1 METODOLOGIA

A metodologia desse projeto será dividida em quatro fases: sistema mecânico, sistema eletrônico, desenvolvimento do sistema de software e integração dos sistemas.

1.1 SISTEMA MECÂNICO

Nessa sessão será desenvolvido o projeto de sistema mecânico que se divide em estrutura e componentes. Para compreensão facilitada do projeto, os cálculos de resistência e equilíbrio foram desconsiderados, já que o equipamento é de pequena escala.

1.1.1 Estrutura

A estrutura ou mesa cartesiana, como é chamada neste trabalho, compõem o sistema mecânico junto com seus componentes, é a parte responsável por manter a união das peças que ela a compõem e também servir de base para o posicionamento dentro da área de testes do túnel de vento. Como o túnel de vento do Laboratório de Sistemas Térmicos da Universidade Federal do Rio Grande é utilizado para atividades de pesquisa em energia renovável e fenômenos de transporte, optou-se por projetar uma estrutura móvel, que pode ser colocada e retirada de dentro do túnel, quando há a necessidade da caracterização do canal aberto. Para que essa operação se torne prática a mesa deve ser leve e resistente.

Por ser um metal leve, durável e resistente, o alumínio se torna uma boa opção para a confecção desta estrutura. Com uma alta relação resistência / peso, confere um excepcional desempenho, além de dar um ótimo acabamento à peça. A estrutura de metal será dividida em duas partes, a base e o pórtico. A base será o componente que terá a função de dar estabilidade a mesa e evitar que esta venha a tombar, terá um formato retangular de 500x400 mm e será feito com um perfil v_slot de 20 x 40 mm em alumínio, conforme apresentado na Figura 3.1 Para fazer a união dos perfis, que serão cortados em tamanho adequado com cantos esquadrejados em ângulo de 45 graus, para dar um melhor acabamento à peça, será utilizada uma placa de conexão interna de 90 graus, conforme Figura 3.3, que unirá os cantos dos perfis por meio de parafusos Allen sem cabeça M5.

Figura 1 – Perfil v_slot 20x40mm em alumínio.
Fonte: www.forsetisolucoes.com.br

Figura 2 – Dimensões do perfil 20x40mm.
Fonte: www.forsetisolucoes.com.br

Figura 3 – Placa de conexão interna de 90º.
Fonte: www.forsetisolucoes.com.br

Figura 4 – Dimensões da placa de conexão interna de 90º.
Fonte: www.forsetisolucoes.com.br

Figura 5 – Estrutura da mesa cartesiana.
Fonte: Próprio autor

Figura 6 – Detalhe do encaixe a 45° da base da estrutura.
Fonte: Próprio autor

O pórtico será feito com o perfil v_slot de 20x20 mm em alumínio conforme apresenta a Figura XX, e terá as dimensões de 500x480 mm. Sua função é sustentar e estabilizar o sistema de transmissão. Para fixação do pórtico à base será utilizada uma placa T simples de aço. Na montagem da parte superior do pórtico a união se dará por meio de parafusos M6 diretamente nos perfis.

Figura 7 – Perfil v_slot 20x20 mm em alumínio.
Fonte: www.forsetisolucoes.com.br

Figura 8 – Dimensões do perfil 20x20mm.
Fonte: www.forsetisolucoes.com.br

Figura 9 – Placa T simples de aço.
Fonte: www.forsetisolucoes.com.br

Figura 10 – Dimensões da placa T simples.
Fonte: www.forsetisolucoes.com.br

1.1.2 Sistema de transmissão

Segundo (Budynas,Richard G. 2016) o parafuso de potência é um dispositivo usado para transformar o movimento angular em movimento linear e, usualmente, para transmitir potência. Para fazer o movimento dos carros do eixo X e do eixo Y, componentes que farão o efetivo deslocamento do equipamento de medição dentro da área de teste do túnel de vento, o

fuso foi escolhido como elemento de transmissão, que vai transformar o movimento giratório do motor de passo em deslocamento linear na estrutura da mesa cartesiana. Acoplados aos motores de passo, por meio de acopladores de eixo e na outra extremidade por mancal para fuso de 8 mm, os fusos farão o transporte dos carros horizontal e vertical.

Para que se tenha um deslocamento mais rápido das castanhas, elemento que estarão em contato direto com o fuso e o carro, foi selecionado o fuso trapezoidal de 8 mm de diâmetro e 8 mm de passo. Será utilizado uma guia para o deslocamento horizontal, para o deslocamento vertical será utilizado um fuso acoplado ao motor é uma guia no lado oposto, apenas para dar suporte ao elemento que fixará o fuso horizontal

1.1.3 Acionador

Os motores elétricos são máquinas capazes de transformar energia elétrica em energia mecânica, essa energia se dá em forma de movimento angular. São equipamentos versáteis, muito eficientes e amplamente utilizados. Para fazer a movimentação dos carros, nos eixos X e Y, optou-se por motores de passo, por que é um motor que possibilita o controle da velocidade e o posicionamento preciso , pois rotaciona em ângulos bem definidos, chamados de passos. O ponto negativo desses motores é que o controle é mais complexo, necessitando de um comando eletrônico digital para fazê-lo funcionar, por outro lado apresenta uma grande precisão no seu movimento. Os motores serão acoplados aos perfis da estrutura de modo a possibilitar o movimento dos fusos e consequentemente o equipamento de medição, que estará preso a ele por meio de uma guia com roldana.

1.2 SISTEMA ELETRÔNICO

O sistema eletrônico de uma mesa cartesiana foi dividido em módulos para um melhor detalhamento: placa de prototipagem eletrônica Arduino, drivers de potência, atuadores elétricos, fonte de alimentação, optoacopladores, encoders e um computador. É importante lembrar que para menor custo do projeto optou-se pela criação de dispositivos.

1.2.1 Placa de prototipagem eletrônica Arduino

A placa de prototipagem eletrônica Arduino é responsável pela recepção e tratamento dos dados provenientes da interface computacional. O controle dos motores está fundamentado

na programação do microcontrolador de acordo com as necessidades definidas inicialmente para operação da mesa cartesiana.

Com o objetivo de elaborar uma interface de prototipagem de baixo custo para uso em projetos escolares, o arduino foi criado por um grupo de pesquisadores na Itália em 2005. Na sua concepção, para ter maior flexibilidade a diversos tipos de projetos, isto é, para que qualquer projetista pudesse personalizá-lo, foi adotado o conceito de hardware livre.

Em sua composição, a placa Arduino UNO contém um microcontrolador ATMEL ATMEGA328, que é um dispositivo de 8 bits da família AVR com arquitetura RISC avançada e com encapsulamento DIP28, além de quatorze portas digitais de entrada e saída, sendo que seis delas com capacidade de PWM (Pulse Width Modulation), seis entradas analógicas e 32 KB de memória flash, 2 KB de RAM e 1 KB de EEPROM.

A alimentação é via porta USB ou por conector tipo Jack de alimentação externa que trabalha entre os limites de 6 V e 20 V sendo recomendável o uso de 7 V a 12 V para não danificar. Para o fornecimento de tensão contínua para alimentação dos circuitos e Shields, a placa Arduino UNO tem um regulador de tensão de 3,3 V.

Além de alimentar, a porta USB é a via de comunicação com o computador para o envio do código de máquina gerado pelo compilador. O código compilado é enviado pelo microcontrolador ATMEL ATMEGA16U2 que está conectado a dois LEDs chamados de TX e RX cuja função é a indicação do envio e recepção dos dados da placa para o computador.

A placa Arduino é programada via IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado), utilizando uma linguagem baseada em C/C++.

- Botão reset: o botão reset serve para reiniciar a placa arduino.
- Conector USB tipo B: o conector USB tipo B serve para conectar a placa Arduino ao computador.
- Conector DC: o conector DC serve para alimentação externa do Arduino.
- Portas digitais: são às portas que trabalham com os sinais digitais que são sinais com valores discretos, no arduino temos às constantes LOW que significa 0 V e também HIGH que significa 5 V, no código são 0 e 1 respectivamente. Dentro das portas digitais temos às portas PWM (Pulse Width Modulation) que são portas que modulam o sinal pela largura do pulso, no arduino podemos variar de 0 a 255. Resumidamente, a placa Arduino possui quatorze portas digitais, sendo que seis delas são portas PWM como dito anteriormente, os pinos delas são 3,5,6,9,10 e 11.

- Portas analógicas: são às portas que trabalham com os sinais analógicos que são sinais com valores contínuos, no arduino podemos variar de 0 a 1023.
- Portas de alimentação: são às portas de saída de tensão do arduino, temos três portas: porta 3,3 V cuja saída trabalha em 3,3 V, porta 5 V cuja saída trabalha em 5 V e a porta Vin cuja saída trabalha com a tensão de entrada do arduino.
- Led porta 13: led conectado a porta 13 do arduino.
- Leds das portas RX TX: leds conectados às portas RX TX, sendo que o led TX serve para indicar a transmissão de dados e o RX para indicar a recepção de dados.
- Processador ATMEGA328: o processador é responsável pelo processamento da lógica de programação.
- Led de power: led que acende quando o arduino está ligado.
- Conector ICSP: o conector ICSP, é o conector in circuit system programmer, que se refere a capacidade de programar Arduinos diretamente dos seus microcontroladores.
- Conjunto microcontrolador e cristal para USB: esse conjunto possui um microcontrolador ATMEGA16U2 e um cristal externo de 16 MHz, e é responsável pelo gerenciamento da porta USB.
- Circuito de proteção de sobretensão na USB: responsável por proteger a entrada USB de sobretensão.
- Controlador de tensão de alimentação: esse controlador é responsável por verificar se a tensão DC está presente, se não estiver, deixa que a tensão da USB alimente o circuito.
- Regulador de tensão de 5 V: serve para regular a tensão em 5 V.
- Regulador de tensão de 3,3 V: serve para regular a tensão em 3,3 V.
- Cristal para clock: serve para gerar o clock.

1.2.2 Drivers de potência

Os drivers de potência são dispositivos que conservam sinais fundamentais de entrada em suas saídas, potencializando e fornecendo maior corrente elétrica para equipamentos atuadores. Os drivers de potência podem ser formados por elementos eletromecânicos (relés) e semicondutores (diodos, transistores, e circuitos integrados).

No presente projeto, os drivers têm a função de, a partir dos sinais originados pelo Arduino, atender a demanda dos motores de passo utilizados.

A equipe do projeto optou por desenvolver para a mesa cartesiana que necessita de

dois atuadores (motores de passo), drivers individuais para cada motor devido ao custo que seria menor.

A construção dos drivers que controlam os motores de passo da mesa cartesiana foi baseada em um circuito eletrônico para motores que trabalham com tensão de 12 V DC e até 10 A de corrente elétrica.

O driver criado foi utilizado para controlar motores de passo e pode operar com tensões entre 8 V e 35 V e e ntregar até 35 V por bobina. A Figura 3.14 mostra o driver com os respectivos componentes.

AQUI EMBAIXO UMA EXPLICAÇÃO DA IMAGEM DAS PORTAS, EXPLICANDO PARA QUE SERVE CADA PORTA COMO SE FOSSE UM DATASHEET DO DRIVER CRIADO

Parâmetro | Magnitude Tensão lógica mínima | 3v Tensão lógica máxima | 5,5v Corrente contínua por fase | 1A Corrente máxima por fase | 2A Tensão de operação mínima | 8v Tensão de operação máxima | 35v

1.2.3 Atuadores

Atuadores são equipamentos ou dispositivos elétricos que convertem energia hidráulica, pneumática ou elétrica em energia mecânica. A energia gerada nos atuadores é transformada em movimento em vários tipos de processos.

Os atuadores utilizados neste projeto foram os elétricos que se dividem em motores elétricos de corrente alternada e contínua, servomotores e motores de passo. Esses atuadores convertem pulsos elétricos recebidos em seus terminais em energia mecânica transformando-a em movimento rotativo do motor, no caso do projeto os motores de passo que controlam o movimento dos fusos da mesa cartesiana. Os atuadores elétricos atendem a comandos manuais ou programáveis, localmente ou remotamente. Eles se destacam por uma transmissão de potência simplificada e eficiente do ponto de vista de energia.

Os motores de passo são atuadores eletromagnéticos com capacidade de converter pulsos elétricos digitais recebidos em movimento de rotação incremental do eixo do motor. Sua aplicação é necessária em movimentos rotativos com alta precisão que necessitam de controle de posição e velocidade. A sua escolha foi definida pela precisão, baixo custo de aquisição e manutenção.

A composição de um motor de passo contém um rotor e um estator que é a parte

fixa do gerador elétrico, nessa está situado um conjunto de bobinas responsáveis pelo giro do rotor. Para que haja o movimento, as bobinas estão ligadas aos terminais do motor, organizadas em pares, interligadas entre si e posicionadas em sentidos opostos para que quando forem energizadas, der o movimento de rotação do motor devido às interações magnéticas.

O atributo que distingue o motor de passo dos demais motores elétricos é a capacidade de realizar passos, que são rotações discretas incrementais e precisas. Os passos são definidos por um número fixo de pólos magnéticos de dente de engrenagens do motor determinando assim, a precisão de ângulo de rotação do motor de passo. Para que haja um controle de quantos passos serão dados, o motor necessita de largura de pulso a fim de enviar a corrente adequada para cada passo. A Figura 3.15 é um exemplo para melhor entendimento do conceito de passo.

A Figura 3.15 mostra um motor de passo com cinquenta dentes de precisão, portanto combinando o número de dentes com as quatro fases de um motor bipolar temos duzentos passos.

NÚMERO PASSOS = NÚMERO DENTES * NÚMERO FASES

A precisão do motor é definida por trezentos e sessenta graus divididos pelo número de passos. Portanto, trezentos e sessenta divididos por duzentos é igual a um vírgula oito graus de precisão. Já o segundo motor de passo possui cem dentes de precisão, portanto a precisão é de zero vírgula nove graus, o que indica que é um motor com maior precisão que o de cinquenta dentes.

Para aumentar a precisão dos motores é possível subdividir o passo de um motor em menores passos fornecendo ao mesmo tempo corrente elétrica em duas fases. Essa forma de trabalho é chamada de micropassos.

Logo, quando é fornecido a mesma magnitude de corrente a duas fases, um campo magnético de mesma magnitude é gerado, resultando em um movimento angular com a metade do passo original, essa configuração é chamada de meio passo. Assim, ao aplicar magnitudes de corrente diferentes a duas fases, o rotor se desloca proporcionalmente ao campo eletromagnético mais forte.

Os motores de passo utilizados no projeto da mesa cartesiana possuem 5 kgf.cm de torque, passo de 1.8° e corrente máxima de 2 Ampères por fase. O incremento de rotação e o torque são definidos conforme o modo de excitação que pode ser por passo completo, meio passo e micropasso explicadas anteriormente. O passo é dividido em quatro, oito ou dezesseis na maioria das vezes, mas também, atualmente se encontra sistemas capazes de dividir um passo em milhares de vezes.

As Tabelas 2, 3 e 4 mostram a sequência de polaridades aplicadas no motor no movimento de sentido horário.

Passo | A+ | B+ | A- | B- | Decimal 1 | 0 | 0 | 0 | 1 | 1 2 | 0 | 0 | 1 | 0 | 2 3 | 0 | 1 | 0 | 0 | 4 4 | 1 | 0 | 0 | 0 | 8

Passo | A+ | B+ | A- | B- | Decimal 1 | 1 | 0 | 0 | 1 | 9 2 | 0 | 0 | 1 | 1 | 3 3 | 0 | 1 | 1 | 0 | 6 4 | 1 | 1 | 0 | 0 | 12

Passo | A+ | B+ | A- | B- | Decimal 1 | 1 | 0 | 0 | 1 | 9 2 | 0 | 0 | 0 | 1 | 1 3 | 0 | 0 | 1 | 1 | 3 4 | 0 | 0 | 1 | 0 | 2 5 | 0 | 1 | 1 | 0 | 6 6 | 0 | 1 | 0 | 0 | 4 7 | 1 | 1 | 0 | 0 | 12 8 | 1 | 0 | 0 | 0 | 8

As Tabelas 5, 6 e 7 mostram a sequência de polaridades aplicadas no motor no movimento de sentido anti-horário.

Passo | A+ | B+ | A- | B- | Decimal 1 | 1 | 0 | 0 | 0 | 8 2 | 0 | 1 | 0 | 0 | 4 3 | 0 | 0 | 1 | 0 | 2 4 | 0 | 0 | 0 | 1 | 1

Passo | A+ | B+ | A- | B- | Decimal 1 | 1 | 1 | 0 | 0 | 12 2 | 0 | 1 | 1 | 0 | 6 3 | 0 | 0 | 1 | 1 | 3 4 | 1 | 0 | 0 | 1 | 9

Passo | A+ | B+ | A- | B- | Decimal 1 | 1 | 0 | 0 | 0 | 8 2 | 1 | 1 | 0 | 0 | 12 3 | 0 | 1 | 0 | 0 | 4 4 | 0 | 1 | 1 | 0 | 6 5 | 0 | 0 | 1 | 0 | 2 6 | 0 | 0 | 1 | 1 | 3 7 | 0 | 0 | 0 | 1 | 1 8 | 1 | 0 | 0 | 1 | 9

A Tabela 8 apresenta alguns parâmetros dos motores de passo.

Parâmetros | Magnitude Número de passos por revolução | 200 (1,8 graus por passo) Corrente de operação | 800 mA Tensão de alimentação | 12 V Configuração das bobinas | Bipolar (4 fios)

1.2.4 Fonte de alimentação

A fonte de alimentação projetada para transformar a tensão elétrica alternada em corrente contínua.

Para a escolha correta da fonte de alimentação é necessário definir a demanda de energia elétrica que os dispositivos que são alimentados pela fonte necessitam.

A Tabela 9 apresenta a demanda de energia elétrica de cada dispositivo.

Componente | Quantidade | Tensão | Consumo | Consumo Total Arduino | 1 | 7 a 12 V | 800 mA | 800 mA Driver | 2 | 8 a 35 V | 2 A | 4 A Motor de passo | 2 | 5 a 36 V | 2 A | 4 A Optoacoplador | 2 | 5 a 35 V | 5mA | 10 mA Resultado | - | 12 V | - | 8,18 A

A Tabela 9 indica que a placa controladora Arduino opera no intervalo de tensão de 7 V a 12 V, os drivers de potência no intervalo de 8 V a 35 V, os motores de passo no intervalo de 5

V a 36 V e o optoacoplador no intervalo de 5V a 35 V.A tensão de saída de 12 V foi determinada como melhor opção para o projeto.

Outro parâmetro que a Tabela 7 apresenta é a corrente elétrica mínima para a operação do circuito. A placa controladora necessita de 800 mA, os drivers de potência necessitam de 2 A cada, como são 2 drivers, 4 A são necessários, os motores de passo consomem 2 amperes cada, como o projeto contém 2 motores, 4 Amperes são necessários e os optoacopladores necessitam de 10 mA cada, como o projeto contém 2 optoacopladores, 20mA são necessários. Decidiu-se acrescentar uma margem de segurança adicional de 30% na corrente, então, considerou-se uma corrente mínima necessária de 10,64 A.

Corrente necessária = 1,3*(Arduino + 2* driver de potência + 2*motor de passo + 2 optoacopladores)

Conforme a determinação da tensão de saída e o cálculo de corrente necessária, é possível determinar que a fonte deve ter 12 V e 10,64 A.

1.2.5 Acopladores ópticos

Os acopladores ópticos ou optoacopladores são dispositivos que realizam a transferência de sinais de um circuito para outro por meio de um feixe de luz sem a ligação elétrica.

A aplicação dentro do projeto é o isolamento elétrico que pode ser estabelecido entre os circuitos de controle de potência, protegendo os circuitos sensíveis a uma alta tensão como a placa controladora Arduino.

A sua composição contém uma fonte emissora de luz (LED) e um sensor fototransistor sensível às variações espectrais da fonte emissora. Seu funcionamento é baseado no efeito fotoelétrico, onde o diodo LED produz um feixe de luz infravermelha polarizando a base do fototransistor impondo uma condução entre base e emissor.

O optoacoplador escolhido para o projeto foi o PC 817 por já estar disponível no Laboratório de Sistemas Térmicos.

1.2.6 Encoders

Geradores de pulsos ou encoders são sensores/transdutores eletromecânicos responsáveis pelo sistema de controle de posição transformando a medida de posição de algum objeto, seja linear ou angular, em sinal elétrico digital que é transmitida à placa controladora, também conseguem converter movimentos circulares ou lineares em pulso elétricos.

Um encoder tem a capacidade de quantização de distâncias, controle de velocidades, medição de ângulos, medição de posição, medição de deslocamento relativo e etc. Sua composição contém um disco com marcações, um emissor e um receptor. Conforme o disco gira, vão sendo contadas às marcações e o emissor envia um sinal à placa controladora que por sua vez executa cálculos de distâncias, velocidades, ângulos, número de rotações e etc.

O princípio de funcionamento é dividido em três tipos, sendo eles: tacômetro, incremental e absoluto.

O encoder do tipo tacômetro possui um sinal de saída digital responsável em emitir um pulso para cada incremento captado no deslocamento. Este é utilizado na medição de velocidade e também no deslocamento angular unidirecional. O encoder incremental conta com sistema eletrônico externo responsável pela interpretação de posição, este dispositivo utiliza dois ou mais elementos geradores de sinal possuindo a capacidade de rotacionar por quantas revoluções forem necessárias.

E o encoder absoluto utiliza várias faixas de saídas com leitura em paralelo e são limitados a uma revolução. Os dados podem ser recuperados, se uma falha no sistema ocorrer, devido a representação binária da posição angular do eixo.

1.2.7 Chaves fim de curso

As chaves fim de curso são dispositivos eletromecânicos usados para limitação de campo de movimento de eixos, como os presentes na mesa cartesiana. Esses componentes têm a capacidade de mudança de estado de conexão em circuitos, alternando o estado de aberto para fechado e vice-versa (ALCIATORE; HISTAND, 2014). Seu estado inicialmente pode ser tanto normalmente aberto como normalmente fechado alterando seu estado por pino, gatilho, roldana, haste alavanca, etc.

Uma chave fim de curso é composta basicamente por três elementos, sendo eles:

a) Caixa: Pode ser metálica ou plástica, dependendo do tipo e abriga os contatos e o atuador.
b) Contato: É usado dentro do circuito a fim de fazer com que a atuação da chave fim de curso interrompa ou acione algum outro dispositivo.
c) Atuador: Recebe a força externa exercida para o acionamento da troca de estado.

1.3 SISTEMA DE SOFTWARE

A seguir será descrito o desenvolvimento do sistema de software.

1.3.1 Plataforma de prototipação Arduino IDE

A plataforma de prototipação Arduino IDE (Integrated development environment ou ambiente de desenvolvimento integrado) é um software que permite o desenvolvimento e envio de códigos compilados direto para o microcontrolador. Essa plataforma tem a flexibilidade de ser utilizada em vários sistemas operacionais e foi desenvolvida na linguagem Java oferecendo suporte de desenvolvimento na linguagem C e C++. O download da plataforma foi realizado através do link: https://www.arduino.cc/en/software

1.3.2 Logica de programação

O desenvolvimento da lógica de programação da placa controladora foi organizado de maneira modular usando orientação a objetos para uma manutenção facilitada e um entendimento mais claro do código.

As operações que o software deve executar foram apresentadas no fluxograma abaixo.

A Figura 3.20 apresenta um fluxograma simplificado da lógica de programação presente na placa controladora Arduino que inicialmente tem como primeiro comando ficar esperando o envio de dados pela porta Serial, se caso há esse envio, então o programa faz a leitura da porta serial, armazenando os dados em variáveis para em seguida realizar o tratamento de dados. Após o tratamento de dados o software dará o comando de movimentação do motor que controla o fuso do eixo horizontal (eixo X), em seguida, assim que o motor do eixo X parar sua rotação, o comando de movimentação do motor do eixo vertical (eixo Y) será acionado. Finalmente, após o fim da execução do motor que controla o eixo Y, o software volta a analisar se é enviado dados pela porta serial.

1.3.3 Diagrama de classes

Diagrama de classe é uma representação da estrutura e relações entre classes que um software possui facilitando e servindo de modelo para criação de objetos. Esse diagrama permite modelar classes com seus atributos e métodos além da relação entre objetos.

Para o desenvolvimento do software da placa controladora cuja linguagem é C++

que é fundamentada em orientação a objetos, foi definido que seria a melhor opção realizar um diagrama de classes antes do desenvolvimento do código. Sendo assim a Figura 3.21 apresentada abaixo é o resultado do que foi modelado.

A seguir será explicado este diagrama classe a classe.

“Sigmoidal” é a classe responsável pela aceleração sigmóide nos motores de passo, essa classe utiliza a função sigmóide que é uma função matemática com o gráfico parecido com a letra S. Para o desenvolvimento da mesa cartesiana houve a preocupação dos projetistas em configurar os motores de passo com uma aceleração variável para maior vida útil dos equipamentos, já que com aceleração constante pode causar forças desnecessárias durante o processo de rotação.

A tabela 10 apresenta a declaração e funcionalidade dos atributos e métodos da classe “Sigmoidal”.

Declaração | Funcionalidade - periodoMinimo: double | Período máximo do passo do motor. - periodoMaximo: double | Período mínimo do passo do motor. - vetorAceleracao: int | Determina se a curva será de aceleração (1) ou desaceleração (-1) - amplitude: double | Amplitude da aceleração, diferença do período máximo menos o período mínimo - declividade: double | Inclinação da curva de aceleração, o quão rápido o motor irá acelerar - xMedio: double | É a metade do número de iterações necessários para percorrer a curva sigmoidal - calculoDoXMedio(): double | Calcula o número de Iterações necessários para percorrer metade da curva sigmoidal. + calculoDoInstante(): double | Calcula um instante na curva tendo como entrada uma iteração

A classe “Sigmoidal” tem como atributos privados: “periodoMaximo” e o “periodoMinimo” que são os períodos que o motor sofrerá os pulsos nas bobinas, “vetorAceleracao” que determina se a curva sigmoidal terá um comportamento de aceleração ou desaceleração, “amplitude” que é a diferença entre o “periodoMaximo” e o “periodoMinimo” e é utilizada na equação do cálculo do X médio e do cálculo do instante, “declividade” que determina a inclinação da curva de aceleração (o quão rápido o motor irá acelerar), “xMedio” que é a metade do número de passos (iterações) que o motor precisará para acelerar e é utilizado no cálculo do instante.

Como operações, a classe “Sigmoidal” tem os métodos de acesso getters e setters que acessam os atributos privados citados acima, esses métodos são necessários para cumprir o princípio de encapsulamento da orientação a objetos protegendo a lógica da classe. A classe

“Sigmoidal” dispõe de duas operações, as quais são definidas por: “calculoDoXMedio”, que calcula o valor do atributo xMedio, “calculoDoInstante”, que calcula o período a ser aplicado em um determinado instante, esse cálculo é utilizado na lógica do método “rotacionarPasso” da classe “Eixo”.

“Pino” é a classe que representa os pinos da placa Arduino, pois devido ao número alto de vezes que foi necessário a utilização dos pinos, foi definido que seria melhor a criação de uma classe para facilitar as operações com pinos.

A tabela 11 apresenta a declaração e funcionalidade dos atributos e métodos da classe “Pino”.

Declaração | Funcionalidade - tipo: TipoPino | Tipo do pino (ANALOGICO, DIGITAL) - numero: byte | Número do pino - modo: byte | Modo do pino (OUTPUT, INPUT) - estado: unsigned int | Estado do pino (LOW, HIGH) e| (0, 255) e| (0, 1023)

A classe “Pino” possui como atributos privados: “tipo”, que é o tipo de entrada do pino que pode ser analogico ou digital, “numero”, que é o número de uma determinada porta na placa, “modo” que define se a porta é de entrada ou saída de dados, e “estado”, que pode estar ligado ou desligado nas portas digitais, ter valores de 0 a 255 nas portas PWM e de 0 a 1023 nas portas analogicas. Como operações, a classe Pino tem os métodos de acesso getters e setters que acessam os atributos privados citados acima.

“Driver” é a classe responsável pelo controle digital do driver de potência, com ela é possível definir o modo do passo dos motores, executar o pulso que dará movimento ao motor de passo, ligar e desligar o driver, deixá-lo no modo “sleep” e também “reseta-lo”.

A tabela 12 apresenta a declaração e funcionalidade dos atributos e métodos da classe “Driver”

Declaração | Funcionalidade - pinoEnable: Pino | Ativar e desativar o driver - pinoReset: Pino | Resetar o driver - pinoSleep: Pino | Ativar e desativar o modo sleep do driver - pinoM0: Pino | Pino M0 do modo de passo do driver - pinoM1: Pino | Pino M1 do modo de passo do driver - pinoM2: Pino | Pino M2 do modo de passo do driver - pinoPasso: Pino | Pino de execução do passo do driver - pinoDirecao: Pino | Pino de configuração da direção do driver - pinoIN1: Pino | Pino IN1 do motor de passo - pinoIN2: Pino | Pino IN2 do motor de passo - pinoIN3: Pino | Pino IN3 do motor de passo - pinoIN4: Pino | Pino IN4 do motor de passo - tipoAcionamento: TipoAcionamento | Tipo de acionamento do driver: (SOFTWARE, HARDWARE) - tipoPasso: TipoPasso | Tipo do passo do driver: (WAVESTEP, FULLSTEP,

HALFSTEP) - direcao: Direcao | Configuração da direção do driver: (ANTIHORARIO, HORARIO) - passoAtual: int | PassoAtual é o indice de acesso as informações do vetor de passos - modoPasso: int | Pino do modo de passo do driver - passosPorVolta: int | Quantidade de passos a cada volta do motor de passo. - vetorPasso: byte | Vetor de bytes de sequência dos passos. - setarEstadoModo(pM0:bool, pM1:bool ,pM2:bool):void | Definir o modo de passo do driver passando como parâmetro os estados dos pinos M0,M1,M2 - setarEstados(b: byte):void | Definir estados dos pinos IN1, IN2, IN3, IN4 do motor de passo - passo():void | Executar um passo do motor. + executarPasso(cont:int, linf:int, lsup:int):void | Executar um passo do motor com a lógica de sequência dos passos. + setarModoPasso(mp:int, b:byte):void | Definir o modo de passo do driver

A classe “Driver” possui os atributos privados: “pinoEnable”, que permite ligar ou desligar o driver, “pinoReset”, que reinicia o driver, “pinoSleep”, que ativa o modo sleep do driver, “pinoM0”, “pinoM1” e “pinoM2”, que definem o modo do Passo, “pinoPasso” e “pinoDirecao” que são os pinos de execução de passo e configuração de direção para tipo de acionamento via hardware, “pinoIN1”, “pinoIN2”, “pinoIN3” e “pinoIN4” que são os pinos referentes ao controle das bobinas do motor de passo para tipo de acionamento via software, “tipoAcionamento” para configuração do tipo de acionamento do driver que pode ser via software ou hardware conforme a necessidade de cada tipo de driver utilizado, “tipoPasso” para configuração do tipo de passo do driver que pode ser (wavestep, fullstep, halfstep), “direcao”, que define o sentido do movimento, “passoAtual” que é o índice de acesso às informações do vetor de passos, “modoPasso”, que define o modo do passo do driver, “passosPorVolta”, que define a quantidade de passos a cada volta do motor como por exemplo, 200 passos de 1,8 graus resultando em 360 graus.

Como operações, a classe ”Driver” tem os métodos: métodos de acesso getters e setters que acessam os atributos privados da classe, os métodos: “setarEstados”, que define os estados dos pinos IN1, IN2, IN3, IN4 do motor de passo, “passo” que executa um passo do motor, “setarModoPasso”, que define o modo de passo do driver e “executarPasso”, executa um passo do motor com a lógica de configurações de passos.

“Eixo” é a classe que foi desenvolvida a lógica de movimentação dos eixos da mesa cartesiana. Com ela é possível rotacionar definindo a coordenada cartesiana de preferência, verificar se a posição do eixo está ativando a chave de fim de curso.

A tabela 13 apresenta a declaração e funcionalidade dos atributos e métodos da classe “Eixo”.

Declaração | Funcionalidade - driver: Driver | Define as configurações do driver que controla os motores de passo, - sigmoidal: Sigmoidal | Define a aceleração sigmoidal dos motores de passo. - pinoCursoMinimo: Pino | Pino do curso mínimo da chave fim de curso - pinoCurso-Maximo: Pino | Pino do curso máximo da chave fim de curso - posicao: int | Pino da posição atual do eixo - estadoFimDeCurso: bool | Pino do estado de fim de curso que pode inicializar LOW ou HIGH - podeMovimentar(direcao:bool):bool | Verificar se o motor está no fim de curso. - rotacionarPassos(direcao:bool,passos:int):void | Rotacionar o motor, tendo como parâmetros de entrada a direção e a quantidade de passos. + rotacionarPara(coordenada:double):void | Rotacionar o motor tendo como parâmetro a coordenada cartesiana do eixo.

A classe “Eixo” possui os atributos privados: “driver”, que é responsável pelas configurações e operações do driver de potência que controla o motor de passo acoplado ao eixo, “sigmoidal” que define a aceleração variável do motor, “pinoCursoMinimo” e “pinoCursoMaximo”, que são os pinos referentes às chaves fim de curso de cada eixo, “estadoFimDeCurso”, que indica o estado que a chave fim de curso inicializará estando desativada.

Como operações, a classe tem os métodos: métodos de acesso getters e setters que acessam os atributos privados da classe, os métodos: “podeMovimentar”, que indica se o motor de passo pode executar o próximo passo ou se já chegou ao fim do curso, “rotacionarPassos”, que rotaciona o eixo tendo como parâmetros de entrada uma direção e uma quantidade de passos a serem executadas e “rotacionarPara”, que rotaciona o eixo tendo como parâmetros de entrada uma coordenada cartesiana.

Além das classes utilizadas, o software possui a sketch principal que é responsável pela configuração das informações iniciais, criação dos objetos na memória, configuração dos atributos de cada objeto, execução da leitura pela porta serial, tratamento de dados recebidos e a movimentação dos motores. A Figura 3.22 apresenta um diagrama geral da organização do software.

O arquivo MesaCartesiana.ino é o principal, ele instancia os objetos das classes: “Pino”, “Sigmoidal”, “Driver” e “Eixo” e também tem a responsabilidade de receber às coordenadas através da comunicação serial e às enviar para o objeto da classe “Eixo” que fará a rotação dos motores.

1.4 INTEGRAÇÃO DOS SISTEMAS

Nesta seção será descrito como os sistemas são integrados mostrando como os três sistemas se comunicam. A Figura 3.23 é um fluxograma que descreve de maneira gráfica a integração dos sistemas.

O sistema de software se comunica com o sistema eletrônico através do envio de dados pela comunicação Serial presente no Arduino IDE para o software presente no microcontrolador ATMEGA da placa Arduino. Por sua vez, o microcontrolador envia um comando elétrico ao sistema eletrônico que acionará os motores de passo que convertem a energia elétrica em mecânica transmitindo o movimento dos fusos da estrutura do sistema mecânico.

APÊNDICES

APÊNDICE A – Lorem Ipsum

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

APÊNDICE B – Modelo de Capa

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

APÊNDICE C – Termo de Fiel Depositário

Pesquisa: ANÁLISE DA MORTALIDADE INFANTIL COM MALFORMAÇÕES CONGÊNITAS.

Pelo presente instrumento que atende às exigências legais, a Sra. Maria Consuelo Martins Saraiva, "fiel depositário" com o cargo de Secretária Municipal de Saúde de Iracema, após ter tomado conhecimento do protocolo de pesquisa intitulado: ANÁLISE DA MORTALIDADE INFANTIL COM MALFORMAÇÕES CONGÊNITAS. Analisando a repercussão desse estudo no contexto da saúde pública e epidemiologia, autoriza Karla Maria da Silva Lima, enfermeira, aluna do Curso de Mestrado Acadêmico em Enfermagem da Universidade Estadual do Ceará (UECE), sob orientação do Prof. Dr. José Maria de Castro, da UECE, ter acesso aos bancos de dados do Sistema de Informação sobre Nascidos Vivos e do Sistema de Informação sobre Mortalidade da Secretaria Municipal de Saúde de Iracema, objeto deste estudo, e que se encontram sob sua total responsabilidade. Fica claro que o Fiel Depositário pode a qualquer momento retirar sua AUTORIZAÇÃO e ciente de que todas as informações prestadas tornar-se-ão confidenciais e guardadas por força de sigilo profissional, assegurando que os dados obtidos da pesquisa serão somente utilizados para estudo.

ANEXOS

ANEXO A – Exemplo de Anexo

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

ANEXO B – Dinâmica das classes sociais

Etiam ac leo a risus tristique nonummy. Donec dignissim tincidunt nulla. Vestibulum rhoncus molestie odio. Sed lobortis, justo et pretium lobortis, mauris turpis condimentum augue, nec ultricies nibh arcu pretium enim. Nunc purus neque, placerat id, imperdiet sed, pellentesque nec, nisl. Vestibulum imperdiet neque non sem accumsan laoreet. In hac habitasse platea dictumst. Etiam condimentum facilisis libero. Suspendisse in elit quis nisl aliquam dapibus. Pellentesque auctor sapien. Sed egestas sapien nec lectus. Pellentesque vel dui vel neque bibendum viverra. Aliquam porttitor nisl nec pede. Proin mattis libero vel turpis. Donec rutrum mauris et libero. Proin euismod porta felis. Nam lobortis, metus quis elementum commodo, nunc lectus elementum mauris, eget vulputate ligula tellus eu neque. Vivamus eu dolor.