin);
}

void setupPWM()
{
   pinMode(10, OUTPUT);

   int myEraser = 7;
   int myPrescaler = 2;

   TCCR2B &= ~myEraser;
   TCCR2B |= myPrescaler;
}

//Ler valores analogicos
float ler_analog(){

  total=0;

  for (i=0; i<numReadings; i++){
    total=total + float(analogRead(A0));
  }

  return total/aux_analog;
}

void setup() {


  ADCSRA &= ~PS_128;
  ADCSRA |= PS_16;
 Serial.begin(115200);

  setupPWM();

  //inicia com controlador em avanço
  programa=0;
  aux_analog=2046.000;

  attachInterrupt(1, blink, RISING);
  Init_Fuzzy();

  Timer3.initialize(500000);
  Timer3.attachInterrupt(enviar);

}
```

```
void enviar(){

  Serial.println(saida_sensor,3);
}

void blink()
{
 static unsigned long last_interrupt_time = 0;
 unsigned long interrupt_time = millis();

 if (interrupt_time - last_interrupt_time > 200)
 {
   programa=3;
 }
   last_interrupt_time = interrupt_time;

}

void loop() {

    switch(programa){

      case 0:
         processCommand(Serial.read());

         saida_sensor=ler_analog();
         err=s_point-saida_sensor;
         gPWMCommand=21.0*err+x;
         x=-19*err + 0.7408*gPWMCommand;
         gPWMCommand_1=gPWMCommand*139.75 +35.355;

         if (gPWMCommand_1>128  )
           {
            gPWMCommand_1=128;}

         else if (gPWMCommand_1<0)
           {
            gPWMCommand_1=0;}

         analogWrite(10, gPWMCommand_1);

         delay(0.488);

        break;

      case 1:
          processCommand(Serial.read());

          saida_sensor=ler_analog();
          err_p=err;
          err=s_point-saida_sensor;
          gPWMCommand_p=gPWMCommand;
          gPWMCommand=31.50*err+x+y;
          x=-60.000*err + 1.7408*gPWMCommand;
          y=28.500*err p - 0.7408*gPWMCommand p;
```

```
        gPWMCommand_1=gPWMCommand*139.75 +35.355;

        if (gPWMCommand_1>128  )
           {
         gPWMCommand_1=128;}

        else if (gPWMCommand_1<0)
           {
        gPWMCommand_1=0;}

        analogWrite(10, gPWMCommand_1);
        delay(8.488);
        break;

    case 2:
        processCommand(Serial.read());

        saida_sensor=ler_analog();
        err=(s_point-saida_sensor);
        s=s+err-0.900*(gPWMCommand-0.4500);

        if (s>0.020)
           {
           s=0.020;}

        else if (s<-0.020)
           {
           s=-0.020;}

      gPWMCommand=((Kp*err) + Ki*s + Kd*(err-err_p));
      gPWMCommand_1=(gPWMCommand*139.75 +35.355);

      if (gPWMCommand_1>255)
           {
         gPWMCommand_1=255;}

       else if (gPWMCommand_1<0)
           {
        gPWMCommand_1=0;}

       analogWrite(10, gPWMCommand_1);
       err_p=err;
       delay(0.888);
       break;

     case 3:
        processCommand(Serial.read());

        saida_sensor=ler_analog();
        erro=(s_point-saida_sensor);
        v_erro=0.05*(err_p-erro);
        err_p=erro;

        fuzzy->setInput(1,erro);
        fuzzy->setInput(2,v erro);
```

```
          fuzzy->fuzzify();

          output1 = (fuzzy->defuzzify(1));
          gPWMCommand_1=((output1)*139.75 +35.355);

           if (gPWMCommand_1>128  )
             {
               gPWMCommand_1=128;}

           else if (gPWMCommand_1<0)
             {
               gPWMCommand_1=0;}

          analogWrite(10, gPWMCommand_1);
          delay(0.888);
          break;

       case 4:
           gPWMCommand=0;
          break;
       }
}
```

Anexo B. Manual de Utilização do Programa

A utilização do programa é bastante simples, no entanto devem ser seguidos alguns procedimentos tanto para o correto funcionamento do sistema físico, como para evitar erros da parte do programa.

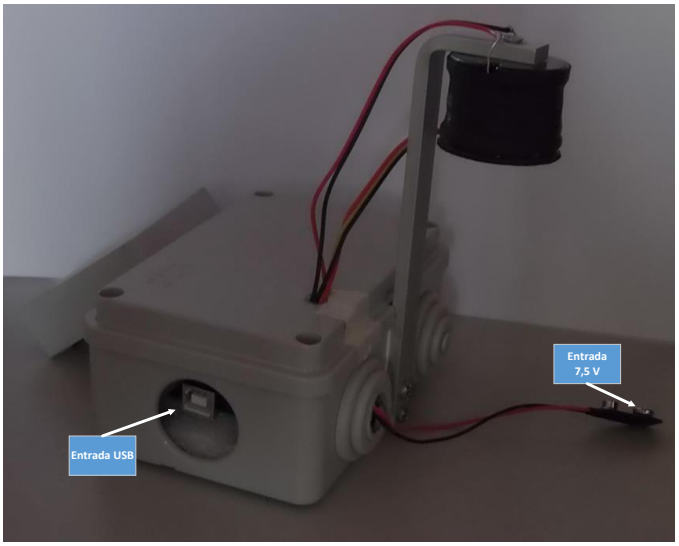

Figura 1 Sistema de Levitação magnética

1. Alimentar o *shield* do Arduíno a 7,5 V através da entrada representada na Figura 1.
2. Ligar a entrada USB do Arduíno ao PC. Verificar se a ligação é feita através do porto COM4. Caso não seja alterar para este porto.
3. Iniciar o programa.
4. Devido ao sistema físico ser pequeno, caso se pretenda ligar uma Webcam para uma melhor apresentação visual, selecionar esta na *combobox* existente no canto superior direito e pressionar o botão “Ligar”.

5. Pressionar o botão “Começar Transmissão” para iniciar a comunicação entre o PC e o Arduíno. A partir deste instante os gráficos começam a ser atualizados com os valores de posição e erro.

6. Neste momento não existe nenhum controlador a atuar no sistema. Selecionar o controlador pretendido das quatro opções disponíveis, como mostra a Figura 2, e colocar o íman permanente aproximadamente na posição de equilíbrio.

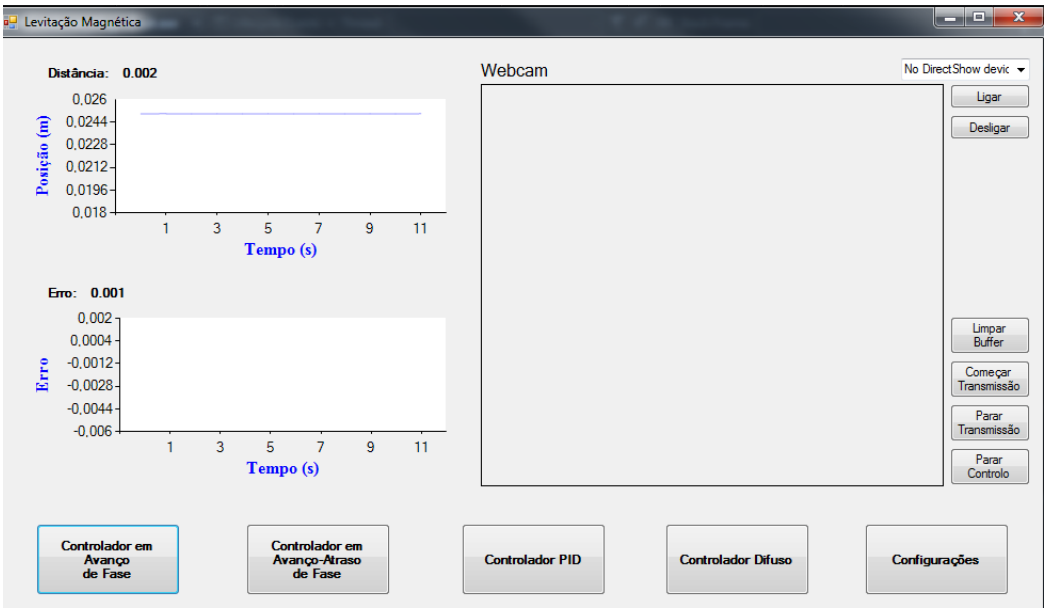

Figura 2 Janela principal do programa

7. Alterar a seleção de controlo, constatando o erro em regime permanente, estabilidade, e resposta a perturbações dos diferentes controladores. O botão referente ao controlador que atua no sistema é apresentado com uma cor diferente dos restantes.
8. Caso se constate que os gráficos possuem um atraso significativo pressionar o botão “Limpar Buffer”.

9. Pressionar o botão “Configurações”. Na janela de configurações alterar o valor de “Setpoint” através dos botões “Sepoint Incrementar” e “Setpoint Decrementar”, apresentados na Figura 3. Cada vez que pressionar algum destes botões o íman desloca-se 0,05 mm, para cima (ao pressionar “Setpoint Decrementar”) ou para baixo

(ao pressionar “Setpoint Incrementar”). Verificar as zonas de funcionamento dos diferentes controladores.

Figura 3 Janela de configurações

10. Para selecionar como valor de referência uma onda quadrada pressionar o botão “Onda Quadrada”. Para o íman ter tempo de estabilizar em cada posição a onda quadrada tem um período de 8 segundos.

11. Para selecionar como valor de referência uma onda sinusoidal pressionar o botão “Onda Sinusoidal”. Tanto esta opção de sinal de referência como a anterior foram projetadas para serem usadas tendo em conta os limites de funcionamento do controlador difuso. Utilizar estas opções num outro controlador leva a que o sistema entre em instabilidade.

12. Para terminar o ensaio retirar o íman, pressionar os botões “Parar controlo” para retirar qualquer controlo ao sistema.

13. Pressionar o botão “Parar comunicação” para terminar a comunicação entre o Arduíno e o sistema de levitação magnética.

Anexo C. Programa de Levitação Magnética e Código Fonte

O código fonte e bibliotecas usadas, assim como um ficheiro executável do programa está disponibilizado em CD anexo a este relatório.