INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DO PIAUÍ-IFPI

COORDENAÇÃO DO CURSO DE LICENCIATURA EM FÍSICA
CURSO DE LICENCIATURA EM FÍSICA

HERYSON DE SOUSA CARVALHO

OSCILADOR HARMÔNICO AMORTECIDO: UM ESTUDO UTILIZANDO A PLATAFORMA ARDUINO

Parnaíba-PI
Julho de 2018

HERYSON DE SOUSA CARVALHO

OSCILADOR HARMÔNICO AMORTECIDO: UM ESTUDO UTILIZANDO A PLATAFORMA ARDUINO

Trabalho de Conclusão de Curso apresentado como exigência parcial para obtenção do diploma do Curso de Licenciatura em Física do Instituto Federal de Educação, Ciência e Tecnologia do Piauí/Campus Parnaíba.

Parnaíba-PI
2018

HERYSON DE SOUSA CARVALHO

OSCILADOR HARMÔNICO AMORTECIDO: UM ESTUDO UTILIZANDO A PLATAFORMA ARDUINO

Trabalho de Conclusão de Curso apresentado como exigência parcial para obtenção do diploma do Curso de Licenciatura em Física do Instituto Federal de Educação, Ciência e Tecnologia do Piauí/*Campus* Parnaíba.

Aprovado em _____/_____/_____.

BANCA EXAMINADORA

Prof.MSc. Itamar Vieira de Sousa Júnior- (Orientador) - Presidente
IFPI/ *Campus* Parnaíba-PI

Prof.MSc. Lucas Izidio de Sousa Sampaio - Examinador
IFPI/*Campus* Parnaíba-PI

Prof. MSc. Francisco Ronan Viana Araújo - Examinador
IFPI/*Campus* São Raimundo Nonato-PI

Parnaíba
2018

Dedico ao meus pais, Gilson Vaz de Carvalho e Benta Firmino de Sousa, as minhas irmãs Danica Vaz de Carvalho e Gilca Vaz de Carvalho. Dedico também a minha namorada Angelina Aguiar Soares pelo total apoio nesta caminhada vitoriosa.

Agradecimentos

Agradeço a Deus pelo dom da vida e sabedoria. Agradeço aos meus pais Gilson Vaz de Carvalho e Benta Firmino de Sousa que durante esta jornada sempre estiveram ao meu lado. Sou grato as minhas irmãs Danica Vaz de Carvalho e Gilca Vaz de Carvalho que sempre acreditaram em mim. Agradeço a minha namorada Angelina Aguiar Soares por estar comigo em todos os momentos desta caminhada vitoriosa.

Agradeço ao meu amigo Francisco José Salvino Rocha pois sem a ajuda dele eu não teria chegado até onde cheguei. Sou grato também ao meu amigo Udson Horlando por ter me recebido nesta cidade e sempre está pronto para me ajudar a qualquer momento.

Agradeço aos meus professores José Venâncio, Alexandro Nascimento, Bruno Sombra, Janete Ribeiro, Vilma Dias, Diego Prudêncio, Marcos Matos, Jeová Calisto e em especial ao meu professor e orientador, Itamar Vieira, que durante o desenvolvimento deste trabalho teve muita paciência, empenho, dedicação e força de vontade. Serei eternamente grato. Sou grato a todos que me ajudaram direta e indiretamente a concluir este trabalho.

Deixo a minha gratidão ainda aos amigos Daminhõa Costa, Maria José, Francisco Felipe, Rafael Magalhães, Samanta Rocha, Robso Raygonslei, Ociel ferreira, Emanuel Felipe, Charliane Melo, Renata Yarima, Gabriela de Asis, Jailda costa, Alberto Sena, Jefferson Wherculis.

Por fim gostaria de agradecer a todos as pessoas que de aluma forma me incentivaram e me deram forças para que eu pudesse concluir este curso. Serei eternamente grato a todos.

Resumo

Movimentos oscilatórios são bastante comuns em nosso cotidiano, e o seu estudo tem contribuído significativamente para a qualidade de vida da sociedade moderna. No presente trabalho propomos um estudo do oscilador harmônico amortecido utilizando a plataforma Arduino, uma ferramenta acessível, de baixo custo e que tem apresentado resultados satisfatórios quando empregada na execução de projetos e experimentos em laboratório, tanto de Física quanto de Química. Iremos explorar esta excelente ferramenta de estudo, analisando suas limitações e propondo alguma forma de utilizarmos ao máximo o que esta tem a oferecer. Empregaremos o Arduino na coleta de dados para que assim possamos realizar uma análise gráfica dos resultados obtidos neste estudo.

Palavras-chaves: Oscilador harmônico amortecido. Plataforma Arduino. Coleta de dados.

Abstract

Oscillatory movements are quite common in our daily lives, and its study has contributed significantly to the quality of life of modern society. In the present work we propose a study of the damped harmonic oscillator using the Arduino platform, an affordable, low cost tool that has presented satisfactory results when used in the execution of projects and experiments in the laboratory of both Physics and Chemistry. We will explore this excellent study tool, analyzing its limitations and suggesting some way to make the most of what it has to offer. We will use the Arduino in the data collection so that we can do a graphical analysis of the results obtained in this study.

Key-Words: Damped harmonic oscillator. Arduino platform. Data collection.

Lista de ilustrações

Sumário

1 INTRODUÇÃO

A sociedade moderna evoluiu a partir das revoluções científicas e atualmente é muito dependente da tecnologia. Com os avanços da ciência foi possível descobrir o nosso lugar no universo, aumentar a expectativa de vida dos seres humanos erradicando doenças, além de gerar meios para produzir cada vez mais conhecimento e avanços tecnológicos. Em contraponto este é um dos poucos momentos da história em que somos capazes de nos destruir por completo, além de alterar as condições ambientais que possibilitam a vida na forma que conhecemos.

O desenvolvimento desta tecnologia demanda muito conhecimento, e por vezes não é transmitido de forma adequada às pessoas que buscam adquiri-lo. Ainda em alguns casos, tais fontes não abordam devidamente determinados conteúdos de estudo. Causando muitas vezes, uma falsa compreensão do assunto ou até mesmo a incompreensão deste.

Apesar de tão importante em nossa sociedade, as ciências de uma forma geral, não despertam o interesse na maioria dos estudantes, seja pelo fato de terem um linguagem relativamente complexa, ou ainda, pelo fato de serem ensinadas de forma desconexa do mundo real.

As Leis Físicas, por exemplo, são em geral ensinadas tendo em vista modelos teóricos que fogem da realidade e não permitem uma conexão direta com o mundo dinâmico em que se encontram inseridos os estudantes do século (RODRIGUES; BUSQUINI; SANTARINE, 2010).

Nesse sentido, o estudo experimental surge como uma ferramenta fundamental de ensino e aprendizado. O professor moderno deve ser capaz de instigar seus alunos a se questionar e buscar suas respostas, sendo a experimentação uma maneira de adquiri-las.

O oscilador harmônico amortecido é um modelo Físico que frequentemente é estudado na disciplina de Física, no entanto, nem sempre são feitas as devidas considerações a seu respeito.

Iremos no presente trabalho abordar o desenvolvimento de um projeto baseado na plataforma Arduino. Será proposto uma análise de oscilações amortecidas (Oscilador Harmônico Amortecido).

A plataforma Arduino tem o seu código fonte totalmente grátis, incluindo o seu projeto eletrônico. Sendo possível portanto, desenvolvermos projetos robustos com um equipamento versátil e de baixo custo. Talvez esse seja um dos motivos para que a mesma seja tão explorada no meio acadêmico quando o assunto é aquisição de dados das mais distintas grandezas físicas.

Inicialmente faremos uma breve abordagem no que diz respeito a natureza dos movimentos oscilatórios, com o foco nos movimentos amortecidos e as suas principais características.

Em seguida apresentaremos ao leitor a plataforma Arduino, destacando os seus principais pontos e até onde ela pode ser útil, informando desde como adquiri-la até como fazê-la funcionar.

Trataremos ainda sobre medidas digitais, tendo em vista que o Arduino faz uso deste tipo de medida. Abordaremos o funcionamento de um encoder óptico linear, que é um componente utilizado para detectar interações em função do deslocamento de uma fita e que será utilizado em um sensor de posição; abordaremos ainda os conceitos básicos de medidas experimentais.

Por fim, teremos o desenvolvimento e a conclusão do experimento proposto neste trabalho. Onde abordaremos desde a construção e a calibração do sensor de posição que utilizaremos até a coleta dos dados, e por fim será feita a análise gráfica do comportamento do nosso sistema e as conclusões obtidas com a execução deste projeto.

2 MOVIMENTO DE PARTÍCULAS SUJEITAS A AÇÃO DE FORÇAS DE RESISTÊNCIA

O movimento de uma partícula em um sistema inercial pode ter o seu deslocamento descrito a partir da segunda lei de Newton nos casos em que consideramos a sua massa constante conforme o tempo passa. Essa equação pode ser expressa da seguinte forma

$$F = \frac{dp}{dt} = \frac{d(mv)}{dt} = m\ddot{x} \tag{2.1}$$

onde $\ddot{x}$ corresponde a aceleração da partícula. Esta é uma equação diferencial de segunda ordem e a sua solução pode ser utilizada para determinarmos a posição e a velocidade da partícula em função do tempo, uma vez que conhecemos a função F. A partir dos valores iniciais de posição e velocidade poderemos avaliar as constantes de integração da equação 2.1. Assim podemos assumir que a posição de uma partícula é determinada a partir dos valores iniciais de velocidade e posição (THORNTON; MARION, 2011).

É importante ressaltar que a força na equação 2.1 não é necessariamente constante e na realidade pode fundamentar-se de várias partes distintas. Assim sendo, podemos admitir que uma partícula em queda livre está sujeita a uma força gravitacional constante dado por $F = mg$, onde g representa a aceleração da gravidade. Vai existir sobre esta partícula uma força de retardo que deverá ser uma função da velocidade instantânea. A figura 1 ilustra um modelo pra a situação descrita acima.

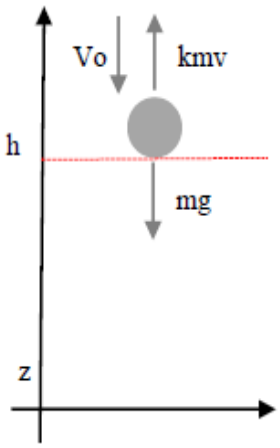

Figura 1: Diagrama de forças que atuam em uma partícula que se desloca em um fluido. Fonte: Elaborada pelo autor.

assim a força total que atua nesta partícula pode ser escrita da seguinte forma:

$$F = F_0 + F_r = mg + F_r(v) \tag{2.2}$$

É comum adotarmos $F_r(v)$ como uma potência da velocidade. Geralmente as forças de resistência reais são bastante complexas, porém a aproximação da lei da potência é útil em várias instâncias, onde a velocidade não varia significativamente. A aproximação da lei da potência, neste caso nos permite escrever a equação 2.2 da seguinte maneira:

$$\vec{F} = m\vec{g} - mkv^n\frac{\vec{V}}{v} \tag{2.3}$$

em que $-mkv$ representa a força de retardo e k é uma constante positiva que especifica a intensidade da força resistiva e $\vec{V}/v$ é um vetor unitário na direção de V. Nos casos em que temos regimes turbulentos podemos utilizar a premissa de que v^2 para fins de simplificação (THORNTON; MARION, 2011).

Considere agora o caso em que uma esfera cai em um fluido qualquer orientada no sentido do eixo z, veja figura 1. Para determinarmos a velocidade e o deslocamento da partícula vamos considerar que a mesma esteja caindo com uma velocidade inicial v_0 de uma altura h e em um campo gravitacional constante. Da equação 2.3 com algumas manipulações matemática temos o seguinte:

$$-dt = \frac{dv}{kv + g}, \tag{2.4}$$

integrando e tomando $V(t = 0) = v_0$ temos,

$$kv + g = \exp^{-kt+kc} => v = -\frac{g}{k} + \frac{kv_0 + g}{k}(\exp^{-kt}), \tag{2.5}$$

integrando mais uma vez e avaliando a constante por meio da definição de $z(t = 0) = h$ obtemos o seu deslocamento

$$z = h - \frac{gt}{k} + \frac{kv_0 + g}{k^2}(1 - \exp^{-kt}). \tag{2.6}$$

Na equação 2.6 vimos que à medida que o tempo se torna muito longo, a velocidade se aproxima do valor limite $-g/k$; este valor é denominado velocidade terminal, v_t (THORNTON; MARION, 2011).

2.1 Partículas sujeitas à uma força restauradora - Osciladores

Movimentos oscilatórios ocorrem constantemente na natureza, como por exemplo um fóton viajando a uma dada frequência de uma fonte emissora até um receptor, as ondas sonoras que chegam até os nossos ouvidos com uma cetra frequência, entre outras. O estudo e a compreensão da natureza dos movimentos oscilatórios tem possibilitado vários avanços para a humanidade, nos possibilitou por exemplo desenvolver sistemas de amortecimento para veículos, desenvolver circuitos eletrônicos e muitas outras situações práticas.

Um modelo extremamente útil para o entendimento das oscilações consiste de um sistema unidimensional hipotético onde temos uma massa presa a uma mola conforme podemos observar na figura 2.

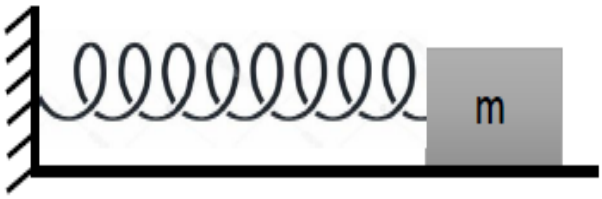

Figura 2: Sistema massa mola proposto. Fonte: Elaborada pelo autor.

Uma vez que essa massa é deslocada da sua posição de equilíbrio, onde a mola encontra-se relaxada, irá surgir uma força (provocada pela mola) na tentativa de restaurar o equilíbrio do sistema e teremos o início de um movimento em direção à posição inicial. No entanto, à medida que a massa do sistema passa pela posição de equilíbrio, esta possui uma velocidade diferente de zero e, portanto, o seu movimento continua a medida que o tempo passa. Se a massa deste sistema estiver livre da atuação de forças externas, este sistema irá se comportará como um oscilador harmônico simples, caso contrário teremos o comportamento de um outro tipo de oscilador, como por exemplo o harmônico amortecido.

Geralmente a força restauradora é uma função complicada do deslocamento, da velocidade, ou até mesmo de alguma derivada temporal de mais alta ordem de posição. Consideramos aqui o caso da força restauradora F ser apenas uma função do deslocamento.

Se $F = f(x)$ descrever a força restauradora possuindo derivadas continuas de todas as funções, onde F_0 é o valor da função função $F(x)$ na origem, e se fizermos uma expansão em serie de Taylor em torno do ponto $(x = 0)$, onde (d^nF/x^n) é a enésima derivada na origem. F_0 é definido como origem do sistema, pois teremos uma derivada

nula nesse ponto.

$$F(x) = F_0 + x\left(\frac{dF}{dx}\right) + \frac{1}{2!}x^2\left(\frac{d^2F}{dx^2}\right) + \frac{1}{3!}x^3\left(\frac{d^3F}{dx^3}\right) + ... \tag{2.7}$$

Caso contrário a partícula iria para longe do ponto de equilíbrio e não retornaria mais. Assim para deslocamentos suficientemente pequenos da partícula, teremos a relação aproximada,

$$F(x) = -Kx, \tag{2.8}$$

onde, a constante k, é da seguinte forma, $k = -dF/dx$. Como a força restauradora sempre tende para a origem, a derivada é negativa e assim temos que a constante k é positiva.

Se apenas a primeira potência do deslocamento vai acontecer em $F(x)$, temos assim uma força restauradora, que nessa aproximação tem caráter linear. Assim a força que tende a trazer a partícula à posição de equilíbrio obedece com uma boa aproximação a lei de *Hooke*.

Os sistemas Físicos descritos nos termos da equação 2.8 obedecem a Lei de *Hooke*. Uma destas classes de processos físicos que pode ser tratada pela aplicação da Lei de *Hooke* é aquela que envolve deformações no regime elástico. Para ilustrar a situação descrita acima podemos usar o modelo físico em que os deslocamentos são pequenos e os limites elásticos não são extrapolados. No entanto devemos ressaltar que esses cálculos são apenas aproximações, pois todas as forças reais de restauração na natureza são essencialmente mais complicadas do que a força simples da Lei de *Hooke*. Como na natureza não dispomos de condições ideais para que as oscilações ocorram, portanto, sempre teremos oscilações amortecidas.

É possível que ocorra algum evento externo, que forneça energia ao sistema, numa taxa igual à taxa de absorção do sistema amortecedor, cancelando o amortecimento. A esse fenômeno damos o nome de oscilações forçadas (THORNTON; MARION, 2011).

2.1.1 Oscilador harmônico simples

Segundo as leis físicas que descrevem o movimento harmônico simples de uma partícula, a mesma seria capaz de executar um movimento oscilatório perpétuo, pois não teríamos interações de nenhuma força externa, o que poderia vir a contribuir para o amortecimento desta partícula. Porém, na prática isso não pode ser observado, pois existem forças dissipativas que atuam contribuindo para que este movimento cesse.

Para chegarmos a um modelo matemático que descreva o movimento harmônico simples (MHS) de uma partícula, substituímos a força da lei de *Hooke* na segunda lei de Newton.

A figura 3 mostra o diagrama de forças que atuam em um sistema massa-mola no momento em que a massa encontra-se livre e causa assim uma deformação na mola, levando o sistema a uma nova posição de equilíbrio.

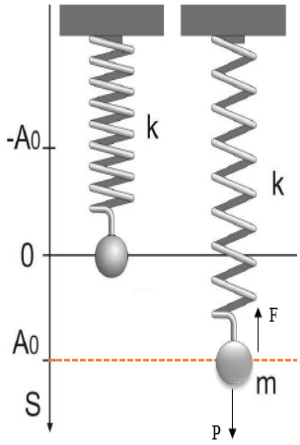

Figura 3: Diagrama de forças que atuam no sistema massa mola proposto. Fonte: (RODREGUES, M)

Assim, tomando $F = -kx$ e $P = m\ddot{x}$ teremos que:

$$-kx = m\ddot{x}, \tag{2.9}$$

se tomarmos $w_0 = \sqrt{k/m}$ temos a seguinte equação:

$$\ddot{x} + w^2 x = 0, \tag{2.10}$$

uma solução para a mesma, pode ser escrita da seguinte forma:

$$x(t) = A\,\mathrm{sin}(w_0 t - \varphi) \tag{2.11}$$

ou

$$x(t) = A\,\mathrm{cos}(w_0 t - \varphi), \tag{2.12}$$

ou ainda uma combinação das duas expressões. Essas são as equações do deslocamento do oscilador harmônico simples. Para obtermos a velocidade e aceleração, basta derivarmos a solução da equação 2.10 em função do tempo.

Se relacionarmos a energia total e a amplitude de movimento da partícula, temos para energia cinética o seguinte:

$$C = \frac{1}{2} k A^2 \,\mathrm{cos}^2(w_0 t - \varphi), \tag{2.13}$$

resolvendo a integral $\int_0^x w(x)dx$ e igualando à energia potencial temos,

$$U = \frac{1}{2}kA^2[\sin^2(w_0t - \varphi)], \tag{2.14}$$

igualando 2.13 e 2.14 para encontrarmos E total, vem que

$$E = \frac{1}{2}kA^2[\cos^2(w_0t - \varphi) + \sin^2(w_0t - \varphi)]$$

$$E = \frac{1}{2}kA^2. \tag{2.15}$$

fica claro a dependência da energia total com o quadrado da amplitude e a independência com o tempo, ou seja, temos a conservação de energia, já que estamos considerando um sistema ideal. Neste caso se fosse possível obter um sistema ideal teríamos um sistema que oscilaria eternamente. Temos aqui o resultado geral para sistemas lineares (THORNTON; MARION, 2011).

2.2 Oscilações amortecidas

Quando temos simultaneamente a ação de uma força restauradora e uma força de resistência proporcional à velocidade, o movimento descrito pela partícula torna-se muito complicado. Nesta seção iremos avaliar qual o comportamento de uma partícula que está nesta situação.

Até agora, analisamos casos de oscilações ocorrendo em sistemas conservativos. Sistemas estes que tem a capacidade de conservar a sua energia interna, permitindo que os seus movimentos sejam perpétuos. Porém, isso não ocorre no mundo a nossa volta devido às forças dissipativas. Nesse tipo de movimento existe uma força de amortecimento, que pode vir a depender da velocidade[1], adotamos esta força, como uma função linear da velocidade. Chamaremos de $-b\dot{x}$ o termo que representa a função de amortecimento do sistema, e que tem dependência direta com a velocidade, Ou seja, teremos uma variação no amortecimento à medida que ocorre alguma variação na velocidade. Para uma partícula de massa m se movimentando sob influência de uma força de restauração linear e uma força resistiva, escreveremos uma equação que descreverá o seu movimento (RODRIGUES; BUSQUINI; SANTARINE, 2010). E esta equação tem a seguinte forma:

$$m\ddot{x} + b\dot{x} + kx = 0, \tag{2.16}$$

[1] Veja a Seção 2.4, Capitulo 2 (THORNTON; MARION, 2011).

se chamarmos de $\frac{b}{2m} = \beta$ e $\frac{k}{m} = w_0^2$ para $m > 0$. temos

$$\ddot{x} + 2\beta\dot{x} + w_0^2 x = 0, \tag{2.17}$$

que é uma equação diferencial homogênea de segunda ordem com coeficientes constantes. Temos como equações características:

$$r_1 = -\beta + \sqrt{\beta^2 - \omega_0^2} \tag{2.18}$$

$$r_2 = -\beta - \sqrt{\beta^2 - \omega_0^2} \tag{2.19}$$

e assim a solução geral da EDO é:

$$x(t) = \exp^{-\beta t}[A_1 \exp(\sqrt{\beta^2 - \omega_0^2}t) + A_2 \exp(\sqrt{\beta^2 - \omega_0^2}t)] \tag{2.20}$$

Temos três casos gerais, de movimentos oscilatórios, que merencem ser estudados, são eles, os movimentos:

Sub amortecimento:

$\omega_0^2 > \beta^2$

Amortecimento Crítico:

$\omega_0^2 = \beta^2$

Sobre Amortecimento:

$\omega_0^2 < \beta^2$

A seguir, é possível observar uma superposição das representações gráficas dos três casos. Estes três movimentos, tem como característica, um aumento na perda de energia, a medida que a viscosidade do fluido cresce.

2.2.1 Movimento sub amortecido

Dos casos anteriormente analisados, apenas o sub amortecimento descreve um movimento oscilatório. No estudo de um movimento dessa natureza, por conveniência devemos, escrever:

$$\omega_1^2 = \omega_0^2 - \beta^2 \tag{2.21}$$

para $\omega_1^2 > 0$.

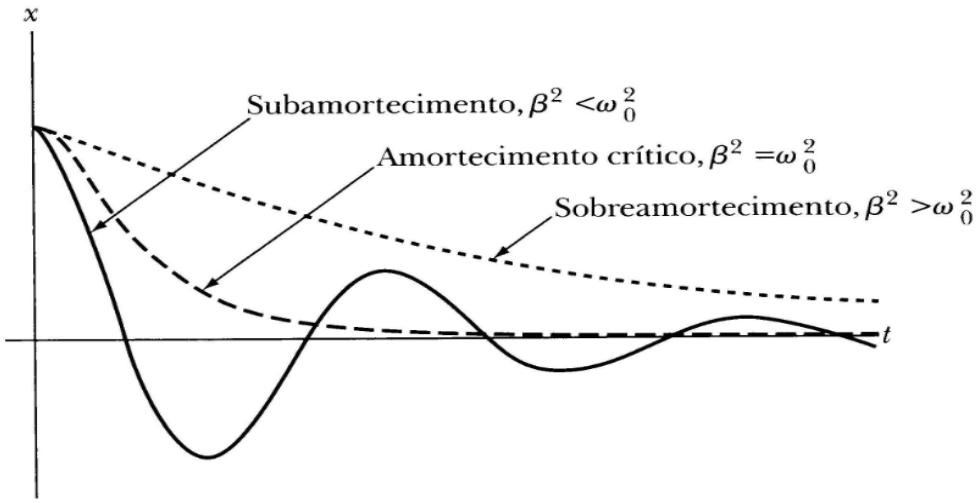

Figura 4: Composição dos movimentos, subamortecido, sobreamortecido e amortecimento crítico. Fonte: (THORNTON; MARION, 2011)

Os expoentes entre os colchetes da equação 2.20 são imaginários e a solução fica da seguinte forma:

$$x(t) = \exp^{-\beta t}[A_1 \exp^{iw_1 t} + A_2 \exp^{-iw_1 t}], \tag{2.22}$$

ou ainda, rescrevendo

$$x(t) = A \exp^{-\beta t} \cos(\omega_1 t - \varphi). \tag{2.23}$$

onde ω_1 representa a frequência angular do oscilador amortecido. Aqui o oscilador não vai ter um período fixo, assim não é possível definir uma frequência, pois o fato de haver amortecimento, ou seja, perda de energia do sistema, implica diretamente nisso. Se $\omega_1 = 2\pi/2t$, onde t é o tempo, notemos que 2T aqui corresponde ao período. Por simplicidade chamamos ω_1 de "frequência angular" do sistema com amortecimento. ω_1 vai ser menor quando não ocorrer amortecimento ($\omega_1 > \omega_0$). Quando o amortecimento tender a zero, teremos:

$$\omega_1 = \sqrt{\omega_0^2 - \beta^2} \cong \omega_0 \tag{2.24}$$

Assim podemos usar o termo frequência angular característica. No entanto só teremos um resultado com boa exatidão, quando o parâmetro de amortecimento tende a zero. O termo ($\exp^{-\beta t}$) faz com que a amplitude do oscilador amortecido caia com o tempo, onde $\beta > 0$ é o envoltório da curva (deslocamento x tempo) é expresso da seguinte forma:

$$x_{en} = \pm A(\exp^{-\beta t}) \tag{2.25}$$

A Figura 5, mostra um movimento sub amortecido (linha cheia) que é um movimento oscilatório, representado pelos traços curtos, este movimento vai decaindo com o envoltório exponencial (traços longos). Se compararmos as curvas observamos que no caso amortecido temos um período maior do que o no não amortecido. Assim temos uma frequência logicamente menor no caso amortecido.

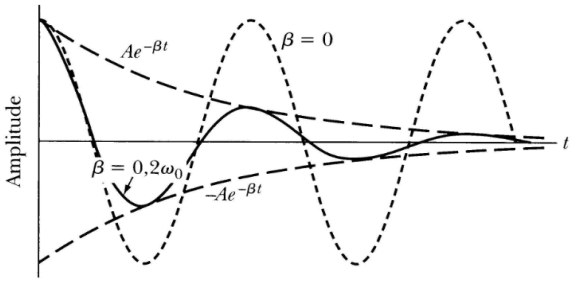

Figura 5: Movimento sub amortecido (linha cheia) é um movimento oscilatório (traços curtos) que diminui com o envoltório exponencial (traços longos). Fonte: (THORNTON; MARION, 2011)

Para dois máximos sucessivos, temos a seguinte relação entre as amplitudes de oscilação:

$$\frac{A(\exp)^{-\beta T}}{A(\exp)^{-\beta(T+\tau_1)}} = (\exp)^{\beta\tau_1} \tag{2.26}$$

onde os pontos de máximos serão para t = T, e $\tau_1 = 2\pi/\omega_1$. Chamaremos de decremento de movimento o termo $(\exp)^{-\beta T\tau_1}$ onde o termo $\beta\tau_1$ é o decremento logaritmo do movimento (THORNTON; MARION, 2011).

De acordo com a figura 6 podemos perceber então, que a energia do oscilador amortecido não é conservada, ela cai conforme o tempo passa. A energia é dissipada pela a entidade que amortece o sistema. A taxa de perda de energia é proporcional ao quadrado da velocidade vetorial e terá valor máximo quando a velocidade for máxima, sendo momentaneamente nula, quando a velocidade da partícula for zero (amplitude máxima).

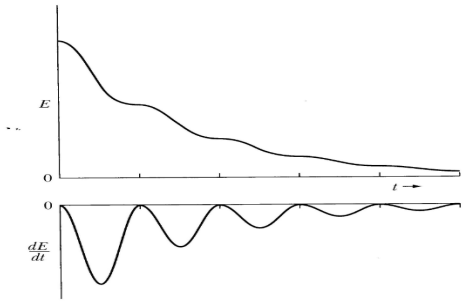

Figura 6: Energia total e a taxa de perda de energia do oscilador amortecido. Fonte: (THORNTON; MARION, 2011)

2.2.2 Movimento criticamente amortecido

Para o caso de uma força de amortecimento suficientemente grande veremos que não ocorrerá um movimento oscilatório ($\beta^2 > \omega_0^2$). Para a velocidade inicial zero, o deslocamento vai ocorrer bem devagar, até atingir o equilíbrio. portanto quando $\omega_0^2 = \beta^2$ ocorre o amortecimento crítico. As raízes da equação característica serão então iguais e a solução será escrita da seguinte forma:

$$x(t) = (A + Bt)\exp^{-\beta t} \tag{2.27}$$

De acordo com as condições iniciais, este oscilador alcançará o equilíbrio mais rápido que os osciladores sub ou sobre amortecidos.

2.2.3 Movimento sobre amortecido

Nestas circunstâncias com o termo $\beta^2 > \omega_0^2$ ocorrerá um amortecimento maior, e aqui teremos um oscilador sobre amortecido. Disso resulta que a equação 2.22 passa a ter seus expoentes entre colchetes, reais. Logo teremos:

$$x(t) = \exp^{-\beta t}[A_1 \exp^{\omega_2 t} + A_2 \exp^{-\omega_2 t}] \tag{2.28}$$

$$\omega^2 = \sqrt{\beta^2 - \omega_0^2} \tag{2.29}$$

Como o movimento nessa ocasião não obedece um período fixo, logo ω_2 não descreve a frequência angular. Um exemplo de movimento sobre amortecido é um sistema onde tenhamos um corpo imerso em um tubo contendo uma substancia viscosa. Isso resultaria em uma queda brusca, da amplitude de oscilação.

Estes modelos teóricos abordados não são complemente observados no mundo real. Porém para certos limites de validade podemos verificar que os resultados obtidos condizem com os esperados. No próximo capítulo iremos apresentar uma ferramenta que pode ser utilizada para a coleta de dados experimentais.

3 AQUISIÇÃO DE DADOS EXPERIMENTAIS E A PLATAFORMA ARDUINO

3.1 Apresentação da plataforma

O Arduino® é uma plataforma de prototipagem cujo código fonte é baseado em software livre. Em suma, podemos dizer que o Arduino é um sistema embarcado (ou sistema embutido) [1], baseado em um microcontrolador AVR. Este equipamento foi desenvolvido principalmente para entusiastas, hobistas entre outros. Seus idealizadores foram: Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, e David Mellis (OLIVEIRA; CARVALHO; ANDRADE, 2014). Esta plataforma conta com arquitetura RISC (*acrônimo de Reduced Instruction Set Computer*; em português, "Computador com um conjunto reduzido de instruções"). Atualmente é fabricada pela empresa ATMEL.

Um microcontrolador é um tipo de circuito integrado que possui no seu interior uma unidade de processamento (Unidade Lógica Aritmética - ULA), memórias e periféricos, que permitem o seu funcionamento requerendo apenas uma pequena quantidade de componentes externos para o seu funcionamento (MCROBERTS, 2011).

A placa Arduino uno, é uma das mais populares e utilizadas. Ela é baseada no microcontrolador ATmega328p,tendo um microcontrolador de 8 bits e encapsulamento DIP28(*Dual In-Line Package*)[2], que dispõe de 32 KB de memória *flash*, onde 512 KB são usados pelo *bootloader*, sendo este, um *firmware* [3] que possibilita a gravação de um novo *firmware* sem a necessidade de um programador externo. Possui 2 KB de memoria RAM e 1 KB de memória EEPROM(Local onde os valores são armazenados quando a placa é desligada). O Arduino opera com um cristal de quartzo de 16MHz que é conectado aos pinos 9 e 10 do ATMEGA328P (CAVALCANTE; TAVOLARO; MOLISANI, 2011).

Fora do Brasil os preços das placas da família Arduino são bem acessíveis, o que nos influencia a comprar diretamente de países como a China, por exemplo.

Quando pretendemos desenvolver um projeto que requisite, mais capacidade de processamento e armazenamento de dados, o Arduino Mega é o mais indicado; pois o microcontrolador utilizado por ele é o ATmega2560, que dispõe de 8 bits, e arquitetura RISC avançada. Esta placa conta com mais recursos que o Arduino Uno.

---

[1] É um sistema microprocessado,onde o computador é totalmente encapsulado ou dedicado ao dispositivo ou sistema que este controla

[2] Pacote com duas fileiras de pinos nos lados cumpridos.

[3] Conhecidos como “software embarcado” estes representam um conjunto de instruções operacionais que são programadas diretamente no hardware de equipamentos eletrônicos

Na figura abaixo é possível vizualizar a estrutura do diagrama de bloco do microcontrolador ATmega2560, utilizado no Arduino Mega.

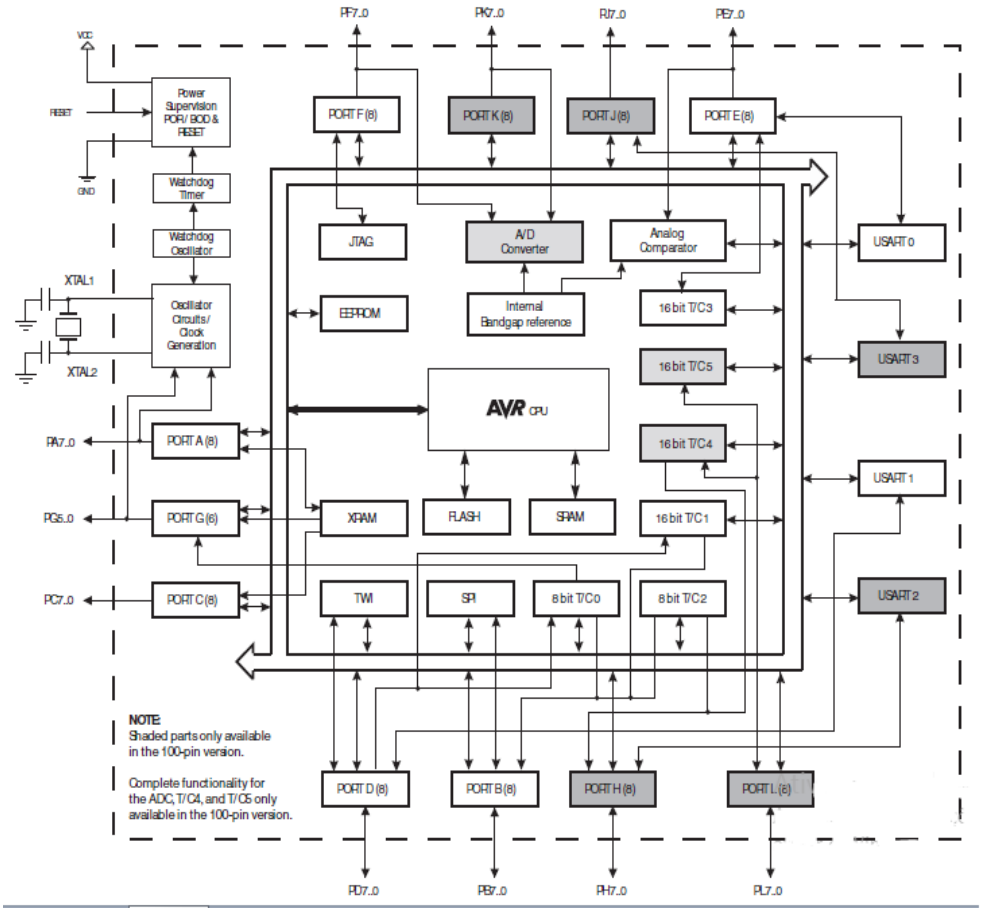

Figura 7: Diagrama de bloco do Micro controlador ATmega2560. Fonte: (ATMEL®, 2018)

O ATmega2560 tem a característica de operar com tensões relativamente baixas, de até $2V$. Conta ainda com 256KB de memória *flash*, onde 8KB são usados pelo *bootloader*, 8 KB de RAM e 4 KB de EEPROM, dispondo também, de 4 fontes de comunicação serial e 16 entradas analógicas e 15 saídas PWM, comunicação SPI, I2C e 6 pinos que podem ser utilizados para interrupções externas (ARDUINO, 2018).

As placas da família Arduino, tornam-se cada dia mais populares tanto no mundo da eletrônica, como no mundo educacional, por possibilitarem aos seus usuários um rápido aprendizado e desenvolvimento de seus projetos. Um outro fator que pode ter proporcionado um rápido crescimento no uso das placas Arduino, é o seu baixo custo, e a liberdade de manipulação que esta placa oferece para o usuário.

Na internet, contamos com uma enorme disponibilidade de informações e códigos de alguns projetos prontos, compartilhados em vários blogs e páginas.Tudo isso disponibilizado gratuitamente.

Um outro exemplo é o movimento *Maker* [4] que acredita no gênio que possa existir

[4] Termo que significa fazer você mesmo. Esse movimento acredita que pessoas comuns são capazes de

dentro de você, e incentiva, projetistas e hobistas a desenvolverem seus próprios projetos (D'AUSILIO, 2012).

Estas placas possuem um *hardware* capaz de fornecer ao microcontrolador todos os componentes necessários ao seu funcionamento (osciladores, reguladores e fontes de tensão, circuitos de proteção e comunicação, dentre outros) (SOUZA et al., 2011). Na figura 8 podemos ver estes componentes.

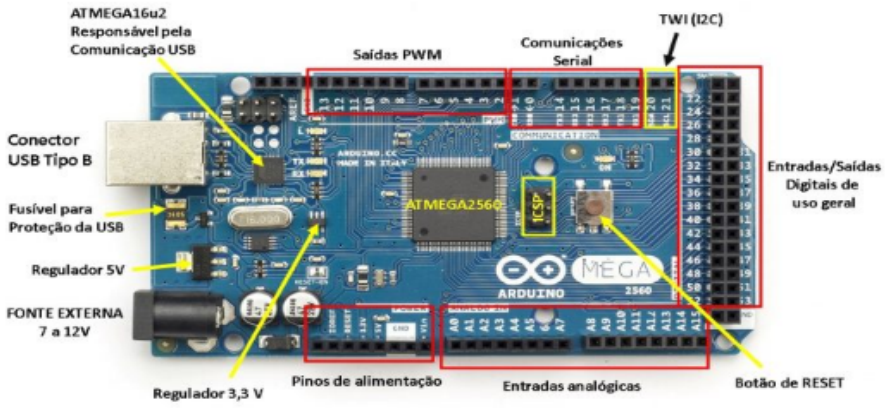

Figura 8: Discrição das partes que compoem a placa Arduino mega. Fonte: (SOUZA, 2018).

A família Arduino apresenta uma variedade de placas. Faremos aqui a análise da placa Arduino Mega. Destacando as principais partes que a constituem.

Descrição:

•54 pinos digitais, (de acordo com a necessidade os mesmos podem ser utilizados como saídas ou entradas).

•16 pinos de entradas analógicas, de 10 bites, onde a tensão recebida por eles que pode variar de 0 a $5V$, é convertida e interpretada em valores binários que vão de 0 a 1023, nos pinos que vão de (A0 até A15).

•15 pinos de saída, que podem ser programados para funcionar como saídas PWM, através da função *analogWrite*.

•6 portas, que podem ser utilizadas para comunicação com *shields* (vin, 2 GND, 5V, Res, 3,3V, IOREF).

•Uma porta USB, onde podemos alimentar a placa, numa tenção que deve variar de (9 a 12) V.

•1 micro processador ATMEGA2560.

•1 micro processador ATMEGA16U2, utilizado para comunicação USB com o computador.

•1 LED que identifica se a placa está ligada.

•1 LED ligado à porta 13, para testes simples.

projetar, desenvolver, e executar projetos sem o auxilio de especialistas dá aria.

•LEDs que mostram a comunicação cereal do Arduino.
•1 botão de Resete.
•1 conector Jack, que pode ser utilizado para alimentação externa da placa.
•1 cristal que oscila a 16MHz.

A seguir, podemos visualizar outros dois modelos de placas da família Arduino em comparação com Mega.

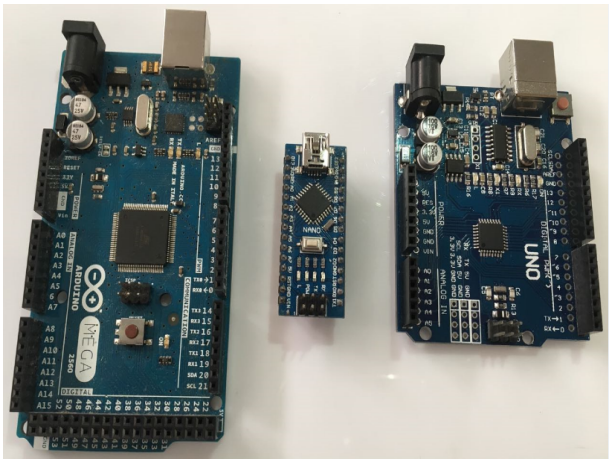

Figura 9: Arduino mega, Arduino nano e Arduino uno. Fonte: Elaborada pelo autor

O número de entradas e saídas do Arduino, bem como suas funcionalidades, podem

ser estendidas com a utilização de placas auxiliares, denominadas *Shields* [5]. Estas contêm outros dispositivos como mostradores de LCD, módulos de comunicação Ethernet[6], sensores de diversos tipos permitindo o rápido desenvolvimento de protótipos.

O uso destas placas auxiliares pode facilitar bastante o processo de prototipagem, por livrar o usuário do trabalhoso processo de confecção e soldagem de placas de circuito impresso (OLIVEIRA; CARVALHO; ANDRADE, 2014).

Esta plataforma utiliza uma linguagem de programação baseada em *Processing* [7] além de um circuito para a comunicação com o computador (SOUZA et al., 2011). Vale frisa que, uma das excelentes vantagens desta placa é o seu ambiente de desenvolvimento baseado na linguagem c [8].

Linguagem esta, conhecida informalmente como C/C++ (linguagem de programação, na qual o código fonte, após ser traduzido pra uma linguagem de nível baixo como a *Assembly*[9]), é compilado, e em seguida é executado diretamente pelo processador. Esta linguagem é bastante difundida sendo relativamente simples, pois permite que leigos em programação desenvolvam pequenos projetos.

Todas as placas Ardunino possuem um *bootloader*, que é *software* pré gravado no microcontrolador (permite que ele se autograve), dispensando a necessidade de um gravador externo. Dessa forma, o *firmware* desenvolvido pelo usuário, é gravado diretamente no microcontrolador a partir de um porta USB.

Toda essa imensa gama de possibilidades, abre infinitas portas e possibilidades, para quem queira utilizar o Arduino, como ferramenta de estudo ou trabalho, entre outras finalidades. Esta fantástica placa, é uma excelente opção, para iniciantes que pretendem adentrar, no mundo da eletrônica digital. Com ele é possível desenvolver desde projetos bem simples até projetos mais complexos.

3.2 Ambiente de desenvolvimento

A IDE[10] é um ambiente de desenvolvimento integrado, onde dispomos de todas as ferramentas necessárias para programar o Arduino. Como o Arduino necessita de instruções lógicas pra realizar algo de útil, é lá que serão desenvolvidos as unidades de códigos ou esboços, que é carregada e executada em sua placa Arduino. Assim como o *hardware*, o *software* também é de código livre (BOUQUET et al., 2017).

[5] Placas usadas como espanação de *hardware* que são acoplados na placa Arduino.
[6] É uma arquitetura de interconexão para redes locais, baseada no envio de pacotes.
[7] Linguagem de código aberto e ambiente de desenviolamento integrado.
[8] No caso do arduino a linguagem c é um pouco diferente da usual por ser aprimorada
[9] Notação que torna o código de maquina legível pra humanos, e que uma arquitetura especifica de computador utiliza.
[10] Ambiente Integrado de Desenvolvimento

É possível fazer o download da IDE, gratuitamente no *site*[11] do Arduino. Na figura abaixo podemos ver a descrição da interface da IDE do Arduino.

Figura 10: IDE do Arduino. Fonte: Elaborada pelo autor

O Editor principal: Ao ser carregado pela primeira vez, o mesmo abre um *esketch* (esboço) em branco pronto para receber as primeira linhas de programação.

A IDE apresenta uma barra de tarefas que tem as seguintes funções:

Verificar – faz a verificação do esboço em busca de erros. Caso encontre algum, este será apresentado na barra de tarefas, localizado na parte inferior da tela.

Novo – Abre um novo esboço.

Abrir – Abre uma lista de esboço salvos anteriormente e exemplos prontos.

Salvar – Salva o esboço.

Carregar – Carrega e envia o esboço para o Arduino.

3.3 *TIMERS* do ATmega2560

Uma maneira de contornar problemas com temporização é utilizando os *TIMERS* do Arduino. O ATMega2560 dispõe de 6 *TIMERS* de 8 *bites*, que são os *TIMERS* 0, 1 e 2; e os *TIMERs* 3, 4, e 5, sendo estes últimos de 16 *bites*. Eles podem ser configurados pra atuarem dentro do limite de precisão na medida de tempo do Arduino. O *TIMER* 2 é um temporizador que permite a contagem de eventos, geração de sinal PWM, medida de

[11] https://www.arduino.cc/en/Main/Software

pulsos, entre outros. A partir da configuração dos seus registradores, onde se pode definir um tempo para que ocorra o seu estouro (ROBOTSHOP, 2018)(QUEIROZ, 2018).

3.3.1 Utilização do *TIMER* 2

O *TIMER* 2 é um registrador com a possibilidade de ser configurado, de modo que, após um dado tempo ocorra o seu estouro, gerando assim interrupções externas por estouro do *TIMER* (ATMEL, 2018). A seguir, temos um esboço que mostra como podemos utilizar este *TIMER* para se gerar um pulso quadrado como um período de 2,048ms na porta digital 3 do Arduino. Portanto o seu estouro vai ocorrer a cada 1,024ms, ou seja, a cada estouro o pino digital 3 inverte o seu estado lógico.

```
1
 int tempo = 0 // variavel que controla o tempo de leitura
3
//----------------- Rotina de Interrupcao ----------------
5 ISR(TIMER2_OVF_vect)
   {
7   TCNT2=240;      // Reinicializa o registrador do Timer2 a cada 1,024ms
    tempo++;        // incremnto de tempo
9   if(tempo==1){   // condi   o de leitura

11  digitalWrite(3, !digitalRead(3)); //Inverte o estado da sa da do pino
   //digital 3
    tempo=0; // zera o tempo
13  }
  }
15//-------------- Configura  es Iniciais ----------------
  void setup()
17{
     pinMode(3,OUTPUT); // modo do pino
19   TCCR2A = 0x00;     //Timer operando em modo normal
     TCCR2B = 0x07;     //Prescaler 1:1024
21   TCNT2  = 240;      //1,024 ms overflow again
     TIMSK2 = 0x01;     //Habilita interrup   o do Timer2
23
//-------------- configura  es do TIMER ----------------
25// Estouro = Timer2_cont x prescaler x ciclo de m quina
  // Ciclo de m quina = 1/Fosc = 1/16E6 = 62,5ns = 62,5E-9s
27// Estouro = (256 - 240) x 1024 x 62,5E-9 = 1,024ms

29} //end setup
```

Na figura 11 podemos visualizar o pulso gerado com o esboço acima.

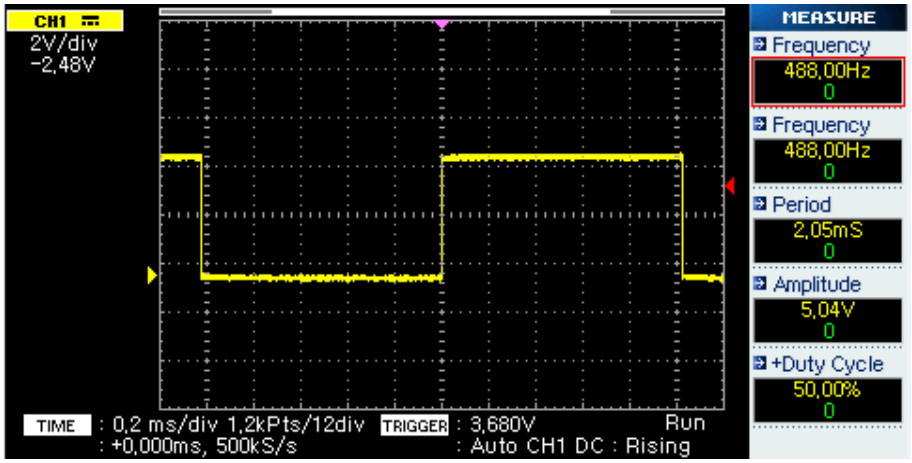

Figura 11: Pulso gerada com o esboço acima. Fonte: elaborada pelo autor

Observe na figura 11 que temos um pulso quadrado que apresenta um período de 2,05ms, este resultado é muito próximo do período esperado. Assim concluímos que a utilização dos *TIMERS* se mostra eficiente e se faz necessária quando buscamos uma base de tempo precisa ao trabalharmos com o Arduino.

3.4 Medias experimentais

Quando realizamos a medida de qualquer grandeza física, deve-se utilizar equipamentos capazes de fornecer uma informação quantitativa dos resultados obtidos. De maneira que o indivíduo que esteja realizando a medição possa expressar numericamente o quanto tal equipamento é confiável, ou seja, o seu grau de precisão ao realizar uma medida. Na falta desses indicadores e dependendo da medida os resultados obtidos podem não ter valor, pois não será possível expressar o erro da medida e compará-la com uma outra. Portanto, vemos a importância de tratarmos com atenção a incerteza das medições (VUOLO, 1992).

É essencial que haja um método confiável, consistente e compreensível para que possamos expressar o grau de confiança nos resultados de uma medição, ou seja, determinarmos o quanto esta medida é incorreta (COLUCI et al., 2013).

O termo incerteza de uma medida, é relativamente novo. Porém, mesmo que saibamos o quanto uma medida é incorreta e sejam tomadas as devidas providências para realizarmos uma medida precisa, sempre teremos dúvida no último algarismo significativo e a incerteza sempre nos acompanhará (JURAITIS; DOMICIANO, 2009).

Durante a execução de uma medida devemos tomar uma padronização, escolhendo unidades para cada grandeza a ser medida. No entanto, até o final do século XVI estas medidas variavam arbitrariamente de um país para outro; houve então a necessidade de

desenvolver um sistema internacional de medidas, a fim de tornar os resultados das medições bem compreendidos e devidamente interpretado, tanto na comunidade científica quanto na industrial, e entre outras. Então, a partir da décima quarta conferência geral sobre pesos e medidas, ficaram definidos as sete grandezas faíscas fundamentais (WALLARD, 2004).

Essas grandezas possuem unidades padrão conhecidas por toda comunidade científica como unidades do Sistema Internacional de Unidades, tendo como sigla (SI), como pode-se observar na lista abaixo:

•Temperatura, e sua unidade de medida no SI é kelvin (K).
•Massa e sua unidade de medida no SI é kilograma (kg).
•Tempo e sua unidade de medida no SI é segundo (s).
•Comprimento e sua unidade de medida no SI é metro (m).
•Intensidade de corrente elétrica e sua unidade de medida no SI é ampére (A).
•Quantidade de matéria e sua unidade de medida no SI é mol (mol).
•Intensidade luminosa e sua unidade de medida no SI é candela (cd).

Para mensurarmos a grandeza de um corpo, devemos realizar uma medida, e os valores numéricos obtidos através destas medições (ato de medir algo) são tratados como dados experimentais. Estes valores trabalhados e apurados após as medidas são os dados da medida. Na figura 12 podemos observar uma medição sendo feita com a utilização de um dilatômetro linear. Com este equipamento é possível, realizarmos medidas com uma precisão de aproximadamente (0,01mm).

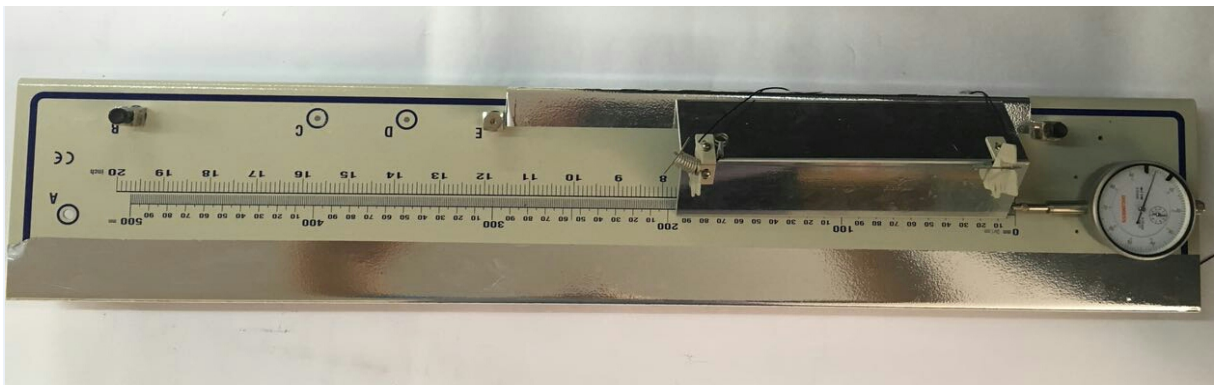

Figura 12: Dilatômetro linear. Fonte: Elaborada pelo autor

3.5 Conversão analógico para digital (A/D)

Os sistemas computacionais processam informações na forma de dados digitais que em geral estão sob a forma de algum código de números binários. Desse modo a informação

digital gera um conjunto discreto e descontínuo de valores possíveis. Estas informações são baseadas em dois níveis lógicos, 0 ou 1, verdadeiro ou falso, ligado ou desligado.

O Arduino é um equipamento capaz de realizar leituras de grandezas Físicas baseadas em sinais digitais utilizando sensores digitais, podendo também utilizar o seu sistema de conversão A/D.

Nesse caso, para que o mesmo possa processar uma informação, advinda do mundo analógico, é necessário que tal informação passe por um tratamento e seja então convertida para uma cadeia de bits. Este processo é denominado conversão Analógico-Digital (A/D). Reciprocamente para que os sistemas digitais possam gerar uma grandeza analógica, estes devem passar pelo processo inverso. É o que chamamos de conversão Digital-Analógica(D/A) (ZUCH, 1979).

Na figura 13 podemos acompanhar de maneira bem simples como se procede a leitura e a conversão de uma grandeza física em um sinal digital (KESTER; BRYANT, 2009).

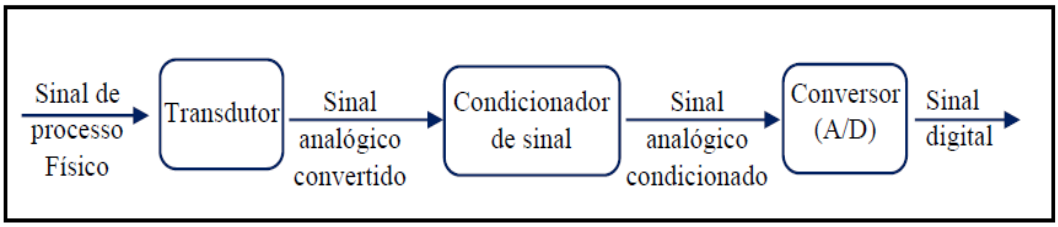

Figura 13: Diagrama de conversão analógico para digital. Fonte: Elaborada pelo autor

Como estão livres de ruídos, consequentemente apresentam uma maior qualidade no envio de informações. Os sistemas digitais atualmente ocupam o espaço que antes era dominado por sistemas analógicos. As técnicas de conversão A/D são variadas e dependem principalmente da precisão e do tempo limite para que a conversão seja realizada, incluindo o também no seu custo.

3.5.1 Técnicas de conversão (A/D)

Existem diversas técnicas de conversões A/D (analógica para digital) citaremos aqui de maneira bem resumida alumas delas:

Codificação paralela- Conversão através de comparadores de tensão conectados em paralelo com o sinal a ser decodificado. Veja na figura 14 o diagrama de esquemático de funcionamento desta técnica.

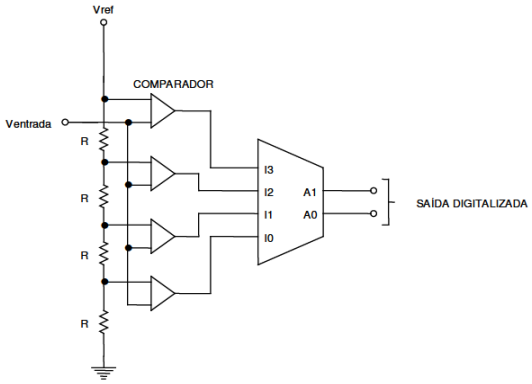

Figura 14: Diagrama esquemático da técnica de codificação paralela. Fonte: (DIGITAL, 2018)

Contador gerador de rampa- Esta técnica consiste em apenas um circuito comparador, sendo que a sua resolução vai depender do número de "degraus" do gerador de rampa. Um número maior ou menor de degraus implica numa maior ou menor resolução do conversor. Veja na figura 15 o diagrama de esquemático de funcionamento desta técnica.

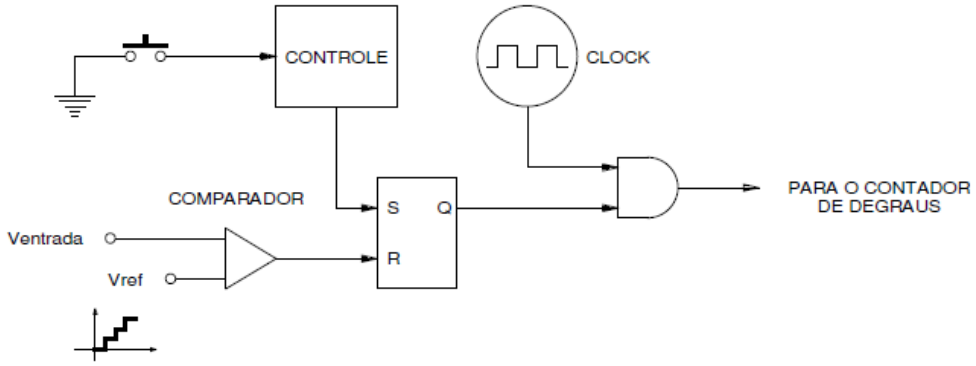

Figura 15: Diagrama esquemático da técnica contador gerador de rampa. Fonte: (DIGITAL, 2018)

Aproximações sucessivas- semelhante ao contador de rampa, difere apenas no fato de que o valor da tensão de entrada é otimizado como menor tempo. Veja na figura 16 o diagrama esquemático de funcionamento desta técnica.

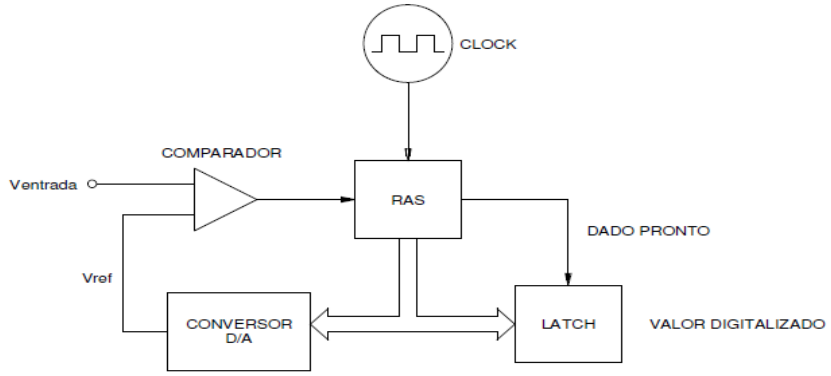

Figura 16: Diagrama esquemático da técnica de aproximação sucessivas. Fonte: (DIGITAL, 2018)

Contamos ainda com os amplificadores operacionais usado para o condicionamento dos sinais de entrada e saída e que simultaneamente trabalham como comparador de sinal. Uma das suas principais características é alta impedância, alto ganho em tenção, baixo ganho em saída.

Já a resolução de um conversor define a sua qualidade, ou seja, quanto maior a resolução maior a precisão na produção de valores convertidos, e como consequência disso teremos o sinal gerado terá uma qualidade melhor. Os valores gerados serão armazenados em pacotes discretos de informação, com sua resolução expressa em níveis da seguinte forma:

$$N = 2^n \tag{3.1}$$

Se estamos utilizando um conversor A/D de 8 bits, de acordo com a equação 3.1 poderemos ter até 256 níveis distintos de resolução. É possível também definir a resolução de um conversor em termos de voltes (V).

Os microcontroladores ATmega2560, trabalham com um conversor analógico digital, que utiliza o modo de conversão, por aproximação sucessiva.

4 METODOLOGIA

De acordo com o discutido até o momento, pretendemos neste trabalho realizar o estudo de sistemas amortecidos, em especial o oscilador harmônico amortecido. Pretendemos desenvolver um sistema masa-mola que se comporte como tal, e que seja submetido a diferentes condições de oscilações. Buscamos alguma relação entre as forças de amortecimento e o quanto estas forças podem influenciar no comportamento do sistema.

Pretendemos verificar as características do nosso sistema, tais como frequência de oscilação natural da mola, período de oscilação e a sua constante elástica. Para esta tarefa, utilizaremos um sistema composto por um oscilador harmônico amortecido que irá oscilar verticalmente.

Realizaremos medidas com o sistema utilizado em diferentes situações. o sistema terá a sua massa imersa em três fluidos distintos, e então será posto para oscilar a partir de uma certa amplitude e será gerado então gráficos que descrevem o comportamento destes sistemas. Uma vez posto a oscilar o sistema terá a posição da sua massa rastreada em função do tempo, o que vai nos permitir gerar gráficos de posição em função do tempo. Para isso, utilizaremos um microcontrolador Arduino mega e um sensor de posição de alta resolução.

De posse dos dados, utilizaremos o *software* (*OriginPro* 8.5.0 SR1) para analisarmos e tratarmos os gráficos obtidos em cada situação.

Esperamos obter gráficos dos movimentos subamortecido, criticamente amortecido e sobre amortecido. Assim como determinar a constante elástica da mola, o valor da gravidade, frequência e período de oscilação da mola do sistema.

4.1 Encoder (Codificador óptico)

Um *encoder* é um exemplo de sensor digital, existindo uma lista variada quanto aos tipos. Estes equipamentos podem ter diversas aplicações em vários projetos.

Esse tipo de sensor funciona como uma chave de luz, sendo capaz de realizar leituras tais como: deslocamento linear e angular, velocidade linear e angular. Estes equipamentos funcionam baseados nas interrupções causadas por um determinado evento (KRASNOW, 2018).

Estes sensores, são constituídos de um emissor e receptor de luz infravermelho, que trabalha convertendo interrupções em pulsos elétricos bem definidos. Quando o feixe de

luz infravermelho atinge o fotorreceptor é então gerado um pulso elétrico que é recebido e tratado pelo Arduino, que o converte para um sinal digital. Este sinal se comporta como uma onda quadrada (NETO; APOLINÁRIO; SOARES, 2017).

Os *encoders* mais simples são utilizados apenas para identificar e contar interrupções. Porém, não podem ser utilizados para o reconhecimento do sentido em que estão ocorrendo as interrupções. Já os *encoders* incrementais, além de possibilitar que se identifique o sentido também permite que se faça a contagem das interrupções geradas. Temos ainda, os *encoders* absolutos, estes são os mais completos além de realizar as mesmas tarefas que os incrementais, com estes ainda podemos determinar a posição de um móvel linear ou ainda a posição angular de um eixo, sendo assim, é mais vantajoso utilizá-los quando buscamos desenvolver um projeto mais elaborado, exemplo disto acontece quando queremos determinar o deslocamento e o sentido de um móvel simultaneamente (SANTOS, 2015).

Sua aplicação se dá basicamente em projetos de eletrônica, sendo também aplicado na industria e na robótica educacional (CARVALHO et al., 2010). Na figura 17, podemos visualizar o diagrama esquemático do funcionamento de um *encoder* óptico linear semelhante ao de uma impressora de jato de tinta, tendo uma fita de leitura óptica viajando no seu interior.

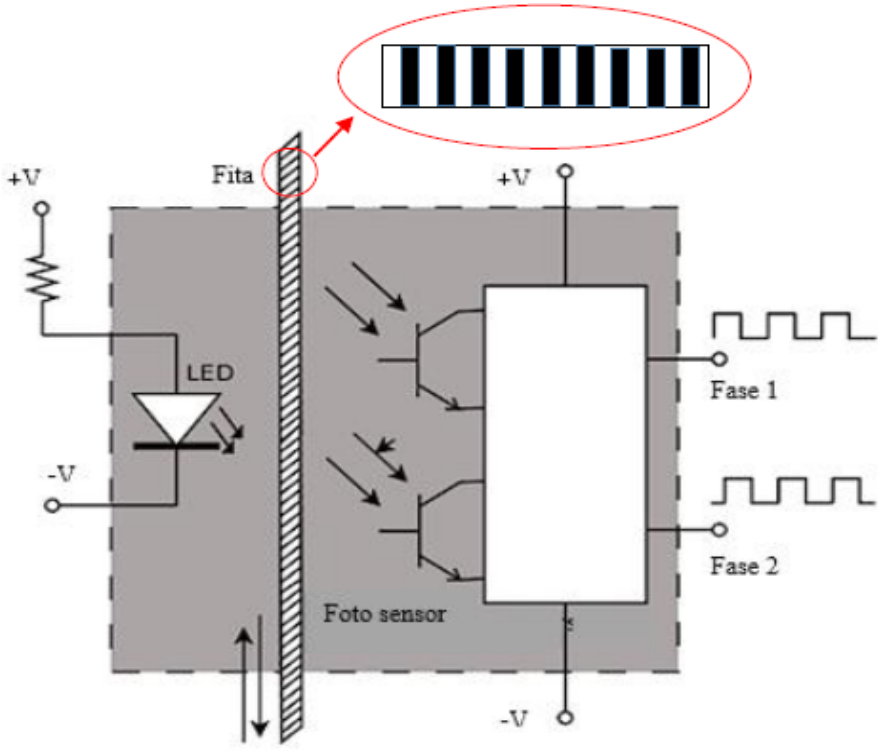

Figura 17: Diagrama de um encoder óptico linear, com uma fita de leitura óptica. Fonte (BALL, 2018)

4.1.1 Construção do sensor de posição

Na confecção do sensor de posição foi utilizado um *encoder* linear retirado de uma impressora jato de tinta semelhante a da figura 18.

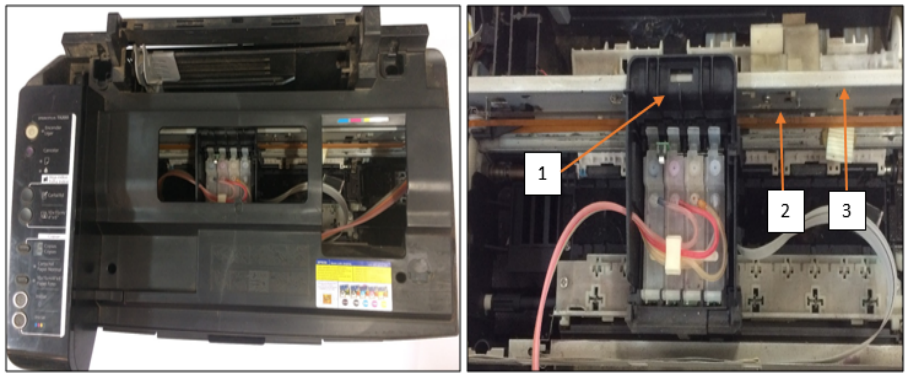

Figura 18: Impressora jato de tinta semi desmontada. Fonte: Elaborada pelo autor

1 —> (Carro do cabeçote de impressão)

2 —> (Fita óptica)

3 —> (Trilho do carro)

Primeiramente, desmontamos a impressora, e em seguida localizamos o *encoder*. Na figura 18 é possível visualizar o carro do cabeçote de impressão, o trilho no qual o mesmo viaja e a fita de código linear utilizada pelo *encoder*.

Este componente tem o seu funcionamento baseado no diagrama da figura 17. O mesmo fica alojado junto ao carro do cabeçote da impressão, que viaja sobre um trilho e tem o controle da sua posição de impressão determinada a partir das interrupções geradas a medida que este equipamento viaja sobre a fita de leitura. As fitas lidas pelos *encoders* são conhecidas como fitas de código linear, *codestrips*, tiras de *encoder*, entre outras (KRASNOW, 2018). Estas fitas funcionam como um "código de barras".

Utilizando um ferro de solda, retiramos com cuidado a placa do circuito que contém o *encoder*. As imagens 19 e 20 mostram o *encoder* e a fita de código linear retirado da impressora.

Com o *encoder* em mãos, devemos primeiramente realizar o mapeamento dos seus terminais e em seguida verificar se o mesmo está funcionando. Podemos verificar o funcionamento do *encoder*, realizando um teste de continuidade utilizando um multímetro digital. O mapeamento dos seus terminais pode ser um pouco mais trabalhoso.

Este componente contem 6 pinos, que chamaremos respectivamente de pinos (A, B, C, D, E, F). Teremos as seguintes ligações nos terminais:

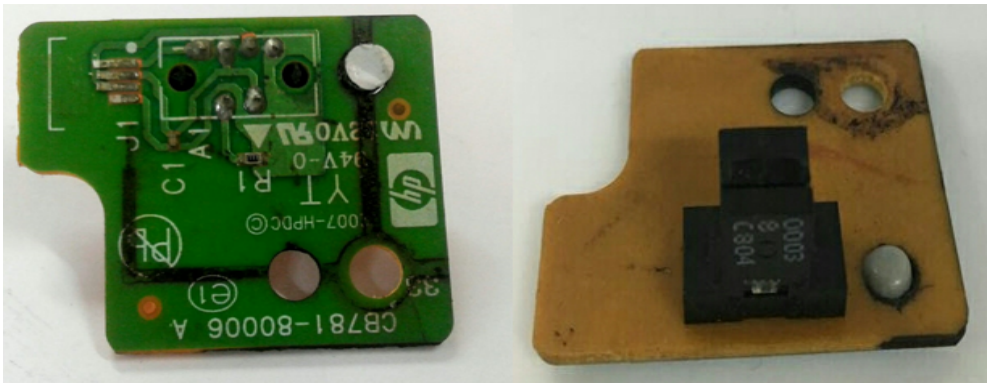

Figura 19: Encoder retirado da impressora. Fonte: Elaborada pelo autor

Figura 20: Fita de código linear. Fonte: Elaborada pelo autor

A —> (LED +) VCC do LED.

B —> (LED -) GND do LED.

C —> (-) GND.

D —> (OUTPUT 1) saída da fase 1.

E —> (VCC 5V) alimentação do encoder.

F —> (OUTPUT 2) saída da fase 2.

O circuito elétrico do sensor, deve conter dois resistores de 80 Ohms entre os seus terminais VCC e GND, e um capacitor de 330uf (BALL, 2018). A figura 21 exibe o diagrama do circuito elétrico utilizado na montagem do sensor.

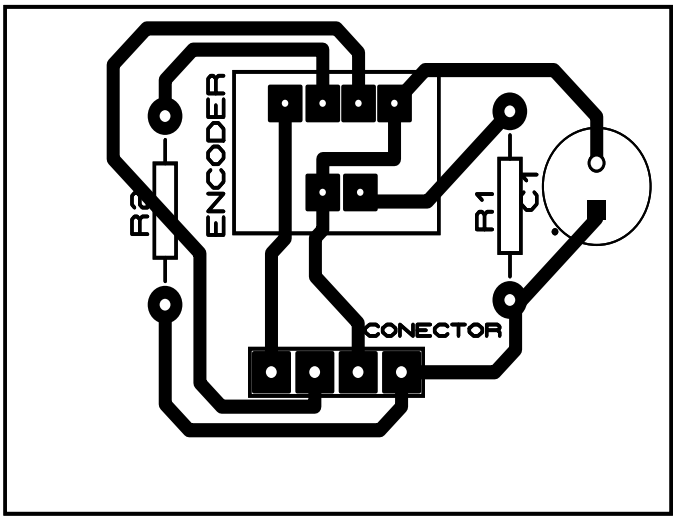

Figura 21: Diagrama do circuito elétrico utilizada na montagem do sensor de posição. Fonte: Elaborada pelo autor

O *encoder* fornece saídas de níveis lógicos que podem ser conectados aos pinos de interrupção do Arduino. A partir das interrupções geradas com a passagem da fita pelo *encoder* será gerado um sinal digital que terá os valores (0 ou 1) em cada fase do sensor, e se comporta como um pulso quadrado. Com o auxilio de um osciloscópio podemos visualizar o pulso gerado. Na figura 22 podemos observar a passagem da fita de código linear pelo *encoder*

Figura 22: Fita de código linear viajando pelo *encoder*. Fonte: Elaborada pelo autor

A figura 23 exibe o pulso quadrado gerado na tela do osciloscópio. Este pulso foi gerado a partir da passagem da fita pelo *encoder*. Repare que este sensor gera dois pulsos.

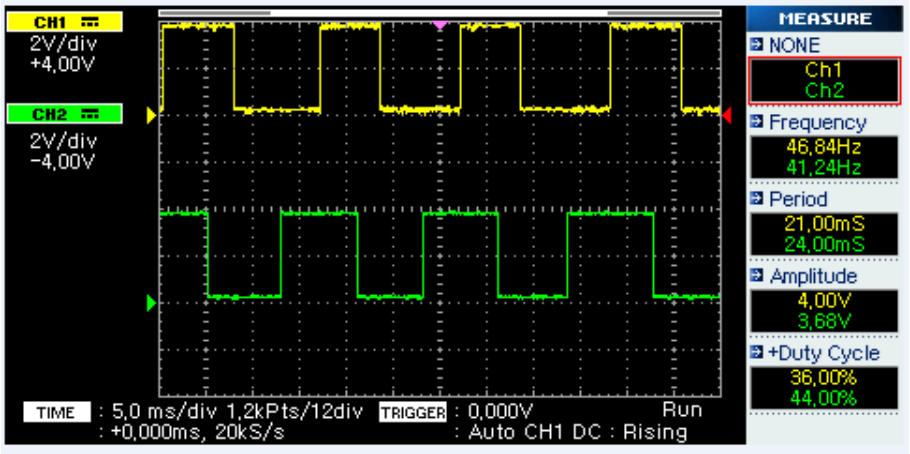

Figura 23: Pulso quadrado sendo gerado na tela do osciloscópio com a passagem de uma fita de código linear pelo encoder. Fonte: Elaborada pela altor

Para a confecção do nosso sensor de posição, foi utilizado uma placa de fibra de vidro cobreada apropriada para confecção deste tipo de circuito. Imprimimos o diagrama elétrico do circuito, e o transcrevemos para a placa cobreada virgem, com a ajuda de percloreto de ferro e com um fio fizemos a corrosão do cobre restando apenas o circuito impresso. Em seguida com um ferro de solda e os demais componentes em mãos montamos a placa. Assim o nosso sensor de movimento está pronto. Veja na figura 22 o sensor.

4.1.2 Calibração do sensor de posição

Na calibração do sensor de posição utilizou-se um relógio comparador de um dilatômetro linear, relógio este que dispõe de uma precisão de (0.01 mm) e é capaz de medir até dez voltas por centímetro deslocado.

Utilizou-se também um suporte de metal onde é possível fixarmos a fita de código linear de modo que a fita possa viajar pelo sensor enquanto o suporte é deslocado sobre o dilatômetro. Nas imagens 24 e 25 podemos visualizar o arranjo completo que foi utilizado na calibração do sensor.

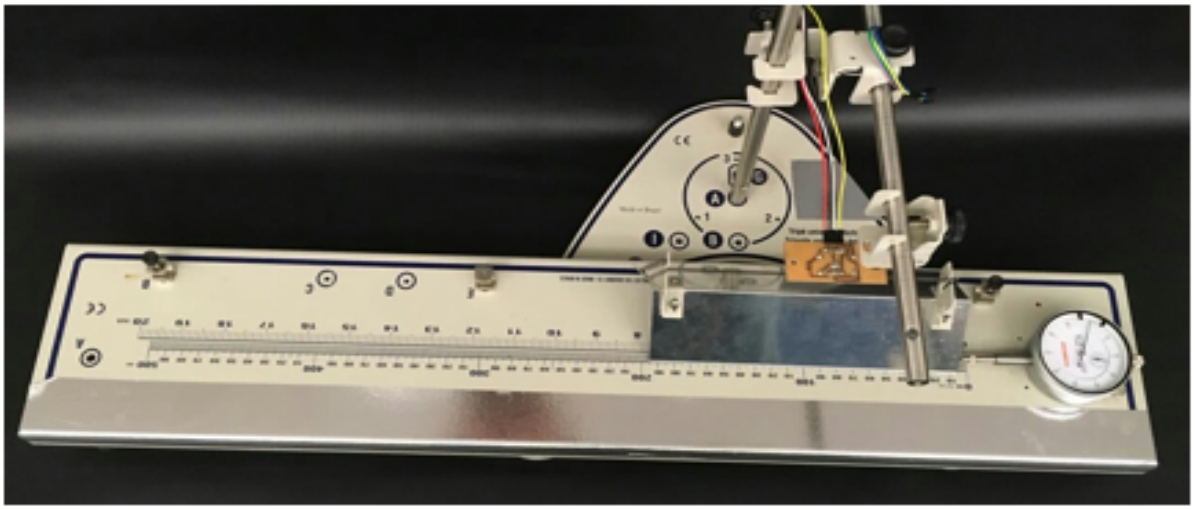

Figura 24: Tripe universal delta Max com sapatas niveladoras, sustentando o sensor de posição. Fonte: Elaborada pelo autor

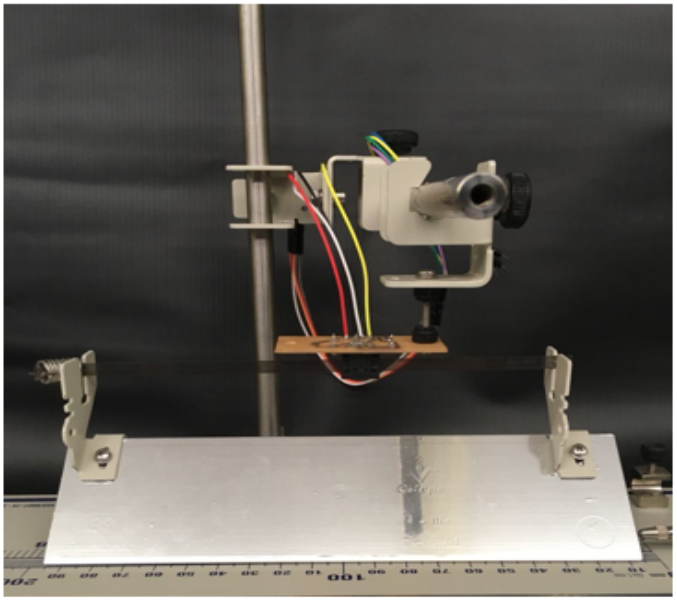

Figura 25: Tripe universal delta Max com sapatas niveladoras, sustentando o sensor de posição. Fonte: Elaborada pelo autor

Foram realizadas 10 medidas consecutivas de deslocamento do suporte com a fita, observamos então a quantidade de interrupções que são geradas a partir do deslocamento do suporte, que sofre um deslocamento de aproximadamente 10mm. Em seguida foi feito a média aritmética da quantidade de interrupções geradas por centímetro deslocado. Constatou-se que em média o sensor registra 12 interrupções a cada centímetro que o suporte é deslocado.

Sabendo que a cada centímetro deslocado são gerado 12 interrupções, realizamos uma regra de três simples e chegamos a um valor que corresponde à quantidade de centímetro deslocados a cada interrupção. Este valor é aproximadamente igual a (0.08333)

milímetros.

Essa constante será inserida no esboço utilizado pelo Arduino, permitindo que o mesmo faça a conversão do número de interrupções em um valor de deslocamento.

A seguir temos um modelo do código utilizado para a calibração do sensor.

```
1
//-------------------- Calibrando o sensor --------------------
3
 const byte encoderA = 2;          // fase 1 no pino digital 2
5 const byte encoderB = 3;          // fase 2 no pino digital 3
 long ValEncoder=0;                // valor do encoder
7 boolean readingA;                 // leitura booleana
 boolean readingB;
9 long imprime=0;
 volatile int encoder0Pos = 0;     // Variavel que trabalha com  //
   interrupcoes(attachInterrupt)
11
 void Interrupt_A(){
13  readingA = digitalRead (encoderA);
 }
15
 void Interrupt_B(){
17  readingB = digitalRead (encoderB);
    if (readingA != readingB){++ValEncoder;}
19    if (readingA == readingB){--ValEncoder;}
 }
21


23 void setup()
 {
25  Serial.begin(9600);


27
  pinMode(encoderA, INPUT);          // modo de entrada pin 2
29  pinMode(encoderA, INPUT_PULLUP); // necess rio para que os pinos n o
    fiquem
                                     //com valores flutuantes
31  pinMode(encoderB, INPUT);          // modo de entrada pin 3
  pinMode(encoderB, INPUT_PULLUP);
33
  attachInterrupt (digitalPinToInterrupt(2), Interrupt_A, CHANGE); // chama
35  //a interrupcao quando ocorre variacao no estado logico dos pinos 2 e 3

37  attachInterrupt (digitalPinToInterrupt(3), Interrupt_B, CHANGE);


39
 } //end setup
```

```
41
43 // Loop infinito
    void loop(){
45
    if(ValEncoder != imprime){
47  Serial.println(ValEncoder*0.08333333); // imprime o valor do deslocamento
      em mm
   //Serial.println(ValEncoder);        //imprime o numero de   interrupcoes
49   imprime=ValEncoder;               //imprime o valor do encoder
     }
51
   } //end loop
53
```

Este esboço será compilado para o Arduino permitindo que o mesmo possa realizar leituras detectando as interrupções geradas por mudanças de bordas (*CHANGE*), que serão geradas a partir da passagem da fita pelo sensor. Logo será possível detectá-las, contá-las, determinar o sentido em que estas ocorrerão e ainda calcular a distância percorrida pelo móvel em milímetros.

O esboço utilizado na calibração tem uma lógica de programação que baseia-se no diagrama da figura 17, ou seja, quando ocorrer o movimento da fita teremos um sinal gerado que indicará mudanças de estado nas fases 1 e 2 do sensor. As fases estarão conectadas as portas digitais (2 e 3) do Arduino.

Como o esboço utilizado faz uma comparação das duas fases para determinar o sentido de movimento e o valor do deslocamento, teremos quatro possibilidades de estado para as fases 1 e 2 do *encoder*.

No repouso o Arduino não faz nenhuma leitura, este fará leituras apenas se ocorrer alguma variação no estado das bordas. Se ocorrer alguma variação, teremos quatro possibilidades de estado, supondo que na primeira teremos respectivamente nas fases 1 e 2, os estados lógicos (1 e 0) ou (0 e 1), o movimento será interpretado como sentido positivo, o Arduino então zera o valor da medida e começa a incrementar valores a medida que a fita viaja pelo sensor, se o sentido do movimento for invertido é realizado o decremento destes valores.

Por outro lado se tivermos respectivamente nas fases 1 e 2, os estados (0 e 0) ou (1 e 1), o movimento será interpretado respectivamente como no sentido negativo e o processo inverso é realizado. Veja na figura 26 o diagrama que mostra a composição dos pulsos gerados com passagem da fita de código linear pelo *encoder*.

Veja na figura 26 que as fases 1 e 2 são comparadas simultaneamente, é baseado

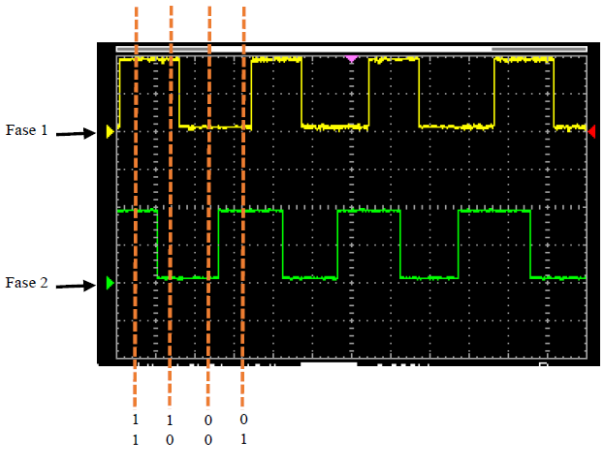

Figura 26: Diagrama esquemático dos pulsos gerados com passagem da fita de código linear pelo *encoder*. Fonte: Elaborada pelo autor

nesta comparação que as medidas de deslocamento são realizadas.

4.2 Montagem do experimento

O experimento proposto no presente trabalho, foi desenvolvido com a utilização dos seguintes componentes: 1 tripé universal delta máx com sapatas niveladoras, 1 placa Arduino mega, cabos conectores, 1 computador, 1 sensor de posição, 2 molas helicoidais, 1 suporte para molas, 1 fita de código linear, 1 fio de multifilamento, 1 proveta de vidro graduada 250ml com base hexagonal, detergente neutro, água, 1 nível de bolha frasco de vidro, 1 esfera de chumbo.

Para realizarmos os testes construiremos um oscilador harmônico amortecido, este por sua vez terá a posição da sua massa rastreada pelo sensor de movimento à medida que a fita viaja pelo sensor.

A massa do sistema será composta pelo conjunto: 2 molas, 1 esfera, 1 fio de multifilamento, suporte para molas e fita de código linear. Foi utilizado uma balança de precisão para determinar a massa do conjunto, e constatou-se que a massa total do conjunto é de aproximadamente (0,056 kg), sendo que cada mola tem uma massa de aproximadamente (0,075kg). Esta será a massa do nosso sistema.

O sistema será posto para oscilar respectivamente no ar, na aguá e no detergente e terá os valores de deslocamento da massa coletados pelo Arduino. Assim buscaremos obter um levantamento gráfico para cada situação. O sistema foi utilizado também pra analisarmos a queda da esfera na água e no Ar, assim como para determinarmos a constante elástica das molas.

4.2.1 Coleta dos dados

Inicialmente montamos a estrutura do sistema, de modo que o oscilador (no caso o sistema masa mola) permaneça com a oscilando orientada na vertical. Verificou-se inicialmente que a mola do sistema sofre uma pequena distensão referente a adição da massa, tendo assim uma nova posição de equilíbrio. Tomaremos esse novo ponto como a origem do nosso sistema. Veja o diagrama da figura 3.

No primeiro teste de oscilação do sistema a massa foi posta para oscilar em interação com o ar. A massa foi manualmente deslocada para baixo aproximadamente (0.05m) da sua posição de equilíbrio e largada para oscilar. Neste e nos demais testes utilizamos um deslocamento inicial de (0,05m), tivemos o máximo de cuidado na montagem e execução do sistema para que o atrito da fita com o sensor fosse o mínimo. Na figura 27 é possível visualizarmos o arranjo utilizado nos testes. Com este arranjo esperemos descrever graficamente o comportamento do oscilador harmônico amortecido.

Figura 27: Tripe universal delta Max com sapatas niveladoras, utilizado como suporte para o oscilador amortecido. Fonte: Elaborado pelo altor.

No segundo teste posicionamos a proveta com água fluido com densidade e viscosidade maior que a do ar, de modo que a massa do sistema fique imersa completamente na aguá. A figura 28 mostra o arranjo utilizado neste teste.

Figura 28: Tripe com sapatas niveladoras, sustentando o oscilador amortecido, com a sua massa emersa no detergente. Fonte: Elaboração pelo altor

No terceiro teste posicionamos a proveta com detergente, um fluido com densidade e viscosidade maior que o ar e a água, de modo que a massa do sistema fique imersa completamente no detergente. Veja na figura 29 o arranjo utilizado neste teste.

Figura 29: Tripe com sapatas niveladoras, sustentando o oscilador amortecido, com a sua massa emersa no detergente. Fonde do altor

Assim finalizamos os testes com o oscilador.

Nos testes de queda livre e na água utilizamos o sistema anterior com algumas alterações, retiramos as molas e o suporte para molas e colocamos o fio de multifilamento em seu lugar a fim de prende-lo á fita de código linear e como a esfera dispõe de um

anel para ser pendurada, prendemos esta á fita. Com esse procedimento pretendemos esperamos evitar o choque da esfera com o fundo da proveta, o que poderia quebra-la.. Com os dados coletados neste teste pretendemos determinar o valor da aceleração da gravidade. Veja na figura 30 o arranjo utilizado no teste de queda livre.

Figura 30: Arranjo utilizado no teste de queda na água. Fonte: elaborada pelo altor

Logo em seguida na figura 31 temos o arranjo utilizado no teste de queda na água.

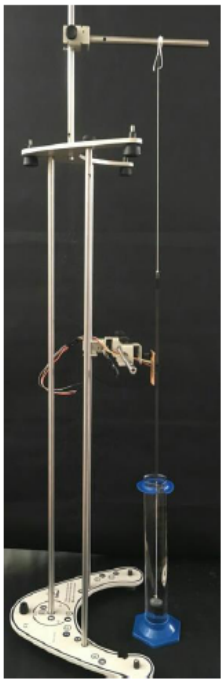

Figura 31: Arranjo utilizado no teste de queda na água. Fonte: elaborada pelo altor

A partir dos resultados do teste de queda na água esperamos obter uma relação para a força de retardo que atua sobre um corpo que se movimenta imerso em um fluido, e com esta relação vem a contribuição para o amortecimento de um oscilador como o que utilizamos nos testes. Assim finalizamos a nossa coleta de dados para o levantamento gráfico do nosso estudo.

Em todos os testes de coleta de dados o Arduino executou o esboço que utiliza uma estrutura de matriz para armazenar os valores de leitura da posição da massa do sistema em sua própria memória RAM. Essa matriz contem 2955 linhas e 1 coluna, ou seja, ela será capaz de armazenar até 2955 valores. Fez-se uso também do *TIMER2* e seus registradores para garantir que a cada (1,024ms) ocorresse o seu estouro.

A utilização do *TIMER* definiu a nossa base de tempo durante a coleta dos dados. Com este procedimento armazenamos todos os valores de leitura na memória RAM do Arduino e em seguida estes valores forão impressos de uma única vez no monitor serial via porta USB. Como o Arduino mega conta com uma memoria RAM de 8KB, temos espaço suficiente para armazenarmos um número razoável de dados (MCROBERTS, 2011).

Como já vimos os registradores do *TIMER2* podem ser configurados de modo que o seu estouro ocorra no tempo desejado. Dentro do código devemos ajustar o valor da variável tempo de acordo com o tempo que desejamos realizar uma leitura. A conta é simples, o tempo de leitura em segundos vai ser igual a (1,024ms/1000)x(2955).

Essa foi uma maneira que achamos conveniente para evitar prejuízos de tempo com comunicação serial e execução das funções de temporização como *delay*, *millis* dentro do *loop* do esboço. A seguir temos o esboço que foi utilizado para realizarmos as leituras de posição em função do tempo. A seguir podemos observar a estrutura do esboço utilizado na coleta dos dados.

```
1  //Este c digo faz a leitura das portas digitais 2 e 3 do Arduino a cada
       //1,024ms (cuja a base tempo    dada pelo timer) e carrega estes  //
       valores (leitura digital) em uma matriz coluna.
   // ---------------------------------------------------------------
3   const byte encoderA = 2;          // Fase 1 do sensor no pino digital 2
    const byte encoderB = 3;          // Fase 2 do sensor no pino digital 3
5   long ValEncoder=0;                // valor da leitura encoder
    boolean readingA;                 // leitura A (pin 2)
7   boolean readingB;                 // leitura B (pin 3)
    long imprime=0;                   // valor de impressao
9   int valores[2500 ] = {};          // matriz com 2500 posicoes
    int i=0;                          // variavel generica
11  int tempo=0;                      // tempo de coleta de dados

13 // ---------------------------------------------------------------
   void Interrupt_A(){
15   readingA = digitalRead (encoderA); // LeituraA igual ao valor do pino 2
```

```
 }

void Interrupt_B(){
  readingB = digitalRead (encoderB); // LeituraB igual ao valor do pino 3
    if (readingA != readingB){++ValEncoder;} // se a leitura A for difernte
    // da B ocorre um incremente dos valores
    if (readingA == readingB){--ValEncoder;} // se a leitura A for igual
    // a B ocorre um decremento dos valores
 }
// --- Rotina de Interrup  o do timer -----

 ISR(TIMER2_OVF_vect)
 {
    TCNT2=240;  // reinicializa o registrador do Timer2 a cada ms
    ++tempo;    // incremento

 if (tempo==1) // 2950 interrup  es do timer = 1,024ms
 {
  tempo=0;            // vari vel que controla o tempo de leitura
  if(ValEncoder!=0) { // leitura diferente de zero inicia-se a leitura

  valores[i] =  ValEncoder; // a cada interrupcao do Timer(aproximadamente
   // 1ms) faz a leitura do valor e carrega em um ponto da matriz

  i++; // incremento dos valores de leitura
   }
  }
 }

  // Estouro = Timer2_cont x prescaler x ciclo de m quina

  // Ciclo de m quina = 1/Fosc = 1/16E6 = 62,5ns = 62,5E-9s

  // Estouro = (256 - 100) x 1024 x 62,5E-9 = 9,98ms


 // ------------------------------------------------------
void setup(){
Serial.begin(250000);    // inicializa  o da comunica  o serial
    TCCR2A = 0x00;       // Timer operando em modo normal
    TCCR2B = 0x07;       // prescaler 1:1024
    TCNT2  = 240;        // valor carregado no timer para que a
                         // interrup  o ocorra a cada 1 ms overflow again
    TIMSK2 = 0x01;       // habilita interrup  o do Timer2

  pinMode(encoderA, INPUT);         // modo de entrada pin 2
  pinMode(encoderA, INPUT_PULLUP); // evita que os pinos tenham valores
```

```
                              //flutuantes
  pinMode(encoderB, INPUT);          // modo de entrada pin 3
61  pinMode(encoderB, INPUT_PULLUP);

63 attachInterrupt(digitalPinToInterrupt(2), Interrupt_A, CHANGE); // chama
    //uma interrup  o quando ocorre uma variacao no estado logico(HIGH ou
    //LOW) do pino (2)
  attachInterrupt(digitalPinToInterrupt(3), Interrupt_B, CHANGE);
65

  }
67 // ------------------------------------------------------------

69 void loop(){
  int imprimir=0;           // comeca imprimindo 0
71

   if (i>2955){             // ler os valores de 1 a 2500
73  TIMSK2 = 0x00;           // desabilita interrupcao do Timer2
   for (int a=0; a<=2954; a++) { // ler os 2955 valores armazenados
75  imprimir = valores[a]; // imprime valores armazenados na matriz
   Serial.println(imprimir*0.084333); // imprime valor deslocado em (mm)
77      }
        while(1)             // prende o loop nesta execucao
79      }
}
81
```

5 RESULTADOS E DISCUSSÕES

Neste capítulo inicialmente faremos uma análise e uma discussão abordando os possíveis atrasos por comunicação serial e erros de temporização no que diz respeito a execução de um esboço pelo Arduino. Prosseguindo com uma análise das características do sistema estudado, o que inclui a medição do valor da gravidade, valor da constante elástica e período de oscilação da mola utilizada, trataremos ainda do comportamento gráfico de cada situação, discutiremos a perda de amplitude de oscilação do sistema assim como sua taxa de perda de energia e que tipo de força pode está atuando para que estas oscilações sejam amortecidas.

5.1 Atrasos por comunicação serial e execução do loop

Ao se trabalhar com circuitos microcontroladores no desenvolvimento de quais quer projetos, devemos tomar alguns cuidados básicos para que tenhamos uma base de tempo eficiente pois ao utilizarmos funções de temporização tais como *delay*, *millis* ou *micros* dentro dos códigos podemos acabar tendo uma perda de tempo ao executá-las, pois o processador leva um certo tempo para processar e executar tais funções. Se existir a perda de tempo, deve-se levar em conta a quantidade relacionada com essa perda, pois a qualidade dos dados coletados disso.

O Arduino utiliza sensores para realizar leituras do mundo externo, essas informações podem ser processadas e enviadas por exemplo para serem armazenadas em uma planilha do Microsoft exel, cartão micro SD entre outros. Esse tráfego de informações é realizado via comunicação serial utilizando uma porta USB. Acreditamos que durante todo esse processo estejamos sujeitos a atrasos na execução dos esboços enquanto o mesmo esteja realizando leituras, processando-as e as enviando simultaneamente.

A fim de analisarmos o quanto os pontos citados anteriormente podem afetar ou não a qualidade das informações coletadas, realizaremos alguns testes onde executaremos um esboço com a funções de temporização *delay* e outro com a função *millis*. Testaremos também como o mesmo se comporta, uma vez que esse esboço é executado e temos a comunicação do Arduino com uma porta serial.

5.1.1 Utilizando a função *delay*

Com o auxilio de um osciloscópio iremos registrar os resultados em imagens e alisar se de fato ocorre algum atraso durante a execução destes códigos, seja pela fato do Arduino está fazendo comunicação com uma parta serial ou pelo fato de está usando funções de temporização no seu loop.

No primeiro testes utilizamos um esboço que tem a função de gerar pulsos quadrados na porta digital 7, ler os valores da porta analógica 5 e escrevê-los no monitor serial. A seguir podemos observar o esboço utilizado.

```
long pino = 5;  // vari vel que faz referencia ao pino 5 Arduino
int linha = 0;  // vari vel que    inicializada em 0
int LABEL = 1;  // vari vel que    insere um r tulo inicializado em 1
int valor = 0;  // vari vel que    inicializada em 0

void setup() {
 Serial.begin(250000); // taxa de comunica  o serial
 Serial.println("LABEL,Hora,valor,linha");
 }
void loop() {
  digitalWrite(7, !digitalRead(7)); // inverte os valores da porta digital
   7
  valor = digitalRead(pino); // valor da porta digital 5
  linha++; // incrementa os valores da porta 5 linha ap s linha pr xima
   linha
  delay(1); // tempo de 1ms
  Serial.print("DATA,TIME,"); // escreve os valores de data e hora
  Serial.print(valor); // escreve os valores da porta 5
  Serial.print(","); // separa os valores por virgula
  Serial.println(linha); // imprime os valores da porta 5 no monitor serial
 }
```

Nos testes posteriores foram alterados apenas os valores da taxa da comunicação serial, utilizamos o menor e o maior valor possível de comunicação, que é (9600 e 250000).

Com o esboço acima em execução esperamos obter um pulso quadrado com período exatamente igual a 2ms, já que o seu *delay* é de 1ms, ou seja, ele inverte o estado do pino 7 a cada 1ms.

Na figura 32 temos os valores do período da onda gerada no teste em que foi utilizado um *delay* de (1ms) e uma taxa de comunicação serial de 250000. Observe que tivemos um erro de (1,04ms) no valor do período esperado .

Já no segundo teste utilizamos uma taxa de comunicação de 9600 para o mesmo esboço. Repare na figura 33 que o erro é ainda maior chagando a (35ms) de atraso.

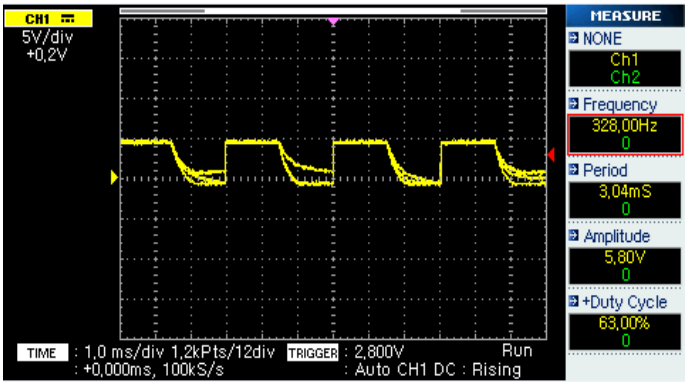

Figura 32: Pulso quadrado gerado, com um *delay* de 1ms e uma taxa de comunicação serial de 250000. Fonte: Elaborada pelo autor

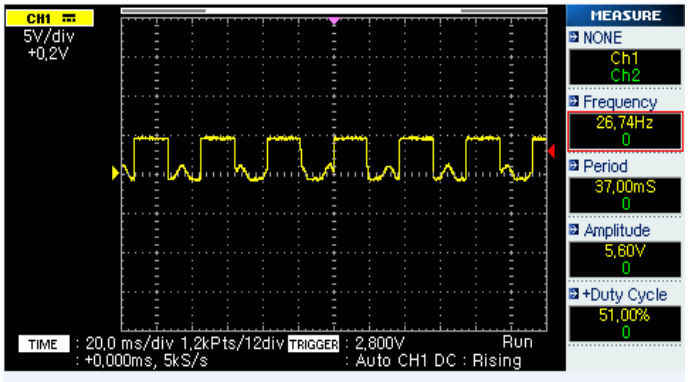

Figura 33: Pulso quadrado gerado, com um *delay* de 1ms e uma taxa de comunicação serial 9600. Fonte: Elaborada pelo autor

As figuras 34 e 35 mostram os resultados dos testes onde foi utilizado um *delay* de 10ms no mesmo esboço e uma taxa de comunicação serial de 250000 e 9600, respectivamente.

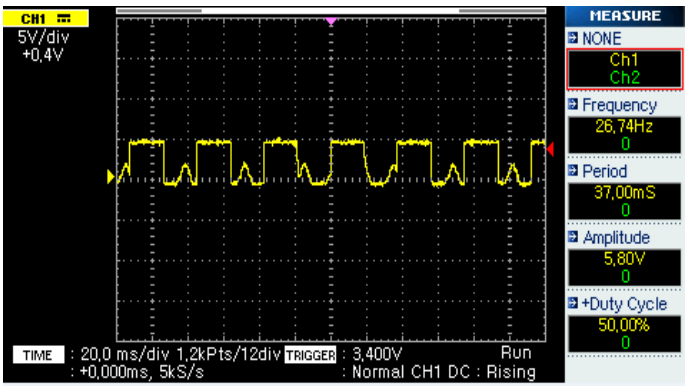

Figura 34: Pulso quadrado gerado, com um *delay* de 10ms e uma taxa de comunicação serial de 9600. Fonte: Elaborada pelo autor

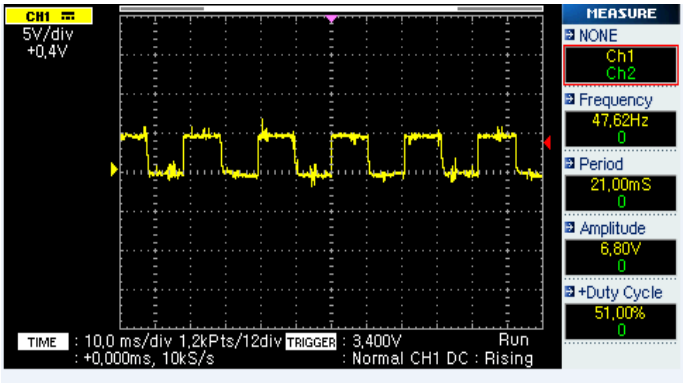

Figura 35: Pulso quadrado gerado, com um *delay* de 10ms e uma taxa de comunicação serial de 250000. Fonte: Elaborada pelo autor

Analisando as figuras 34 e 35 percebemos que à medida que o intervalo de tempo de execução do *loop* aumenta o atraso diminui.

E para finalizarmos, veja na figura 36 o pulso gerado ao executarmos o *loop* somente para inverter o estado do pino 7.

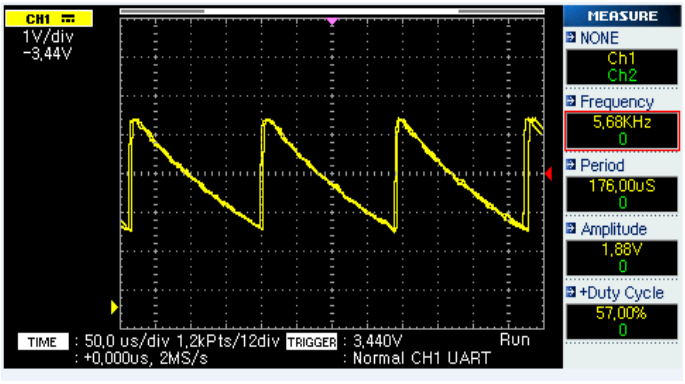

Figura 36: Pulso gerado apenas com a execução do *loop* . Fonte: Elaborada pelo autor

Podemos observar na figura 36 que a execução do *loop* leva um tempo.

5.1.2 Utilizando a função *millis*

No exemplo seguinte foi utilizado um esboço exemplo que faz uso da função *millis* em seu *loop*, esse esboço foi utilizado pra gerarmos um pulso quadrado, seu período de oscilação pode ser alterado modificando o intervalo de tempo deste esboço. Logo em seguida temos as imagens que mostram os resultados observados.

```
1 long pino = 5;                // pino anal gico 5
  int linha = 0;                // variavel linha
3 int LABEL = 1;                // R tulo
  int valor = 0;                // valor do pino
5 const int ledPin =  13;       // inverti o estado da porta cada estouro do
    //TIMER
```

```
int ledState = LOW;             // estado da porta
unsigned long previousMillis = 0; // armazenar o tempo que a porta  //
  inverteu seu estado pela ultima vez
const long interval = 1 // intervalo de tempo em (milissegundos)


void setup() {

Serial.begin(9600);             // taxa de comunica  o serial
Serial.println("CLEARDATA"); // limpa as linhas da planilha do Plx-DAQ
Serial.println("LABEL,Hora,valor,linha"); // escreve nas primeiras //
  linhas
//do Plx-DAQ("LABEL,Hora,valor,linha")
pinMode(ledPin, OUTPUT);  // modo do pino

}

void loop() {

  valor = digitalRead(pino); // variavel valor igual ao valor do pino 5
  linha++;// incremento de valor linha apos linha
  unsigned long currentMillis = millis();//tempo em milissegundos

  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;

    if (ledState == LOW) { // condicao do estado da porta digital 13
      ledState = HIGH;
    } else {
      ledState = LOW;
    }
   digitalWrite(ledPin, ledState); // define o estado da variavel
   Serial.print("DATA,TIME,");    // escreve o data e tempo
   Serial.println(valor);         // escreve o valor da porta 5
   Serial.print(",");             // separa por linha
   Serial.println(linha);         // imprime o valor na linha
  }

}
```

Na figura 37 temos o resultado do primeiro teste com o esboço que utiliza a função *millis*, onde utilizamos um tempo de 1ms e uma taxa de comunicação serial de 9600. De acordo com o esboço utilizado neste teste, esperávamos obter um pulso quadrado como

um período de 2ms, no entanto obtivemos um pulso com período de 37ms, ou seja, 35 vezes maior.

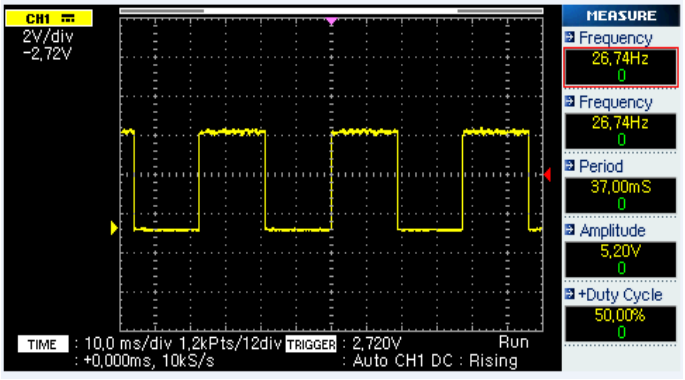

Figura 37: Pulso quadrado gerado com um intervalo de tempo de 1ms e uma taxa comunicação serial de 9600. Fonte: Elaborada pelo autor

Na figura 38 temos o resultado do segundo teste com o esboço que utiliza a função *millis*, onde utilizamos um tempo de 1ms e agora uma taxa de comunicação serial de 250000.

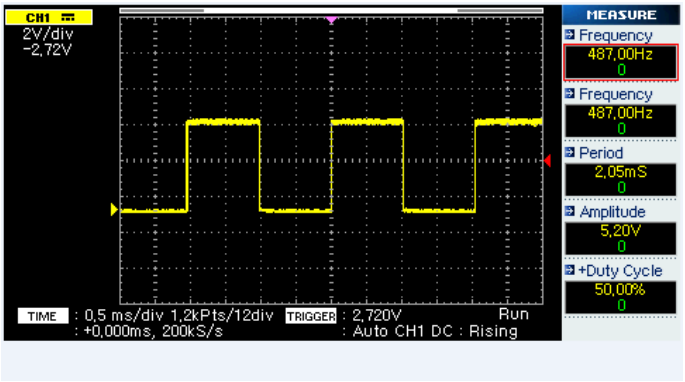

Figura 38: Pulso quadrado gerado com um intervalo de tempo de de 1ms e uma taxa comunicação serial de 250000. Fonte: Elaborada pelo autor

Podemos observar na figura 38 que o período encontrado é de 2,05ms, valor bem próximo do esperado, porem não exato.

Portanto, de acordo com os resultados obtidos nos testes, concluímos que de fato temos prejuízo de tempo durante a comunicação serial do Arduino e que as funções de temporização testadas *delay* e *millis* levam um determinado tempo para serem executadas dentro do *loop*, assim como o próprio *loop* também leva um tempo para ser executado, mesmo que sem nenhuma função de temporização inserida neste.

Fica claro que de fato estamos sujeitos a erros de temporização quando executamos um esboço que contenha funções de temporização no seu *loop*. É bom ressaltar que esse tempo é pequeno e dependendo do projeto que se pretenda desenvolver talvez isso não

represente um problema, porém em projetos que requeiram uma base de tempo precisa devemos buscar outra alternativa.

Por esse motivo utilizamos os *TIMER2* do Arduino para definirmos a nossa base de tempo e optamos por armazenar os nossos dados na memória RAM do Arduino durante a coleta dos dados.

5.2 Análise gráfica dos dados obtidos no teste de queda livre

Faremos uma análise gráfica dos dados obtidos com os testes a fim de chegarmos a alguma conclusão no que diz respeito ao comportamento de um oscilador harmônico amortecido.

Começamos com a análise do gráfico de queda livre da esfera, a fim de determinarmos o valor da gravidade local. Neste teste utilizamos o mesmo esboço utilizado na coleta dos dados de oscilação da esfera no ar, água e detergente.

A esfera foi inicialmente abandonada de uma altura de $(0,35m)$ e em seguida teve a sua posição rastreada pelo sensor de posição conectado ao Arduino. A taxa de coleta de dados utilizada foi de $1,024ms$ ou seja, os valores da posição da esfera são coletados neste intervalo de tempo. Veja na figura 39 o gráfico de queda livre da esfera.

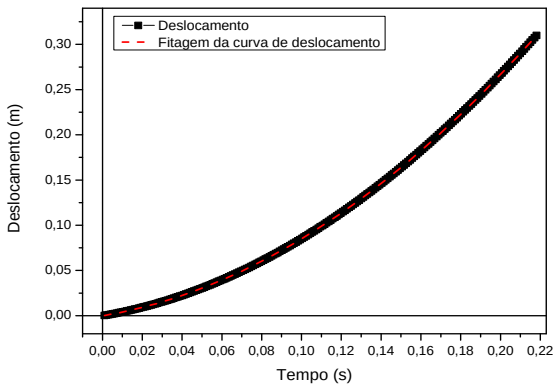

Figura 39: Representação gráfica do movimento de queda livre da esfera. Fonte: Elaborada pelo altor

Com o *fitting* da curva do gráfico de queda livre obtivemos a tabela da figura 39. De acordo com a tabela 40 constatamos que a aceleração da gravidade medida é de aproximadamente $9,7m/s^2$, valor muito próximo do medido a nível do mar e à latitude de 45 que é de aproximadamente $9,8m/s^2$. Este foi o valor que obtivemos com a utilização do nosso sistema de coleta de dados.

	A	B	C	D
1	Equation	y = Intercept + B1*x^1 + B2*x^2		
2	Weight	No Weighting		
3	Residual Sum of Squares	3,73083E-7		
4	Adj. R-Square	1		
5			Value	Standard Error
6	Deslocamento	Intercept	-1,20397E-4	8,7461E-6
7		B1	0,3688	1,84291E-4
8		B2	4,82601	8,14525E-4

Figura 40: Representação gráfica do movimento de queda livre da esfera. Fonte: Elaborada pelo altor

Veja que se:

$$Y = Y_0 + V_0 * t + \frac{g * t^2}{2} \tag{5.1}$$

descrever a queda livre de um corpo, por comparação podemos concluir da tabela na figura 40 que a aceleração da gravidade é dada por:

$$g = 2 * B2 \cong 9,7m/s^2 \tag{5.2}$$

Esse valor foi utilizado no cálculo da constante elástica das molas e do período natural de oscilação.

Para calcularmos as contantes elásticas das molas utilizamos o arranjo da figura 27, onde foi possível medir a distensão das molas, uma vez que adicionamos a massa do sistema nas mesmas.

No caso da mola 1, constatou-se que esta sofre um distensão de aproximadamente $0,237m$. De acordo com a lei de *Hook* $F = kx$ basta isolarmos k na expressão para encontramos a contante elástica da mola do sistema. Como já conhecemos os valores da massa do sistema, da gravidade e do deslocamento é possível calcularmos k. Logo o cálculo da contante elástica da mola fica da seguinte forma:

$$k_1 = F/x = mg/x = (0,0485).(9,7)/(0,237) = 1,98N/m \tag{5.3}$$

levando em consideração a massa da mola.

Já a mola 2, sofrem um distensão de aproximadamente $0,248m$. Assim temos que a sua constante elástica é de:

$$k_2 = F/x = mg/x = (0,0485).(9,7)/(0,248) = 1,89N/m \tag{5.4}$$

Como o sistema utilizou molas em paralelo, temos uma contante k_{eq} equivalente que pode ser expressa da seguinte forma:

$$k_{eq} = k_1 + k_2 = 1,89 + 1,98 = 3,87N/m \tag{5.5}$$

A frequência natural de oscilação do sistema deve portanto ser:

$$w_0 = \frac{1}{2\pi}\sqrt{k/m} = 1,32Hz \tag{5.6}$$

já o seu período de oscilação deve ser de aproximadamente:

$$T = 1/1,32 \cong 80s \tag{5.7}$$

5.3 Análise gráfica dos dados obtidos no teste de queda na água

A partir dos resultados obtidos com os testes de queda da esfera na água buscaremos associar a força de retardo que atua na esfera em interação com a água, a perda de energia e amplitude do sistema. Realizamos quatro testes de queda da esfera na água buscando sempre as mesmas condições iniciais em ambos os casos, com isso podemos observar que os resultados obtidos em todos estes testes tendem para um mesmo valor. Veja na figura 41 como as curvas obtidas nos testes se comportam.

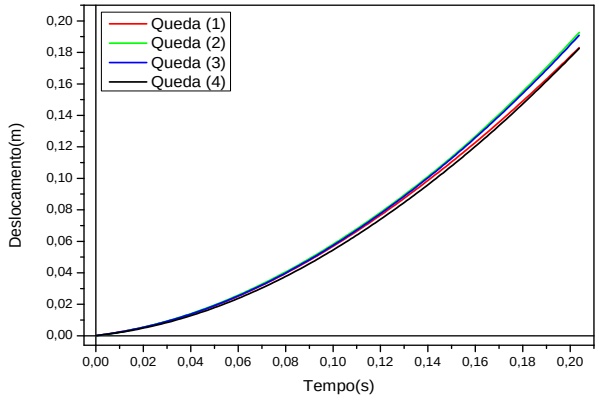

Figura 41: Representação gráfica do movimento de queda da esfera na água.

Repara na figura 41 que todas as curvas comportam-se de maneira semelhante, o que nos leva a crê que a força de retardo existe e tendem para o mesmo valor.

De acordo com a solução da equação 2.3 obtemos a equação 2.6 que representa o deslocamento de uma partícula em interação com um fluido. A partir da equação 2.6 obtemos o *fitting* da curva obtida segundo o teste de queda na água, assim obtivemos um valor para a constatante k, onde $-kmv$ representa a força de retardo que atua na esfera.

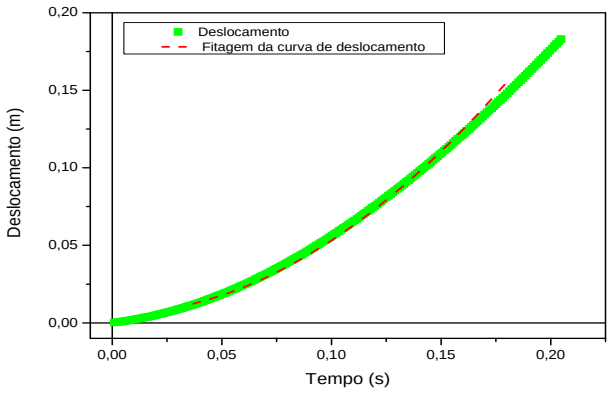

Figura 42: Representação gráfica do movimento de queda da esfera na água.

Veja a figura 42 onde temos a representação da segunda curva de queda da esfera na água sendo fitada.

Já a tabela 43 traz o volar da constante k. Constante essa que é utilizada no cálculo da força de retardo que um fluido (neste caso a água) exerce sobre um corpo em queda.

	A1(Y)	A2(Y)	A3(Y)	A4(Y)
1	Model	NewFunction (User)		
2	Equation	-g*((1- exp(-K*x)/K^2) - x/K)		
3	Reduced Chi-Sqr	9,46444E-6		
4	Adj. R-Square	0,99466		
5			Value	Standard Error
6	Deslocamento	K	0,99971	1,31192E-5

Figura 43: Tabela usada no calculo da constante k.

Como em todos os teste as curvas obtidas tendem a ter o mesmo comportamento, concluímos que o valor de K converge para um mesmo valor. Assim temos um valor para a constante de amortecimento do fluido K.

5.4 Análise gráfica dos testes de oscilação na água, ar e detergente

Começamos com a análise dos gráficos que exibem o comportamento do oscilador amortecido no teste em que a sua massa é posta para oscilar em interação com o ar. Na sequência analisaremos os casos em que a massa do sistema oscila na água e no detergente.

Nas figuras 44 e 45 podemos observar a representação gráfica do movimento harmônico amortecido descrito pela esfera quando esta oscila no ar. Veja na figura 27 que exibe o arranjo utilizado para realizarmos o teste de oscilação no ar.

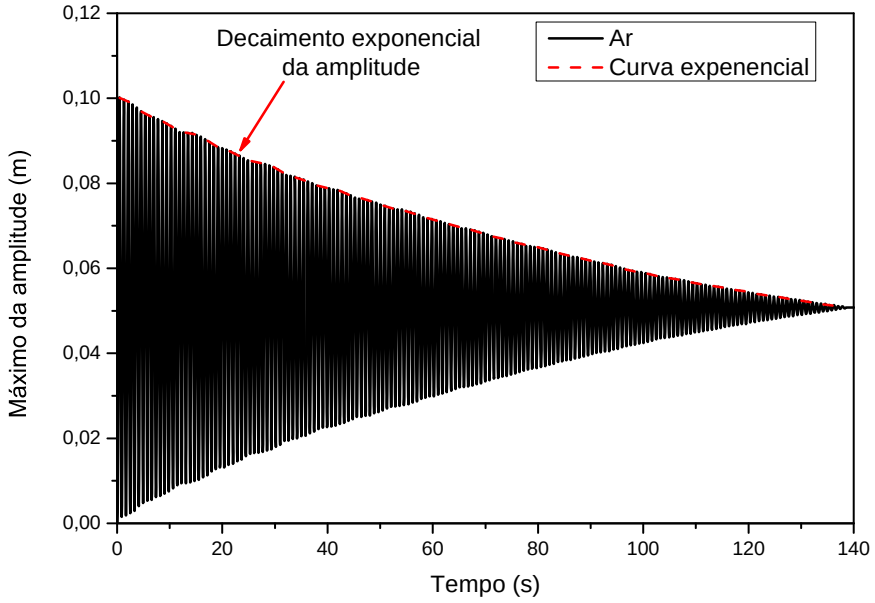

Figura 44: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa no ar.

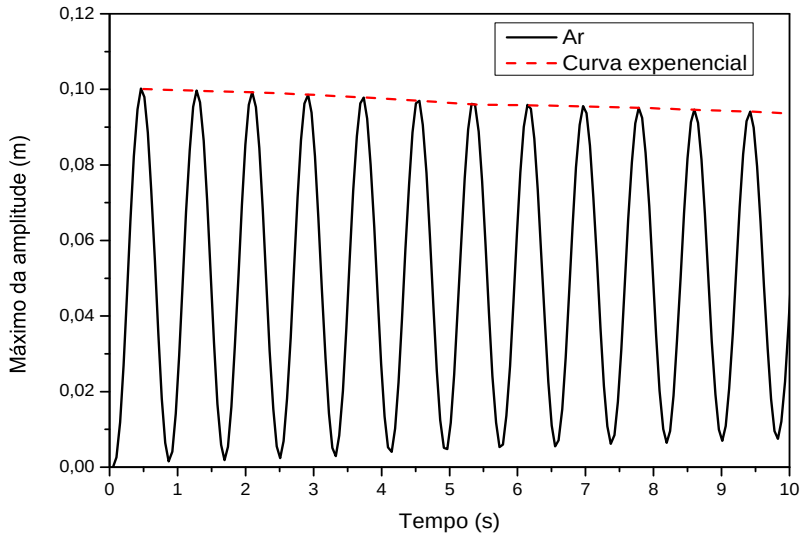

Figura 45: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa no ar, durante os 10 primeiros segundos de oscilação.

Observarmos no primeiro gráfico um decaimento exponencial da amplitude de oscilação do sistema, até que este atinja a posição de equilíbrio novamente a medida que nos aproximamos de um tempo finito. Este fenômeno é atribuído a atuação das forças dissipativas sobre o oscilador. Embora os atritos refentes ao ar e à fita de leita representem valores bem pequenos, ainda sim, contribuem significativamente para a perda de energia do sistema. A partir desta análise concluímos que estando o oscilador sujeito a interações externas, este sofre uma perda de energia, o que leva a uma diminuição da sua amplitude

de oscilação a medida que o tempo passa. Verificamos ainda que o período de oscilação medido está bem próximo do valor teórico esperado. De acordo com a figura 46 o período de oscilação medido é de aproximadamente $0,82s$ enquanto que o valor esperado é de aproximadamente $0,80s$, vale ressaltar que neste cálculo utilizamos o valor da gravidade medida, $9,7m/s^2$.

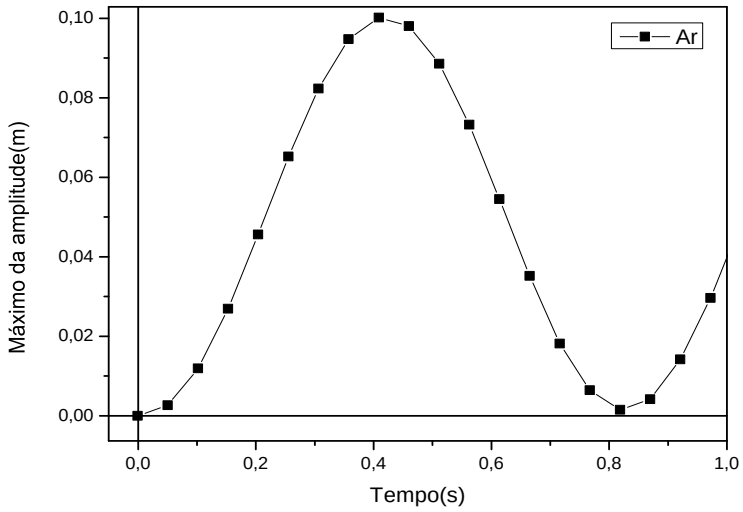

Figura 46: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa no ar no seu primeiro segundo de oscilação.

Nas figuras 47 e 48 podemos observar a representação gráfica do movimento oscilatório amortecido descrito pelo sistema quando este é posto para oscilar na água. Aqui também é possível observamos uma queda de amplitude que é maior se comparada com com o caso da figura 44, e consequentemente neste caso a perda de energia será também maior.

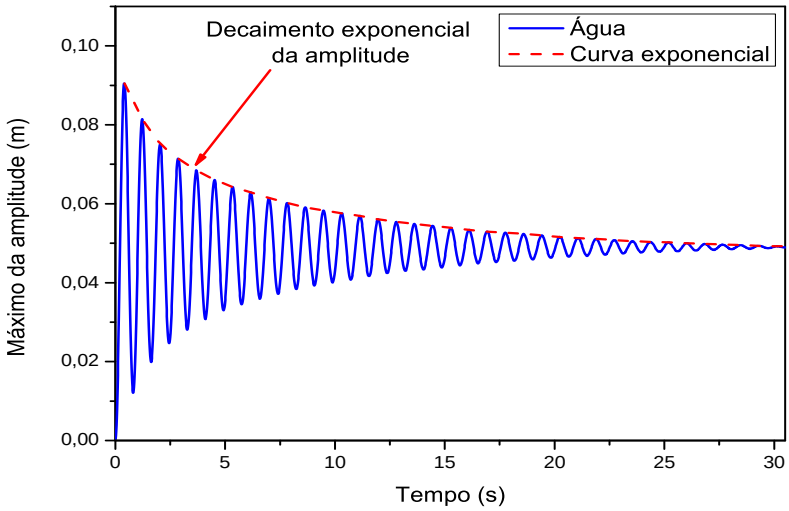

Figura 47: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa na água.

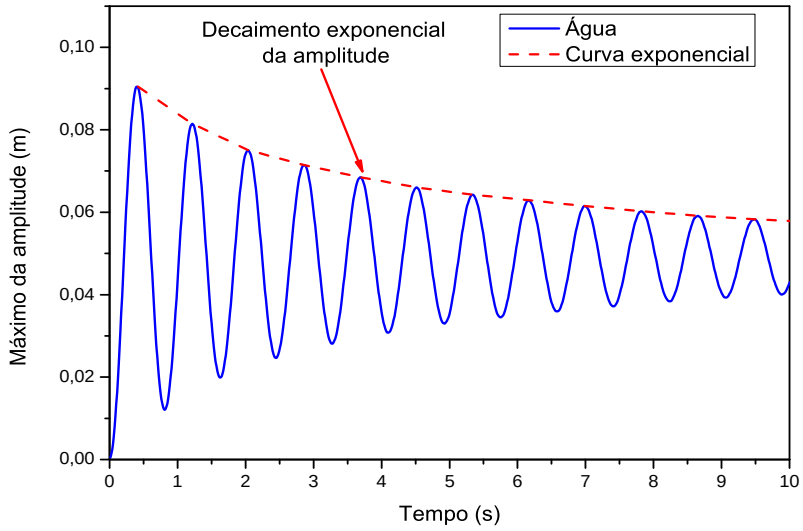

Figura 48: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa na água, durante os 10 primeiros segundo de oscilação.

Os resultados deste teste mostram que a queda de amplitude é bem mais significativa que no caso da figura 47.

Baseado nos resultados obtidos com os testes de queda da esfera na água, acreditamos que a perda de energia do sistema esteja associado as características físicas de cada fluido, como densidade e viscosidade, ou seja, a interação do sistema com um fluido de viscosidade alta vai resultar numa perda de energia mais rapidamente do que se o mesmo estiver interagindo com um fluido menos viscoso. Com tudo propomos que este fator de amortecimento depende das características de cada fluido.

Na figura 49 podemos observar a representação gráfica do movimento descrito pela massa do sistema quando esta é posta para oscilar em um fluido como o detergente, sendo que este fluido é mais viscoso e denso que a água e o ar.

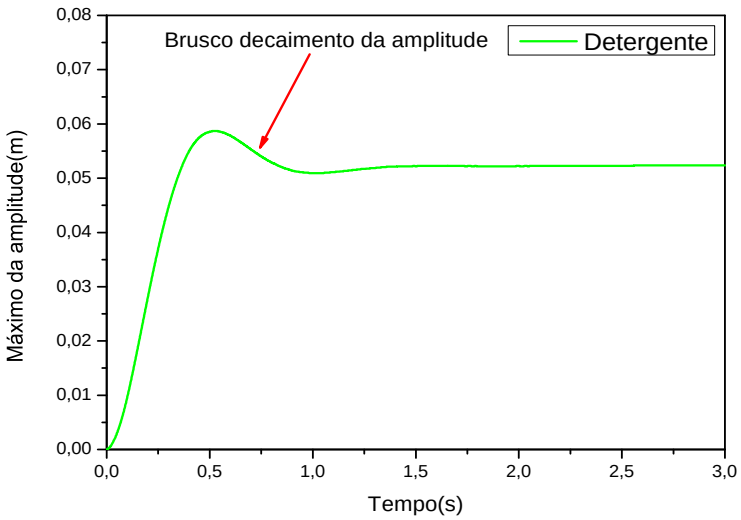

Figura 49: Representação gráfica do movimento amortecido descrito pela esfera oscilando imersa no detergente, durante os dez primeiros segundo de oscilação.

Repare que aqui temos uma queda ainda mais brusca na amplitude de oscilação devido a força de retardo que neste fluido tem um valor superior a da água. Este sistema descreve um comportamento que tende para um oscilador criticamente amortecido.

Para finalizarmos, as figuras 50 e 51 mostram a superposição dos três gráficos obtidos nas três diferentes condições de oscilação a que o sistema foi submetido.

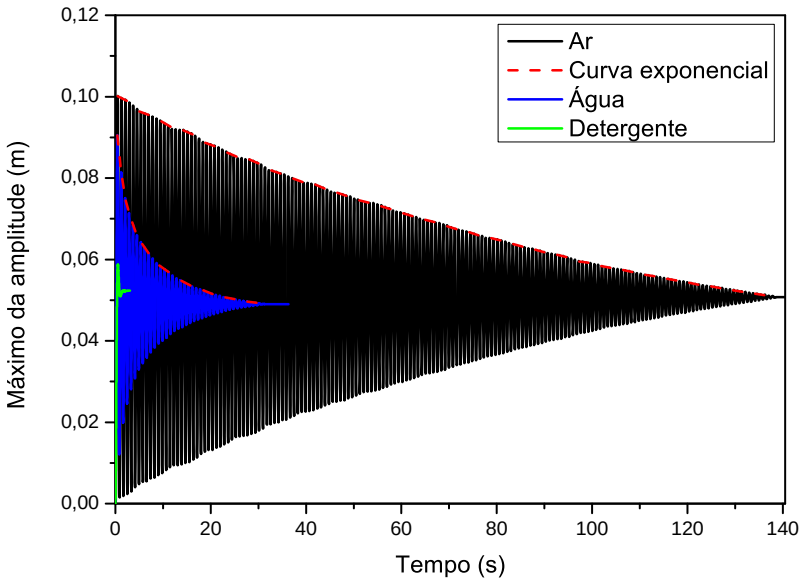

Figura 50: Superposição das representações gráficas dos movimentos amortecidos descrito pela esfera quando esta, está oscilando imersa no detergente, ar e água.

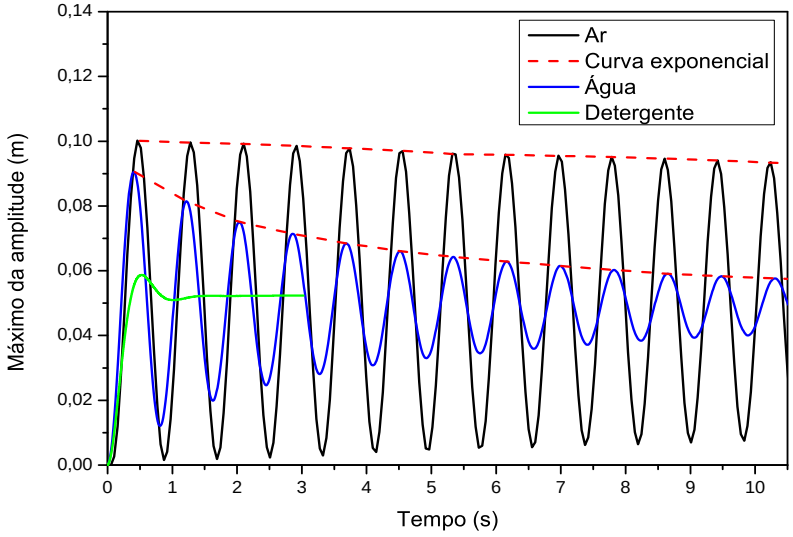

Figura 51: Superposição das representações gráficas nos dez segundos iniciais de movimento dos sistemas.

Observe que de acordo com a figura 51, percebemos com os gráfico sobrepostos o quanto variam as amplitudes para cada situação. Veja que a medida que temos a massa do sistema interagindo com um fluido de viscosidade maior temos uma diminuição ainda mais significativa das amplitudes de oscilação do sistema. No caso do detergente não temos nem uma oscilação completa, o que indica que quanto mais denso e viscoso é o fluido maior a taxa de perda de energia do oscilador. Assim concluímos a analise gráfica dos dados obtidos com os testes realizados.

Conclusão

Com o desenvolvimento e execução do presente trabalho foi possível destacarmos o quanto a experimentação é importante para um estudante no processo de aprendizado e como isso pode vir a somar conhecimentos para o mesmo.

A todo momento surgem novas tecnologias, entre elas temos o Arduino, uma ferramenta moderna e versátil que nos permite infinitas possibilidades de uso. O contato com este equipamento foi decisivo e permitiu perceber o quanto podemos ir além das monótonas salas de aula na busca pelo saber. Uma ferramenta acessível e que pode vir a contribuir enormemente para os estudos tanto em laboratório quanto em sala de aula, principalmente na Física. Foi apoiado nesta excelente ferramenta que desenvolvemos e concluímos este trabalho.

Com o auxílio desta fantástica ferramenta foi possível realizarmos um estudo detalhado do comportamento e das características do oscilador harmônico amortecido, bem como calcular o valor da aceleração da gravidade local, determinar o período de oscilação natural do oscilador e encontrar uma relação para o fator de amortecimento do mesmo.

Tivemos algumas dificuldade no desenvolvimento deste trabalho. A principal dificuldade foi dominar a programação do Arduino. Tivemos também algumas dificuldades durante a coleta dos dados, pois tivemos que ajustar o sistema de modo que durante a coleta dos dados o equipamento não sofresse influências externas o que poderia vir a afetar significativamente a qualidade dos dados coletados.

A partir dos resultados obtidos concluimos que o Arduino sendo utilizado de maneira adequada pode nos fornecer resultados bastantes satisfatórios quando empregado em projetos de baixo custo em laboratório. Mostramos que é possível desenvolver um bom trabalho utilizando equipamentos reciclados e um pouco de engenhosidade. Observamos que os resultados obtidos após a execução deste trabalho estão em conformidade com os resultados teóricos esperados o que nos mostra que é possível utilizar o Arduino como ferramenta de estudo.

Em suma esperamos que após a execução deste trabalho os estudantes de graduação, em especial os de Física tenham a curiosidade e o interesse em buscar desenvolver outros projetos na área da educação utilizando o Arduino, e que este trabalho possa contribuir de alguma maneira para todos que buscam conhecimento de qualidade.

Referências

ARDUINO, . *ARDUINO MEGA 2560 REV3*. 2018. [Online; accessed 23-julho-2018]. Disponível em: <https://store.arduino.cc/arduino-mega-2560-rev3>. Citado na página 26.

ATMEL, . *Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V*. 2018. [Online; accessed 23-julho-2018]. Disponível em: <http://ww1.microchip.com/downloads-/en/DeviceDoc/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf>. Citado na página 31.

BALL, M. *Arduino Sketch para gerenciar alta resolução - codificadores lineares de alta velocidade*. 2018. [Online; accessed 24-julho-2018]. Disponível em: <http://arduino-pi.blogspot.com/2014/04/arduino-sketch-to-manage-high.html>. Citado 3 vezes nas páginas 8, 38 e 40.

BOUQUET, F. et al. Project-based physics labs using low-cost open-source hardware. *American Journal of Physics*, AAPT, v. 85, n. 3, p. 216–222, 2017. Citado na página 29.

CARVALHO, E. A. et al. Medição de velocidade angular com alta resolução usando encoders de baixa resolução e pll. *Controle & Automação (Impresso)*, p. 616–625, 2010. Citado na página 38.

CAVALCANTE, M. A.; TAVOLARO, C. R. C.; MOLISANI, E. Physics with arduino for beginners. *Revista Brasileira de Ensino de Física*, SciELO Brasil, v. 33, n. 4, p. 4503–4503, 2011. Citado na página 25.

COLUCI, V. R. et al. Ilustração de incertezas em medidas utilizando experimentos de queda livre. *Rev. Bras. Ens. Fis*, v. 35, n. 2, p. 2506, 2013. Citado na página 32.

DIGITAL, E. . . L. *CONVERSÃO ANALÓGICO-DIGITAL*. 2018. [Online; accessed 23-julho-2018]. Disponível em: <https://www2.pcs.usp.br/~labdig/pdffiles_2009/labdig2-/pcs2308-p1.pdf>. Citado 2 vezes nas páginas 8 e 35.

D'AUSILIO, A. Arduino: A low-cost multipurpose lab equipment. *Behavior research methods*, Springer, v. 44, n. 2, p. 305–313, 2012. Citado na página 27.

JURAITIS, K. R.; DOMICIANO, J. B. Introdução ao laboratório de física experimental: métodos de obtenção, registro e análise de dados experimentais. *Londrina: Eduel*, 2009. Citado na página 32.

KESTER, W.; BRYANT, J. Voltage-to-frequency converters. *MT-028 Tutorial*, 2009. Citado na página 34.

KRASNOW, B. *Linear position tracking with a quadrature encoder*. 2018. [Online; accessed 23-julho-2018]. Disponível em: <http://benkrasnow.blogspot.com/2010/02-/linear-position-tracking-with.html>. Citado 2 vezes nas páginas 37 e 39.

MCROBERTS, M. Arduino básico. *São Paulo: Novatec*, v. 1, 2011. Citado na página 25.

NETO, J. T. de C.; APOLINÁRIO, F. R.; SOARES, A. de A. Sistema photogate de seis canais analógicos para laboratórios didáticos de física. *Caderno Brasileiro de Ensino de Física*, v. 40, n. 1, 2017. Citado na página 38.

OLIVEIRA, A. A.; CARVALHO, F. K.; ANDRADE, L. C. V. de. Agosto de 2014. Rio de Janeiro, 2014. Citado 2 vezes nas páginas 25 e 29.

ROBOTSHOP, . *Arduino 101: Timers and Interrupts*. 2018. [Online; accessed 23-julho-2018]. Disponível em: <https://www.robotshop.com/letsmakerobots/arduino-101-timers-and-interrupts>. Citado na página 31.

RODRIGUES, M. G.; BUSQUINI, J. A.; SANTARINE, G. A. Oscilador harmônico amortecido e séries infinitas. *Revista Brasileira de Ensino de Física*, Sociedade Brasileira de Física, p. 4304–1, 2010. Citado 2 vezes nas páginas 13 e 20.

SANTOS, M. F. Bancada bidimensional para calibração de sensores de fluxo óptico. NATAL - RN, 2015. Citado na página 38.

SOUZA, A. R. d. et al. The arduino board: a low cost option for physics experiments assisted by pc. *Revista Brasileira de Ensino de Física*, SciELO Brasil, v. 33, n. 1, p. 01–05, 2011. Citado 2 vezes nas páginas 27 e 29.

SOUZA, F. *Arduino MEGA 2560*. 2018. [Online; accessed 24-julho-2018]. Disponível em: <https://www.embarcados.com.br/arduino-mega-2560/>. Citado 2 vezes nas páginas 8 e 27.

THORNTON, S. T.; MARION, J. B. *Dinâmica clássica de partículas e sistemas*. [S.l.]: Cengage Learning São Paulo, 2011. Citado 7 vezes nas páginas 8, 15, 16, 18, 20, 22 e 23.

VUOLO, J. H. Fundamentos da teoria de erros-editora edgard bl ucher. *São Paulo,*, 1992. Citado na página 32.

WALLARD, A. News from the bipm—2003. *Metrologia*, v. 41, n. 1, p. 99, 2004. Disponível em: <http://stacks.iop.org/0026-1394/41/i=1/a=M01>. Citado na página 33.

ZUCH, E. L. Principles of data acquisition and conversion. *Data Acquisition and Conversion Handbook*, Mansfield, MA: Datel, p. 13–18, 1979. Citado na página 34.