UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Botucatu

FCA
FACULDADE DE CIÊNCIAS AGRONÔMICAS
UNESP | BOTUCATU

JOUBERT ALVES DE CARVALHO NETO

SISTEMA DE ILUMINAÇÃO DE GRANJAS CONTROLADAS POR LÓGICA *FUZZY*

Botucatu

2021

JOUBERT ALVES DE CARVALHO NETO

SISTEMA DE ILUMINAÇÃO DE GRANJAS CONTROLADAS POR LÓGICA FUZZY

Dissertação apresentada à Faculdade de Ciências Agronômicas da Unesp *Campus* de Botucatu, para obtenção do título de Mestre em Agronomia (Energia na Agricultura).

Orientador: Prof. Dr. Fernando de Lima Caneppele

Botucatu

2021

C331s

Carvalho Neto, Joubert Alves de
Sistema de iluminação de granjas controladas por lógica fuzzy / Joubert Alves de Carvalho Neto. -- Botucatu, 2022
136 p. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências Agronômicas, Botucatu
Orientador: Fernando de Lima Caneppele

1. Frango de corte. 2. Bem-estar animal. 3. Energia elétrica. 4. Luz. 5. Inferência fuzzy. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências Agronômicas, Botucatu. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Botucatu

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: SISTEMA DE ILUMINAÇÃO DE GRANJAS CONTROLADAS POR LÓGICA FUZZY

AUTOR: JOUBERT ALVES DE CARVALHO NETO
ORIENTADOR: FERNANDO DE LIMA CANEPPELE

Aprovado como parte das exigências para obtenção do Título de Mestre em AGRONOMIA (ENERGIA NA AGRICULTURA), pela Comissão Examinadora:

Prof. Dr. FERNANDO DE LIMA CANEPPELE (Participaçao Virtual)
Engenharia de Biossistemas / Faculdade de Zootecnia e Engenharia de Alimentos USP

P/ Prof. Dr. LUÍS ROBERTO ALMEIDA GABRIEL FILHO (Participaçao Virtual)
Departamento de Gestão, Desenvolvimento e Tecnologia / Faculdade de Ciências e Engenharia - FCE - UNESP - Tupã/SP

P/ Prof. Dr. JAIR ANTÔNIO CRUZ SIQUEIRA (Participaçao Virtual)
Centro de Ciências Exatas e Tecnológicas / Universidade Estadual do Oeste do Paraná

Botucatu, 17 de dezembro de 2021

A todos os professores

e estudantes,

dedico

AGRADECIMENTOS

Aos meus pais, Joubert Junior (Beto) e Albertina Kortz (Bel), pelo apoio incondicional.

Ao professor Dr. Fernando de Lima Caneppele, por ter me aceitado para orientação, pelos seus ensinamentos, conselhos e pela paciência.

Ao Programa de Pós-Graduação em Agronomia – Energia na Agricultura da Faculdade de Ciências Agronômicas – FCA da Unesp, *campus* de Botucatu, pela oportunidade de realizar o mestrado.

À minha namorada, Gabriela Vilela Freitas, pelo companheirismo e apoio durante as horas mais difíceis.

Aos meus amigos, Jayme de Campos Junior e Dayene Correia Castilho, pelo incentivo, apoio e encorajamento durante as dificuldades.

A todos os professores e funcionários da FCA, em especial à professora Dra. Silvia Regina Lucas de Souza.

E a todos que estiveram presentes de alguma forma durante esta jornada.

“Para mim, é muito melhor compreender o Universo como ele realmente é do que persistir no engano, por mais satisfatório e tranquilizador que possa ser.”

SAGAN, C. **O mundo assombrado pelos demônios:** a ciência vista como uma vela no escuro. Tradução de Rosaura Eichemberg. São Paulo: Companhia das Letras, 2006. P.25.

RESUMO

A iluminação em granjas para a produção do frango de corte não somente influencia o comportamento dos frangos, como também seu sistema fisiológico e sua engorda. Além do bem-estar animal, outro fator que deve ser considerado na iluminação de uma granja é o consumo de energia elétrica, cujo impacto pode ser significativo na receita do criador. Entretanto, o processo de controle de iluminação em granjas é pouco explorado se comparado aos processos de ventilação e refrigeração, calefação, abastecimento de água e ração. Por meio desta pesquisa, buscou-se desenvolver um sistema de controle automático de iluminação para granjas e analisar a possibilidade de integrá-lo com luz natural pelo processo de inferência *fuzzy*. Para construí-lo, foi proposta uma modelagem teórica dos materiais necessários à sua construção, assim como suas respectivas simulações computacionais para previsão e análise de seu comportamento. Os materiais foram responsáveis pelas etapas de entrada, processamento e saída de sinais. O sinal de entrada, em porcentagem, trata-se do erro da intensidade de luz em relação ao *setpoint* que incide no interior da granja, e o material responsável por sua coleta foi o módulo GY-302. Após coleta, o sinal foi processado por um sistema de inferência *fuzzy* com o auxílio do *software* MATLAB. Fazendo uso da inferência *fuzzy* avaliou-se ciclicamente o grau de pertinência do sinal de entrada em relação ao valor de *setpoint* para, após o processo de “defuzzificação”, gerar valores de correções em porcentagem, que foram incrementos e decrementos do valor do tempo de disparo para ativar o Triac. Eles foram tratados como sinais de saída e, consequentemente, direcionados ao *Dimmer Shield*, que foi responsável pelo controle de intensidade luminosa da lâmpada LED. Uma vez montado o protótipo, foram feitos testes para investigar seu comportamento em diferentes situações. O primeiro teste consistiu em analisar se o protótipo alcançava os valores de *setpoint* da intensidade luminosa sem a interferência de luz externa. As lâmpadas LED foram iniciadas com 0 e 100% de sua potência elétrica, e durante esses testes o protótipo alcançou os valores de *setpoint*. No segundo teste, o protótipo foi exposto à fonte de luz externa ao sistema, gerada por uma lâmpada incandescente para simular a incidência de luz natural. A intensidade da lâmpada incandescente foi controlada por um *dimmer* analógico (manual) e verificada por um luxímetro comercial. Nesses testes, o protótipo foi forçado a iniciar com o valor de *setpoint* e a tensão elétrica nesse instante foi aferida. Em um determinado instante, a luz externa era inserida no sistema

e, em outro, a luz externa era retirada. O protótipo indicou graficamente o aumento da intensidade luminosa quando a luz externa era inserida, corrigindo-a até alcançar novamente o *setpoint*. Durante o tempo em que o *setpoint* foi alcançado, a tensão das lâmpadas LED foi aferida novamente, a qual mostrou uma diminuição de seus valores, ou seja, a diminuição da potência consumida por elas. O protótipo também se mostrou efetivo no controle de mudanças de *setpoint*, assim como no controle das horas necessárias para a escuridão na granja.

Palavras-chave: Frango de corte. Bem-estar animal. Energia elétrica. Luz. Inferência Fuzzy.

ABSTRACT

Lighting on farms for broiler production not only influences the behavior of broilers, but also their physiological system and their fattening. In addition to animal welfare, another factor that must be considered when lighting a farm is the consumption of electricity, which can have a significant impact on the creator's income. However, the lighting control process in farms is little explored compared to the processes of ventilation and cooling, heating, water supply and feed. Through this research, we sought to develop an automatic lighting control system for farms and analyze the possibility of integrating it with natural light through the fuzzy inference process. To build it, a theoretical modeling of the materials needed for its construction was proposed, as well as their respective computer simulations for prediction and analysis of their behavior. The materials were responsible for the input, processing and output stages of signals. The input signal, in percentage, is the error of light intensity in relation to the setpoint that affects the interior of the farm, and the material responsible for its collection was the GY-302 module. After collection, the signal was processed by a fuzzy inference system with the help of MATLAB software. Using fuzzy inference, the degree of relevance of the input signal in relation to the setpoint value was cyclically evaluated to, after the "defuzzification" process, generate correction values in percentage, which were increments and decrements of the value of the shot to activate the Triac. They were treated as output signals and, consequently, directed to the Dimmer Shield, which was responsible for controlling the luminous intensity of the LED lamp. Once the prototype was assembled, tests were carried out to investigate its behavior in different situations. The first test consisted of analyzing whether the prototype reached the light intensity setpoint values without the interference of external light. The LED lamps were started with 0 and 100% of their electrical power, and during these tests the prototype reached the setpoint values. In the second test, the prototype was exposed to a light source external to the system, generated by an incandescent lamp to simulate the incidence of natural light. The intensity of the incandescent lamp was controlled by an analog dimmer (manual) and verified by a commercial lux meter. In these tests, the prototype was forced to start with the setpoint value and the electrical voltage at that moment was measured. At a given moment, the external light was inserted into the system and, at another, the external light was removed. The prototype graphically indicated the increase in light intensity when the external light was inserted,

correcting it until reaching the setpoint again. During the time that the setpoint was reached, the voltage of the LED lamps was measured again, which showed a decrease in its values, that is, a decrease in the power consumed by them. The prototype also proved effective in controlling setpoint changes, as well as controlling the hours required for darkness on the farm.

Keywords: Broiler chicken. Animal welfare. Electricity. Light. Fuzzy inference.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O Brasil é atualmente o maior exportador de proteína animal do mundo. Em 2018, somente com a exportação de frango, o país arrecadou um valor bruto de 14,452 bilhões de dólares. O principal importador de frango do Brasil é a China, que até fevereiro de 2019 comprou 41,7 mil toneladas desse produto, seguido pela Arábia Saudita, que no mesmo período adquiriu cerca de 38,3 mil toneladas (EMBRAPA, 2019; REUTERS, 2019).No entanto, inicialmente a produção de carne de frango no país não era executada de forma exclusiva por se concentrar no sistema familiar, ou seja, era desenvolvida por famílias que concomitantemente à carne de frango produziam leite, carne bovina e suína. Outra característica desse período é o perfil de subsistência da produção, pois somente os frangos excedentes eram vendidos (ZEN *et al.*, 2014).

Esse perfil começou a mudar na década de 1960, com a implantação do formato de integração, resultado de uma parceria entre frigoríficos e produtores. Apesar de possibilitar o aumento da produção, é importante destacar que esse formato não colocou os produtores em uma posição financeira confortável, pois os frigoríficos ganharam um enorme poder sobre eles (CAMPOS, 2016). Contudo, isso trouxe grandes avanços tecnológicos como ventilação, abastecimento de água e ração, isolamento da construção (ZEN et al., 2014; PECHE, 2017). Mesmo assim, ainda há itens que não tiveram seus possíveis potenciais explorados, e um exemplo é a iluminação (PECHE, 2017).

Oferecer uma iluminação adequada em uma granja é muito importante, pois possibilita o bem-estar do frango, além de melhorar índices de produção como crescimento e engorda (PECHE, 2017; COBB-VANTRESS, 2018). Ainda, para melhorar índices econômicos referentes à eficiência energética, é recomendado por Castro e Rancura (2018) que projetos de luminotécnica integrem luz natural e artificial.

Com o objetivo de fazer essa integração entre luz natural e artificial em uma granja, neste trabalho é proposto um sistema de iluminação controlado por lógica *fuzzy*. Esse sistema será composto de um Arduino, sensores de luz e periféricos conectados ao Arduino para o controle das lâmpadas.

1.1 Objetivos

A questão-problema do trabalho foi:

a) É possível fazer um sistema de controle de iluminação que possa integrar luz natural com artificial em uma granja de frangos de corte?

O objetivo deste trabalho é desenvolver um protótipo que possa controlar de forma precisa a iluminação artificial de uma granja de aves de corte de modo integrado à luz natural fornecendo a quantidade de luminância necessária às aves para seu desenvolvimento e bem-estar.

1.1.1 Objetivos específicos

Este trabalho visou cumprir os seguintes objetivos:

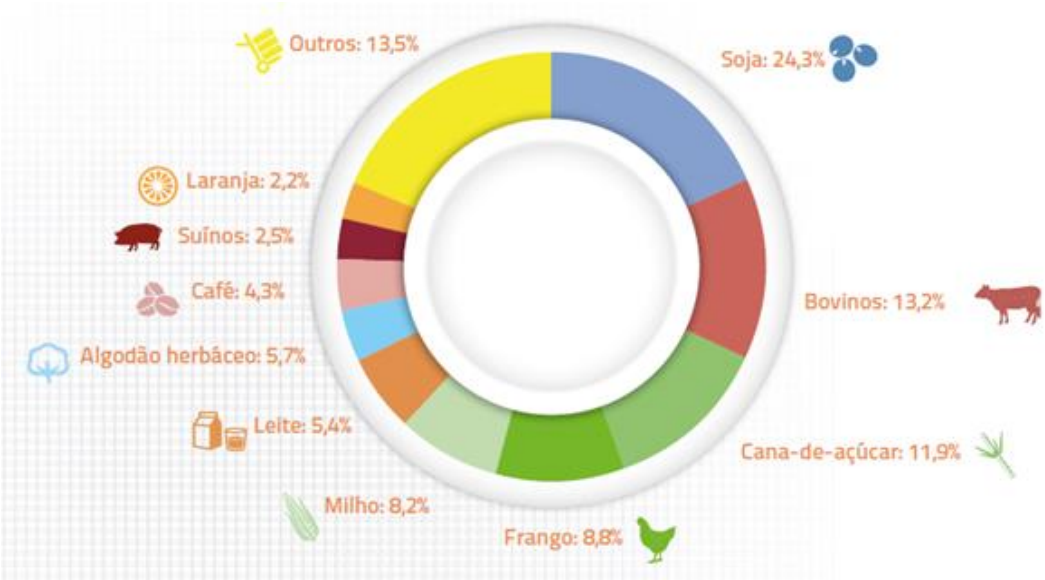
a) definir a quantidade de amostras para a filtragem do sinal de entrada por intermédio da técnica de média móvel;
b) verificar os valores de entradas recebidos pelo módulo GY-302 em comparação com o luxímetro comercial.
c) elaborar o sistema de inferência *fuzzy* definindo as funções de pertinência, assim como os valores das suas variáveis linguísticas, as regras da inferência e o processo de “defuzzyficação”;
d) elaborar com o auxílio do *software* a programação *fuzzy*, analisar o seu comportamento e configurar a comunicação entre o *software* e a plataforma Arduino;
e) analisar o comportamento do Triac e definir os parâmetros para o seu funcionamento por meio de simulações;
f) analisar o comportamento do protótipo final em situações diversas.

2 REVISÃO DE LITERATURA

2.1 Avicultura

Mesmo com uma queda nas exportações de 2018, que resultou na arrecadação de US$ 773 milhões a menos em comparação a 2017, o Brasil é o maior exportador mundial de carne bovina e de frango. A produção de carne de frango em 2018 foi de 4.018 milhões de toneladas, cujo valor comercial bruto atingiu US$ 14,452 bilhões. Desse total, US$ 6,412 bilhões foram obtidos por meio de exportação (EMBRAPA, 2019).

Figura 1 – Valor bruto da produção brasileira em 2018 (US$ 164,23 bilhões)



Fonte: EMBRAPA (2019).

O principal destino das exportações brasileiras de carne de frango tem sido o continente asiático. Em março de 2019, a Ásia importou cerca 196,5 mil toneladas de frango, o que representa cerca de 58,7% do volume de exportação desse período (REUTERS, 2019).

Para conseguir atingir esses mercados, o Brasil está se adaptando a costumes, particularidades e demandas técnicas de cada país. Por exemplo, os países islâmicos exigem que, no abate do animal, seja realizado um ritual chamado de *Halal*. Por esse motivo, os principais frigoríficos brasileiros adaptaram o seu formato de abate para atender a essa exigência (DALL'AZEN; WEISE, 2014).

Em razão da pesquisa, da ciência e da tecnologia introduzidas na indústria, a produção de carne de frango no Brasil cresceu 4,5 vezes entre 1990 e 2010. Esse incremento provocou uma alteração nos hábitos do mercado interno, cujo consumo por habitante aumentou de 13,1 kg para 44,06 kg (ESPINDOLA, 2012). Watanabe (2016) também relaciona esse crescimento a um segundo fator, como consequência da crise econômica enfrentada pelo Brasil, já que o preço do frango é bem menor que o da carne bovina e suína.

No entanto, inicialmente, antes da década de 1970, a produção de carne de frango no Brasil era feita por meio de sistema familiar, o qual é utilizado até hoje por alguns criadores no Brasil. Além de criar frangos, eles exerciam outras atividades agropecuárias como produção de leite e de carne bovina e suína. A criação de frangos era praticamente de subsistência, visto que somente os frangos excedentes eram comercializados (ZEN *et al.*, 2014).

Atualmente, o sistema de avicultura industrial do Brasil é estruturado em um formato chamado de *integração*, em que há a integração entre a indústria (frigoríficos) e o produtor. Esse sistema começou a ser utilizado na década de 1960 e possibilitou um grande avanço da avicultura brasileira (WATANABE, 2016). Nele, os frigoríficos entregam os pintinhos, os remédios e as rações aos produtores que, dessa forma, são obrigados a vender o frango exclusivamente ao frigorífico que entregou os insumos. Descontados posteriormente os custos de produção, o produtor recebe a sua devida remuneração (CAMPOS, 2016).

Essa relação, no entanto, também envolve todo o conflito de interesses relacionado a qualquer sociedade. Campos (2016) afirma que esse sistema oferece aos frigoríficos um enorme poder para imporem os seus critérios de qualidade, o que muitas vezes resulta em altos investimentos por parte dos avicultores. Segundo o autor, para definir bem as obrigações de cada parte (frigorífico e produtor) é que foi criado o projeto de Lei 6.459/2013.

Além de ter que aceitar as imposições de seus parceiros comerciais, os avicultores ficam sujeitos aos mais diversos tipos de risco econômico. No boletim informativo do Sistema Faep (EXTENSÃO..., 2018), é apresentado um exemplo de um frigorífico que exigiu do avicultor a instalação de mais cinco ventiladores em seu aviário, além dos 10 que já estavam em funcionamento. Essa imposição representou um grande ônus ao avicultor, uma vez que, nesse mesmo período, foi cancelado o subsídio energético

governamental denominado *tarifa rural noturna*, que cobra apenas 60% do valor da tarifa de energia elétrica consumida no período das 21:30 à 6:00h.

Assim, faz-se evidente a importância da produção de carne de frango no Brasil, em vista do crescimento do consumo nacional e o direcionamento das exportações, principalmente para países asiáticos, que ocorre enquanto os avicultores se empenham em adaptar o processo de produção para atender às exigências de seus clientes. A mudança de formato do sistema familiar para o sistema de integração, que proporcionou o aumento da produção, trouxe também conflitos de interesses, com destaque para o aumento das exigências técnicas e a elevação de custos, o que torna o avicultor um elo fraco.

2.2 Eficiência energética

Eficiência energética envolve implantar técnicas de administração, economia e engenharia nos sistemas energéticos. Esse conceito ficou em evidência no Brasil depois da crise energética enfrentada em 2001, que expôs a necessidade de racionalizar energia elétrica (VIANA *et al.*, 2012).

No período de 1990 a 2000, a demanda por energia elétrica no Brasil cresceu 49%, em contrapartida da capacidade instalada, que cresceu apenas 35%. Já no início dos anos 2000, 90% da energia elétrica do país era gerada por usinas hidrelétricas, mesmo período em que os baixos índices pluviométricos foram insuficientes para manter os reservatórios em níveis satisfatórios. Esse quadro fez com que o sistema não suportasse a demanda energética (TOLMASQUIM, 2000; LAGO, 2020).

Durante a crise, o governo implementou um plano de racionamento cujo tópico mais evidente era a exigência de que a população economizasse 20% do consumo geral de energia elétrica. Esse racionamento obteve um bom resultado e foi encerrado em 2002; porém, como consequência, ocasionou grandes impactos na economia, como a queda de 41% na oferta de empregos no período de abril a maio daquele ano (OLIVEIRA, 2021; FUTEMA, 2001).

Segundo Viana *et al.* (2012), tal impacto na economia se justifica porque todos os processos inerentes às atividades humanas carecem de energia e, por consequência, de recursos energéticos. Esses recursos nada mais são que reservas ou fluxos de energia – tratados como renováveis e fósseis (não renováveis) – dispostos

naturalmente e utilizados pela humanidade de modo a atender, assim, às suas necessidades.

Enquanto as reservas de energia fóssil são inevitavelmente finitas e reduzidas à medida que são consumidas, os recursos energéticos renováveis são dados por fluxos naturais (VIANA *et al.*, 2012). Entretanto, segundo os mesmos autores, as fontes renováveis podem perder sua característica de renovação quando são exploradas de modo que seu ciclo natural de reposição não seja respeitado.

O estilo de vida da sociedade industrial demanda um consumo excessivo de recursos não renováveis. Se isso não for controlado, pode levar a humanidade a uma catástrofe ambiental (GUEDES; REIS; JOUCOSKI, 2015). Esse modo de viver faz com que temas relacionados à política energética e ambiental, como fontes renováveis de energia e diversificação de matrizes energéticas, se tornem mais relevantes no mundo para a promoção do desenvolvimento sustentável. Além disso, esses temas ficam em evidência porque a sociedade e o governo já se conscientizaram de que o meio ambiente e os recursos que ele nos disponibiliza são escassos e finitos (LOPES; TAQUES, 2016).

Para implantar políticas de eficiência energética, o governo brasileiro tem instituições que a regulamentam nas suas mais diversificadas fontes energéticas: petróleo, carvão, eletricidade etc. Para gerenciar a eficiência relacionada à energia elétrica, o governo tem o Ministério de Minas e Energia, a Agência Nacional de Energia Elétrica (Aneel) e a Eletrobras, que é responsável pelo Programa Nacional de Conservação de Energia Elétrica (Procel) (VIANA *et al.*, 2012).

Para se ter uma ideia da importância desse programa, por meio de suas atividades foram poupados cerca de 22,99 bilhões de kWh em 2018, o que corresponde a 4,87% do consumo de energia elétrica do país e a uma economia de 5,378 bilhões de reais (ELETROBRAS, 2019). A maioria dos resultados do Procel foi obtida por intermédio do Programa Brasileiro de Etiquetagem (Selo Procel), em que um selo é concedido aos equipamentos mais eficientes no consumo de energia elétrica. O Procel emprega recursos da Eletrobras e os investimentos das concessionárias distribuidoras de energia elétrica nos programas de eficiência energética (VIANA *et al.*, 2012; ELETROBRAS, 2019).

Outro ponto que foi exposto após certos períodos de estiagem no país, em que os níveis das represas atingiram pontos críticos, é a urgência da diversificação da matriz

energética brasileira, pois atualmente 70% desta é formada por hidrelétricas (SANDY *et al.*, 2018).

Para diversificar a matriz energética nacional, pode-se utilizar os seis tipos de energias alternativas renováveis mais empregadas no Brasil: biomassa, eólica, geotérmica, hidráulica, marítima e solar. Entretanto, esses tipos de energia ainda causam baixos impactos ambientais quando comparados a fontes não renováveis (NASCIMENTO; ALVES, 2016).

Além de diminuir o consumo de energia nos seus mais variados modos, a aplicação dos métodos de eficiência energética possibilita retardar a aplicação de recursos financeiros voltados para a ampliação da capacidade instalada e da produção de energia, bem como oportuniza a redução dos impactos ambientais gerados por esses processos (SANDY *et al.*, 2018).

2.3 Luminotécnica

De acordo com Cavalin e Cervelin (2014, p. 107), “luminotécnica é o estudo das técnicas das fontes e iluminação artificial através da energia elétrica”. No Brasil, os requisitos referentes à iluminação, ambientação e segurança são determinados pela norma técnica ABNT NBR ISO/CIE 8995-1, da Associação Brasileira de Normas Técnicas – ABNT (2013). Essa norma estabelece os valores de iluminâncias médias e mínimas dos espaços internos dos mais variados setores para a implantação de luminotécnica (ABNT, 2013).

No entanto, a existência da norma não tem sido o suficiente para promover todas as melhorias possíveis, uma vez que os sistemas de iluminação no Brasil se mostram inadequados por ainda utilizarem luminárias e lâmpadas ineficientes ou pela falta de uso de técnicas e/ou hábitos corretos (VIANA *et al.*, 2012).

Do total de energia elétrica consumida no Brasil, 15% são referente à iluminação; desta, o setor industrial consome de 2 a 8%. A instalação luminotécnica na indústria é um grande foco de ineficiência energética em razão de vários fatores técnicos (MAMEDE, 2017). Entretanto, não existe a possibilidade de se aplicar os cálculos para luminotécnicas sem conhecer previamente as grandezas fundamentais a que a elas estão associadas (CREDER, 2016).

2.3.1 Grandezas luminotécnicas

Segundo ABNT (2013), existem várias grandezas que podem ser utilizadas em projetos luminotécnicos. Para efeito deste trabalho, serão abordadas algumas das grandezas básicas, conforme a seguir.

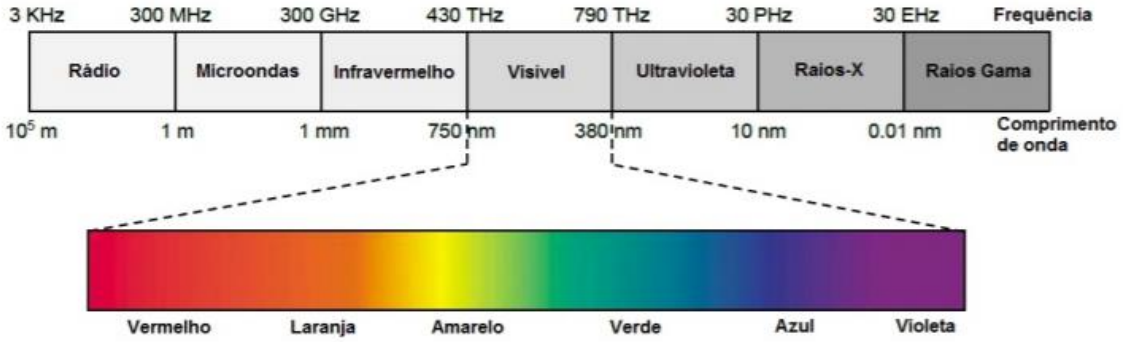
2.3.1.1 Luz

De acordo com Osaki (2018, p. 15):

> Luz, ou radiação visível, trata-se da energia em forma de ondas eletromagnéticas capazes de excitar o sistema humano olho-cérebro, produzindo uma sensação visual. As ondas eletromagnéticas não necessitam de um meio para sua transmissão. Apesar de passarem através dos meios sólido, líquido e gasoso, propagam-se mais eficientemente no vácuo, onde não há nada para absorver a energia radiante.

Um dos modos de medição da luz é por meio do espectro eletromagnético, como é mostrado na Figura 2.

Figura 2 – Espectro eletromagnético da luz



Fonte: PATHAK *et al.* (2015).

Observa-se a faixa visível para o olho humano: entre 380 e 780 nm.

2.3.1.2 Fluxo luminoso

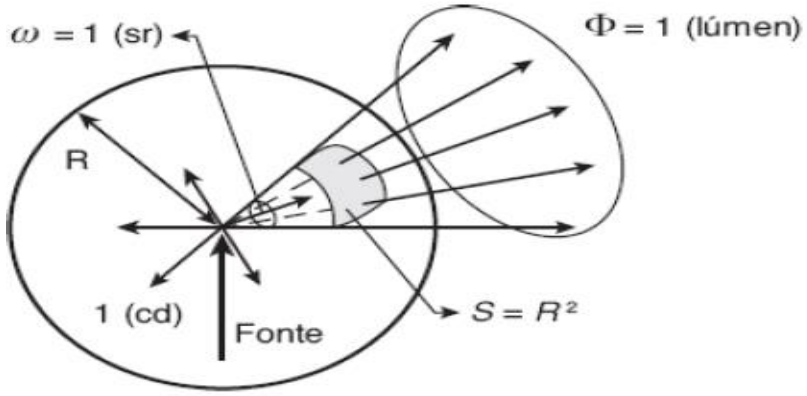
Segundo Mamede (2017, p. 114):

> É a potência de radiação emitida por uma fonte luminosa em todas as direções do espaço. Sua unidade é o lúmen, que representa a quantidade de luz irradiada através de uma abertura de 1 m^2 feita na superfície de uma esfera de 1 m de raio por uma fonte luminosa de intensidade igual a 1

> candela, em todas as direções, colocada no seu interior e posicionada no centro.

Conforme é ilustrado na Figura 3 a seguir, se for inserida uma fonte luminosa, com intensidade de 1 candela dentro de uma esfera de raio de 1 m, essa intensidade será projetada para todas as direções. Dentro do ângulo que fizer projetar 1 m^2 terá 1 lúmen (CREDER, 2016).

Figura 3 – Definição de lúmen



Fonte: CREDER (2016).

2.3.1.3 Iluminância: lux (lx)

Segundo Creder (2016, p. 411), "iluminância, anteriormente chamada de iluminamento, é definida como a relação entre o fluxo luminoso, em lumens, que incide perpendicularmente sobre uma superfície plana, pela área dessa superfície em m^2".

A iluminância é calculada por meio da equação (1) (CREDER, 2016):

$$E = \frac{\psi\ (lm)}{S\ (m^2)} \quad (1)$$

Em que E é a iluminância (lm/m^2), ψ é a luminância (lm) e S é a área (m^2).

A iluminância é umas das principais grandezas, uma vez que possibilita o cálculo da quantidade de lâmpadas que atenderá as necessidades visuais do ambiente.

2.3.1.4 Eficiência luminosa

Conforme Viana *et al*. (2012, p. 125), "é o quociente do fluxo luminoso total emitido por uma fonte de luz em lúmens e a potência por ela consumida em Watts".

A eficiência luminosa é calculada por meio da equação (2) (CREDER, 2016):

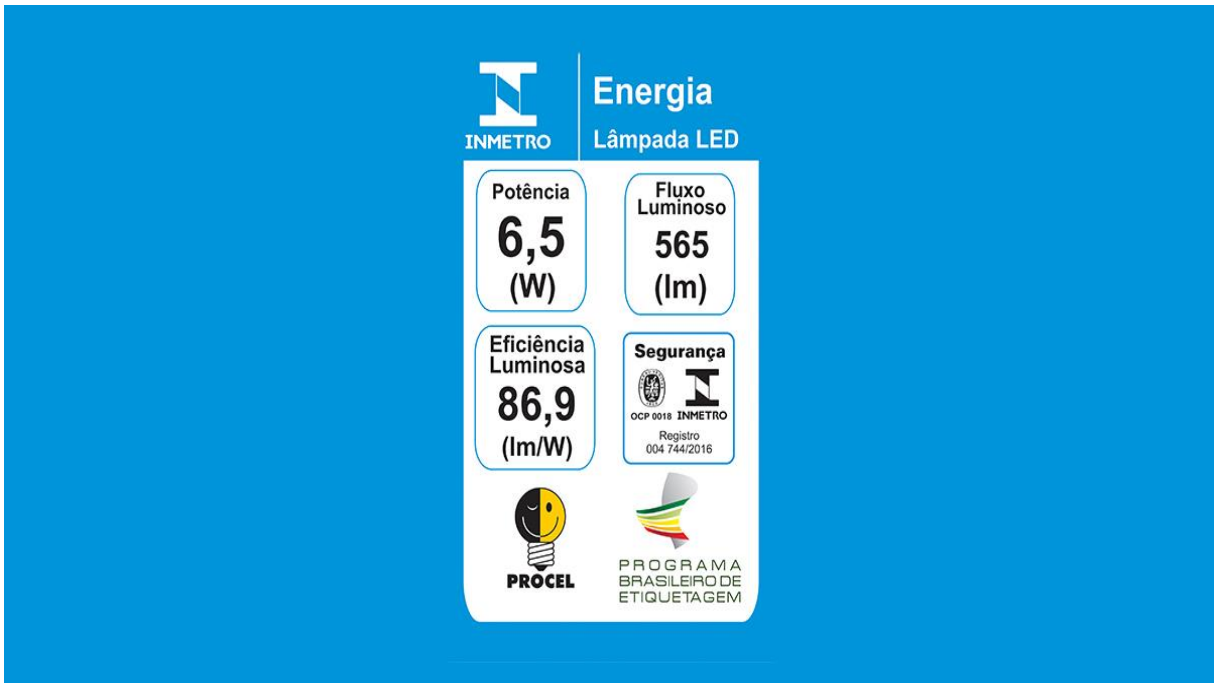
$$n = \frac{\psi\ (lm)}{P\ (w)} \qquad (2)$$

Em que n é a eficiência luminosa (lm/w), ψ é a luminância (lm) e P é a potência (w).

Para exemplificar o que é eficiência luminosa, serão utilizados a seguir os dados técnicos de duas lâmpadas diferentes (Figura 4; Figura 5). Ambas ilustram grandezas que possibilitam o cálculo da eficiência luminosa, por meio da equação (2).

A Figura 4 representa uma lâmpada LED com fluxo luminoso de 565 lm e potência de 6,5 W; o valor de eficiência luminosa obtido por meio da equação (2) foi de 86,9 lm/W. Já a Figura 5 representa uma lâmpada fluorescente; o valor referente à eficiência luminosa, usando a mesma equação, é de 60 lm/w. Essa comparação demonstra que a lâmpada LED é mais eficiente que a lâmpada fluorescente.

Figura 4 – Dados técnicos da lâmpada LED



Fonte: G-LIGHT (2017).

Figura 5 – Dados técnicos da lâmpada fluorescente

Fonte: FOXLUX (2021).

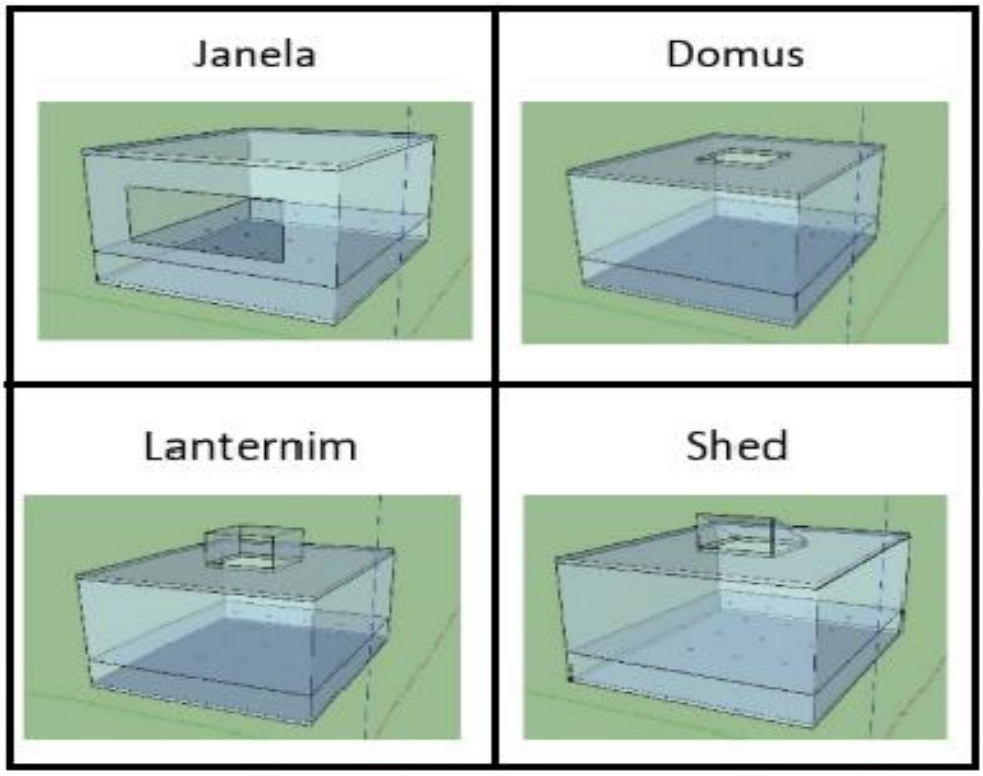
2.3.2 Luz natural em luminotécnica

A luz natural pode fornecer parte ou toda a iluminância necessária a um ambiente fechado. Porém, sempre essa configuração necessitará de uma iluminação artificial adicional em decorrência da mudança sazonal da luz natural (ABNT, 2013).

Castro e Rancura (2018) recomendam que um projeto de luminotécnica integre luz artificial e natural. Entretanto, isso requer cautela, pois a luz natural em excesso pode tornar o ambiente visualmente desconfortável.

Uma das maneiras de se aproveitar a luz natural é por intermédio da construção de espaços para a luz conseguir penetrar o ambiente. Existem pelo menos duas formas de se construir esse espaço, conforme pode ser observado na Figura 6 a seguir.

Figura 6 – Tipologias de aberturas analisadas em relação ao ambiente



Fonte: MAPELLI; LARANJA; ALVAREZ (2018).

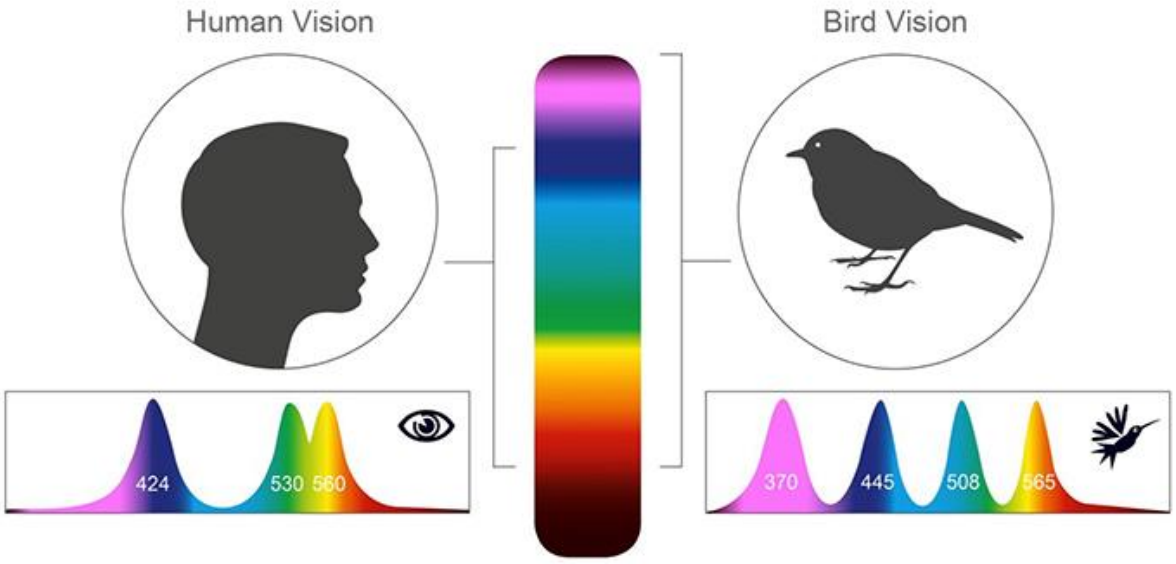
A forma “janela” é criada por meio de aberturas laterais de espaços horizontais. Outras formas são os espaços superiores, chamados de aberturas zenitais, que se configuram de três formas: *domus*, lanternim e *shed* (MAPELLI; LARANJA; ALVAREZ, 2018).

2.4 Luz e aves

O olho humano é tricromático, pois apresenta três tipos de cones que são sensíveis ao roxo, ao verde e ao azul. Já o olho das aves é tetracromático, ou seja, apresenta cinco tipos diferentes de cones, sendo o 4.º sensível à luz ultravioleta e o 5.º capaz de detectar movimentos muito rápidos. Especificamente nos frangos, os olhos ficam posicionados lateralmente em relação ao seu crânio, o que lhes proporciona um ângulo visual amplo (PECHE, 2017).

Na Figura 7, é possível notar mais detalhadamente a diferença entre a percepção de cores entre humanos (tricromáticos) e aves (tetracromáticas).

Figura 7 – Percepção de cores entre humanos e aves



Fonte: ROMANZOTI (2019).

Propiciar uma iluminação adequada a um frango, e consequentemente a uma granja, é de fundamental importância para que esses animais consigam exercer suas atividades cotidianas. Outro fator importante é a alternância de períodos com luz e escuridão para simular as condições da natureza, com vistas a colaborar com o desenvolvimento de suas funções fisiológicas, hormonais, entre outras (PECHE, 2017).

Essa influência pode ser atestada, por exemplo, nos primeiros períodos de vida do frango, no qual ele precisa de uma estimulação luminosa correta para que o consumo alimentar ocorra normalmente e, por consequência, o seu sistema digestivo e imunológico se desenvolvam corretamente. Outro fator importante é que a luz deve ser distribuída uniformemente na granja (COBB-VANTRESS, 2018).

Nos primeiros modelos de aviários, os frangos recebiam uma alta incidência de luz natural, pois as laterais do ambiente eram abertas. Com os avanços dos estudos técnicos e zootécnicos, as granjas começaram a fazer mudanças em sua configuração. Um exemplo disso são as granjas no formato *dark house*, (Figura 8), que utilizam o sistema de pressão negativa e controlam a entrada de luz por meio de cortinas pretas nas laterais do galpão. Similar a esse formato existem os galpões no formato *blue house* (Figura 9), que utilizam em suas laterais cortinas azuis, e os galpões no formato *yellow house* (Figura 10), com cortinas amarelas em suas laterais (VERCELLINO, 2012).

Figura 8 – Granja *Dark House*

Fonte: LENHARD (2017).

Figura 9 – Granja *Blue House*

Fonte: CAMPOS (2017).

Figura 10 – Granja *Yellow House*

Fonte: CASA DAS CERCAS (2017).

Os principais itens monitorados nas granjas são a ventilação e a refrigeração, a calefação, o abastecimento de água e ração, o isolamento da construção, a cama e a iluminação. Entretanto, de todos esses itens, o que menos é levado em consideração pelos criadores é o sistema de iluminação. Se for necessário mudar qualquer processo da granja, por exemplo, a refrigeração ou a substituição de bebedouro, o produtor consultará um técnico da área; contudo, quando se trata da troca de fontes luminosas com o objetivo de diminuir o consumo de energia, dificilmente um especialista será contatado (PECHE, 2017).

No entanto, apesar dessa situação, existem técnicas de iluminação de granjas, como o programa de luz. Trata-se de “uma técnica importante de manejo para a produção do frango de corte e se compõe de, pelo menos de três aspectos, a saber: comprimento da onda, intensidade da luz, duração e distribuição do fotoperíodo” (SCHWEAN-LARDNER; CLASSEN, 2010, p. 2).

O programa de luz é aplicado levando em consideração as mudanças que acontecem com o frango em razão da idade e do peso final. Existem programas de luz que controlam o ganho de peso do frango entre 7 e 21 dias, pois o ganho excessivo traz problemas como morte súbita, problemas nas pernas etc. Pesquisas demonstram

que programas de luz que apresentam um intervalo de escuridão de seis horas melhoram o sistema imunológico do frango (COBB-VANTRESS, 2018).

Schwean-Lardner e Classen (2010) explicam que taxas exageradas de iluminação – por exemplo, 23 horas – podem ser prejudiciais na obtenção de uma taxa de crescimento adequada e de bem-estar animal, bem como podem resultar em mortalidade e ganho de peso. Outro fator é que os programas de luz podem ser iguais entre as diferentes raças e sexos do frango.

Para o frango ganhar peso de forma adequada até a idade de sete dias, é recomendado uma intensidade luminosa de 25 lux. Passado esse período, ou quando o frango atingir 160 gramas, é recomendado que a intensidade luminosa seja reduzida gradativamente para 5 ou 10 lux, sem que varie para acima de 20% em ambos os casos. Outro fator importante é o controle, em horas, da exposição do frango à luz (COBB-VANTRESS, 2018).

Alguns exemplos de duração, em horas, do fotoperíodo são apresentados nas tabelas de 1 a 4 a seguir.

Tabela 1 **– Programa de luz padrão: opção 1**

Idade (dias)	**Horas de escuro**	**Alteração de horas**
0	0	0
1	1	1
130-180 g	6	5
5 dias antes do abate	5	1
4 dias antes do abate	4	1
3 dias antes do abate	3	1
2 dias antes do abate	2	1
1 dia antes do abate	1	1

Fonte: Adaptado de COBB-VANTRESS (2018).

Nota: Peso no abate < 2,5 kg.

Tabela 2 **–** Programa de **luz padrão: opção 2**

Idade (dias)	**Horas de escuro**	**Alteração de horas**
0	0	0
1	1	1
130-180 g	6	6
21	5	1
28	4	1
35	3	1
2 dias antes do abate	2	1
1 dia antes do abate	1	1

Fonte: Adaptado de COBB-VANTRESS (2018).

Nota: Peso no abate < 2,5 kg.

Tabela 3 – Programa de luz padrão: opção 3

Idade (dias)	**Horas de escuro**	**Alteração de horas**
0	0	0
1	1	1
130-180 g	8	7
21	7	1
28	6	1
35	6	1
42	4	1
49	3	1
3 dias antes do abate	3	1
2 dias antes do abate	2	1
1 dia antes do abate	1	1

Fonte: Adaptado de COBB-VANTRESS (2018).

Nota: Peso no abate: 2,5 kg-3 kg.

Tabela 4 **– Programa de luz padrão: opção 4**

Idade (dias)	Horas de escuro	Alteração de horas
0	0	0
1	1	1
130-180 g	10	9
22	9	1
28	8	1
35	7	1
42	6	1
49	5	1
5 dias antes do abate	5	0
4 dias antes do abate	4	1
3 dias antes do abate	3	1
2 dias antes do abate	2	1
1 dia antes do abate	1	1

Fonte: Adaptado de COBB-VANTRESS (2018).

Nota: Peso ao abate > 3 kg.

É importante observar que a duração, em dias, do fotoperíodo tem relação com o peso final desejado para o abate.

O sistema de iluminação não deve apenas propiciar boas condições para o bem-estar do frango, mas também melhorar os índices ambientais e econômicos. Para isso, é aconselhável usar tecnologias mais eficientes e com embasamento técnico e científico (PECHE, 2017).

2.5 Lógica *fuzzy* ou lógica nebulosa

A teoria dos conjuntos baseia-se no conceito de um indivíduo, uma variável ou um elemento que pertence ou não a um determinado conjunto. As regras que determinarão se esse elemento pertence ou não ao conjunto serão muito claras, de modo que sempre haverá apenas duas possibilidades para ele: pertence ou não pertence; simplificando, “sim” e “não”. Mesmo usando a probabilidade, as possibilidades de “sim” e “não” desse elemento não mudam, pois a probabilidade apresenta apenas uma determinada chance de o elemento pertencer ou não ao conjunto (CHEN; PHAM, 2001).

Similar à teoria dos conjuntos, os computadores foram baseados no sistema binário, em que as possibilidades de estado de um *bit* podem ser somente “0” ou “1”. O sistema binário é baseando na lógica booleana, em que as respostas podem ser dadas como verdadeira ou falsas (OLIVEIRA, 2018).

Em contrapartida, a lógica *fuzzy* trabalha diferentemente da lógica booleana, pois ela representa o quão o elemento está próximo de pertencer ao conjunto, ou seja, aceita incertezas.

> O computador trabalha naturalmente com duas situações, o falso e o verdadeiro, o “0” e o “1”, enquanto o ser humano trabalha com nuances, com gradações, com indefinições e incertezas, em uma palavra, com a “nebulosidade”. E é com a Lógica Nebulosa que vamos aproximar a capacidade de decisão do computador da capacidade de decisão do ser humano (OLIVEIRA, 2018, p 113).

Lotfi Zadeh foi quem apresentou os conceitos da lógica *fuzzy* em 1965 e, mais tarde, desenvolveu vários métodos para se trabalhar com *fuzzy*. Vários anos ainda foram necessários para que outros cientistas a compreendessem e aplicassem esse novo conceito (KASABOV, 1998).

No método tradicional dos conjuntos, seus elementos são apresentados da seguinte forma:

1) μA (u) = 1, se *u* é um elemento do conjunto A; e
2) μA (u) = 0, se *u* não for um elemento do conjunto A.

Já nos conjuntos *fuzzy* é aceito que um elemento possa pertencer parcialmente a um ou mais conjuntos. Além disso, seu grau de pertinência é definido pela função a seguir (KASABOV, 1998).

1) μA (u): U→ [0,1].

Em que U é chamado de universo e A é um subconjunto difuso de U (KASABOV, 1998).

Oliveira (2018) demonstra um exemplo que evidencia as diferenças entre a lógica booleana e a lógica *fuzzy*. Na primeira parte desse exemplo (Figura 11), a lógica booleana é utilizada para estabelecer se uma pessoa é alta. A medida utilizada para classificar a pessoa como alta é de 1,80 m e pode ser determinada pela seguinte função (OLIVEIRA, 2018):

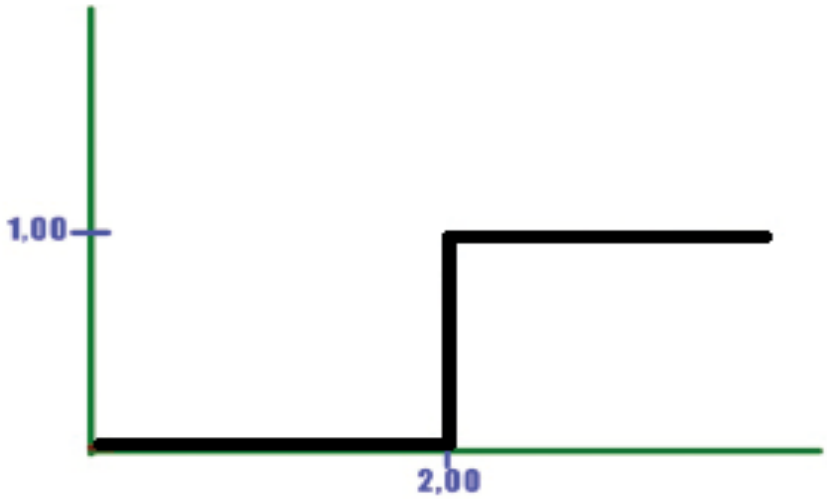
$$F_{(X)} = \begin{cases} 0 \leftrightarrow X \notin A \\ 1 \leftrightarrow X \in A \end{cases} \quad (3)$$

Figura 11 – Conjunto booleano (com limite rígido)

Fonte: OLIVEIRA (2018).

Nota-se que o rapaz de bermuda (Figura 11) tem a altura próxima de 1,80 m; porém, ele não foi classificado como alto pela lógica booleana. A Figura 12 demonstra graficamente essa função.

Figura 12 – Gráfico do conjunto booleano



Fonte: OLIVEIRA (2018).

Na Figura 12, demonstra-se claramente que o valor F(x) somente pode ocupar os valores 0 ou 1. Já na outra parte do exemplo (Figura 13) é utilizada a lógica *fuzzy* para estabelecer se a pessoa é alta. Nela, aceita-se que todas as pessoas são altas, porém,

o que vai ser avaliado é o grau ou a pertinência da pessoa em pertencer ao conjunto “Alto”, que é determinado pela função (4) a seguir (Oliveira, 2018):

$$G_{(X)} = \begin{cases} 0 \leftrightarrow X < 1{,}20m \\ \frac{X}{2} \leftrightarrow 1{,}20m \leq X < 2m \\ 1 \leftrightarrow X \geq 2m \end{cases} \quad (4)$$

A menina da Figura 13 tem um grau de pertinência ao conjunto “Alto” de 60%; Já em relação ao homem de terno, levando em conta que ele tem uma altura de 1,90 m, pode-se definir que seu grau de pertinência é de 95%.

Figura 13 – Conjunto nebuloso (com graus de pertinência)

Fonte: OLIVEIRA (2018).

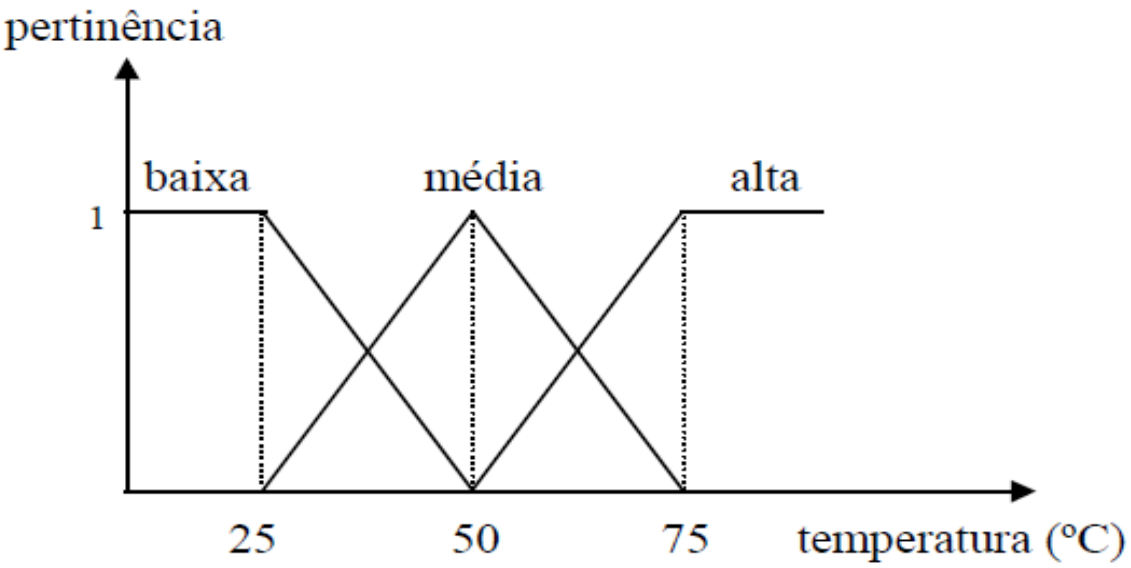
Tanscheit (2004) descreve que, na lógica *fuzzy*, as funções de pertinência são as bases para as variáveis linguísticas e podem assumir valores genéricos como baixo, médio e alto. Essas variáveis apresentam termos utilizados por seres humanos para descrever de forma aproximada processos complexos ou mal definidos. Sua estrutura é definida pelos seguintes termos descritos:

1) N: nome da variável;
2) T(N): conjunto de termos de N, ou seja, o conjunto de nomes dos valores linguísticos de N;
3) X: universo de discurso;

4) G: regra sintática para gerar os valores de N como uma composição de termos de T(N), conectivos lógicos, modificadores e delimitadores;
5) M: regra semântica para associar a cada valor gerado por G um conjunto fuzzy em X.

A exemplificação dos termos citados anteriormente pode ser observada na Figura 14.

Figura 14 – Funções de pertinência para a variável temperatura



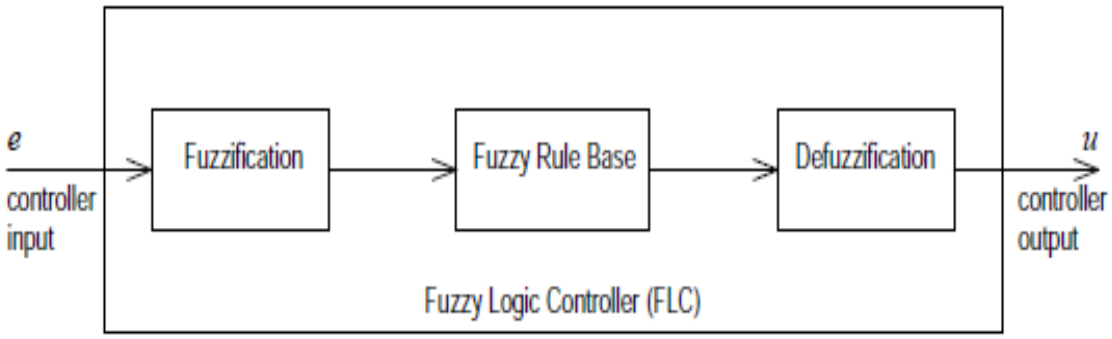
Fonte: TANSCHEIT (2004).

O nome da variável N é *temperatura*; suas variáveis linguísticas – T(N) – são os termos que os seres humanos utilizam comumente para designar sensações térmicas como baixas, médias e altas. O universo de discurso (X) é o *range* que a temperatura pode alcançar, que nesse caso específico é de 0 a 100 °C. A regra sintática (G) pode ser explicada como termos que definem os valores de saída, por exemplo, temperatura não muito alta; assim sendo, a temperatura pode ser definida como temperatura não alta. A regra semântica (M) é o que associa a variável linguística ao seu significado; por exemplo, o valor “média”, que é uma função triangular que se inicia em 25 °C, tem seu pico em 50 °C e termina em 75 °C.

2.5.1 Sistema de inferência *fuzzy*

Na Figura 15, são demonstradas as três partes básicas de um sistema de inferência *fuzzy*, ou controlador lógico *fuzzy*.

Figura 15 – Estrutura geral de um controlador lógico *fuzzy*

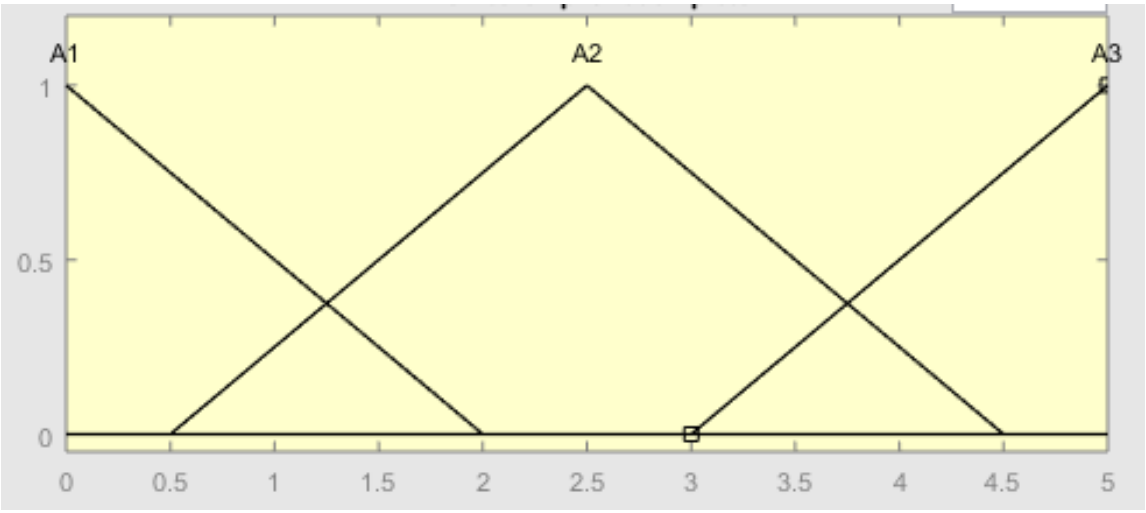


Fonte: CHEN; PHAM (2001).

A primeira parte de "fuzzificação" é a responsável por gerenciar os sinais de entradas do sistema, no qual o sinal físico entra e esse bloco o processa para que ele se adeque às variáveis linguísticas do conjunto *fuzzy* (CHEN; PHAM, 2001).

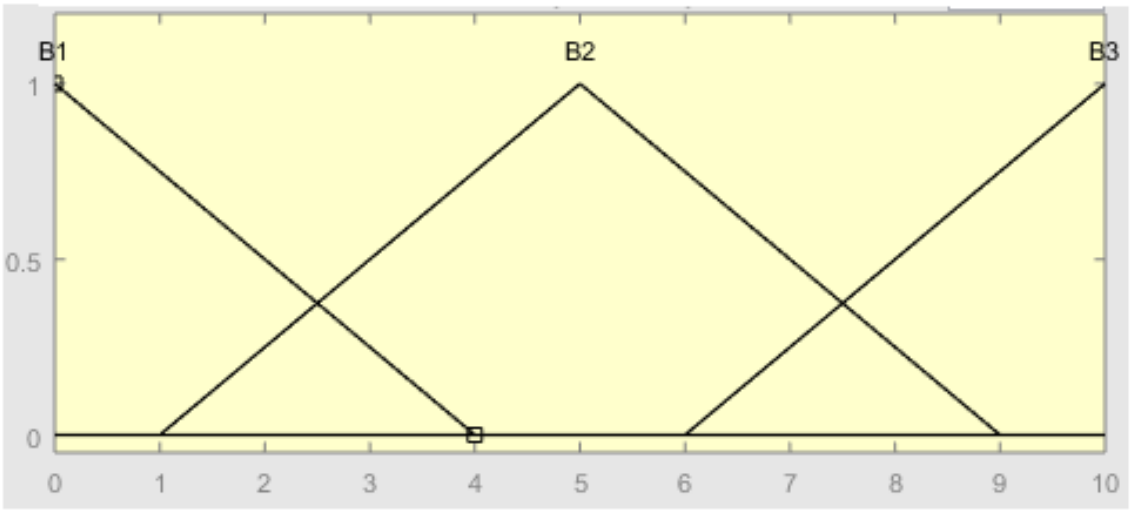
Para exemplificar esse conceito de fuzzificação, é proposto na Figura 16 a seguir um processo genérico em que há duas variáveis de entrada e uma de saída.

Figura 16 – Representação das variáveis linguísticas do Processo A



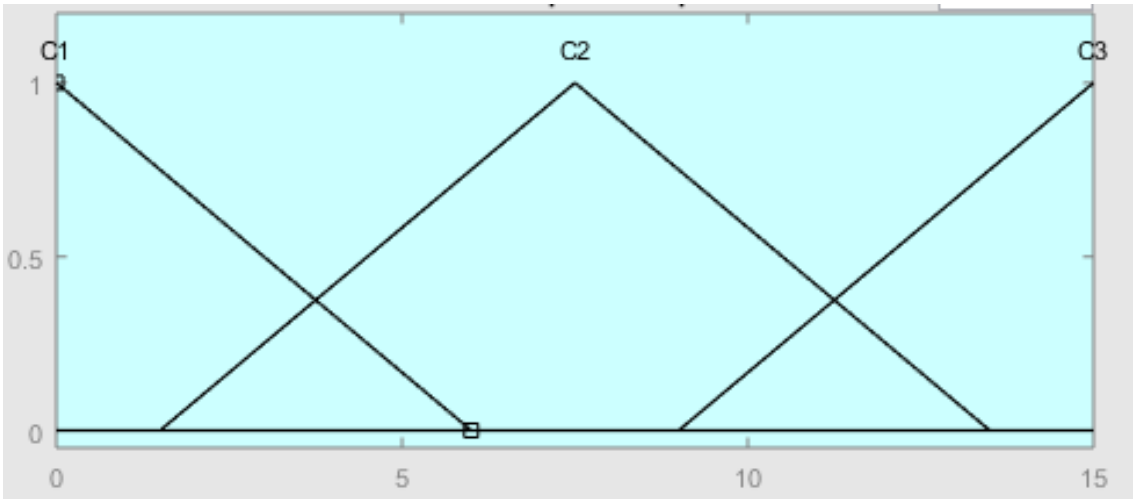
Fonte: o autor.

Figura 17 – Representação das variáveis linguísticas do Processo B



Fonte: o autor.

Figura 18 – Representação das variáveis linguísticas do Processo C



Fonte: o autor.

Os processos A e B são processos de entrada e suas variáveis linguísticas são, respectivamente, A1, A2, A3, B1, B2 e B3. Já o processo C é uma variável de saída e suas variáveis linguísticas são C1, C2 e C3.

O bloco *Fuzzy Rule Base* (ou “inferência”, em português) é responsável por fazer as operações com os conjuntos *fuzzy*. Nele, estão todas as regras de associação de entradas com suas respectivas consequências de saídas (TANSCHEIT, 2004):

1) SE processo A é A1 E Processo B é B1, ENTÃO C é C1;
2) SE processo A é A1 E Processo B é B2, ENTÃO C é C1;
3) SE processo A é A1 E Processo B é B3, ENTÃO C é C2;

4) SE processo A é A2 E Processo B é B1, ENTÃO C é C1;
5) SE processo A é A2 E Processo B é B2, ENTÃO C é C2;
6) SE processo A é A2 E Processo B é B3, ENTÃO C é C2;
7) SE processo A é A3 E Processo B é B1, ENTÃO C é C2;
8) SE processo A é A3 E Processo B é B2, ENTÃO C é C3;
9) SE processo A é A3 E Processo B é B3, ENTÃO C é C3.

As regras do processo genérico foram estabelecidas por meio do operador lógico “E” ou “AND” de forma similar à lógica booleana. Porém, o operador lógico da regra “E” é descrito da seguinte maneira:

a) E-fuzzy(A,B)=min(A,B).

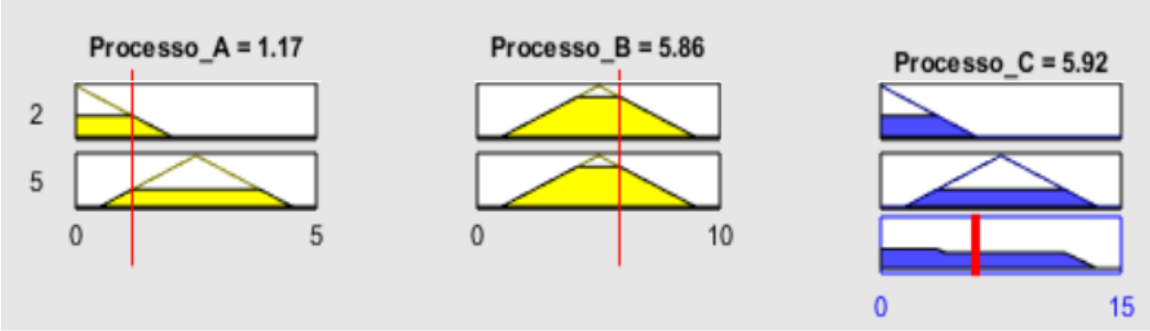
A operador lógico “E”, em lógica *fuzzy*, estabelece que o resultado adotado é o menor valor entre A e B.

O bloco de defuzzificação é o elo que liga os conjuntos de regras do bloco inferência ao sistema a ser controlado. Esse processo de ligação consiste em transformar os sinais do bloco inferência, que são nebulosos, em sinais que o sistema possa aceitar (CHEN; PHAM, 2001).

Se o processo A estiver com um sinal de 1,17 e o processo B estiver com um sinal de 5,86, as regras ativadas serão a 2 e a 5, respectivamente:

2) SE processo A é A1 E Processo B é B2, ENTÃO C é C1;
5) SE processo A é A2 E Processo B é B2, ENTÃO C é C2.

Figura 19 – Representação gráfica das regras ativas e defuzzificação



Fonte: o autor.

Pelo operador lógico “E”, o resultado adotado na regra 2 é o A1 e esse valor será inserido em C1. Na regra 5, o menor valor é A2 e este será inserido em C2. Desse

modo, C1 e C2 serão unidos e formarão a figura geométrica que corresponde à resposta *fuzzy*.

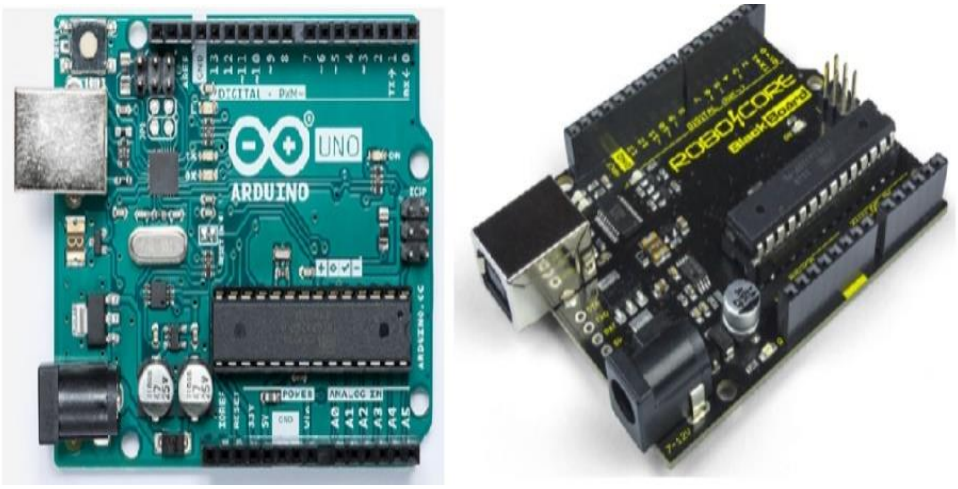
Nesse caso, a defuzzificação utilizada para transformar a resposta *fuzzy* em um valor que o sistema entende será o método centroide, que nada mais é do que usar o centro geométrico da figura gerada pela união de C1 e C2; logo, o valor é de 5,92.

2.6 ARDUINO

O Arduino é uma plataforma de *software* e *hardware open-source*. Na prática, tanto *softwares* como *hardwares* são liberados para que qualquer interessado possa modificá-lo, construí-lo e distribuí-lo na própria plataforma (ARDUINO, 2020).

Um exemplo dessa política é a placa “BlackBoard UNO R3”, fabricada pela empresa brasileira Robocore. Ela é totalmente compatível com os *softwares*, os *hardwares* e os periféricos desenvolvidos para o Arduino.

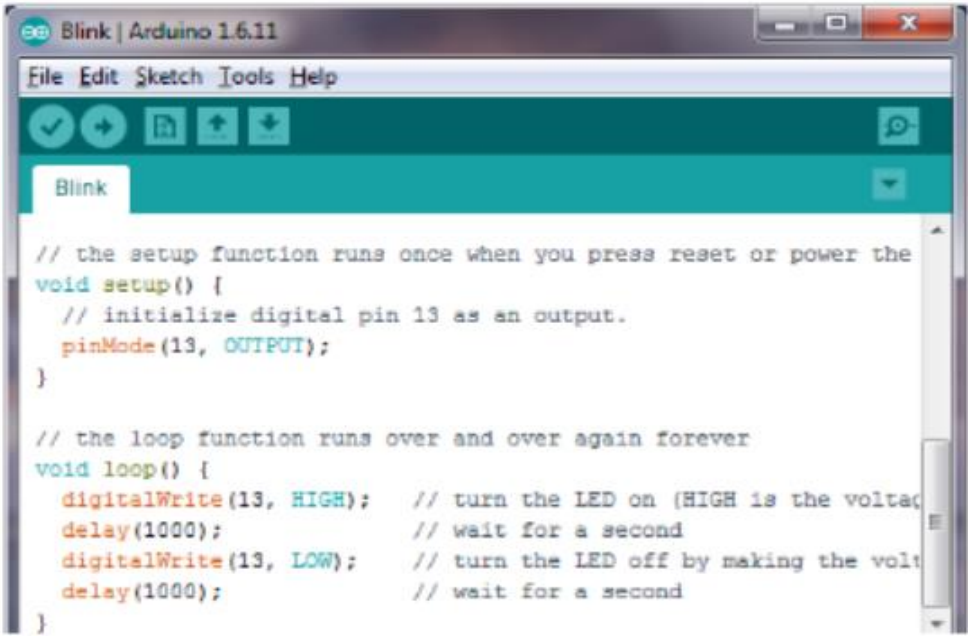
Figura 20 – Arduino Uno e BlackBoard UNO R3



Fonte: ARDUINO (2020); ROBOCORE (2021).

As placas do Arduino conseguem ler sinais digitais, analógicos e dados seriais dos mais diversos tipos de sensores. Além disso, apresentam saídas digitais, PWM e dados seriais para alimentar os mais diversos tipos de dispositivos de saídas. Para programar o Arduino e controlar essas variáveis de entrada/saídas, é necessário usar o *software* Arduino (IDE) (Figura 21) (ARDUINO, 2020).

Figura 21 – Visualização do Arduino IDE



Fonte: o autor.

O Arduino IDE é programável por meio da linguagem C. Esse *software* também funciona como monitor serial para uma prévia de interface homem-máquina (IHM). Outro ponto é que o IDE apresenta uma vasta biblioteca de exemplos de *softwares*, sendo útil aos mais variados níveis de programadores.

A plataforma Arduino tem microcontroladores embarcados em cada tipo de placa, na qual aqueles são o cérebro. Isso porque são eles os responsáveis pela interação da placa com as informações fornecidas pelo ambiente externo. Os microcontroladores utilizados pelo Arduino são da família Atmel e o modelo varia conforme a placa. Um exemplo é que a placa Arduino Uno usa o microcontrolador Atmega 328p (STEVAN JUNIOR; SILVA, 2015).

Pérez (2019) aponta para a necessidade de se conhecer e estudar as características de arquitetura do microcontrolador da Atmel, pois esse conhecimento será muito útil para o desenvolvimento de sistemas gerenciados pelo Arduino.

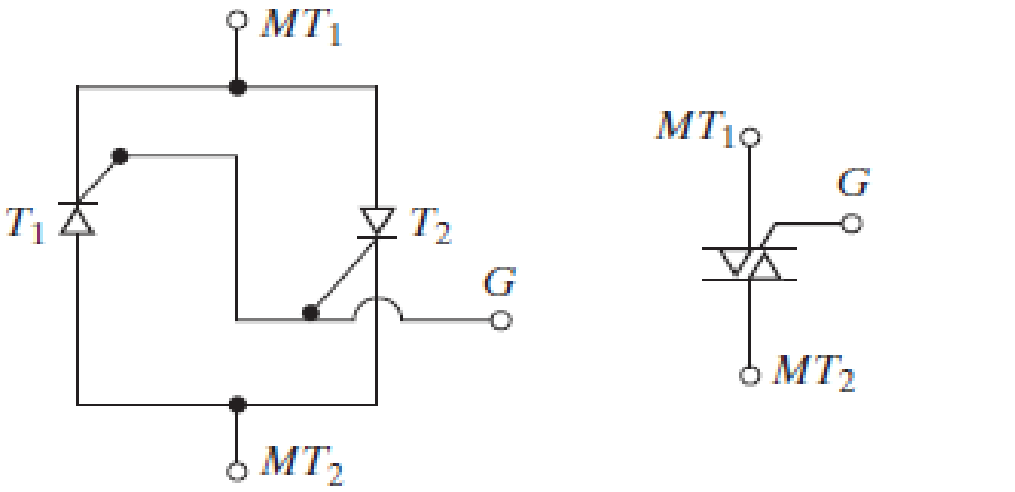
2.7 Triac

O diodo de quatro camadas e o SCR são exemplos de tiristores unidirecionais, isto é, ele possibilita que a corrente elétrica percorra somente um único caminho ou sentido. Contudo, outros membros da família dos tiristores, por exemplo, o Diac e o

Triac, podem conduzir corrente elétrica em duas direções. Por isso, também são chamados de tiristores bidirecionais (BOYLESTAD; NASHELSKY, 2013).

O Triac, por ser um tiristor bidirecional, é muito utilizado em circuitos de controle de rede em corrente alternada (CA). Para explicar o seu funcionamento, muito se utiliza como exemplo a montagem de SCRs em antiparalelo e os terminais gatilhos (G), ou *gate* em inglês, conectados em comum (Figura 22a). Outro detalhe é que seus terminais são chamados de MT1 e MT2, pois como o Triac é um dispositivo em que a corrente circula em ambas as direções, os termos *ânodo* e *cátodo* não fazem sentido e o terminal gatilho pode ser disparado tanto por um pulso positivo como por um pulso negativo (RASHID, 2014).

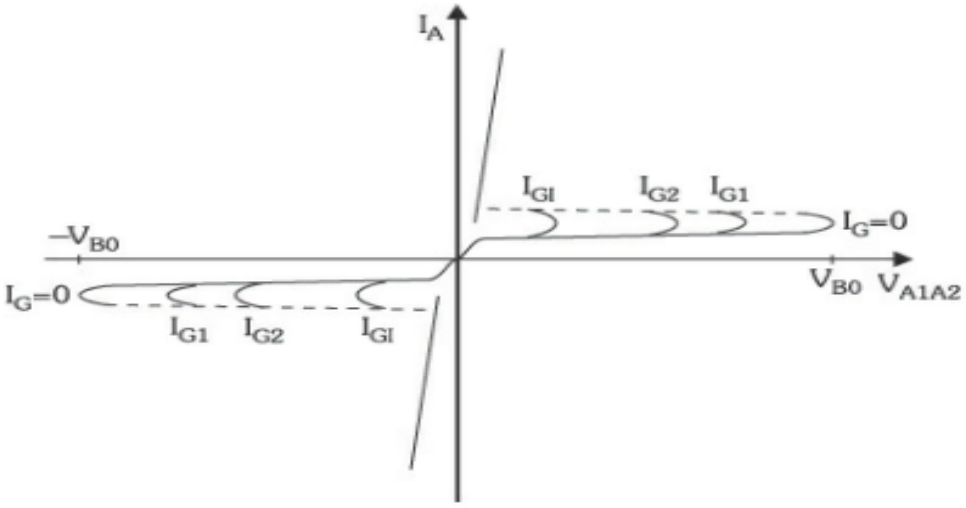
Figura 22 – Características de um Triac



Fonte: RASHID (2014).

Uma vez inserido um pulso (I_G) no gatilho, ou quando se ultrapassa o valor de tensão de *breakover* (V_{BO}) nos terminais MT1 e MT2, o Triac entra no estado de condução (fecha a trava) e assim possibilita que a corrente elétrica passe por ele como se fosse um botão fechado. A curva característica do Triac e sua possibilidade de condução tanto na polaridade positiva como na negativa são demonstradas na Figura 23 (ALMEIDA, 2018).

Figura 23 – Curva característica e símbolo do Triac

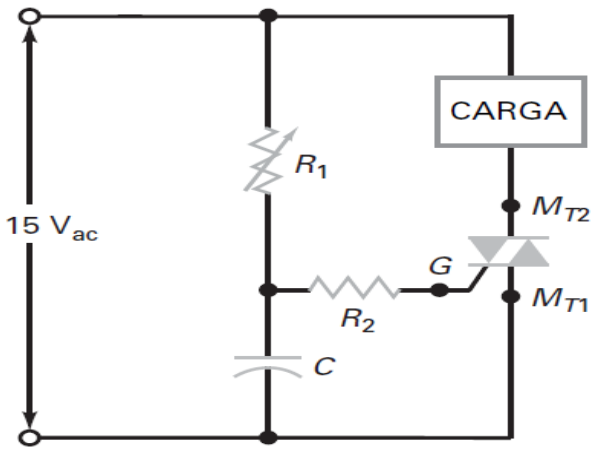


Fonte: ALMEIDA (2018).

Esse pulso inserido no gatilho do Triac pode ser modulado para que ele ocorra de forma sistemática. Um dos métodos para fazer essa modulação é por meio da utilização de um circuito RC, assim como é mostrado na Figura 24 (MALVINO; BATES, 2016).

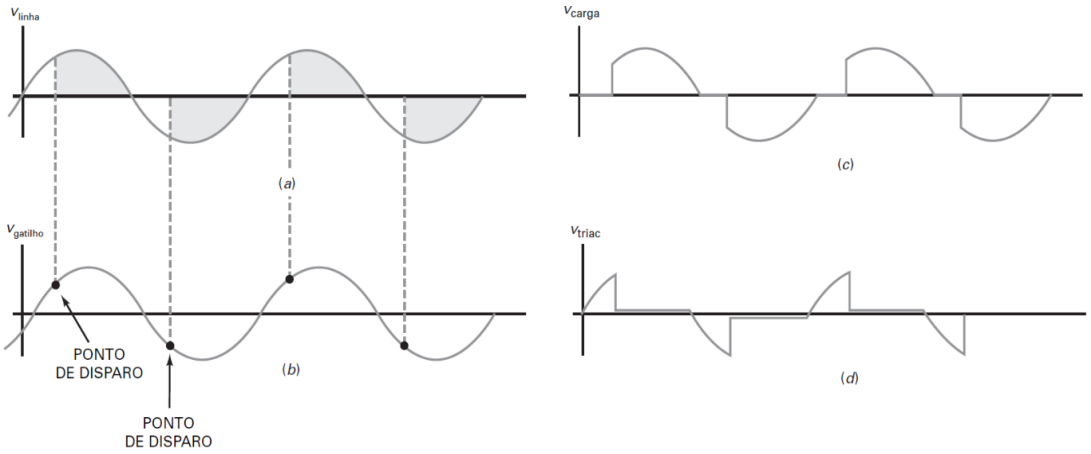
O capacitor do circuito apresentado na Figura 24 carregará em uma tensão alta o bastante para fornecer uma corrente de disparo no gatilho do Triac. Logo após esse disparo, o capacitor se descarregará, porém, o Triac entrará em modo de condução, conforme é mostrado na Figura 25. Ele ficará nesse modo de condução até que a tensão da linha retorne a zero e o processo de carregamento do capacitor se reinicie, repetindo todo o processo de disparo e modo de condução do Triac (MALVINO; BATES, 2016).

Figura 24 – Circuito de controle de fase com Triac



Fonte: MALVINO; BATES (2016).

Figura 25 – Controle de fase com Triac



Fonte: MALVINO; BATES (2016).

2.8 Média móvel

Um determinado sinal vindo de um sensor inserido em um sistema não é puro, pois sempre estará acompanhado de ruídos que podem ocasionar leituras errôneas, o que pode gerar resultados incorretos nas saídas dos sistemas. Desse modo, para amenizar esses ruídos e produzir sinais estáveis, é necessária a implementação de filtros, e um deles é o filtro de média móvel (FERON; FRICK, 2015).

O sistema de média móvel pode ser feito por *hardware* ou por *software*, sendo ambos de fácil formulação e implementação. Especificamente a inserção por *software* será feita por meio da utilização de um código simples, independentemente das linguagens de programação utilizada (DE LA VEGA, 2018). A seguir, está disposta a equação (5) (FERON; FRICK, 2015), utilizada para representar o filtro de média móvel.

> O filtro de média móvel é obtido calculando-se a média de um conjunto de valores, adicionando um novo valor ao conjunto e descartando o mais velho, porém, não é apenas uma média de um conjunto isolado. O filtro de média móvel é representado pela equação (5), onde 'n' é o índice dos valores de entrada e saída, N + 1 é o número de amostras utilizadas para a filtragem, y[n] é o sinal filtrado e x[n-k], representam as amostras do sinal analógico, estes valores serão somados segundo as especificações do filtro. (FERON; FRICK, 2015).

$$y[n] = \frac{1}{N+1}\sum_{k=0}^{n} x[n-k] \qquad (5)$$

3 MATERIAL E MÉTODOS

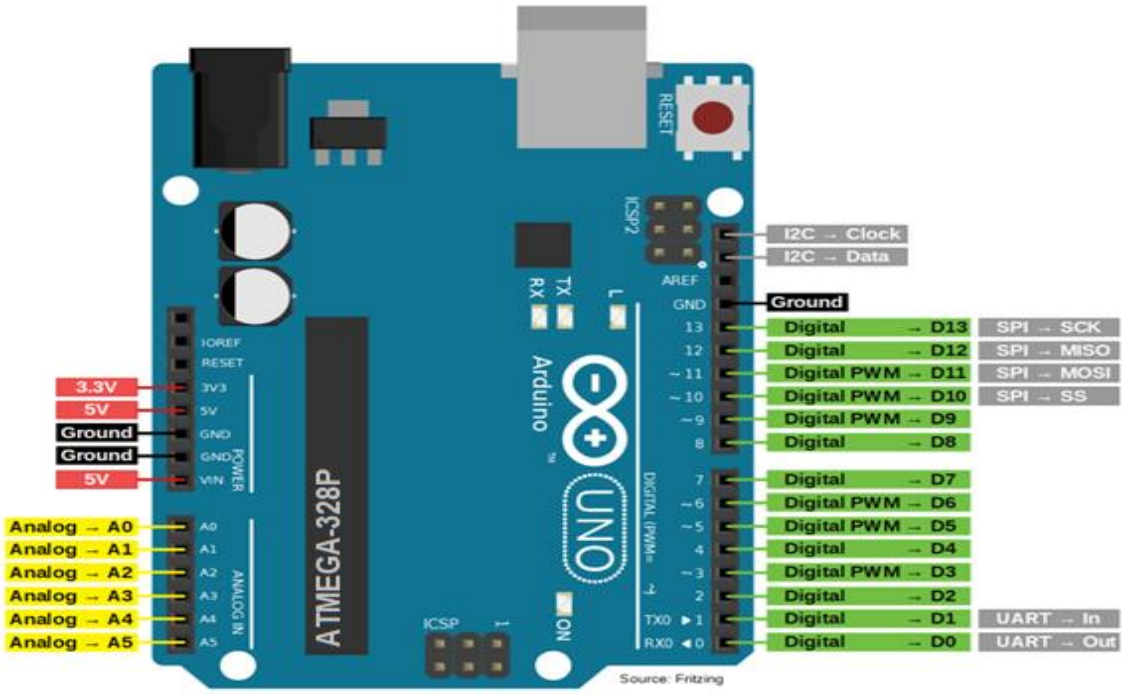
3.1 Material

Para a elaboração deste trabalho, um protótipo foi montado com a utilização dos *hardwares* Arduino Uno e Arduino Mega, como também de alguns periféricos compatíveis com a sua plataforma, além dos *softwares* MATLAB, Arduino IDE e Proteus. A seguir estão descritos os detalhes técnicos de cada componente.

O *hardware* Arduino Uno foi o responsável por receber o valor de “fuzzyficação” do MATLAB e gerenciar o periférico que controla a intensidade de luz da lâmpada.

Na Figura 26, estão dispostas as principais características do Arduino Uno. Destaca-se que ele tem o microcontrolador ATMEL ATMEGA 328p. O Arduino é energizado por uma tensão de 5V, mas isso também pode ser feito por uma fonte externa de 7 a 12V. Ele também apresenta 14 pinos para I/O, sendo que seis destes podem trabalhar como saídas PWM e outros seis como entradas analógicas.

Figura 26 – Interfaces de comunicação e alimentação do Arduino Uno

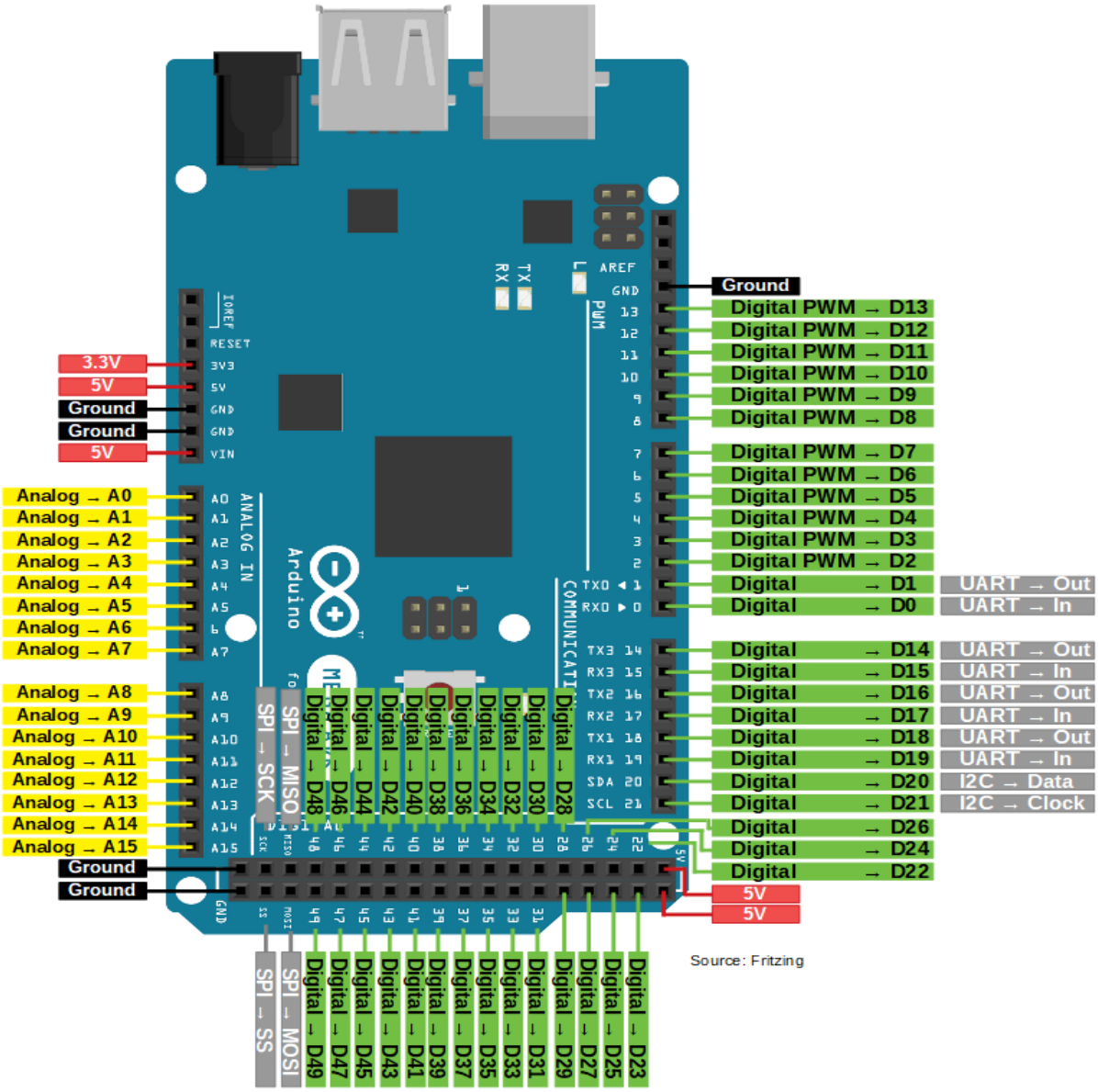


Fonte: DIYI0T (2021a).

O *hardware* Arduino Mega2560 (Figura 27) foi o responsável por receber o sinal do Módulo GY-302, filtrá-lo e enviá-lo para o sistema *fuzzy* no MATLAB.

Na Figura 27, são apresentadas as principais características do Arduino Mega. Um ponto em destaque é que ele tem o microcontrolador ATMEL ATmega2560 e pode ser energizado da mesma maneira que o Arduino Uno. Um contraste em relação ao Arduino Uno é que o Mega apresenta 54 pinos para I/O, sendo que 15 podem trabalhar como saídas PWM. Além disso, ele apresenta 16 portas que podem ser utilizadas com entradas analógicas.

Figura 27 – Interfaces de comunicação e alimentação do Arduino Mega2560



Fonte: DIYI0T (2021b).

O Dimmer Shield (Figura 28) foi o dispositivo responsável pelo controle da potência entregue para as lâmpadas, e por consequência, pelo controle da iluminação.

Ele é basicamente um *shield* que foi desenvolvido para ser acoplado em uma placa Arduino. O principal elemento desse dispositivo é o Triac BTA08-600B.

Outra característica do Dimmer Shield são as quatro entradas na régua de *borne*, sendo dois para a entrada de tensão (Vac) e dois para a saída de tensão (LOAD). A placa apresenta dois botões que podem ser utilizados para aumentar e diminuir o tempo de disparo do Triac, conforme a programação feita no Arduino.

Figura 28 – Dimmer Shield

Fonte: PEREIRA (2016).

Para a análise e a verificação dos valores de tensões de saídas do Arduino e do Dimmer Shield foi utilizado o multímetro digital da marca Thompson modelo 880 (Figura 29). Esse modelo faz leitura de tensão em corrente contínua (CC) de até 1.000 V e de corrente alternada de até 750 V. Além de tensão, ele também faz leituras de outras grandezas elétricas, como corrente (CC e CA) e continuidade elétrica.

Figura 29 – Multímetro Digital Thompson 880

Fonte: EFÁCIL (2021).

No protótipo, foram utilizadas duas lâmpadas LED dimerizáveis da marca GALAXY LED, linha Concept (Figura 30). Elas são compatíveis com soquete E27 e apresentam bulbo A60. Além disso, elas consomem cada uma 9,5 W de potência elétrica, são bivolts, apresentam fluxo luminoso de 810 lm e eficiência luminosa de 85 lm/W.

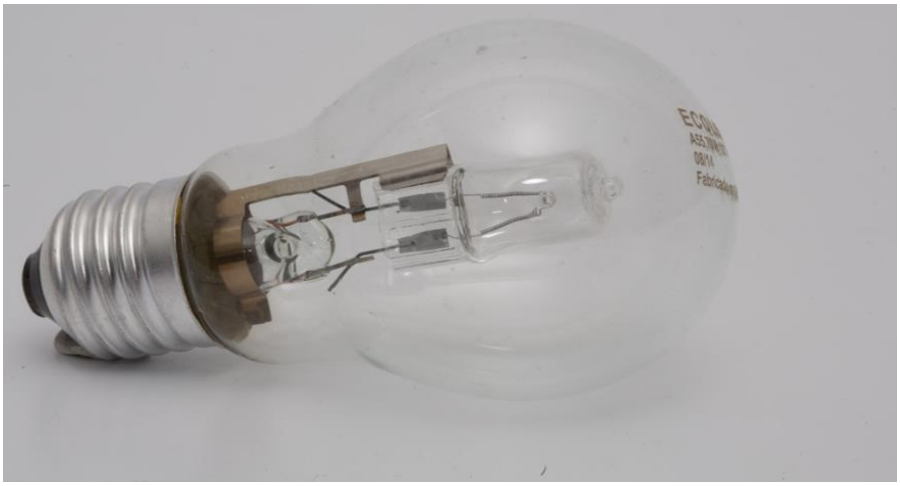
Figura 30 – Lâmpada LED dimerizável

Fonte: GALAXYLED (2021).

Para simular a incidência de luz solar, foi utilizada uma lâmpada halógena da marca Ecolume, modelo A55 (Figura 31). Ela consome 70 W de potência elétrica, tem tensão de 220 V e fluxo luminoso de 1.200 lm, bem como apresenta uma eficiência

luminosa de 17,14 lm/W. Do mesmo modo que as lâmpadas LEDs, ela é compatível com o soquete E27.

Figura 31 – Lâmpada halógena



Fonte: ECOLUME (2021).

O Dimmer analógico QD34 (Figura 32) foi utilizado para controlar a intensidade luminosa da lâmpada halógena. Esse dispositivo é da marca Qualitronix e é compatível com equipamentos de potência 400 W para tensão de 127 V, e 600 W para tensões de 220 V.

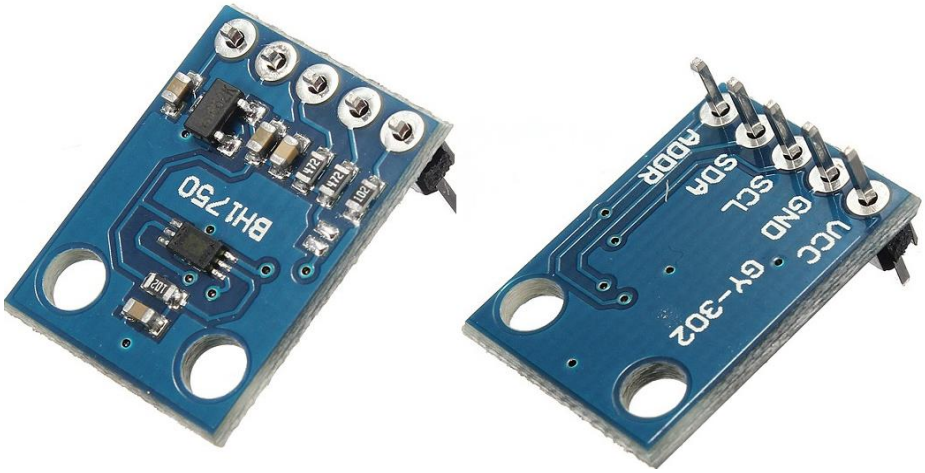
Figura 32 – Dimmer analógico QD34

Fonte: QUALITRONIX (2021).

O módulo GY-302 (Figura 33) foi o componente responsável por detectar e quantificar a incidência de luz no sistema. Seu principal componente é o circuito integrado BH1750, que é um sensor de luminosidade digital.

Para a utilização do módulo GY-302, foi necessário instalar a biblioteca “BH1750” no IDE do Arduino, que apresenta três exemplos. O exemplo *Advanced BH1750* foi utilizado como base da programação do Arduino.

Figura 33 – Módulo GY-302



Fonte: FILIPEFLOP (2021).

O luxímetro digital ITR-3000 (Figura 34) foi utilizado para a verificação e adequação dos sinais do módulo GY-302.

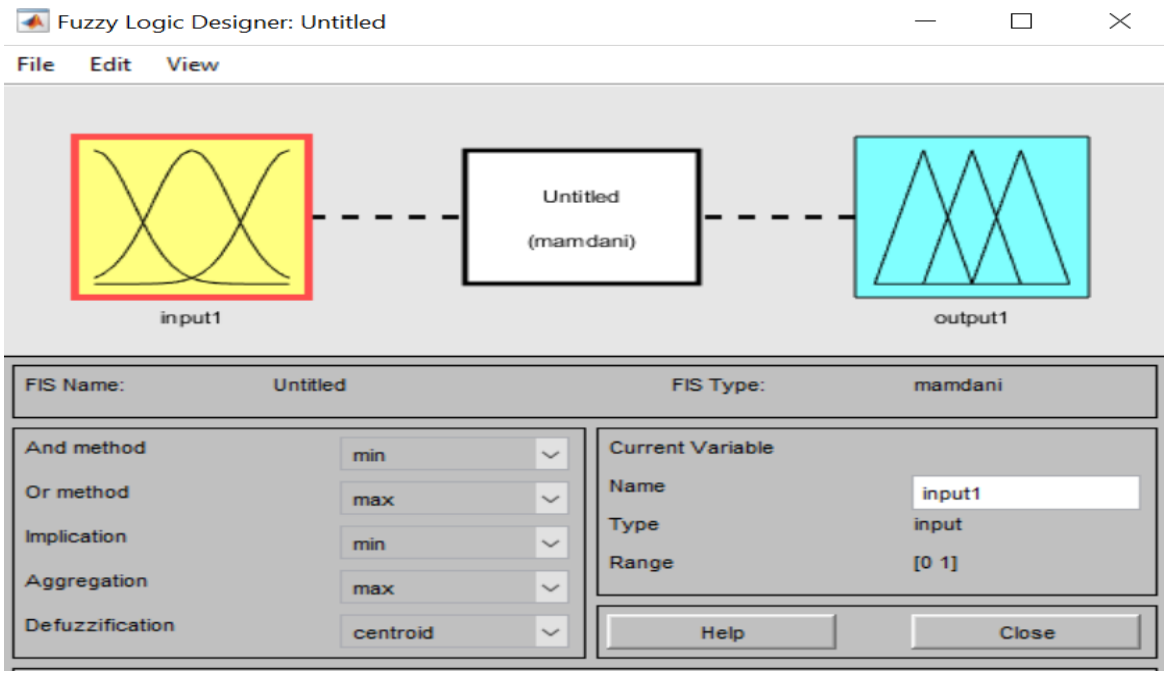
Figura 34 – Luxímetro digital ITR-3000

Fonte: INSTRUBRAS (2021).

No *software* MATLAB, versão R2015a, cuja licença pertence à Universidade de São Paulo – USP e é vinculada ao Grupo de Pesquisa Agroenerbio, foi desenvolvido o *software* em que foram executadas todas as funções relacionadas à lógica *fuzzy*. Para isso, foi utilizada a função "*Fuzzy Logic Designer*" (Figura 35) e a janela "Editor" para digitar o *software* principal.

Nessa "*toolbox*", foram inseridas as variáveis linguísticas da entrada e saída e suas respectivas funções de pertinência, assim como suas regras de associação. Com todos esses dados inseridos, o *software* fez todo o processo de decisão *fuzzy* (fuzzificação, inferência e defuzzificação).

Figura 35 – Função "*Fuzzy Logic Disigner*", do MATLAB

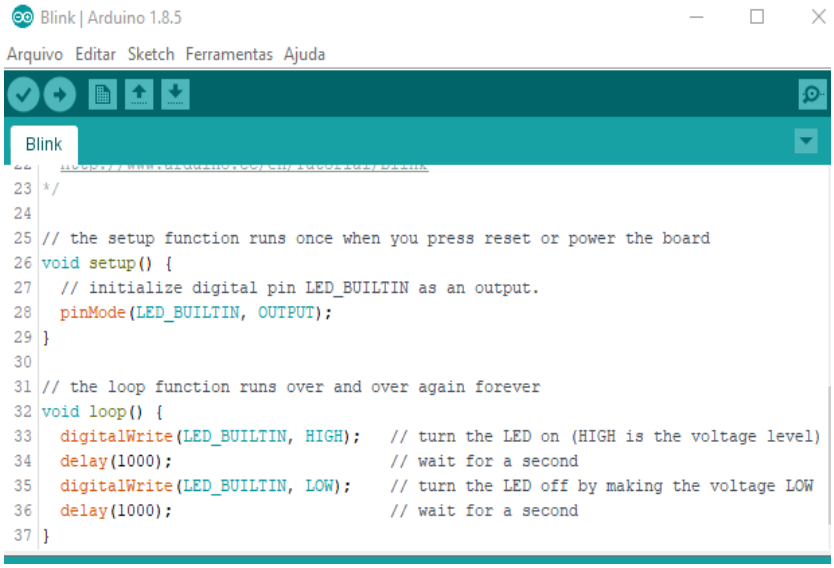


Fonte: o autor.

Outra ferramenta é o Arduino IDE, *software* livre, no qual foi desenvolvido o *software* que fez a comunicação serial entre o Arduino e o MATLAB. A tela do Arduino IDE pode ser visualizada na Figura 36 adiante.

O Arduino IDE é programado por meio da linguagem C. Para colaborar com o processo de geração do *software* foi utilizada sua vasta biblioteca de exemplos. Um destes, o *software* "*Blink*", é ilustrado na Figura 36.

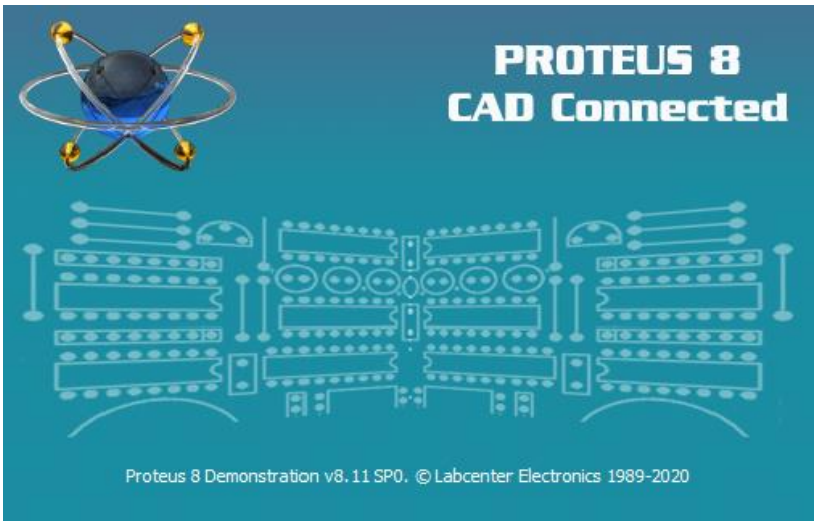
Figura 36 – Exemplo "*Blink*" da Arduino IDE



Fonte: o autor.

O *software* Proteus, versão 8, versão de testes, da desenvolvedora Labcenter, foi o *software* em que as simulações envolvendo elétrica/eletrônica foram realizadas. A versão utilizada será a de demonstração, tornando seu uso limitado, porém, gratuito.

Figura 37 – Proteus 8 *Demonstration*



Fonte: o autor.

3.2 Métodos

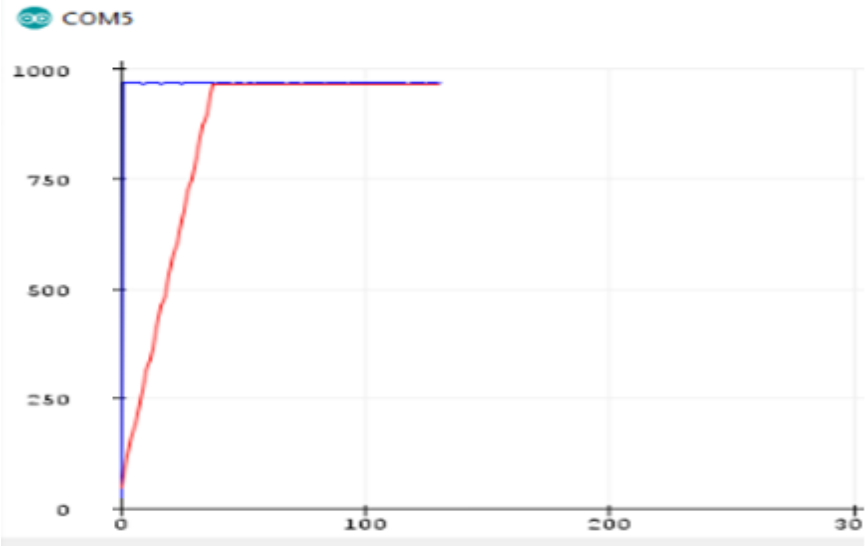
3.2.1 Implementação da média móvel

Neste trabalho, foi proposto que o sinal de entrada do módulo GY-302 fosse tratado usando-se o filtro de média móvel, implementado por meio de uma rotina na programação do Arduino. A fórmula que serviu de base foi a da equação (5).

No entanto, antes de aplicar esse filtro no módulo GY-302, foram feitos testes em que o potenciômetro foi utilizado para simular um sensor. O sinal de entrada do potenciômetro equivale ao *range* de tensão de 0 a 5V. Após passar pelo conversor analógico-digital do Arduino, os valores de tensão se convertem para um *range* de valores digitais de 0 a 1.023. Os resultados foram plotados na janela “monitor serial” do *software* Arduino IDE (Figura 38). Em azul, está representado o sinal original, e, em vermelho, o sinal filtrado. Nesse primeiro momento, os testes foram feitos com 20 e 40 amostras.

Um detalhe importante é que, na programação, cada quantidade de amostras – 20 e 40 – será substituída por uma nova quantidade de amostras de forma cíclica.

Figura 38 – Monitor Serial do Arduino IDE



Fonte: o autor.

Uma característica dessa janela é que as variações dos sinais são mostradas em tempo real.

3.2.2 Teste de módulo GY-302

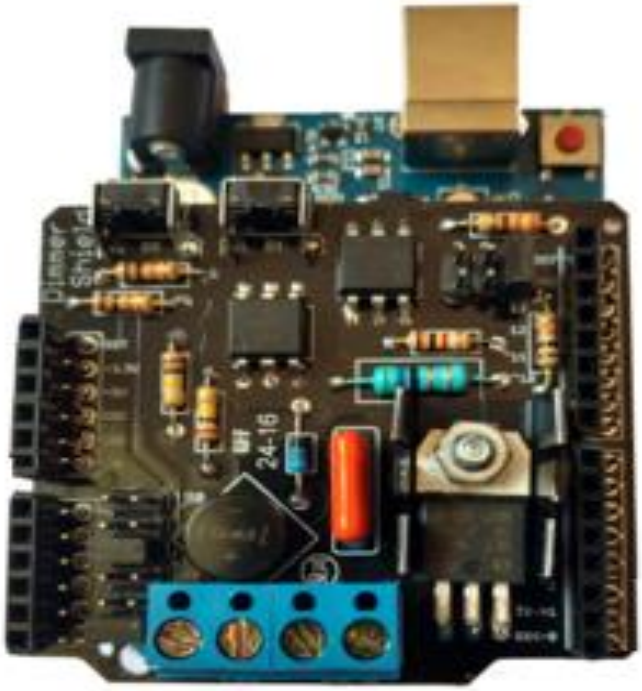
Para realizar os testes com o módulo GY-302 foi utilizada a lâmpada LED instalada em uma luminária, a qual foi colocada em uma base (Figura 39 a) para permanecer distante 1,70 m do sensor GY-302 (Figura 39 b). Para a verificação da coleta dos valores pelo sensor, foi utilizado o luxímetro e os resultados de ambos foram inseridos e demonstrados em uma tabela.

Figura 39 – Estrutura do teste do sensor GY-302

Fonte: o autor.

No controle de intensidade da lâmpada LED, para as leituras do GY-302 e do luxímetro, foi utilizado o Dimmer Shield acoplado ao Arduino, conforme Figura 40 a seguir.

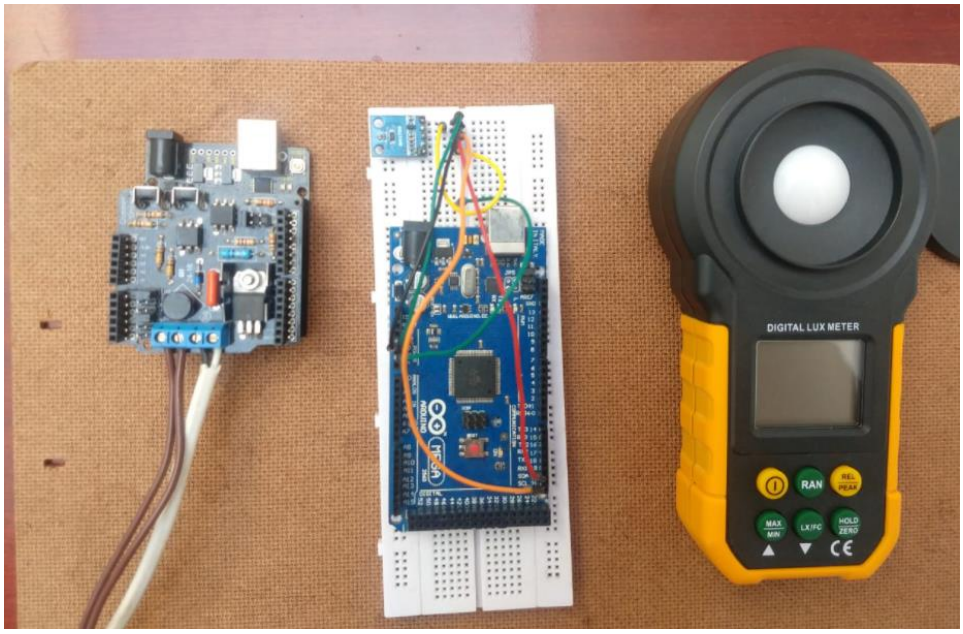
Figura 40 – Dimmer Shield acoplado no Arduino



Fonte: PEREIRA (2016).

Os componentes para os testes foram dispostos conforme indica a Figura 41.

Figura 41 – Materiais para o teste do GY-302



Fonte: o autor.

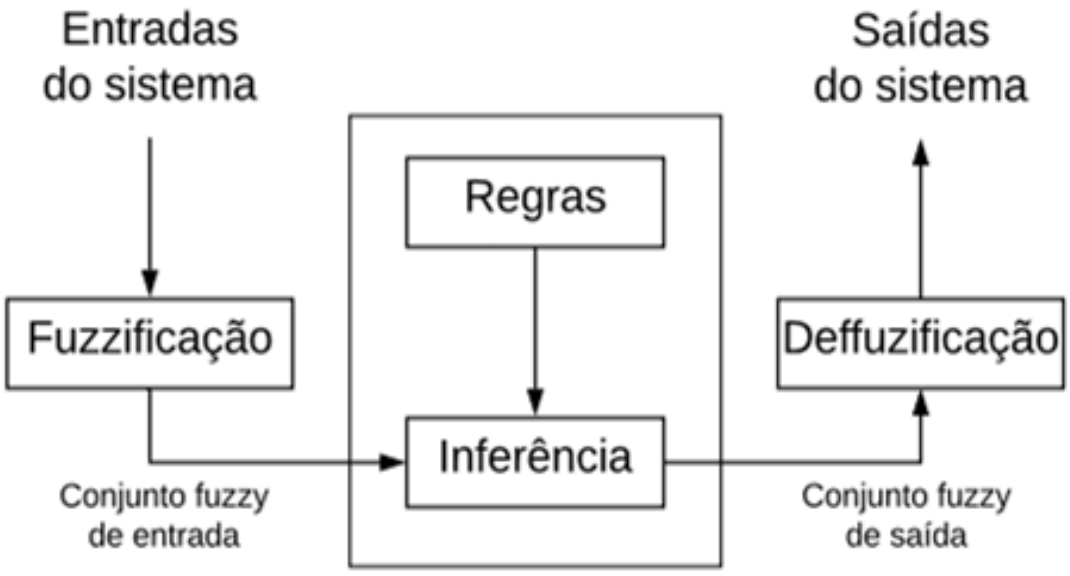
Nesse controle, foi utilizada uma programação em que o tempo de disparo do Triac será controlado pelos botões do próprio Dimmer Shield. Um botão aumentará o tempo de disparo do Triac em 2%, e o segundo botão diminuirá também em 2%.

Os testes foram realizados no período noturno para que não houvesse a interferência de luz natural, de modo a facilitar o controle de fontes de luz artificial. A própria a tela do computador utilizada foi disposta de forma que a luz não incidisse no sensor.

3.2.3 Desenvolvimento do sistema *fuzzy*

Para o desenvolvimento de um sistema *fuzzy*, as etapas básicas de um sistema de inferência devem ser seguidas conforme pode ser verificado na Figura 42.

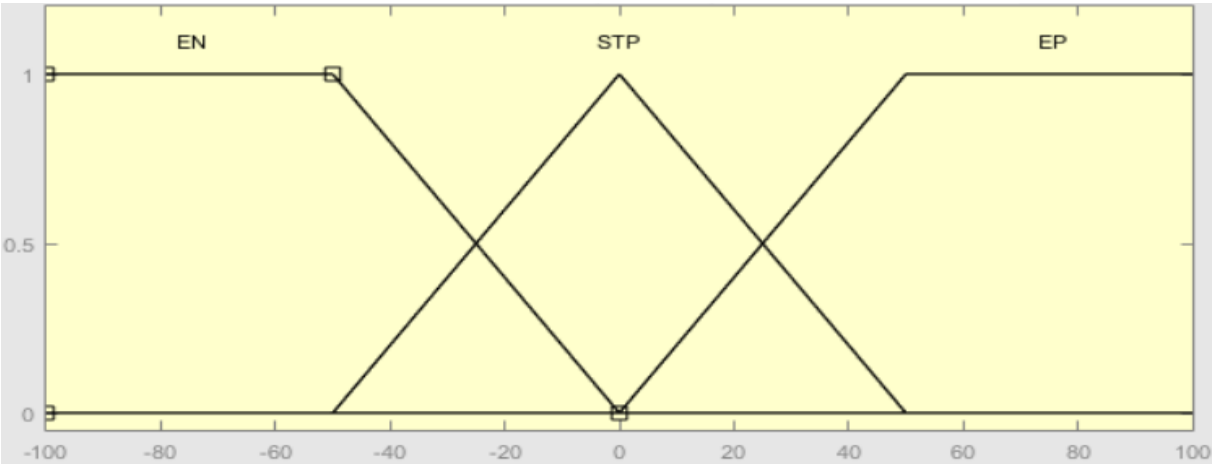
Figura 42 – Diagrama das etapas de um sistema *Fuzzy*



Fonte: COELHO *et al.* (2018).

A grandeza de entrada foi o percentual do erro (PE) da intensidade luminosa, obtida pela leitura do sensor GY-302 em relação aos *setpoints* 25 ou 10 lux. As variáveis linguísticas foram erro negativo (EN), *setpoint* (ST) e erro positivo (EP), sendo que EN e EP foram elaboradas com o uso da função trapezoidal e STP por meio da função de pertinência triangular. Seus valores correspondentes são demonstrados na Figura 43 a seguir.

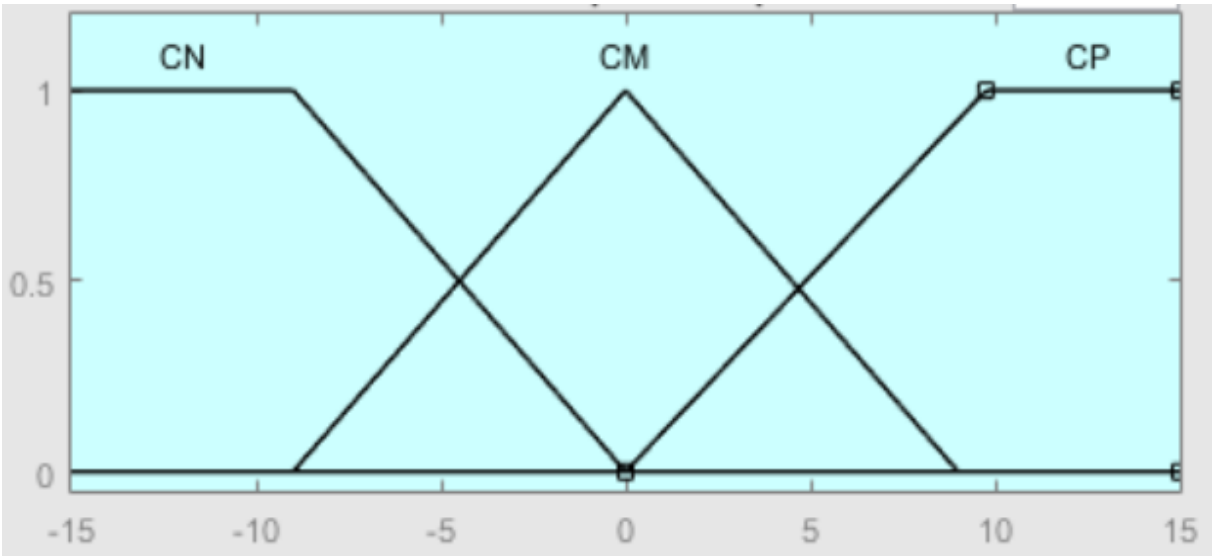
Figura 43 – Função de pertinências da entrada “PE”



Fonte: o autor.

Já a grandeza de saída foi chamada de correção do erro (CE), também em porcentagem, que serão os valores de incremento e decremento ao valor de intervalo de tempo em que o pulso elétrico será enviado ao *gate* do Triac, controlando assim a potência elétrica das lâmpadas. Esses valores serão enviados ao Arduino Uno para controlar o Dimmer Shield. Suas variáveis linguísticas foram correção negativa (CN), correção média (CM) e correção positiva (CP), sendo que CN e CP foram elaboradas com funções de pertinências trapezoidais e CM com função de pertinência triangular. Seus valores correspondentes são demonstrados na Figura 44.

Figura 44 – Função de pertinências da Saída “CE”



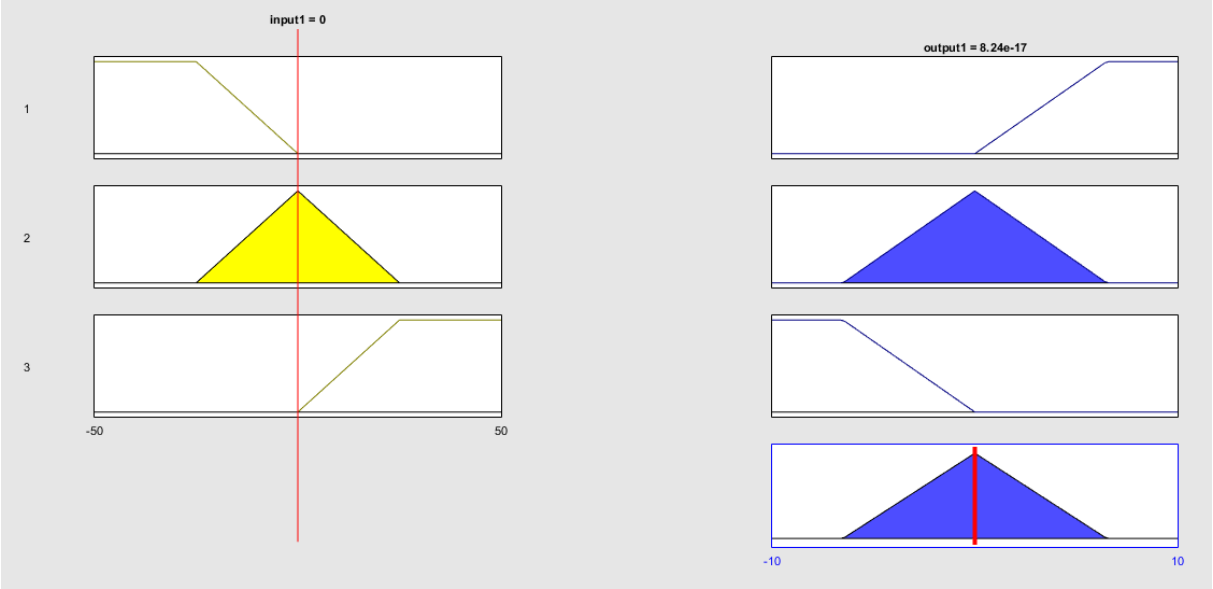
Fonte: o autor.

A inferência utilizada no controle do protótipo foi a Mamdani e as regras usaram os operadores “SE” e “ENTÃO”. Sua base foi descrita da seguinte forma:

1) SE entrada PE é EN ENTÃO saída CE é CP;
2) SE entrada PE é STP ENTÃO saída CE é CM;
3) SE entrada PE é EP ENTÃO saída CE é CN.

A defuzzificação que foi utilizada no protótipo será a de centro de área ou centroide (Figura 45).

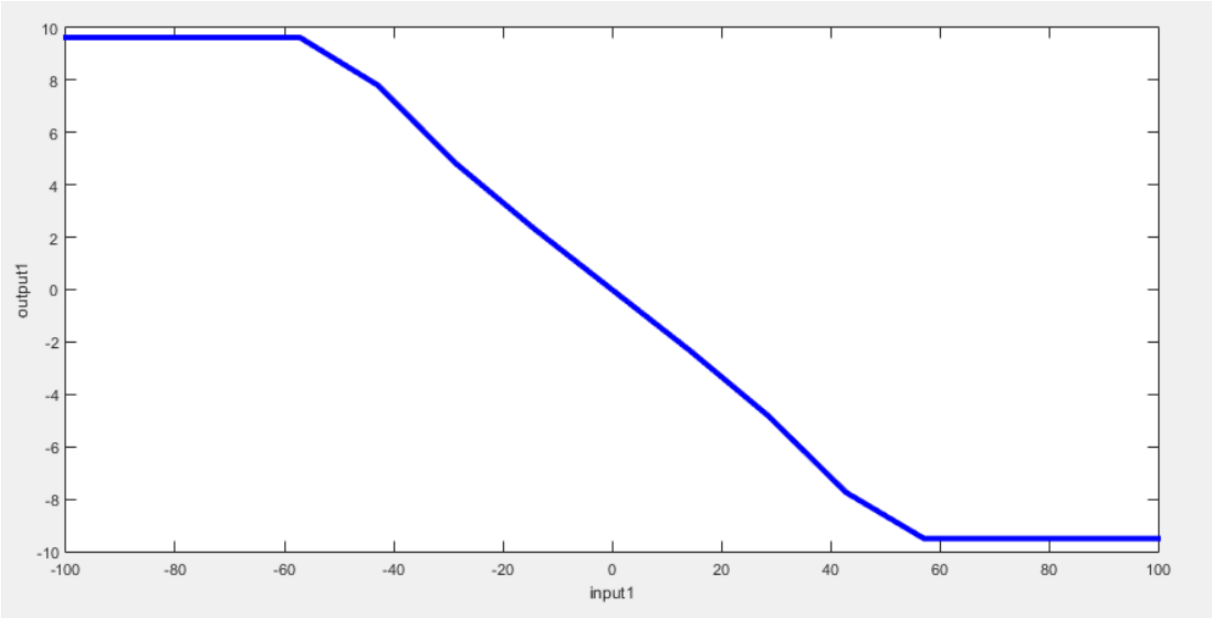
Figura 45 – Defuzzificação do protótipo



Fonte: o autor.

O comportamento do sistema é demonstrado na Figura 46 a seguir, em que o eixo horizontal indica o valor de entrada (erro em %) e o eixo vertical indica os valores de saída (correção em %).

Figura 46 – Entradas e suas respectivas saídas do sistema *fuzzy* principal



Fonte: o autor.

3.2.3.1 Testes de compatibilidade de programação e comunicação com o Arduino

O sistema *fuzzy* foi construído no *software* MATLAB, no qual pode ser feito de duas maneiras. Uma delas é por meio da ferramenta *Fuzzy Logic Designer* (FLD) e a outra é escrever a programação na janela “Editor”. Este trabalho utilizará as duas ferramentas.

Por apresentar uma interface um pouco mais amigável, o FLD foi utilizado de modo a agilizar a montagem e, por consequência, a compreensão do comportamento do sistema *fuzzy*.

Depois de compreendido, e de o sistema se mostrar satisfatório no FLD, a próxima etapa foi a de escrever a programação na janela “Editor”, a qual é responsável por se comunicar com os Arduinos.

A comunicação entre os Arduinos e o MATLAB foi feita usando as portas seriais, também conhecidas como *comunicação serial*, que fisicamente estão conectadas à entrada USB dos Arduinos. Essas portas têm a função de enviar apenas dados entre os componentes, por exemplo, letras, números, palavras e caracteres.

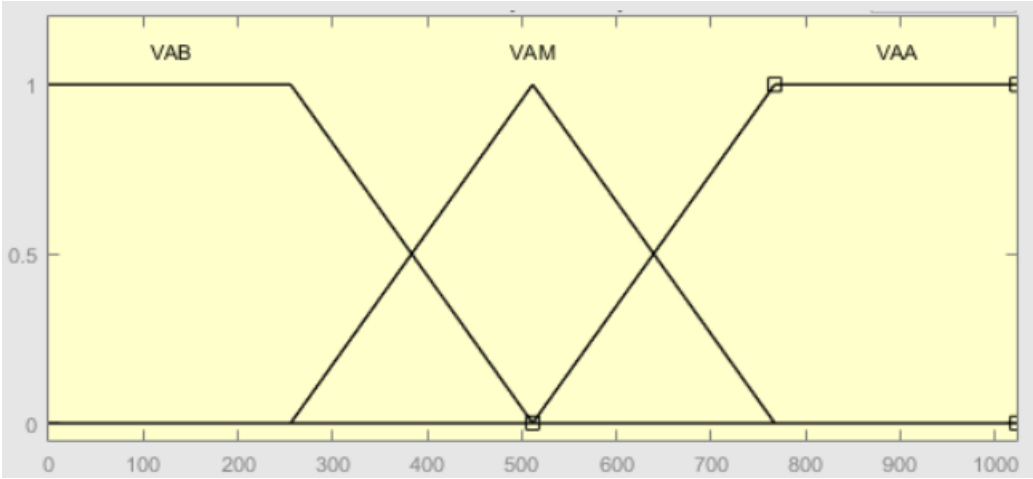
É importante destacar que, antes da implementação do sistema definitivo no protótipo, alguns testes tiveram que ser realizados para assegurar que a leitura e o envio de sinais entre o MATLAB e os Arduinos ocorressem de forma adequda. Para

esses testes foi utilizado um sistema *fuzzy* secundário. Nele, as grandezas a serem “fuzzyficadas” foram o valor inserido na porta analógica do Arduino (VA) e a tensão elétrica no LED (TL); respectivamente, o sinal de entrada e o sinal de saída.

Para as variáveis linguísticas da entrada foram utilizados os termos valor analógico baixo (VAB), valor analógico médio (VAM) e valor analógico alto (VAA). Nessa etapa de simulação, as variáveis VAB e VAA foram elaboradas com o uso de funções de pertinência trapezoidal, e a variável VAM usou função de pertinência triangular (Figura 47). O *range* do sinal de entrada é de 0 a 1.023, que corresponde à conversão do sinal da entrada analógica do Arduino em valores digitais.

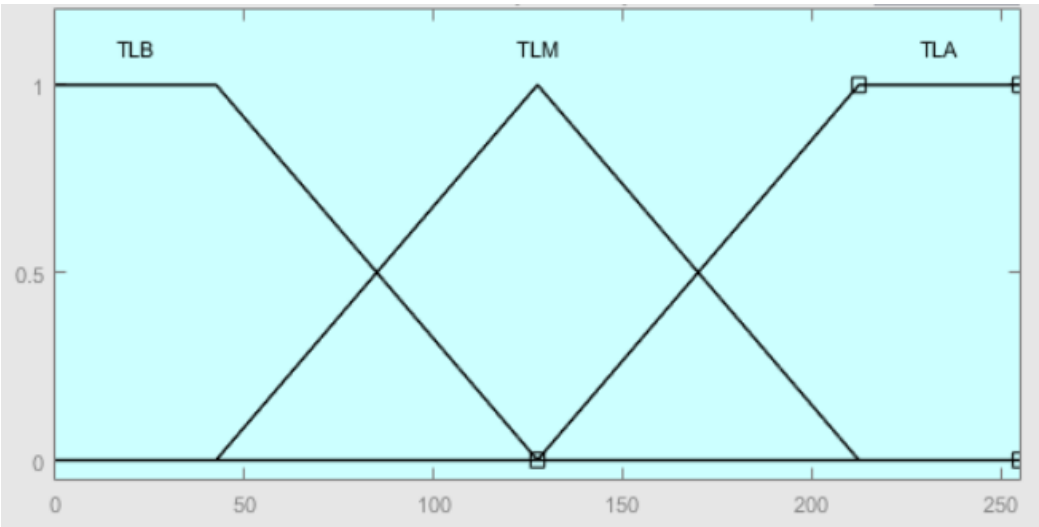
Para as variáveis linguísticas de saída foram utilizados os termos tensão elétrica do LED baixa (TLB), tensão elétrica do LED média (TLM) e tensão elétrica do LED alta (TLA). Similarmente ao sinal de entrada, as variáveis de saída TLB e TLM foram elaboradas com o uso da função de pertinência trapezoidal, e a variável TLM usou a função de pertinência triangular (Figura 48). O seu *range* do sinal de saída é de 0 a 255, que corresponde aos limites do pulso PWM do Arduino.

Figura 47 – Função de pertinências da entrada “VA”



Fonte: o autor.

Figura 48 – Função de pertinências da Saída “TL”



Fonte: o autor.

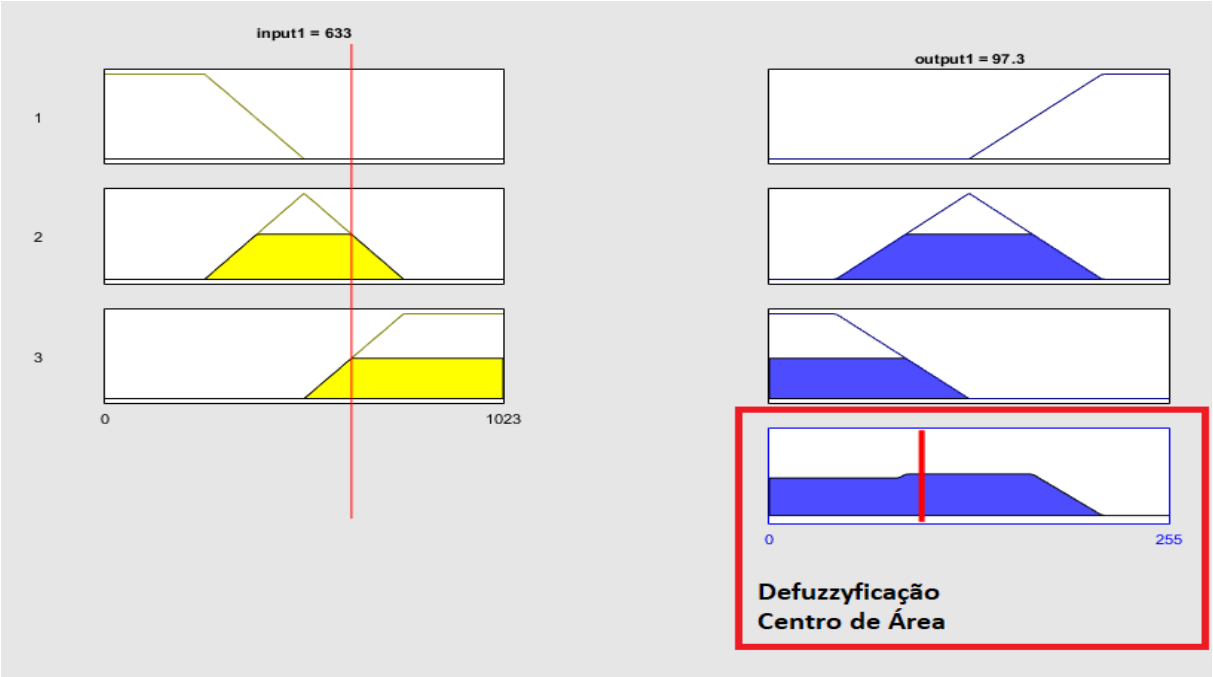
A segunda etapa corresponde às regras ou ao *Fuzzy Rule Base*, que definiu as regras do sistema *fuzzy*, mais precisamente como os sinais de entrada interagiram com os sinais de saída. A inferência utilizada foi a Mamdani.

As regras utilizaram os operadores “SE” e “ENTÃO” e sua base foi descrita da seguinte forma:

1) SE entrada VA é VAB ENTÃO saída TL é TLA;
2) SE entrada VA é VAM ENTÃO saída TL é TLM;
3) SE entrada VA é VAA ENTÃO saída TL é TLB.

O último passo foi a *defuzzificação*, que consiste em transformar a resposta *fuzzy* em um valor compreensível para o sistema. Neste trabalho, a *defuzzificação* escolhida foi a centroide ou centro de área, mas outras poderiam ser utilizadas de acordo com a inferência de Mamdani, por exemplo, média dos máximos e primeiro máximo.

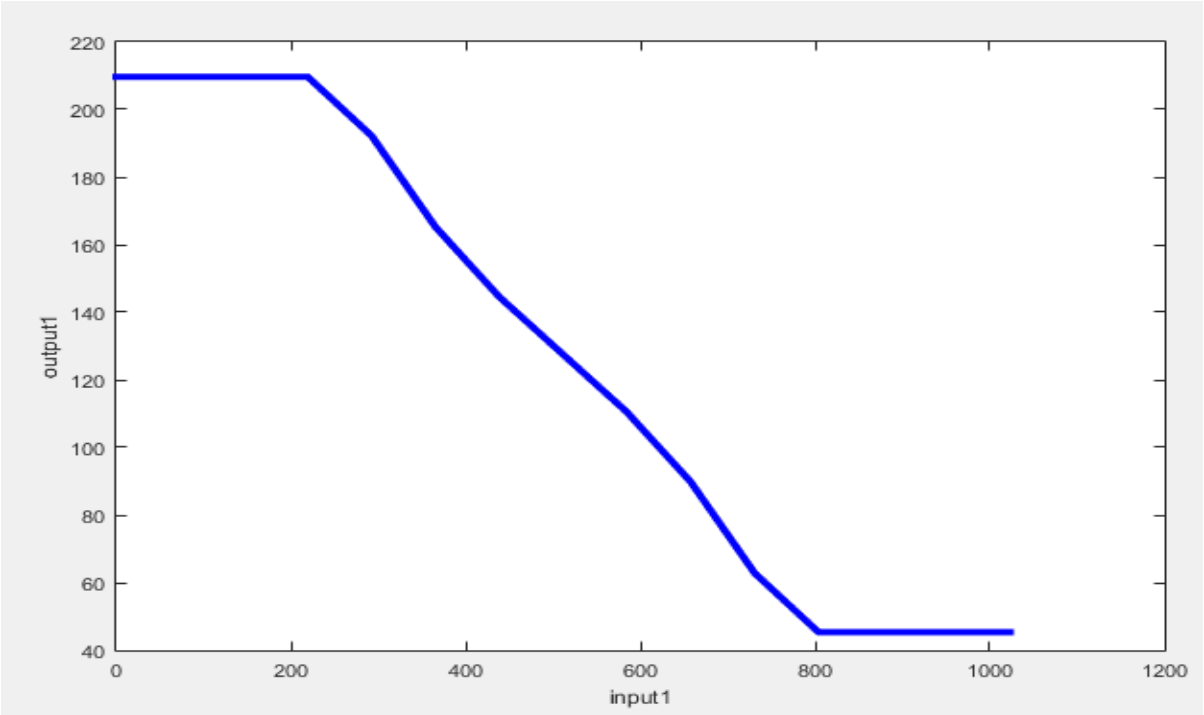
Figura 49 – “Defuzzyficação” do sistema *fuzzy* secundário



Fonte: o autor.

Nesse caso em questão, quando a entrada recebeu um valor de 633, a saída retornou um valor de 124 (em PWM). O comportamento do sistema é demonstrado na Figura 50.

Figura 50 – Entradas e suas respectivas saídas do sistema *fuzzy* secundário



Fonte: o autor.

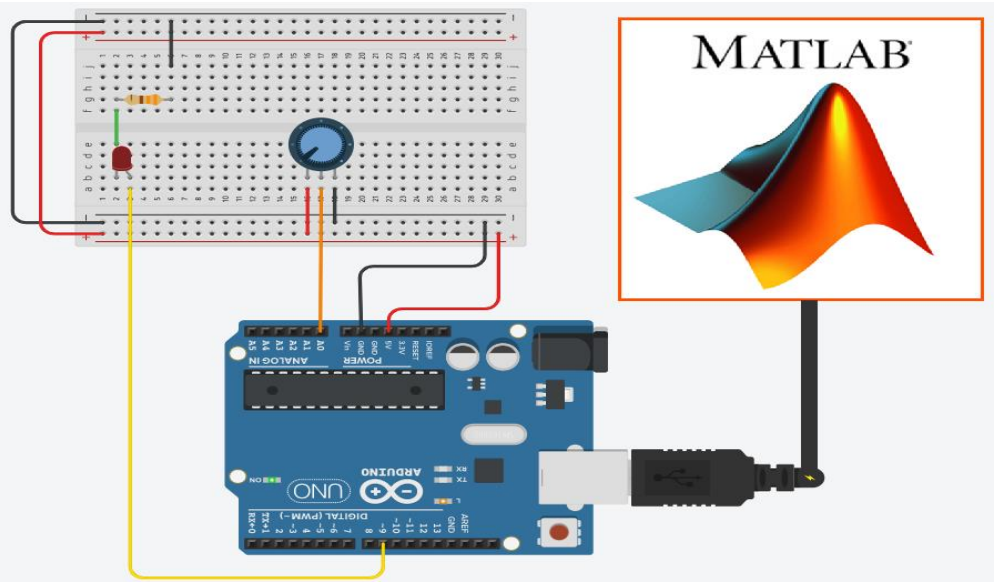
Além de verificar a comunicação serial entre o Arduino e o MATLAB, essa etapa serviu para verificar a compatibilidade dos *softwares* elaborados no FLD e no Editor.

3.2.3.2 Montagem da simulação *fuzzy* de teste

Como foi descrito anteriormente, o Arduino enviará os sinais de entrada, que serão obtidos por intermédio de um potenciômetro pelo MATLAB, o qual realizará todo o sistema *fuzzy*. Logo depois, o Arduino receberá o valor da *defuzzificação*, enviado pelo MATLAB, que aplicará esse valor em sua saída.

O potenciômetro foi empregado nessa etapa para simular um sensor. Em razão de ele ser uma resistência variável (manualmente), é possível ver o comportamento do sistema *fuzzy* com vários valores de entrada. A configuração do sistema físico e da comunicação serial é demonstrada na Figura 51.

Figura 51 – Comunicação serial entre Arduino e MATLAB

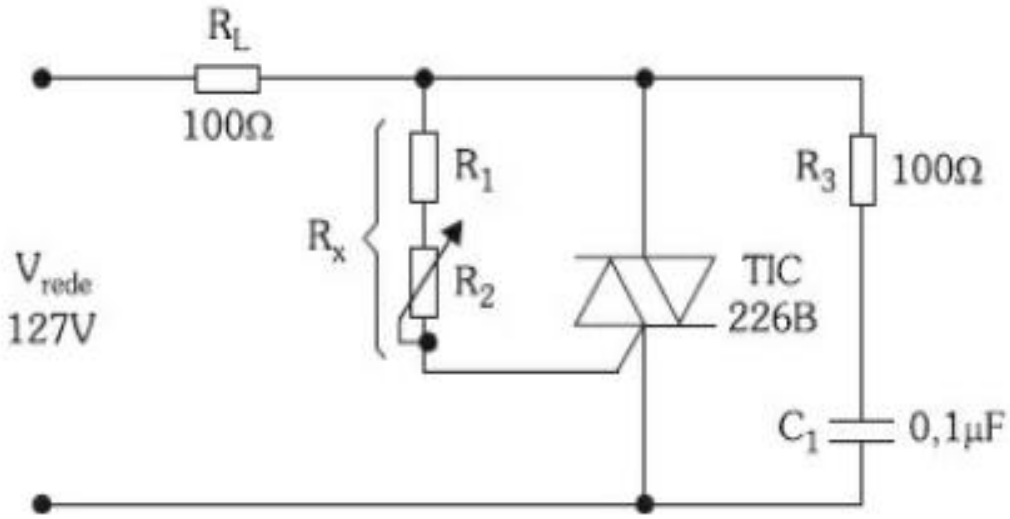


Fonte: o autor.

3.2.4 Simulação Triac

O componente que realizou o controle de potência das lâmpadas, e por consequência de sua intensidade luminosa, foi o Triac que está acoplado ao Dimmer Shield. Para possibilitar a execução dessa função, ele pode ser instalado em um circuito eletrônico conforme o exemplo de Almeida (2018) (Figura 52).

Figura 52 – Circuito de Controle de fase com Triac



Fonte: ALMEIDA (2018).

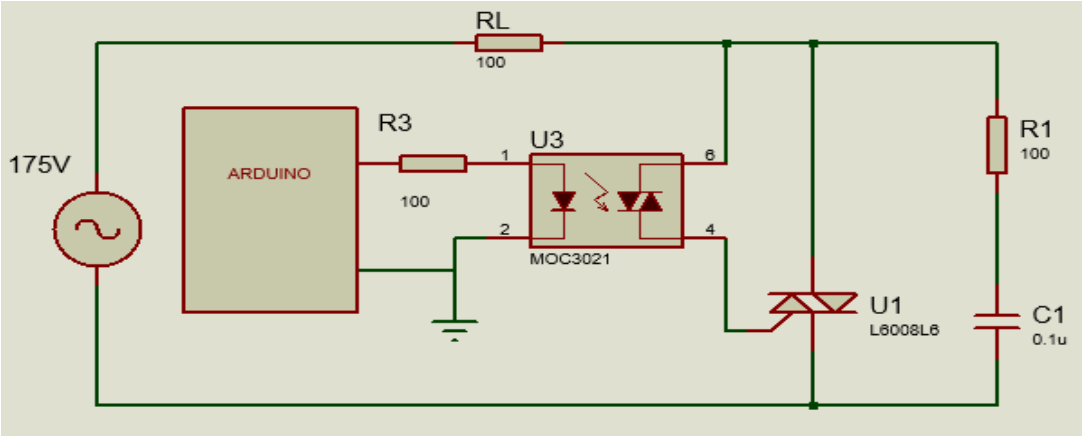
Os principais componentes desse circuito são o resistor R_L, que representa a carga principal a ser alimentada pelo circuito, o Resistor R_X, que é a soma do resistor R_1 e do potenciômetro R_2, e o Triac (TIC 226B).

Nesse circuito, o ângulo de disparo do Triac é controlado ao se modificar manualmente o valor do potenciômetro R_2 da resistência R_X. Um fator importante é que o ângulo de disparo do Triac não pode exceder 90º nessa configuração; os cálculos para parametrizar R_X têm que levar esse fator em consideração.

Para fazer com que o ângulo de disparo seja controlado automaticamente e exceda o limite de 90°, foi necessário substituir R_X por um componente que envie pulsos elétricos para o *gate* do Triac em intervalos programados e em sincronia com a frequência da rede elétrica.

A técnica proposta por este trabalho foi a de usar o microcontrolador Atmega 328-P, que está embarcado no Arduino, para enviar os pulsos e assim controlar o ângulo de disparo do Triac. Outro componente importante nessa nova configuração é o acoplador óptico MOC3021, que tem como uma de suas funções isolar o Arduino da rede elétrica principal. As modificações no circuito são demonstradas na Figura 53 a seguir.

Figura 53 – Triac controlado pelo Arduino

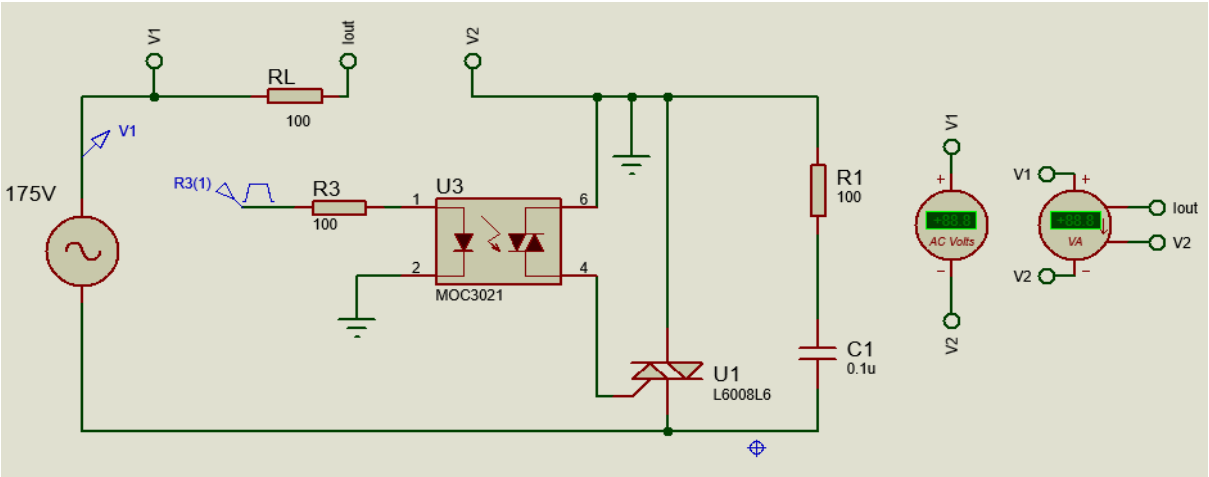


Fonte: o autor.

Nas simulações que foram realizadas no *software* Proteus, o Arduino foi substituído por um gerador de pulsos (Figura 54). Essa substituição facilitará as configurações de tempo para os disparos do Triac. Outra modificação foi que no circuito serão inseridos um voltímetro (AC) e um wattímetro para medir, respectivamente, a tensão e a potência na carga (R_L). Outros dados importantes da simulação é que a tensão de pico (V_P) da fonte foi de 175 V e sua frequência foi de 60 H_z.

Na Figura 54, pode-se observar os terminais I_{out} e V_2 ligados virtualmente pelo wattímetro.

Figura 54 – Triac controlado pelo gerador de pulso



Fonte: o autor.

O gerador de pulso foi configurado para uma frequência de 120 H_z, pois ele enviará os sinais tanto no semiciclo positivo quanto no negativo, para assim possibilitar um

maior controle do sinal senoidal. Para calcular o tempo necessário para se obter um determinado ângulo foi utilizada uma adaptação da equação (6) a seguir (MARKUS, 2011).

$$T_\alpha = \frac{\alpha * T}{360°} \tag{6}$$

Em que T_α é o tempo para o ângulo desejado, α é o ângulo desejado, T é o período da rede (16,66 ms para 60 Hz).

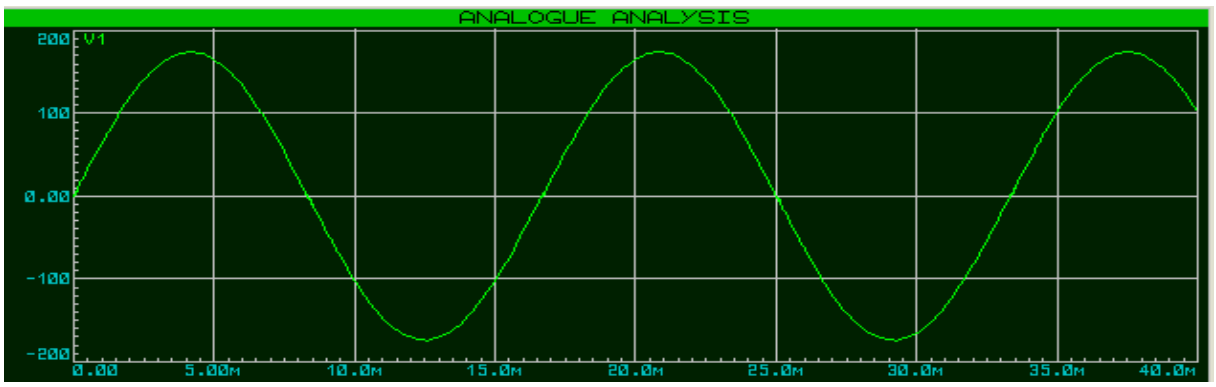
O valor de tensão eficaz e de potência de cada sinal de onda foi mostrado pelo voltímetro e pelo wattímetro inserido em R_L. Para a verificação desses valores foi feita uma comparação pelos valores teóricos obtidos pelas equações (7) e (8). Essas fórmulas são para serem usadas em circuitos conforme indicado pela Figura 52, mas essa etapa servirá para analisar se elas são compatíveis com o circuito indicado pela Figura 54 (ALMEIDA, 2018).

$$V_{eficaz} = V_p * \sqrt{\frac{1}{2} - \frac{\alpha}{2\pi} + \frac{sen(2\alpha)}{4\pi}} \tag{7}$$

$$P = \frac{V_{eficaz}^2}{R_L} \tag{8}$$

Para demonstrar o comportamento das ondas que alimentam a carga R_L foi utilizada a ferramenta “ANALOGUE ANALYSIS” (Figura 55), que se trata de um *display*.

Figura 55 – Ferramenta ANALOGUE ANALYSIS



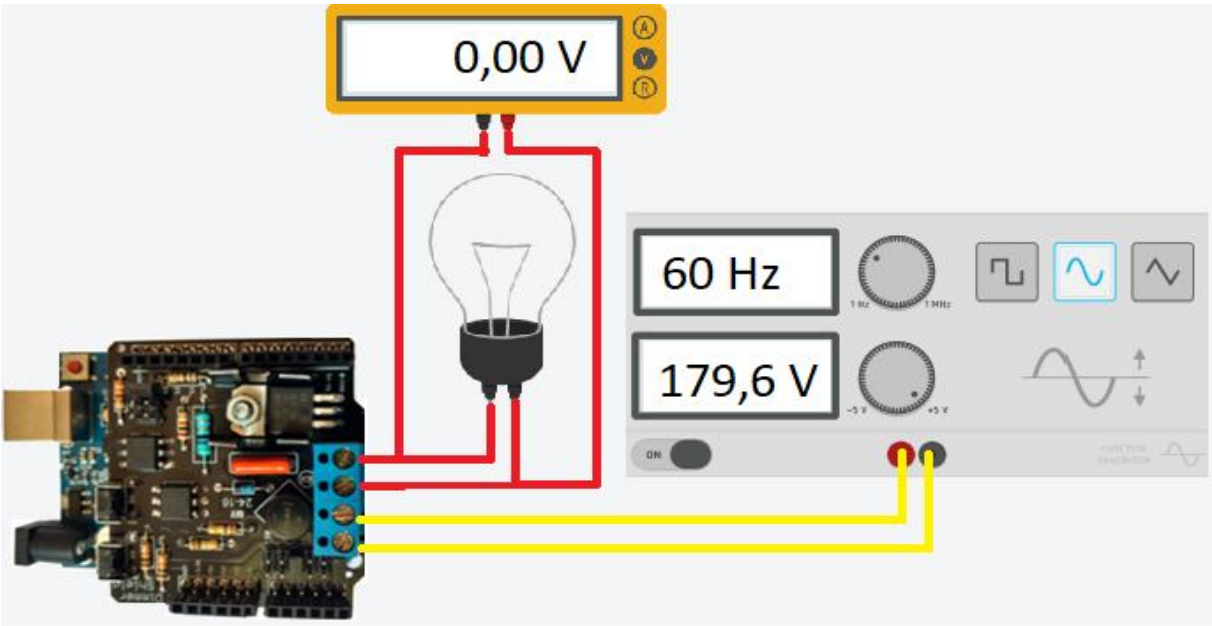
Fonte: o autor.

O tempo de cada ciclo, que foi impresso no *display*, foi configurado para 40 ms e os picos se mantiveram entre 200 e –200 V.

3.2.5 Testes com o Dimmer Shield

Para fazer a verificação e a comparação dos valores de tensão que saíram do Dimmer Shield com o item 3.2.4, um multímetro em paralelo foi conectado com a lâmpada, cuja configuração é demonstrada na Figura 56. Os valores de tempo para controlar o disparo do Triac foram inseridos via monitor serial do IDE do Arduino.

Figura 56 – Verificação dos valores reais do Triac



Fonte: o autor.

3.2.6 Montagem do protótipo final

Após os parâmetros do sistema *fuzzy* terem sido definidos no item 3.2.3, o próximo passo foi o de criar a programação no MATLAB na janela Comand Window.

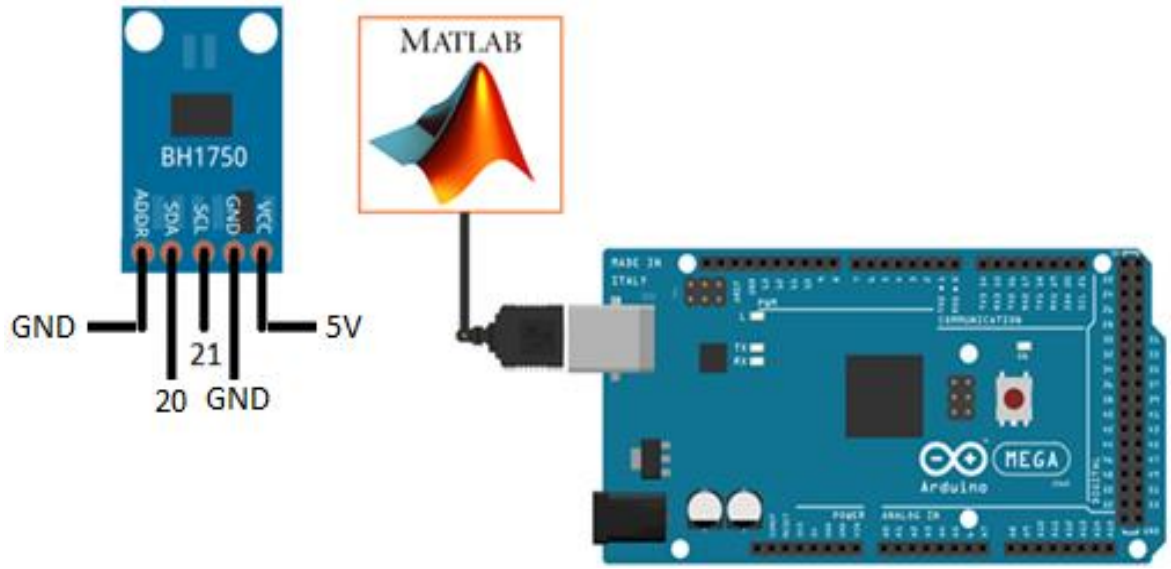
Os dados dos valores de entrada foram enviados pelo Arduino Mega, que está conectado ao sensor, e depois de concluído todo o processo de inferência *fuzzy*, o sinal de saída foi enviado ao Arduino Uno, que está conectado ao Dimmer Shield. Toda essa transmissão de dados foi realizada por comunicação serial.

Optou-se por trabalhar com dois Arduinos porque o Dimmer Shield requer controle preciso do tempo para o disparo no *gate* do Triac, e quando as funções que envolvem

tempo no Arduino são acionadas, ocorre um processo chamado de interrupção, no qual o microcontrolador interrompe todas as suas funções para executar apenas a contagem de tempo. Como isso pode comprometer o processo de leituras do sinal pelo sensor, um Arduino foi utilizado com o objetivo de controlar os sinais de entrada e o outro com o de controlar os sinais da saída.

A montagem física do protótipo é ilustrada nas figuras 57 e Figura 58. Os testes foram realizados na mesma base indicada pela Figura 39, mas com a seguinte modificação: a fonte de luz foi posicionada a 2,50 m do módulo GY-302.

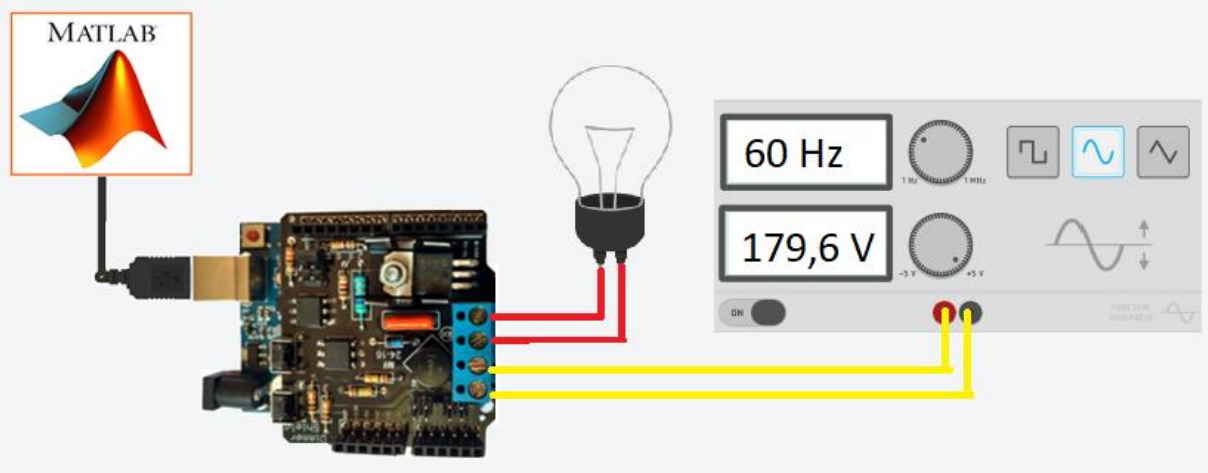
Figura 57 – Arduino responsável pela entrada de sinais



Fonte: o autor.

As marcações feitas nos pinos do sensor GY-302 (Figura 57) mostram em quais pinos do Arduino eles foram conectados.

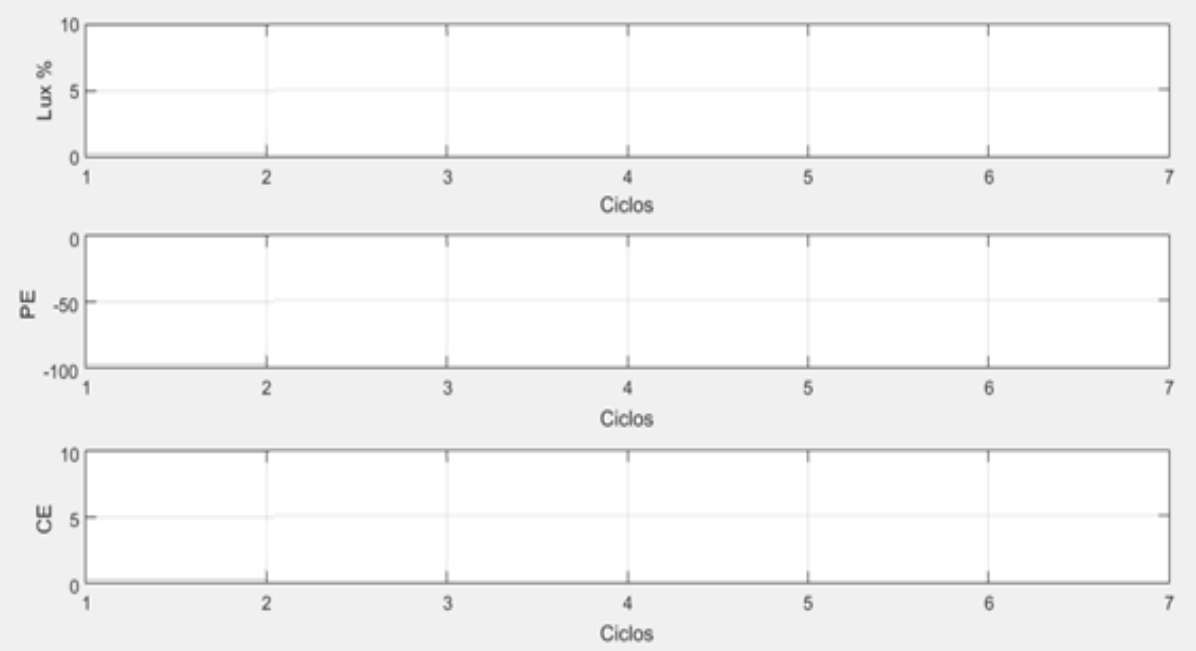
Figura 58 – Arduino responsável pela saída de sinais



Fonte: o autor.

Os resultados do comportamento do protótipo foram demonstrados em formato de gráficos (Gráfico 1) nos quais a linha superior indica a intensidade luminosa (lux), a linha do meio indica o valor do erro (%) em relação ao *setpoint* e a linha inferior indica o valor (%) para a correção do erro.

Gráfico 1 – Comportamento do protótipo



Fonte: o autor.

Os testes foram realizados em várias etapas, sendo a primeira, rampa, necessária para analisar se o protótipo alcançava os *setpoints* sem a interferência de luz externa, apenas com o uso das duas lâmpadas LEDs. Nessa etapa, os primeiros testes

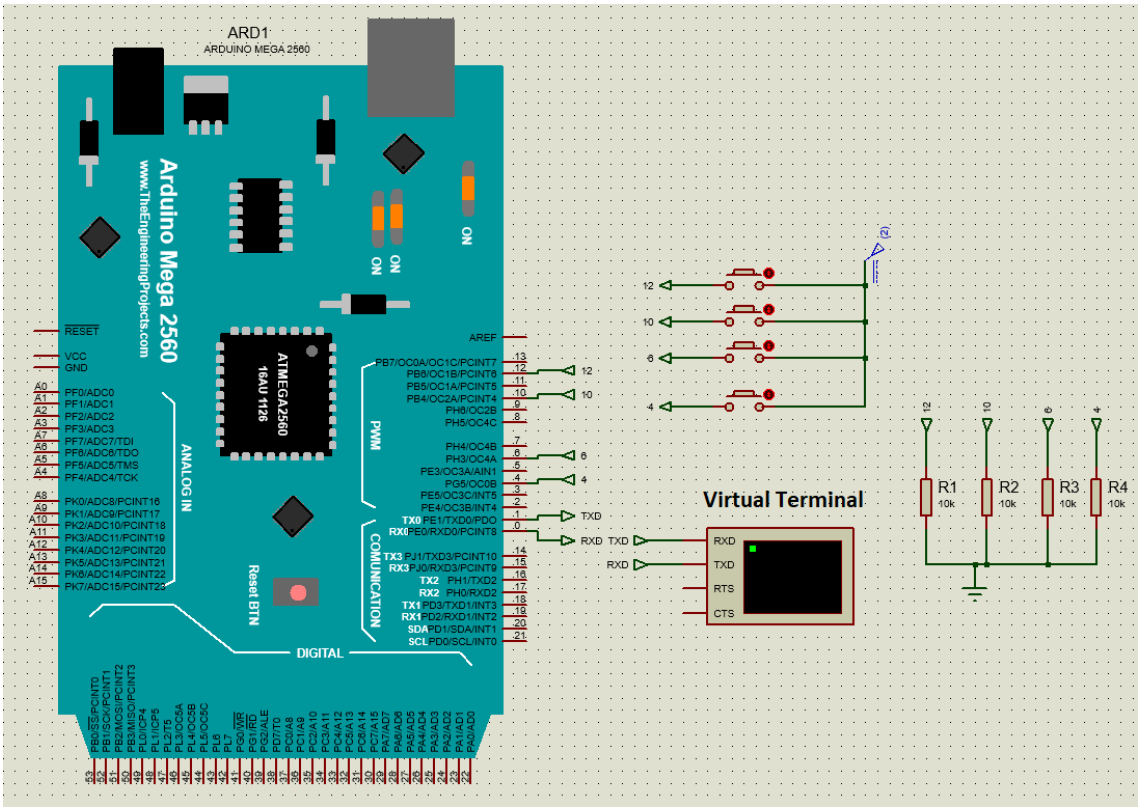
iniciaram com o Triac forçando as lâmpadas a iniciarem apagadas; em outros testes, o Triac forçou as lâmpadas a iniciarem com 100% de potência.

Na segunda etapa, o sistema foi exposto à interferência de luz externa, fornecida pela lâmpada incandescente. Os testes foram feitos de forma que o Triac fornecesse a potência necessária para forçar o sistema a iniciar com o valor de *setpoint* (10 lux e 25 lux) e o Dimmer analógico QD34 controlasse a intensidade da lâmpada incandescente, simulando assim a incidência de luz natural no sistema.

Um outro item importante é que as lâmpadas LEDs ficaram a 2,5 m do chão, simulando a altura do pé direito de uma granja, e o módulo GY-302 ficou posicionado a 30 cm do chão.

Além do controle de iluminação, o protótipo precisa controlar o intervalo em que as lâmpadas ficam desligadas (fotoperíodo). Para testar e demonstrar o controle do fotoperíodo, foram realizadas simulações por intermédio do *software* Proteus (Figura 59).

Figura 59 – **Configuração do circuito de simulação do fotoperíodo**



Fonte: o autor.

Optou-se pela simulação para facilitar a visualização dos resultados no virtual terminal. Essa ferramenta utiliza também a comunicação serial. As informações que

aparecerem em seu *display* somente são habilitadas nas simulações, pois é por meio da porta serial que o Arduino se comunica com o MATLAB e esses dados extras bloqueariam o funcionamento do protótipo.

No entanto, durante o seu funcionamento, esses dados não serão enviados para comunicação serial e somente servirão para mudar os valores das variáveis, da programação e dos *setpoints* (0, 10 e 25 lux) em intervalos de tempo programados.

Um detalhe importante é que durante as simulações não foi necessária a conexão com o MATLAB.

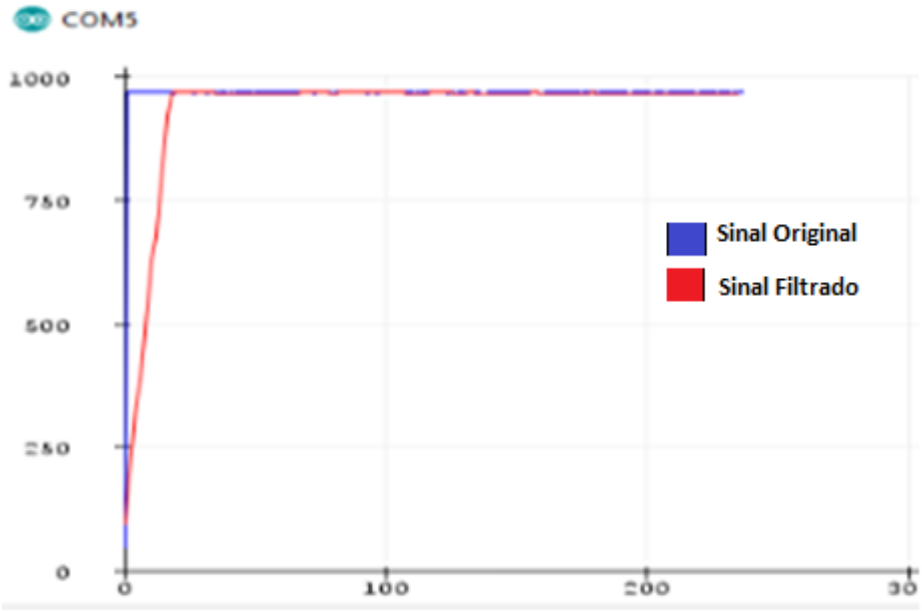
4 RESULTADOS E DISCUSSÃO

4.1 Filtro de média móvel

Do Gráfico 2 ao 7, serão apresentados o sinal original, em azul, gerado pelo potenciômetro, e o sinal filtrado pela média móvel, em vermelho. Esse sinal original tem um *range* de 0 a 1.023; respectivamente, 0 a 5 V. Para os exemplos mostrados do Gráfico 2 ao 4 foram utilizadas 20 amostras.

Nota-se no Gráfico 2 que existe uma defasagem entre os dois sinais, o que se deve ao tempo necessário para que o filtro faça os cálculos com as 20 amostras.

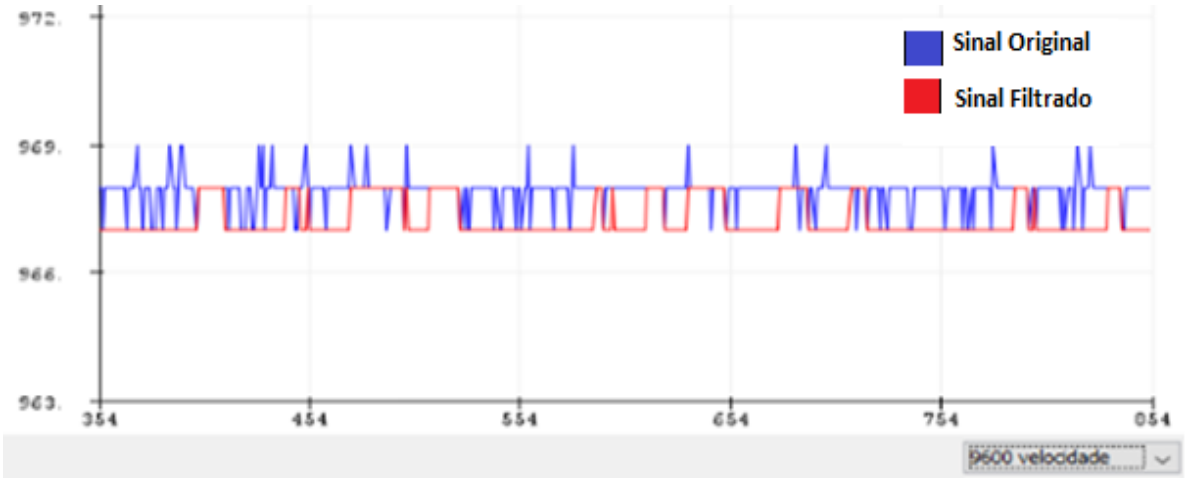
Gráfico 2 – Média móvel com 20 amostras



Fonte: o autor.

No Gráfico 3, pode-se observar que o sinal filtrado (vermelho) é suavizado no decorrer do tempo, em comparação com o sinal original. Ainda, observa-se que o sinal original apresenta muitos ruídos; logo, se ele fosse empregado diretamente como um sinal de entrada de algum sistema, fatalmente este leria um valor incorreto.

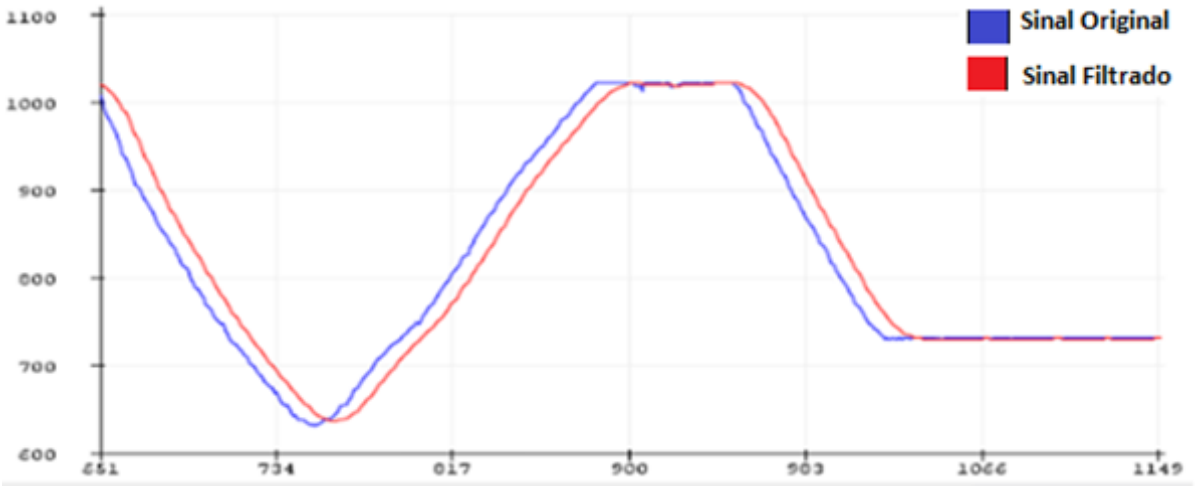
Gráfico 3 – Comparação de sinal original com sinal filtrado por 20 amostras



Fonte: o autor.

No Gráfico 4 é possível observar um sinal original oscilante e a defasagem do sinal filtrado.

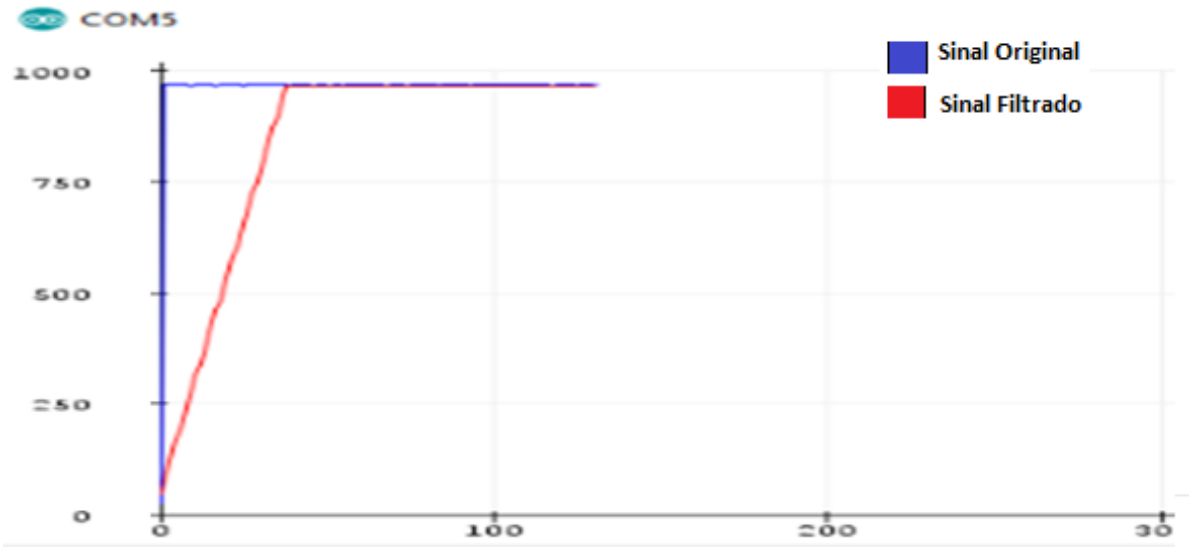
Gráfico 4 – Comportamento do filtro com sinal original oscilante



Fonte: o autor.

Para os exemplos apresentados nos gráficos 5 e 7 foram utilizadas 40 amostras.

Gráfico 5 – Gráfico de média móvel com 40 amostras

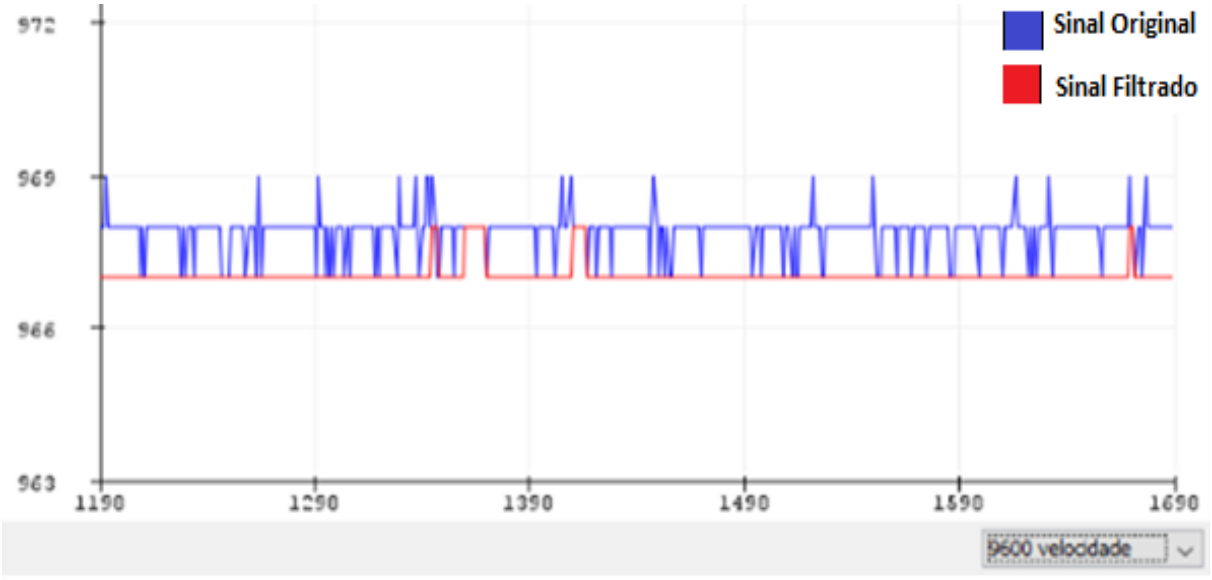


Fonte: o autor.

É perceptível que a defasagem do sinal original com o sinal filtrado é maior em comparação com o do **Erro! Fonte de referência não encontrada.** 2, que apresenta 2 0 amostras. Isso se deve também ao tempo necessário ao filtro para fazer os cálculos, mas agora com o dobro de amostras.

No Gráfico 6, também pode-se observar como o sinal filtrado (vermelho) é suavizado em comparação com o sinal original. Por causa das 40 amostras, sua suavização ficou melhor do que a apresentada pelo Gráfico 3.

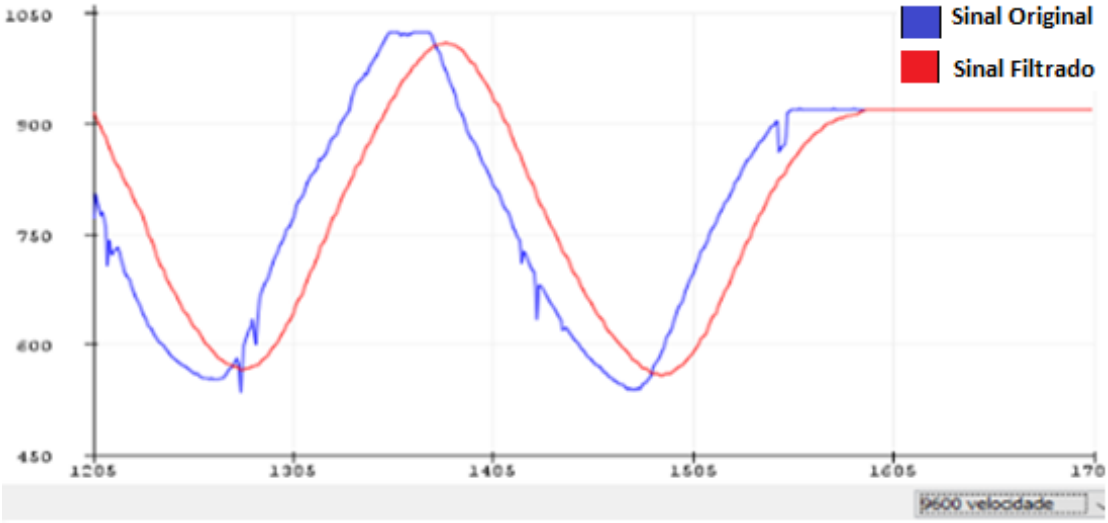
Gráfico 6 – Comparação de sinal original com sinal filtrado por 40 amostras



Fonte: o autor.

No Gráfico 7, pode-se notar um sinal original oscilante e a defasagem do sinal filtrado; esta, porém, bem maior em comparação ao Gráfico 4, visto que apresenta uma resposta mais lenta às oscilações.

Gráfico 7 – Comportamento do filtro com sinal original oscilante



Fonte: o autor.

Com base nos testes, decidiu-se por inserir o filtro de 40 amostras no programa do Arduino responsável pela leitura do módulo GY-302.

4.2 Comportamento do MÓDULO GY-302 em comparação com o luxímetro

Nessa etapa de testes, o filtro de média móvel com 40 amostras foi inserido na programação do Arduino Mega que está conectado ao sensor. Contudo, para uma acareação, os valores sem o uso do filtro de média móvel também foram utilizados.

Os valores da leitura do Módulo GY-302, com e sem o filtro de média móvel, e do luxímetro estão dispostos na Tabela 5.

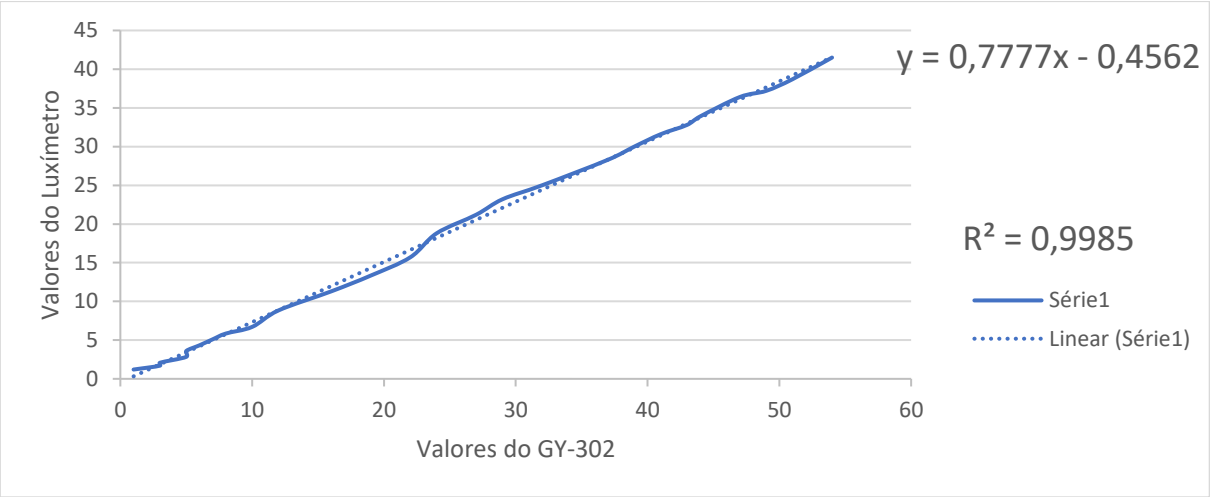
Tabela 5 – Valores do GY-302 x Luxímetro

GY-302 sem média Móvel (lx)	**GY-302 com média móvel (lx)**	Luxímetro **(lx)**
54	54	41,5
51	51	38,7
49	49	37,2
47	47	36,4
44	44	33,9
43	43	32,8
41	41	31,6
39	39	30
37	37	28,3
32	32	25
29	29	23,2
27	27	21,2
24	24	18,8
22	22	15,7
19	19	13,34
16	16	11,3
12	12	8,84
10	10	6,71
8	8	5,85
7	7	5,1
6	6	4,33
5	5	3,6
5	5	2,84
3	3	2,08
3	3	1,7
1	1	1,2

Fonte: o autor.

Na Tabela 5, é possível verificar que os valores do GY-302 e do luxímetro estão diferentes, o que ocasionará um funcionamento irregular do sistema. Para sanar esse problema foi plotado um gráfico (Gráfico 8) com os valores do GY-302 sem o filtro de média móvel, bem como do luxímetro, por meio do qual foi gerada uma equação para a correção da leitura.

Gráfico 8 – GY-302 x Luxímetro



Fonte: o autor.

A equação (9) foi extraída do Gráfico 8 por intermédio do *software* Microsoft Excel.

$$Vc = 0{,}7777x - 0{,}4562 \quad (9)$$

Em que Vc é o valor corrigido da leitura do GY-302 e x é a leitura do GY-302.

A equação foi inserida na programação da leitura do GY-302 em uma nova variável e os novos valores foram comparados com o luxímetro, conforme pode ser observado na Tabela 6 a seguir.

Tabela 6 – Correção do GY-302

Correção sem média móvel (lx)	Correção com média móvel (lx)	Luxímetro (lx)
43	43	43,3
40	40	40,9
39	39	39,4
37	37	37,7
35	35	36,1
34	34	35,1
31	31	31,7
25	25	26,4
22	22	22,5
19	19	20,5
18	18	18,16
15	15	15,34
14	14	13,27
11	11	11,17
9	9	8,85
7	7	7,07
6	6	6,13
5	5	5,44
4	4	4,55
3	3	3,76
2	2	2,31
1	1	1,78

Fonte: o autor.

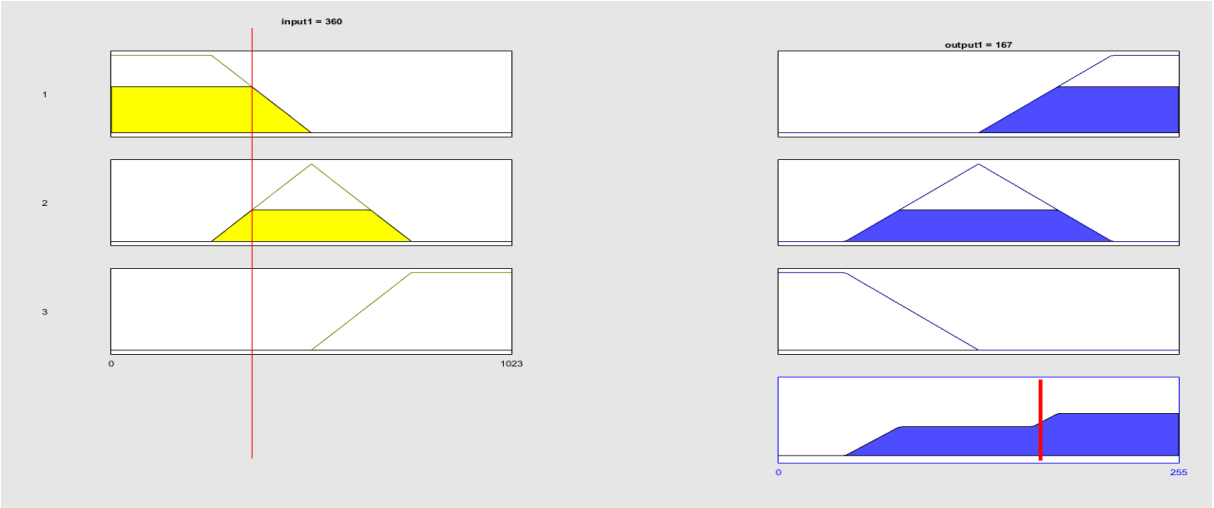
Na Tabela 6, é possível averiguar que a equação (9) corrigiu os valores obtidos pelo módulo GY-302 e os aproximou dos valores obtidos pelo luxímetro comercial.

Um detalhe apresentado pelas tabelas 5 e 6 é o de que os valores sem e com a implementação do filtro de média móvel foram os mesmos. Uma hipótese da ocorrência dessa situação é fato de o sensor não utilizar a entrada analógica do Arduino para a inserção de seus valores, e sim o protocolo de comunicação I2C. Isso é um indicativo de que não é necessário utilizar a correção por filtro de média móvel, dado que pode ser útil a trabalhos futuros que fizerem uso desse sensor.

4.3 Simulação *fuzzy* de teste

Para a simulação no FLD, foi estabelecido um sinal de entrada de 361. O seu processo *fuzzy* pode ser observado na Figura 61.

Figura 61 – Sistema *fuzzy* secundário simulado no FLD

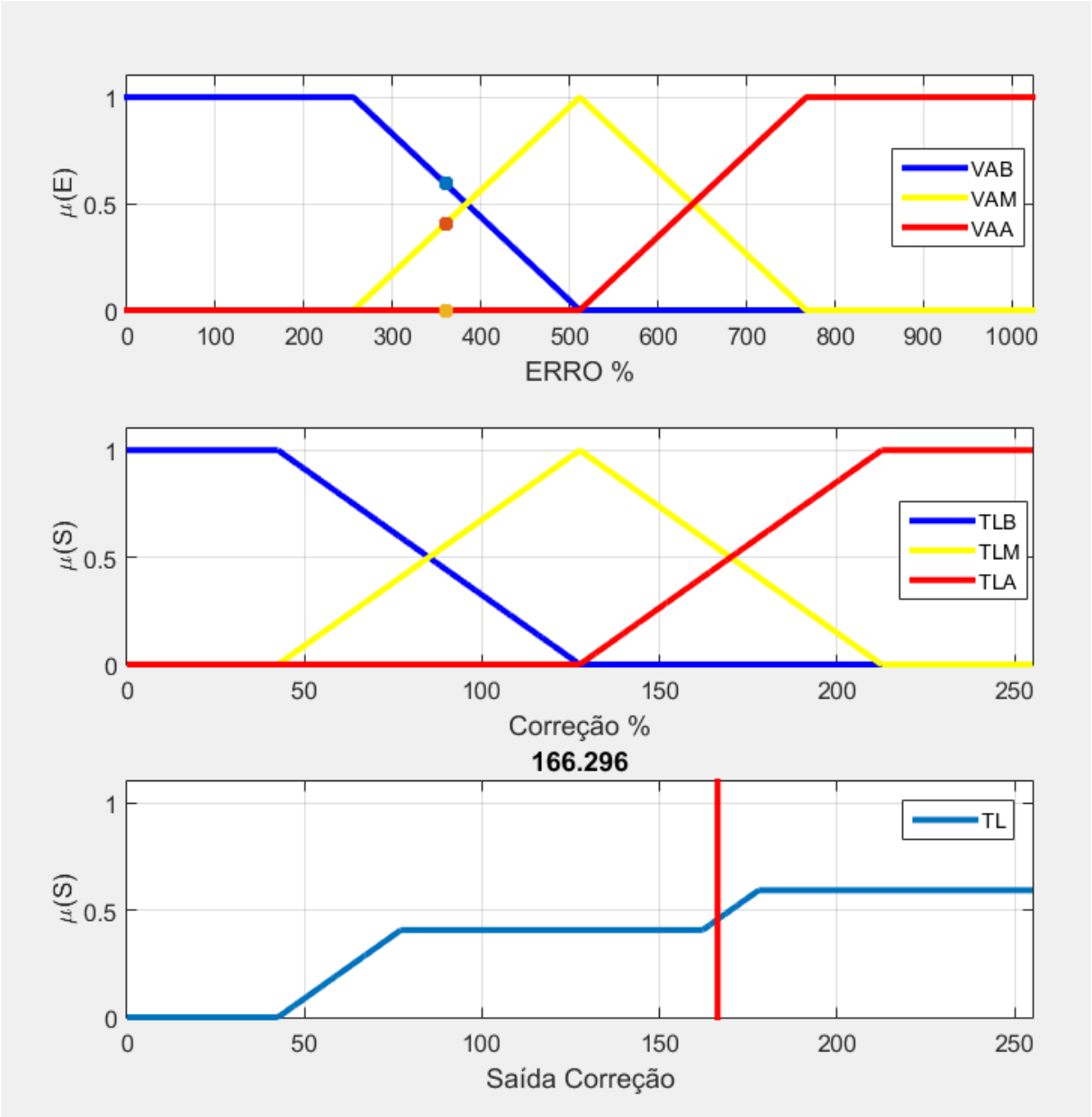


Fonte: o autor.

Após obter os resultados por meio do FLD, a próxima etapa foi compará-los com a programação *fuzzy* feita no Editor. Como esse programa está conectado diretamente ao Arduino, o valor de entrada de 360 foi obtido ao se ajustar o potenciômetro. O seu valor foi exibido na janela "*Command Window*", o que indica que a comunicação serial estava funcionando.

Assim que o valor foi exibido na janela "*Command Window*", o sistema *fuzzy* foi iniciado, conforme o Gráfico 9.

Gráfico 9 – Sistema *fuzzy* secundário simulado no "Editor"



Fonte: o autor.

O sinal de saída de 166.296 foi enviado ao Arduino e convertido em valor PWN, que saiu no pino 9 conectado ao LED. Os valores de PWN (0 e 255) representam um *range* de tensão de 0 a 5V, que é o *range* de tensão de saída do Arduino.

Ao converter 166,296 PWM para volts, obteve-se o valor de tensão de 3,26V, que foi aferido com o uso do multímetro. Os terminais do multímetro foram conectados entre o pino 9 e o gnd do Arduino, cujo valor obtido foi de 3,24V. Esse valor de tensão mostra que a comunicação MATLAB e Arduino ocorreu de forma satsfatória.

Os valores e gráficos das saídas, com entrada de 360, geraram valores muito próximos. Entretanto, para averiguar se os sistemas *fuzzy* estão compatíveis, bem

como verificar o comportamento deles e a comunicação entre MATLAB e Arduino, foram inseridos outros valores em suas entradas. Esses valores e seus respectivos valores de saídas, FLD e Editor, estão dispostos na Tabela 7.

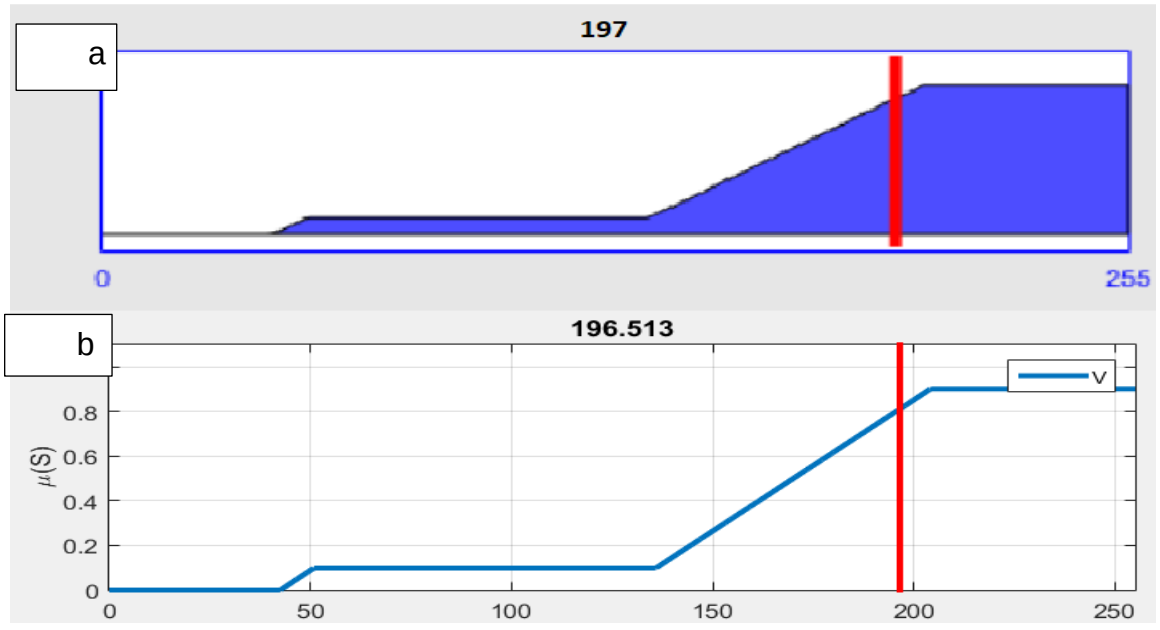
Tabela 7 – Entradas e suas respectivas saídas no FLD e no Editor

Valor de entrada	Saída no FLD	Saída no Editor	Tensão Teórica V	Tensão Real V	Erro Tensão Teórica X Real %
281	197	196,513	3,85	3,82	0,09
431	146	146,06	2,86	2,84	0,09
900	45,3	46,02	0,88	0,89	0,10

Fonte: o autor.

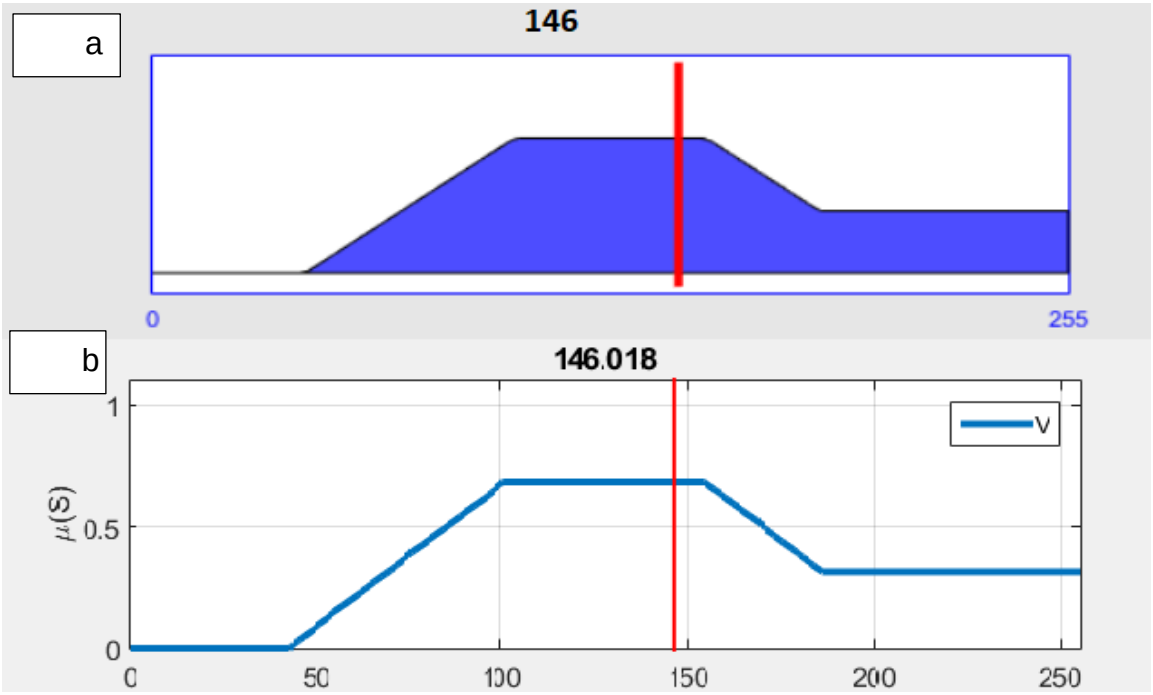
As respectivas “defuzzyficações” foram apresentadas nos gráficos 10, 11 e 12. Destaca-se que a designação “a” corresponde a FLD e “b” à programação feita no Editor.

Gráfico 10 – Valores saídas e gráficos referente ao valor de entrada 281



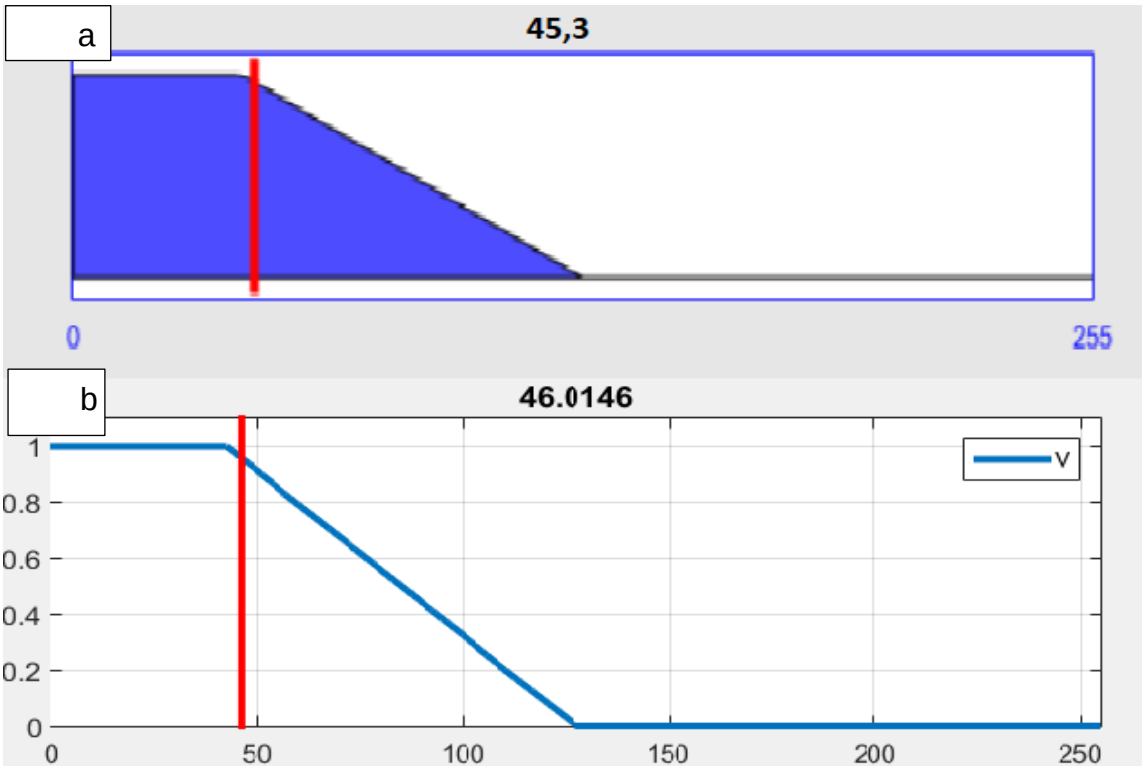
Fonte: o autor.

Gráfico 11 – Valores saídas e gráficos referente ao valor de entrada 431



Fonte: o autor.

Gráfico 12 – Valores saídas e gráficos referente ao valor de entrada 900



Fonte: o autor.

Os valores e os gráficos gerados em ambos os sistemas, FLD e Editor, foram bem aproximados. Isso indica que o sistema *fuzzy* feito no Editor foi programado de forma adequada.

4.4 Comparação dos resultados teóricos do Triac com o gerador de pulsos

Para as simulações do comportamento do Triac foram usados os ângulos 0, 30°, 60°, 90°, 120º e 150º, conforme disposto na Tabela 8, e seus respectivos tempos, que foram obtidos pela equação (6).

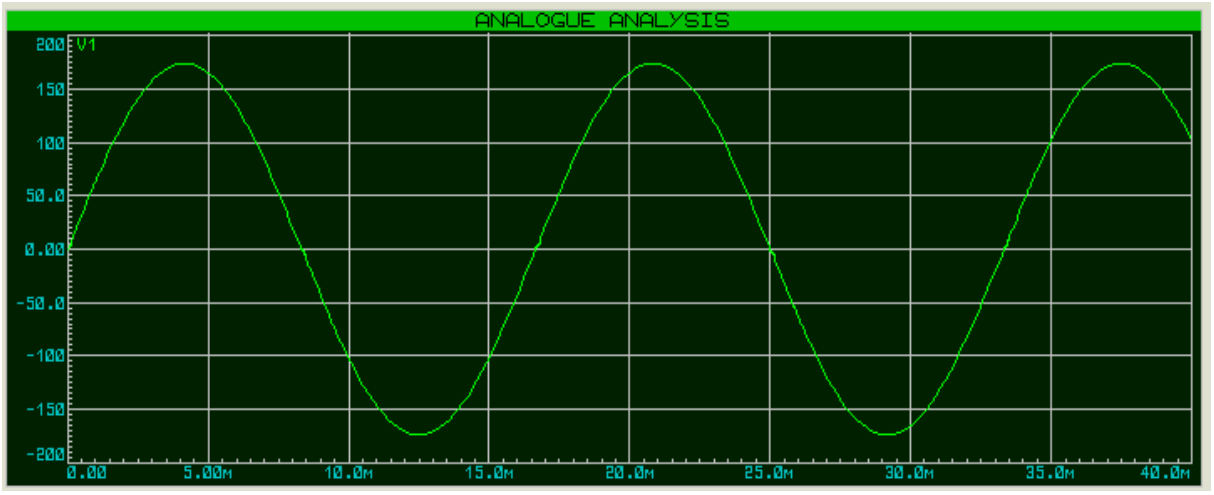
Tabela 8 – Ângulos e seus respectivos tempos para o disparo do Triac

Graus	Radianos	T_α(ms)
0	0	0
30°	π/90	1,39
60°	π/3	2,78
90°	π/2	4,17
120°	2π/3	5,56
150°	5 π/3	6,94

Fonte: o autor.

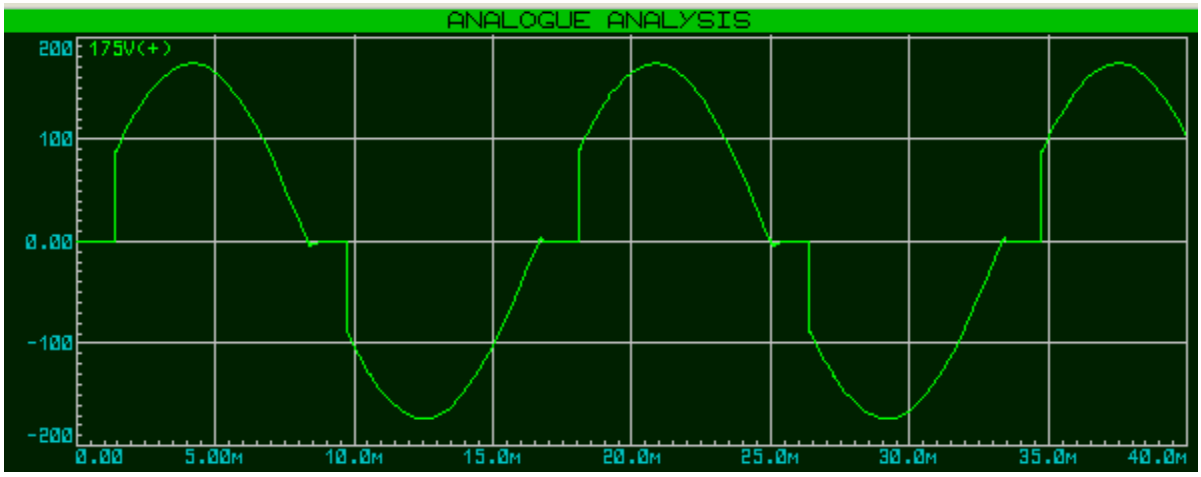
Os valores de tempo da Tabela 8 foram inseridos no gerador de pulsos com vistas a se obter o ângulo de disparo desejado. O comportamento das ondas (tensão) referente aos tempos de disparo pode ser observado nos gráficos 13 a 18.

Gráfico 13 – Disparo do Triac a 0°



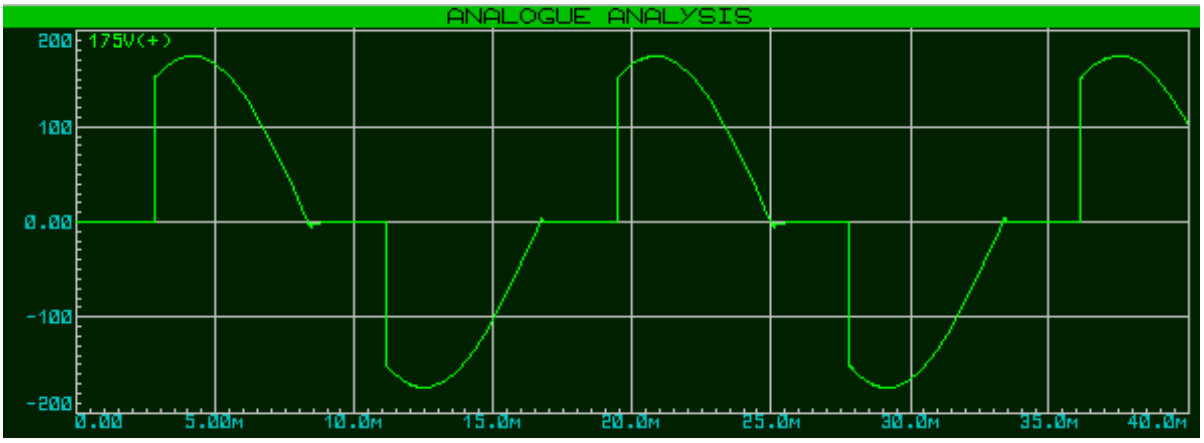
Fonte: o autor.

Gráfico 14 – Disparo do Triac a 30°



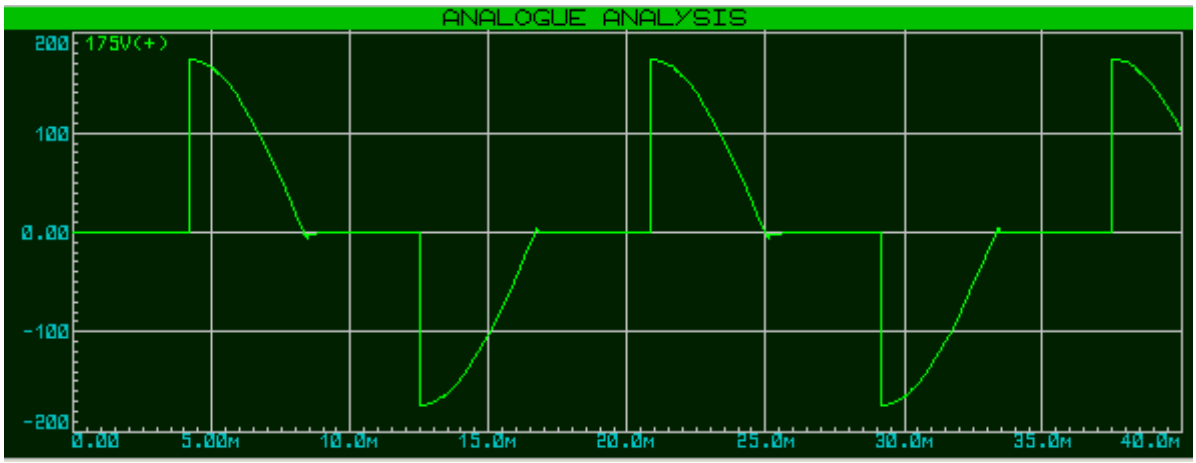
Fonte: o autor.

Gráfico 15 – Disparo do Triac a 60°



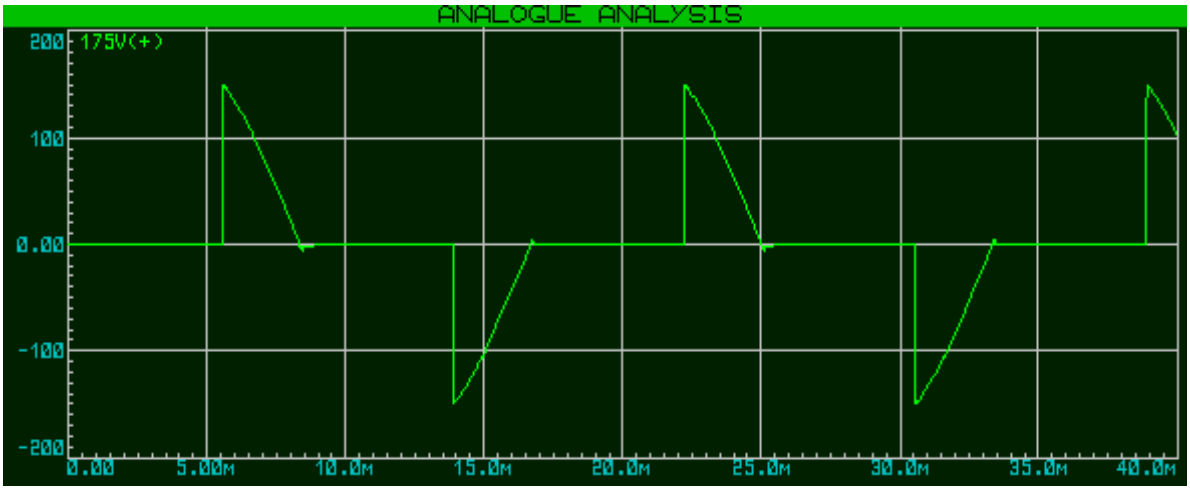
Fonte: o autor.

Gráfico 16 – Disparo do Triac a 90°



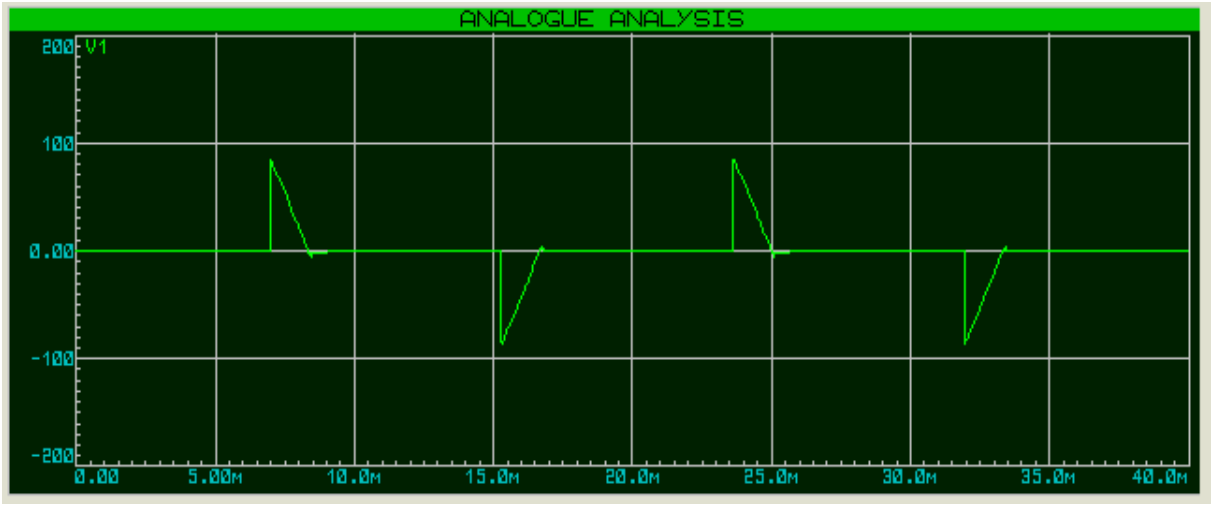
Fonte: o autor.

Gráfico 17 – Disparo do Triac a 120°



Fonte: o autor.

Gráfico 18 – Disparo do Triac a 150°



Fonte: o autor.

Nota-se, nessa seleção de gráficos, que quanto maior o tempo (ângulo) de disparo, menor é a tensão que alimenta a carga R_L.

Os valores de tensão eficaz e de potência de cada sinal de onda foram obtidos pelo voltímetro e pelo wattímetro inserido em R_L. Para averiguá-los, foi feita uma comparação com base nos valores teóricos obtidos pela equação (7) e (8) (Almeida 2018), conforme demonstrado pela Tabela 9.

$$V_{eficaz} = V_p * \sqrt{\frac{1}{2} - \frac{\alpha}{2\pi} + \frac{sen(2\alpha)}{4\pi}} \quad (7)$$

$$P = \frac{V_{eficaz}^2}{R_L} \quad (8)$$

Tabela 9 – Tensões e potências dos ângulos

Ângulo (graus)	Ângulo (radianos)	Veficaz (V) fórmula (7)	Veficaz (V) simulação	Erro (%)	Potência (W) fórmula (8)	Potência (W) simulação	Erro (%)
0	0	123,74	122,5	1,00	153,12	150 ,5	1,74
30°	π/6	121,95	120	1,59	148,72	146	1,86
60°	π/3	110,99	109,5	1,35	123,19	120	2,66
90°	π/2	87,5	86,3	1,37	76,56	73,9	3,59
120°	2π/3	54,71	54,25	0,84	29,96	28,9	3,66
150°	5π/6	21,01	20,55	2,19	4,41	4,23	4,25

Fonte: o autor.

Nota-se, pela Tabela 9, que os valores teóricos ficaram próximos dos valores obtidos pela simulação. Isso indica que o Triac funcionará corretamente quando os pulsos elétricos forem inseridos em seu *gate*.

4.5 Comportamento real do Triac

Uma vez analisados os resultados teóricos e os da simulação, buscou-se averiguar o funcionamento do Dimmer Shield (circuito físico do Triac) com duas lâmpadas LED instaladas em sua saída, com o objetivo de comparar os dados obtidos com os valores teóricos. Em paralelo com a lâmpada, foi instalado o multímetro para averiguação dos valores de tensão.

Os testes foram feitos com o uso das tensões 127 e 220 V. A tensão eficaz aferida na tomada de 127 V, no momento do teste, foi de 123,49 V, e na tomada de 220 V foi de 210 V. Os valores das variações da tensão no que concerne ao disparo do Triac referentes, respectivamente, à tensão eficaz de 123,49 V e à tensão eficaz de 210 V, bem como os valores teóricos podem ser observados nas tabelas 10 e 11.

Tabela 10 – Valores teóricos e valores do Dimmer Shield em 123,49 volts

Ângulo (°)	Tempo para o disparo (ms)	Tensão teórica (V)	Tensão multímetro (V)	Erro (%)
30,25	1,4	121,66	120	1,38
47,63	2,2	116,76	113	3,33
64,67	2,99	107,94	102	5,82
81,37	3,77	95,28	88	8,27
97,73	4,52	79,53	73	8,95
113,75	5,26	61,84	57	8,49
129,43	5,99	43,63	41	6,41

Fonte: o autor.

Tabela 11 – Valores teóricos e valores do Dimmer Shield em 210 volts

Ângulo (°)	Tempo para o disparo (ms)	Tensão teórica (V)	Tensão multímetro (V)	Erro (%)
0	0	210	210	0,00
30	1,39	206,95	203	1,95
60	2,78	188,36	182	3,49
90	4,17	148,48	139	6,82
120	5,55	92,85	86	7,97
150	6,94	35,55	41	– 13,29

Fonte: o autor.

Os valores indicados pelas tabelas 10 e 11 mostram que o Dimmer Shield está funcionando de forma adequada, visto que apresenta uma pequena diferença entre a tensão teórica e a real.

4.6 Comportamento do protótipo

Uma vez verificada a perfeita comunicação entre o MATLAB e os Arduinos (item 4.3), na próxima etapa foi inserido no protótipo o sistema de inferência *fuzzy* descrito no item 3.2.3.

Nos gráficos 19 a 31, estão dispostos o comportamento do sistema de inferência *fuzzy*. As grandezas de cada linha do gráfico já foram descritas no item 3.2.6.

Os testes, que foram separados em duas etapas, foram estruturados de modo a averiguar o comportamento do protótipo em diversas situações. A primeira etapa verificou o seu comportamento sem a interferência de luzes externas ao sistema; a

segunda, o seu comportamento com a interferência de luz externa. Os testes foram feitos com o uso de tensão eficaz de 210 volts.

O módulo GY-302 foi posicionado a 30 cm do chão pelo fato de essa ser a altura média de um frango adulto. Entretanto, o frango em uma granja necessita de dois *setpoints* em etapas distintas de sua vida. Conforme já foi descrito anteriormente, nos seus primeiros sete dias ele requer 25 lux, contudo, a sua altura nesse período é de 5 cm; passando esse período, ele precisa de uma incidência de 10 lux, quando sua altura será de 30 cm.

Desse modo, para conferir se ocorrem alterações na leitura feita a 30 cm para o *setpoint* de 25 lux, fez-se um experimento com o luxímetro e o modulo GY-302 que consistiu em posicionar o primeiro a 5 cm do chão e o segundo a 30 cm. O próximo passo foi verificar o valor da leitura do GY-302 quando o luxímetro indicava 25 lux. O valor informado pelo módulo GY-302 foi de 33 lux, o qual foi adotado como *setpoint* nos experimentos futuros.

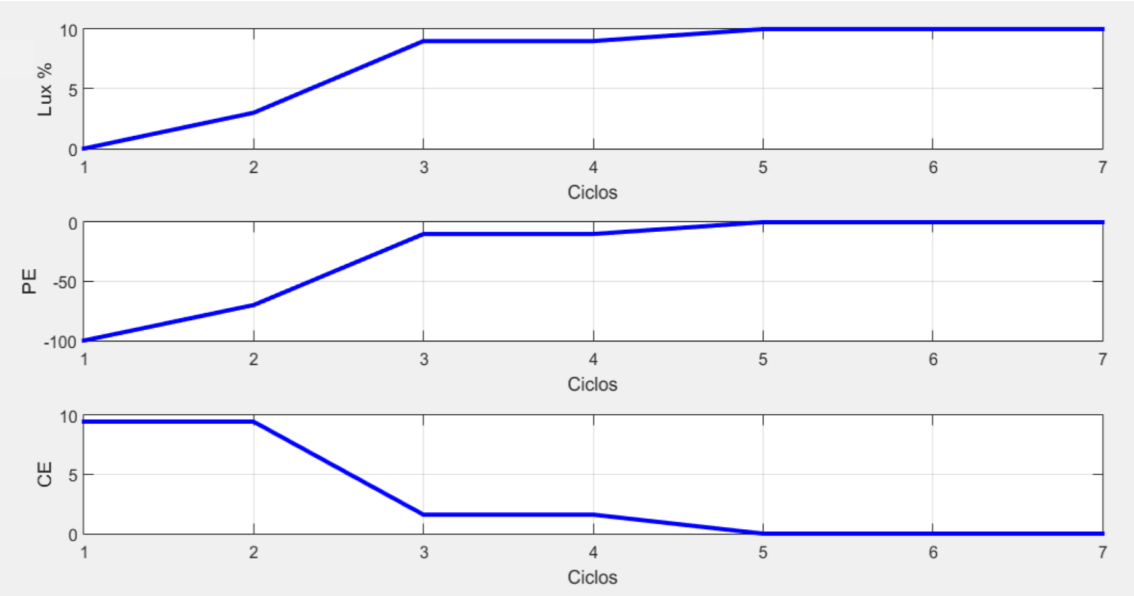
Outro detalhe importante é que o luxímetro comercial foi usado para acarear os resultados, com o objetivo de verificar se o sistema realmente alcançou os valores de *setpoint*.

No primeiro e no segundo teste da primeira etapa, a proposta foi de que no início as lâmpadas se mantivessem desligadas; uma vez iniciados os testes, buscou-se averiguar se o protótipo atingia os *setpoints* de 10 e 33 lux. No Gráfico 19, pode-se constatar o comportamento do protótipo com o *setpoint* de 10 lux.

Um detalhe importante é que, nos testes em que o *setpoint* é 10 lux, o luxímetro comercial está a 30 cm de altura do solo, a mesma distância do GY-302; já nos de 33 lux, o luxímetro comercial está a 5 cm do solo.

Analisando o Gráfico 19 a seguir, é possível verificar que o protótipo atingiu o *setpoint* no ciclo 5, ou seja, em 15 segundos. Outro fator importante é que o luxímetro comercial indicou 9,64 lux, valor dentro da tolerância de 20% indicada por Cobb-Vantress (2018).

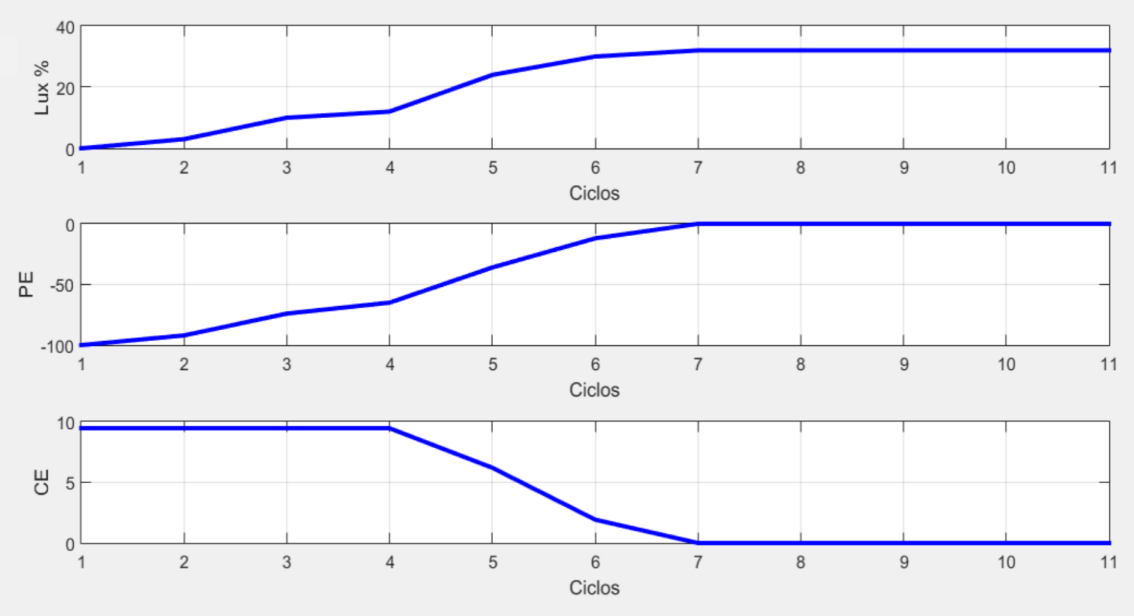
Gráfico 19 – Rampa de 0 até 10 lux



Fonte: o autor.

Continuando o teste com as lâmpadas LEDs desligadas, no Gráfico 20 pode-se observar o comportamento do protótipo com o *setpoint* de 33 lux.

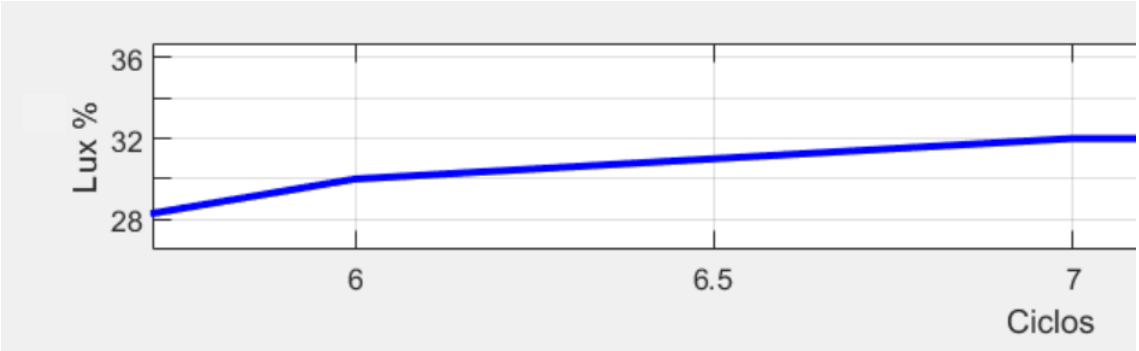
Gráfico 20 – Rampa de 0 até 33 lux



Fonte: o autor.

Como a análise do Gráfico 20 impossibilita a observação do valor exato da linha “lux”, o Gráfico 21 a seguir mostra essa linha de forma ampliada. Analisando-a, é possível verificar que o protótipo atingiu um valor próximo do *setpoint* de 33 lux no ciclo 7, ou seja, em 21 segundos. O luxímetro comercial indicou o valor de 24,7 lux.

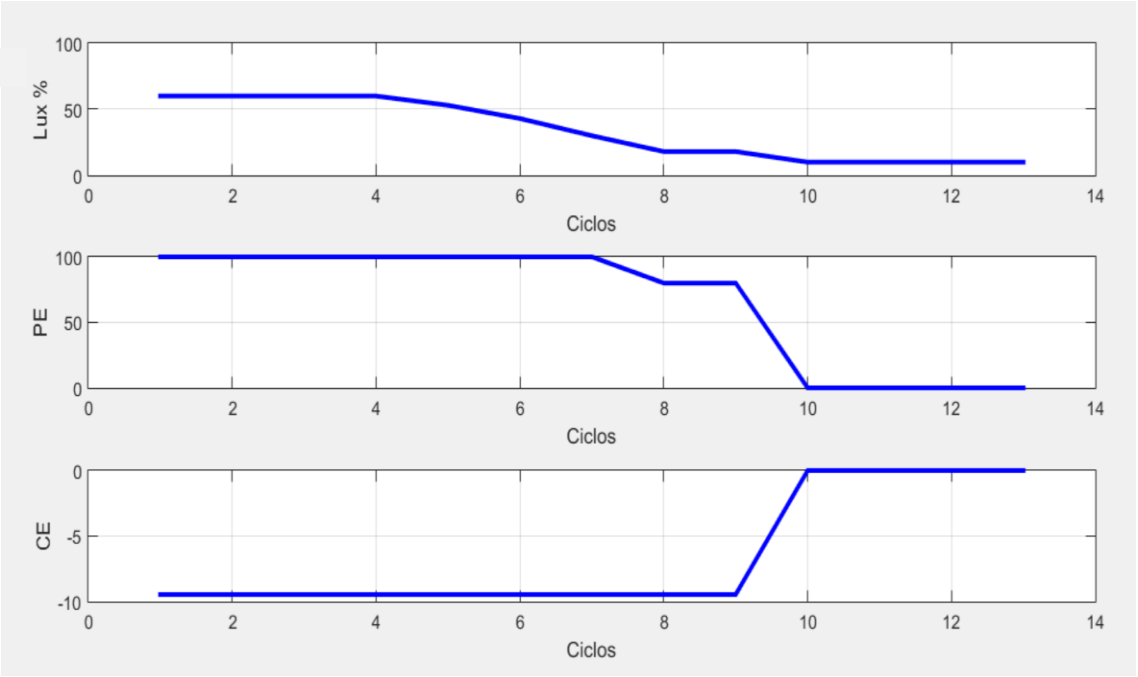
Gráfico 21 – Ampliação da entrada lux



Fonte: o autor.

No terceiro e quarto teste da primeira etapa de testes, foi proposto que as lâmpadas de início se mantivessem ligadas a 100% de potência elétrica; uma vez iniciado o teste, buscou-se verificar se o protótipo atingia os *setpoints* de 10 e 33 lux. No Gráfico 22, pode-se observar o comportamento do protótipo com o *setpoint* de 10 lux.

Gráfico 22 – Rampa de 58 lux até 10 lux

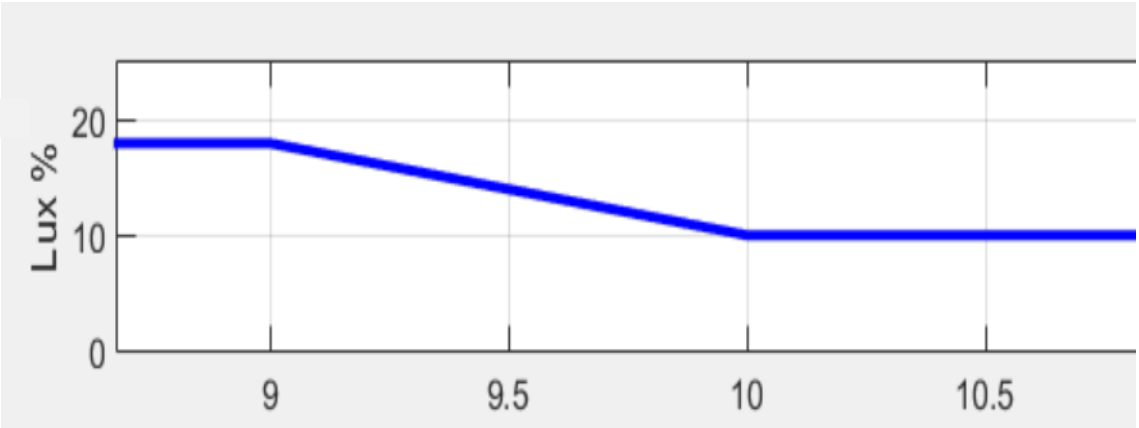


Fonte: o autor.

Assim como no Gráfico 20, a análise do Gráfico 22 impossibilita a observação do valor exato da linha “lux”. Por essa razão, Gráfico 23 mostra essa linha de forma ampliada. Analisando o Gráfico 23, é possível verificar que o protótipo atingiu o

setpoint no ciclo 10, ou seja, em 30 segundos. O luxímetro comercial indicou o valor de 10,13 lux.

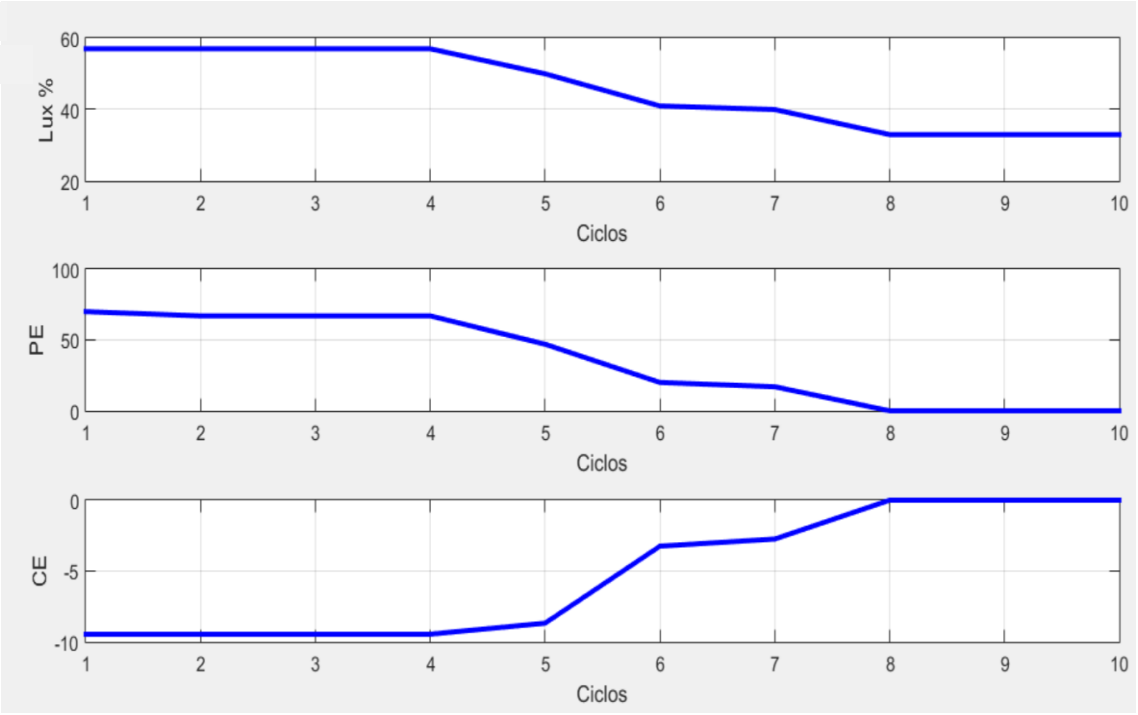
Gráfico 23 – Ampliação da entrada lux



Fonte: o autor.

No Gráfico 24, pode-se observar o comportamento do protótipo com o *setpoint* de 33 lux e as lâmpadas a 100% de potência elétrica.

Gráfico 24 – Rampa de 58 lux até 33 lux

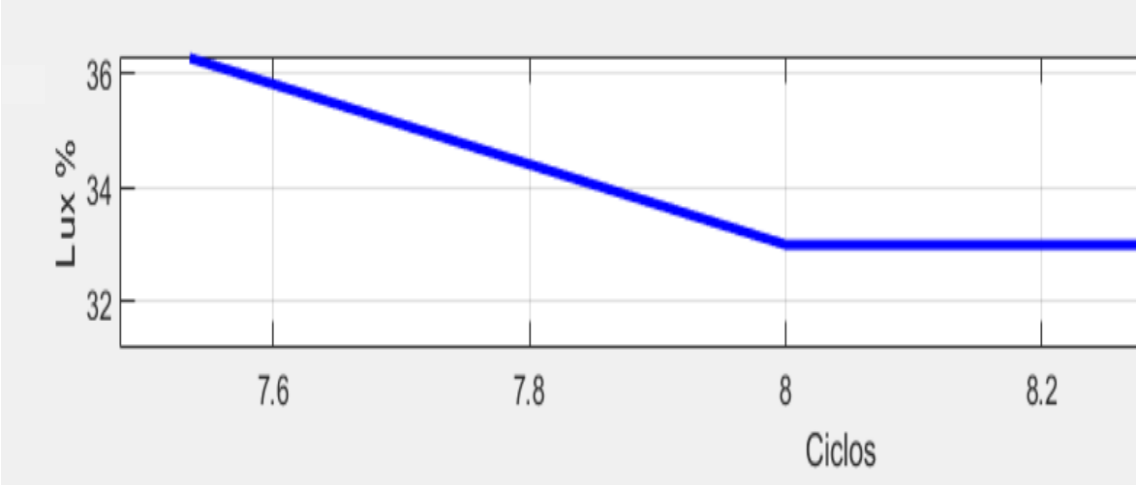


Fonte: o autor.

No Gráfico 25, pode-se analisar a linha “lux” do Gráfico 24 ampliada, no qual é possível constatar que foi atingido o valor de 33 lux no ciclo 8, ou seja, em 24

segundos. Analisando ainda esse gráfico, pode-se verificar que o protótipo atingiu o *setpoint* de 33 lux no ciclo 8, ou seja, em 24 segundos. O luxímetro comercial indicou o valor de 25,5 lux.

Gráfico 25 – Ampliação da Entrada lux da Gráfico 24



Fonte: o autor.

Na segunda etapa de testes, o protótipo foi testado com a luz de uma lâmpada incandescente incidindo no sistema, que servirá para simular a incidência de luz natural. Como ela não estava conectada ao sistema de controle (Triac), foi utilizado o Dimmer analógico QD34 (Figura 32) para controlar a intensidade luminosa.

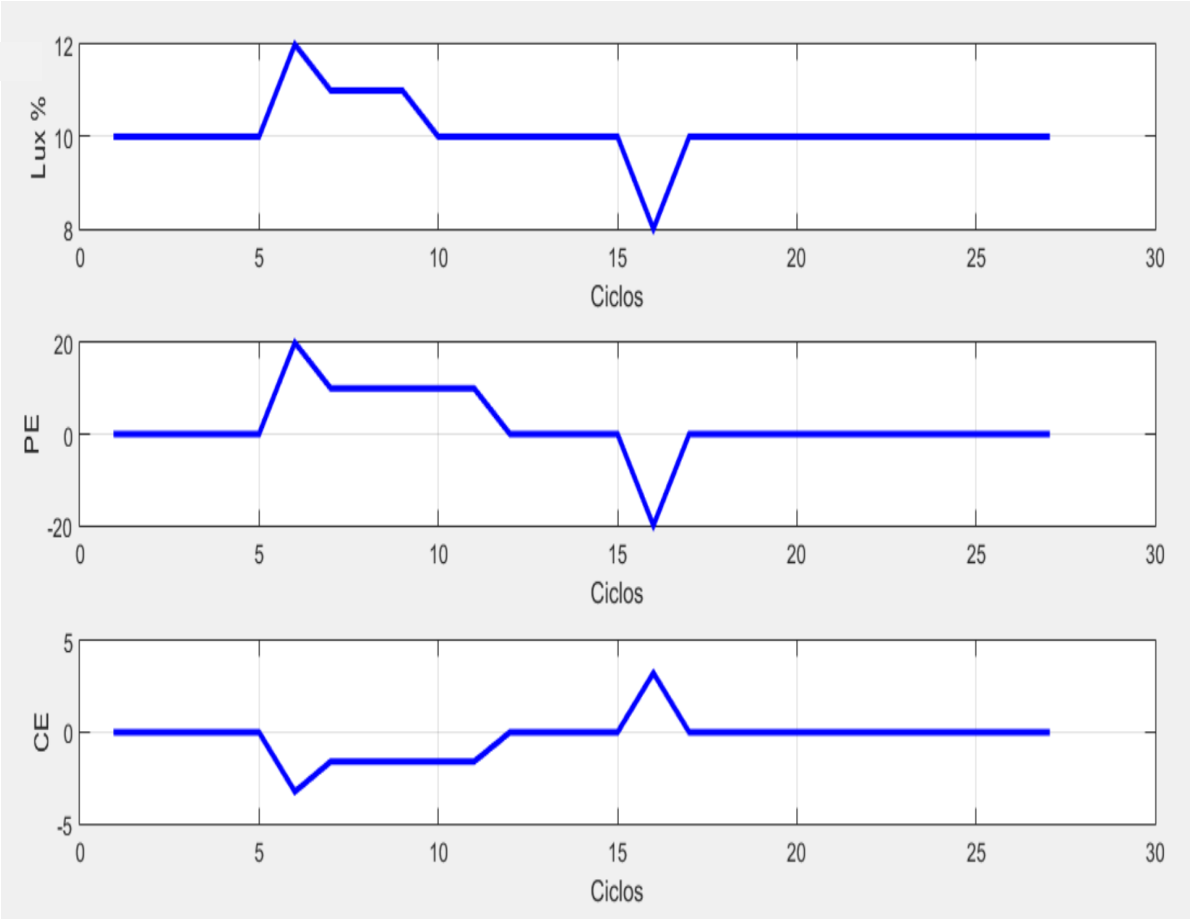
Em todos os testes, de início as lâmpadas LEDs estavam com o *setpoint* já estabelecido e a lâmpada incandescente desligada. Em determinado ciclo, a lâmpada incandescente era ativada na sua intensidade luminosa pré-estabelecida, voltando a ser desligada em outro ciclo. Durante as atividades entre os ciclos, a intensidade luminosa e a tensão das lâmpadas LEDs foram aferidas.

No teste representado pelo Gráfico 26, o *setpoint* utilizado foi o de 10 lux e a lâmpada incandescente foi estabelecida em 1,67 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 10,35 lux e a tensão medida era de 53 volts.

A lâmpada incandescente foi ligada no ciclo 5, o que aumentou a entrada para 12 lux. O protótipo se estabeleceu no ciclo 10, no qual foram aferidos 37 volts pelo multímetro e 9,95 lux pelo luxímetro comercial.

No ciclo 15, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 8 lux. O protótipo se estabeleceu no ciclo 17, em que foram aferidos 42 volts pelo multímetro e 9,42 lux pelo luxímetro comercial.

Gráfico 26 – *Setpoint* 10 lux com lâmpada externa em 1,67 lux



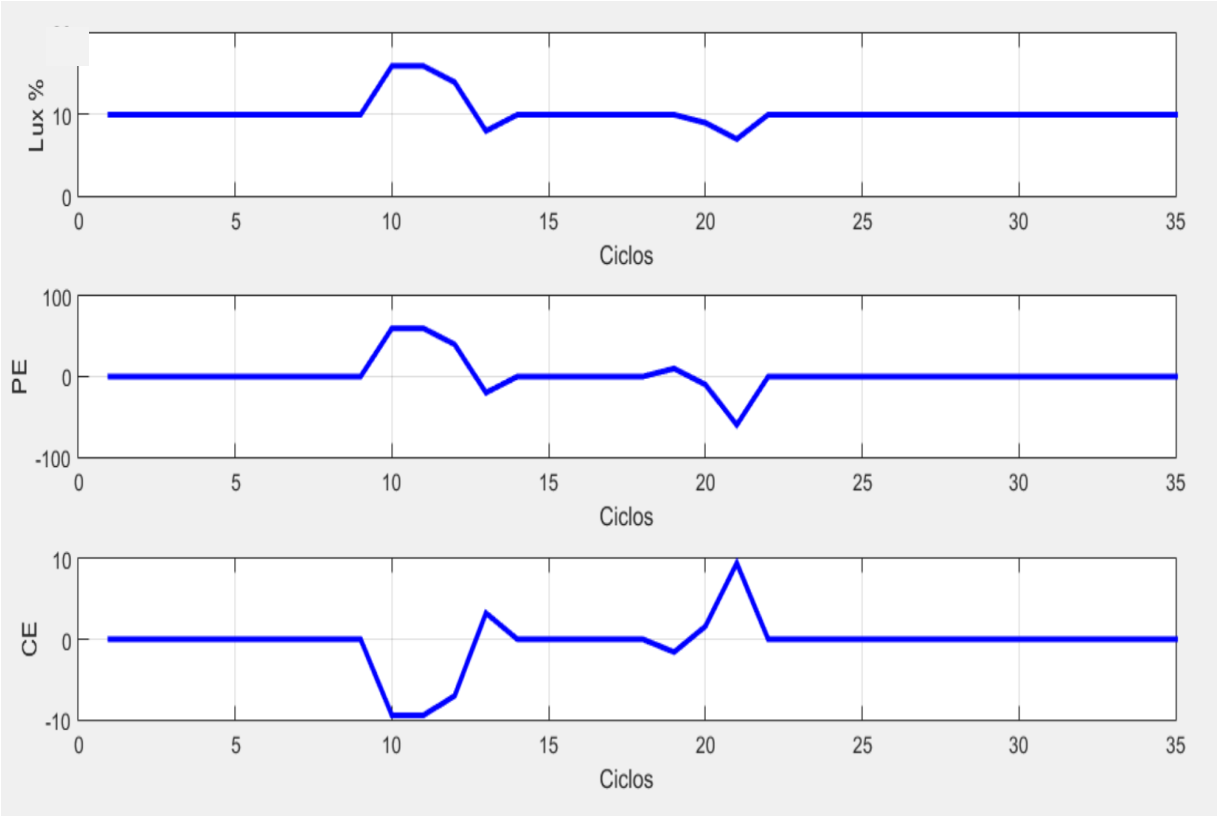
Fonte: o autor.

No teste representado pelo Gráfico 27, utilizou-se o *setpoint* de 10 lux e a lâmpada incandescente foi estabelecida em 5,31 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 10,26 lux e a tensão medida era de 52 volts.

A lâmpada incandescente foi ligada no ciclo 8, o que aumentou a entrada para 15 lux. O protótipo se estabeleceu no ciclo 13, no qual foram aferidos 25 volts pelo multímetro e 10,13 lux pelo luxímetro comercial.

No ciclo 20, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 5 lux. O protótipo se estabeleceu no ciclo 22, em que foram aferidos 41 volts pelo multímetro e 9,38 lux pelo luxímetro comercial.

Gráfico 27 – *Setpoint* 10 lux com lâmpada externa em 5,31 lux



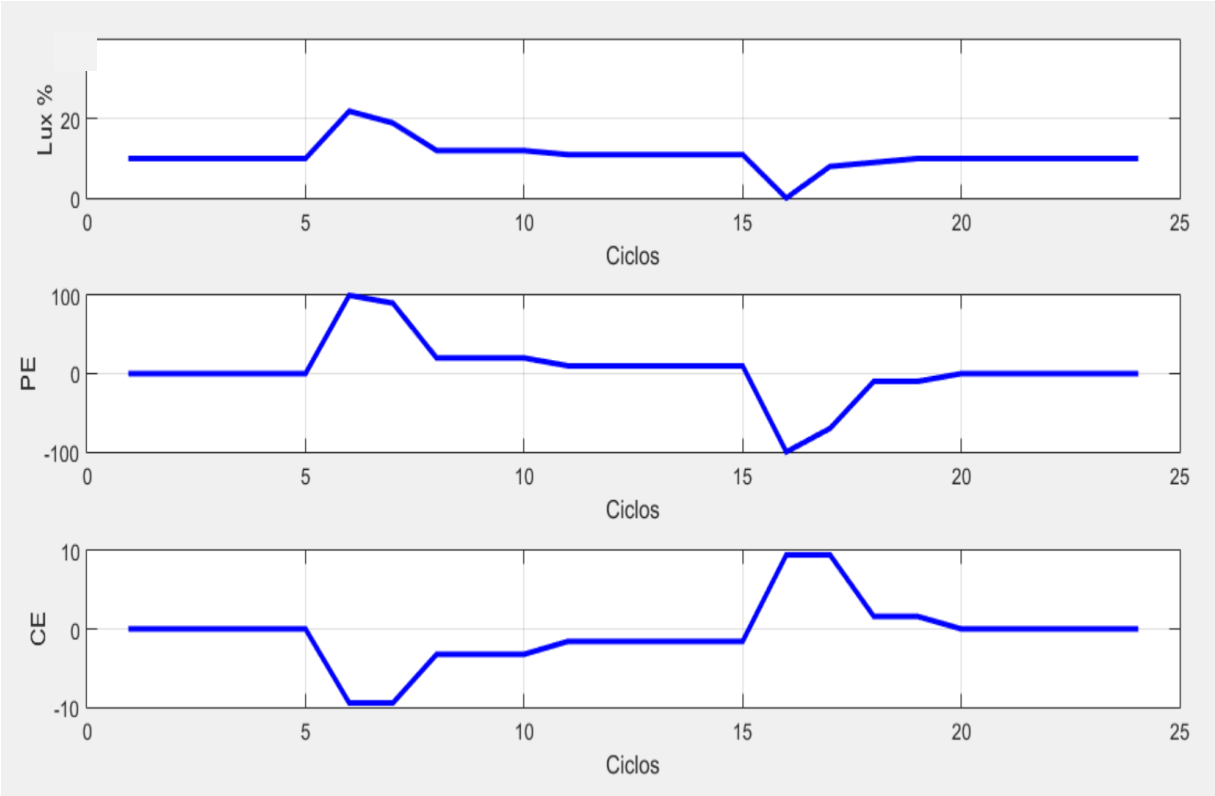
Fonte: o autor.

No teste representado pelo Gráfico 28, foi utilizado o *setpoint* de 10 lux e a lâmpada incandescente foi estabelecida em 10,43 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 10,25 lux e a tensão medida era de 53 volts.

A lâmpada incandescente foi ligada no ciclo 5, o que aumentou a entrada para 21 lux. O protótipo se estabeleceu no ciclo 8, no qual foram aferidos 16 volts pelo multímetro e 10,48 lux pelo luxímetro comercial. Contudo, mesmo com 16 volts, as lâmpadas LEDs se mantiveram desligadas.

No ciclo 15, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 0 lux. O protótipo se estabeleceu no ciclo 18, em que foram aferidos 44 volts pelo multímetro e 9,68 lux pelo luxímetro comercial.

Gráfico 28 – *Setpoint* 10 lux com lâmpada externa em 10,43 lux



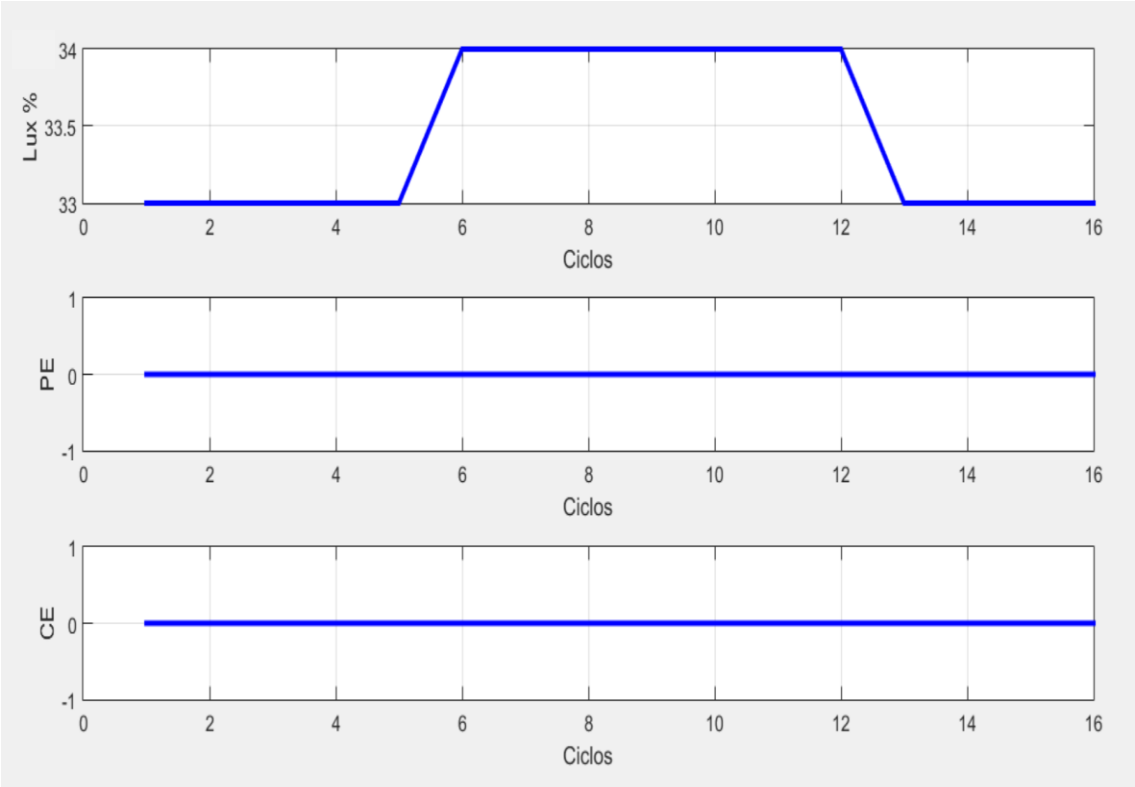
Fonte: o autor.

No teste representado pelo Gráfico 29, o *setpoint* utilizado foi o de 33 lux, o que representa 25 lux a 5 cm do chão. A lâmpada incandescente foi estabelecida em 1,22 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 24,6 lux e a tensão medida era de 132 volts.

A lâmpada incandescente foi ligada no ciclo 5, o que aumentou a entrada para 34 lux. O protótipo se estabeleceu no ciclo 6, no qual foram aferidos 132 volts pelo multímetro e 24,6 lux pelo luxímetro comercial.

No ciclo 12, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 33 lux. O protótipo se estabeleceu no ciclo 13, em que foram aferidos 132 volts pelo multímetro e 24,6 lux pelo luxímetro comercial. Não houve alteração de tensão com a entrada em 1,22 lux.

Gráfico 29 – *Setpoint* 33 lux com lâmpada externa em 1,22 lux



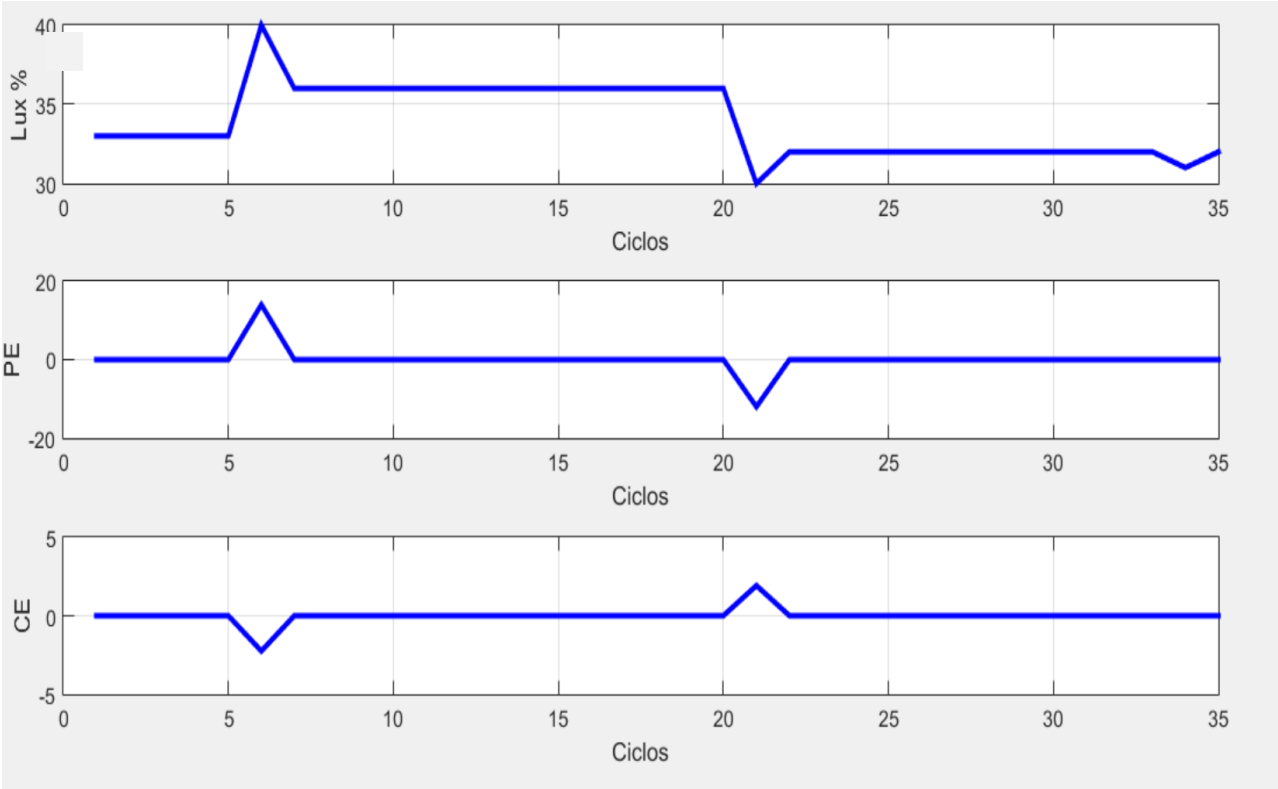
Fonte: o autor.

No teste representado pelo Gráfico 30, foi usado o *setpoint* de 33 lux e a lâmpada incandescente foi estabelecida em 5,53 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 25,3 lux e a tensão medida era de 131 volts.

A lâmpada incandescente foi ligada no ciclo 5, o que aumentou a entrada para 40 lux. O protótipo se estabeleceu no ciclo 8, no qual foram aferidos 120 volts pelo multímetro e 28 lux pelo luxímetro comercial.

No ciclo 20, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 30 lux. O protótipo se estabeleceu no ciclo 23, em que foram aferidos 128 volts pelo multímetro e 24,3 lux pelo luxímetro comercial.

Gráfico 30 – *Setpoint* 33 lux com lâmpada externa em 5,53 lux



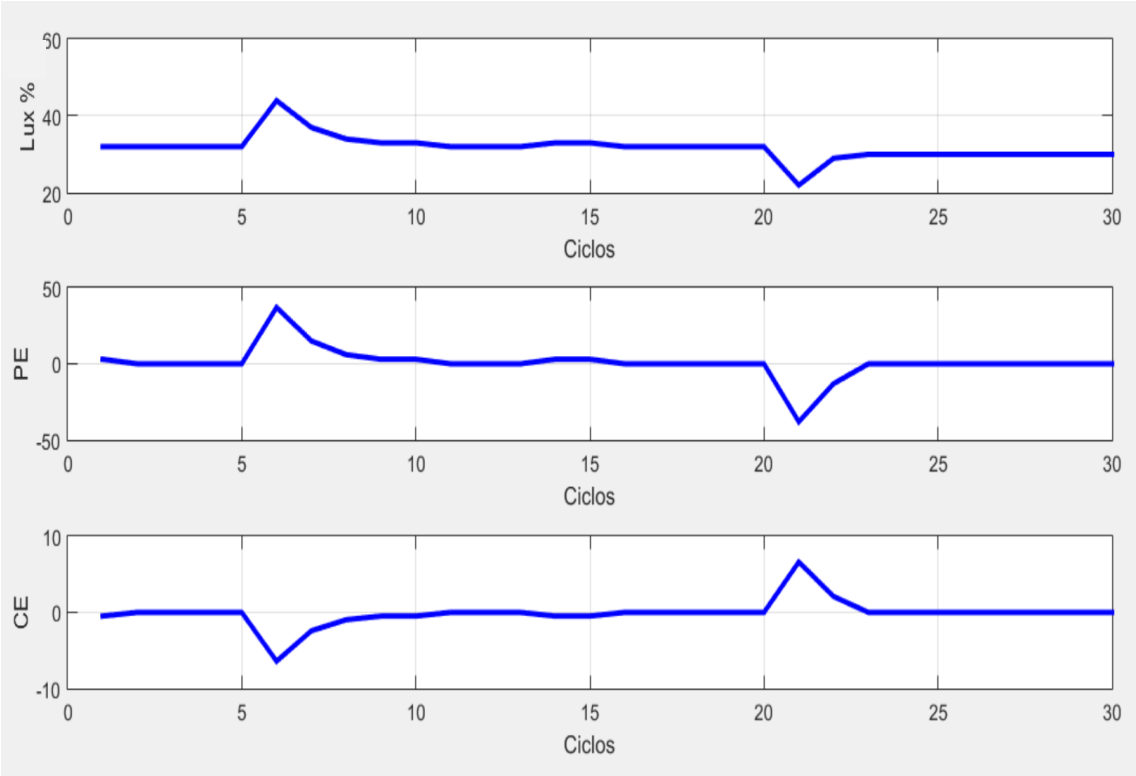
Fonte: o autor.

No teste representado pelo Gráfico 31, foi usado o *setpoint* de 33 lux e a lâmpada incandescente foi estabelecida em 10,33 lux. No momento em que foi ligado o protótipo, o luxímetro comercial indicava 25,2 lux e a tensão medida pelo multímetro era de 129 volts.

A lâmpada incandescente foi ligada no ciclo 5, o que aumentou a entrada para 45 lux. O protótipo se estabeleceu no ciclo 8, em que foram aferidos 98 volts pelo multímetro e 27 lux pelo luxímetro comercial.

No ciclo 20, a lâmpada incandescente foi desligada, o que diminuiu a entrada para 21 lux. O protótipo se estabeleceu no ciclo 23, no qual foram aferidos 122 volts pelo multímetro e 23 lux pelo luxímetro comercial.

Gráfico 31 – *Setpoint* 33 lux com lâmpada externa em 10,33 lux

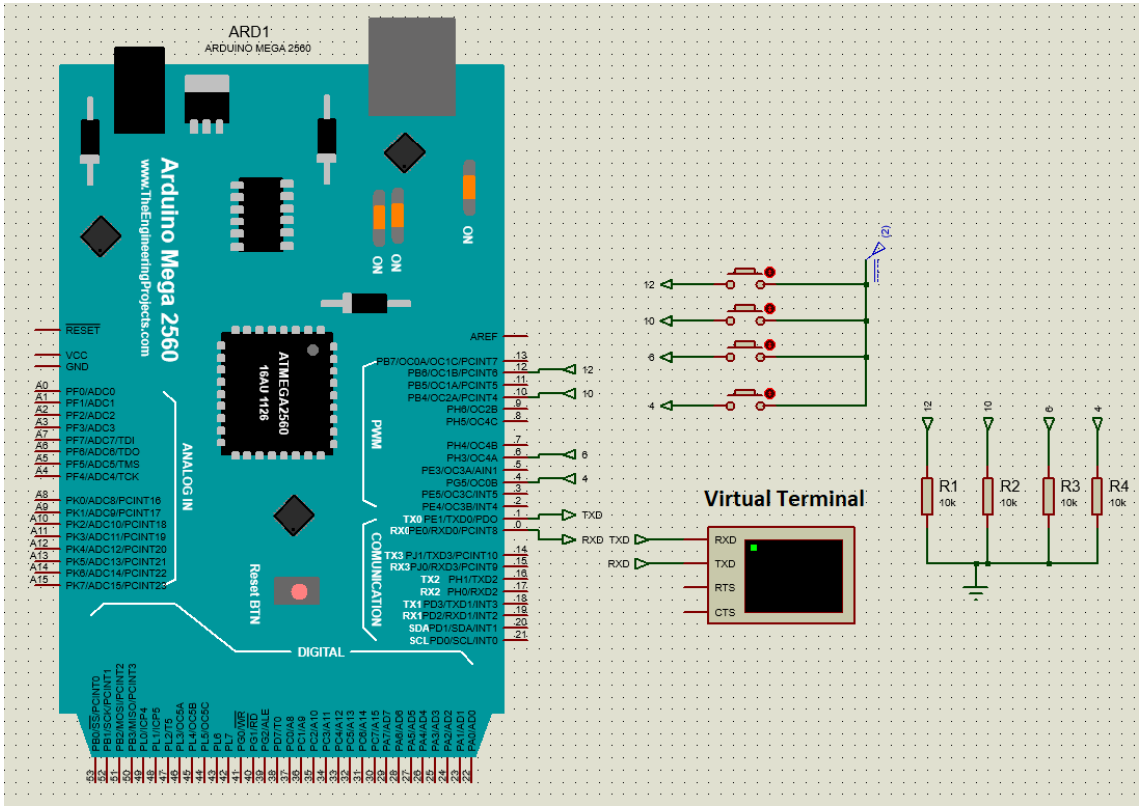


Fonte: o autor.

4.7 Comportamento do protótipo em relação ao controle de tempo

Na Figura 62 a seguir, pode-se observar a configuração das conexões do Arduino Mega para a simulação do controle de tempo em relação aos programas de luz.

Figura 62 – Configuração para simulação de controle de tempo



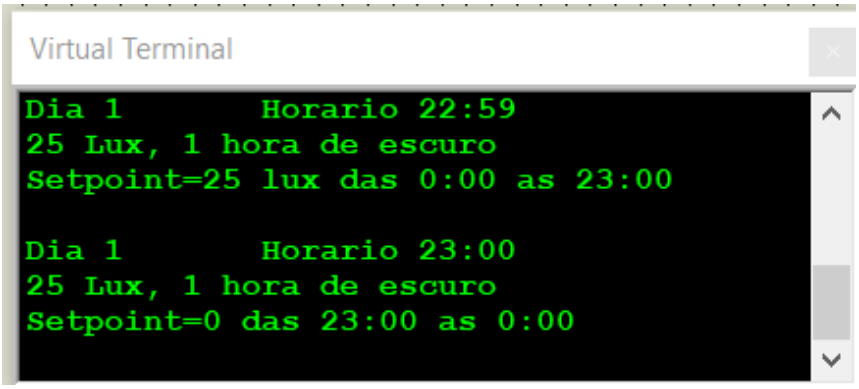
Fonte: o autor.

Nas figuras 63 a 67, foram indicadas as transições de dias e horas do programa luz da Tabela 4. As figuras indicam as informações do *display* do virtual terminal do simulador Proteus. O virtual terminal funciona com comunicação serial; por esse motivo, buscou-se demonstrar o controle por simulações caso essas informações fossem habilitadas no funcionamento do protótipo.

Nas figuras 63 e 64, há a indicação do processo do primeiro até o sétimo dia. Nesse período, segundo a Tabela 4, haverá uma hora de escuridão que será alocada entre 23:00 e 00:00h.

Na Figura 63, há a demonstração da transição das 22:59h para as 23:00h, que indica, respectivamente, os *setpoints* de 25 lux e 0 lux.

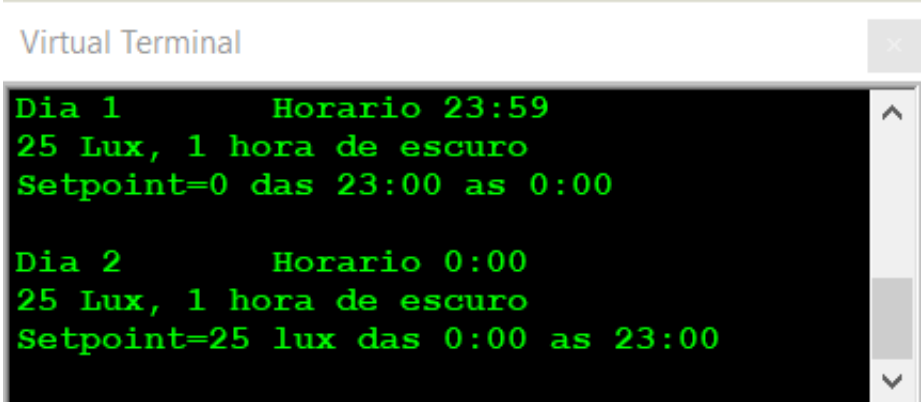
Figura 63 – Dia 1: transição das 22:59h para as 23:00h



Fonte: o autor.

Já na Figura 64, demonstra-se a transição das 23:59h para as 00:00h, que indica, respectivamente, os *setpoints* de 0 lux e 25 lux.

Figura 64 – Dia 1: transição das 23:59h para as 00:00h

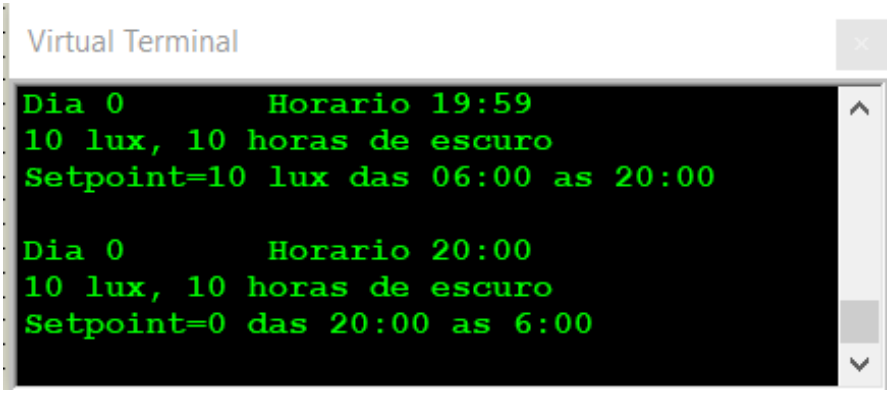


Fonte: o autor.

Conforme os dados da Tabela 4, quando o frango atingir 130 ou 180 gramas, ou depois do sétimo dia, a intensidade luminosa deve ser diminuída de 25 para 10 lux, assim como o período de escuridão deve aumentar para 10 horas. O programa fará a transição por dia automaticamente, porém, por causa do peso do frango, o botão conectado ao pino 12 do Arduino precisará ser acionado.

A Figura 65 mostra o que ocorre quando o botão é acionado: mesmo no dia 0, o *setpoint* e o período de escuridão foram alterados. Outro detalhe é a transição das 19:59h para as 20:00h, que são, respectivamente, os *setpoints* de 10 lux e 0 lux.

Figura 65 – Dia 0: peso de 130 gramas atingido

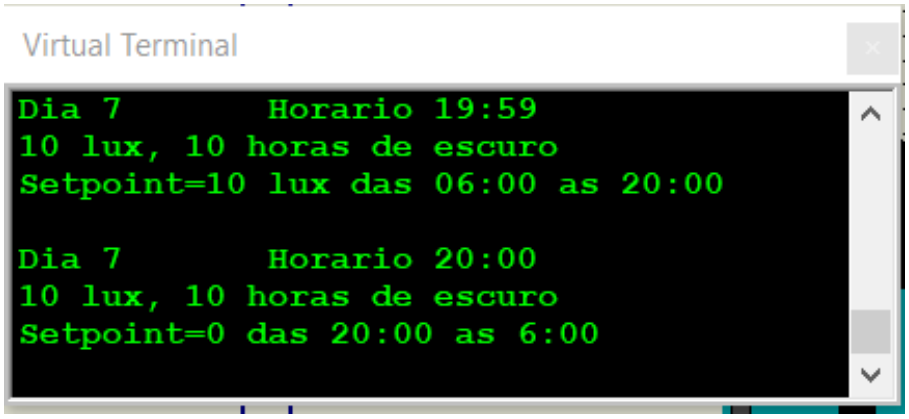


Fonte: o autor.

Nas figuras 66 e 67, há a indicação do processo do sétimo dia até o vigésimo segundo dia. Nesse período, segundo a Tabela 4, haverá 10 horas de escuridão e ela será alocada entre 20:00h e 6:00h.

Na Figura 66, é demonstrada a transição das 19:59h para as 20:00h, que são, respectivamente, os *setpoints* de 10 lux e 0 lux.

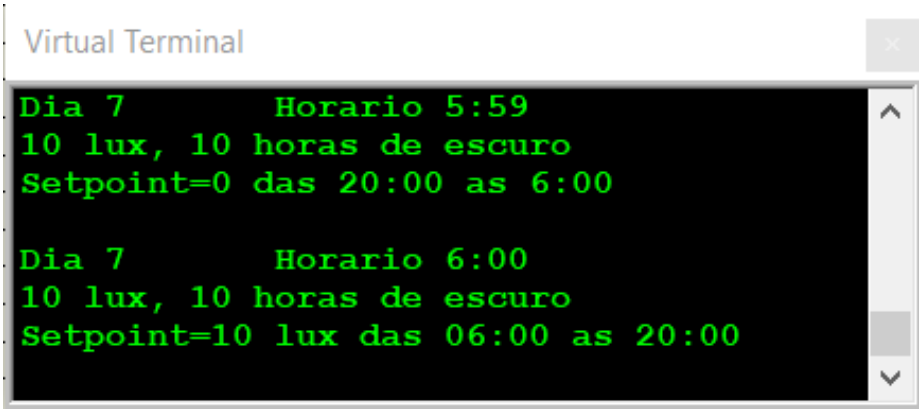
Figura 66 – Dia 7: transições das 19:59h para as 20:00h



Fonte: o autor.

Já na Figura 67, pode-se observar a transição das 5:59h para as 6:00h, que são, respectivamente, os *setpoints* de 0 lux e 10 lux. O mesmo ocorreu quando o botão do pino 12 foi acionado.

Figura 67 – Dia 7: transições das 5:59h para as 6:00h



Fonte: o autor.

Os valores e as transições apresentados pelas figuras 63 a 67 demonstram que a programação pode mudar os valores de *setpoints* e os períodos de escuridão especificados pelas tabelas de programas de luz.

5 CONCLUSÕES

Em um sistema, é muito importante saber que as etapas de entrada, de processamento e de saída estão se comportando de forma adequada, pois qualquer erro em qualquer uma dessas fases pode comprometer todo o sistema.

Em relação aos sinais de entrada, o filtro de média móvel se mostrou uma ferramenta simples para evitar que o sistema leia e envie um sinal corrompido. Porém, durante os experimentos, o módulo GY-302 demonstrou que essa técnica pode ser excluída em seu uso.

Além da média móvel, fazer a comparação entre o GY-302 e o luxímetro se mostrou também de extrema importância, pois sem essa ação os sinais enviados não seriam precisos.

No tratamento de sinais, a principal função do sistema *fuzzy* é tornar o processo de tomadas de decisão mais próximo ao dos humanos. Logo, o FLD se mostrou uma ferramenta muito importante para essa função, pois ele facilita a execução e compreensão do sistema, o que acelerará a escrita da programação principal. A sua compatibilidade com a programação feita no Editor oferece mais credibilidade para a etapa de tratamento de sinais.

Outro ponto importante foi a comunicação serial, visto que possibilitou uma boa interação entre o Arduino e o MATLAB e, consequentemente, um controle melhor do circuito com o Triac.

Na simulação do sinal de saída, o circuito do Triac se mostrou efetivo no controle de tensão enviada à carga, e por consequência, da potência consumida por ela. Outro fator importante é que o gerador de pulsos possibilitou exceder o limite de 90°, ampliando o *range* de controle da tensão.

Destaca-se também a importância de conhecer o comportamento do circuito do Triac, pois é por meio dele que os parâmetros de programação do Arduino serão feitos de forma a se conseguir a melhor interação e funcionalidade com o sistema *fuzzy*.

O sistema de inferência *fuzzy* implementado no protótipo se mostrou efetivo no controle de intensidade luminosa, pois durante a alteração da potência da lâmpada nos testes os *setpoints* foram alcançados, indicados pelos testes da rampa.

Além disso, o protótipo se mostrou eficiente em alcançar os valores de *setpoint* quando uma luz externa que incidiu no sistema ainda assim foi aproveitada. Nos testes, foi demostrado que houve redução do valor de tensão das lâmpadas com a luz

externa, e por consequência, redução de potência elétrica consumida. Por fim, o protótipo mostrou o controle dos programas de luz, assim como o fotoperíodo.

Futuramente, nas etapas subsequentes, esse sistema será aplicado em uma granja real, para assim analisar a sua eficácia e viabilidade. Além disso, propõe-se inserir esse sistema em outras áreas como comércios, prédios públicos, *shoppings* etc.

REFERÊNCIAS

8 ERROS clássicos que você deve evitar ao construir um galinheiro. **Casa das Cercas**, 2017. Disponível em: http://blog.casadascercas.com.br/cuidados-ao-construir-um-galinheiro/. Acesso em: 15 maio 2021.

ALMEIDA, L. A. D. **Dispositivos semicondutores**: tiristores – controle de potência em CC e CA. 13 ed. São Paulo: Erica, 2018.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR ISO/CIE 8995-1**: Iluminação de ambientes de trabalho parte 1 Interior. Rio de Janeiro: ABNT, 2013.

BOYLESTAD, L.; NASHELSKY, L. **Dispositivos eletrônicos e Teoria de Circuitos**. 11 ed. São Paulo: Pearson Education do Brasil, 2013.

CAMPOS, A. A indústria do frango no Brasil. **Monitor**, São Paulo, n. 2, p. 3-18, jun. 2016.

CAMPOS, J. Frango paga salário de metade da população em cidade de 6 mil habitantes. **Gazeta do Povo**, Curitiba, 31 ago. 2017. Disponível em: https://www.gazetadopovo.com.br/agronegocio/frango-paga-salario-de-metade-da-populacao-em-cidade-de-6-mil-habitantes-9z6yx0a5f7cwt8amzhcwanfpp/. Acesso em: 15 maio 2021.

CASTRO, A. P. A. S.; RANCURA, R. L. **Conforto ambiental**: acústico e lumínico. Londrina: Educacional S.A., 2018.

CAVALIN, G.; CERVELIN, S. **Instalações elétricas prediais**. 22 ed. São Paulo: Érica: Saraiva, 2014.

CERTIFICADO do Inmetro é obrigatório para lâmpadas de LED! **G-Light**, 11 maio 2017. Disponível em: https://www.glight.com.br/blog/certificado-inmetro-e-obrigatorio-para-lampadas-de-led/. Acesso em: 10 jan. 2021.

CHEN, G.; PHAM., T. T. **Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems**. Boca Raton: CRC Press, 2001.

COBB-VANTRESS. **Manual de manejo de frango de corte**. Siloam Springs: Coob North America, 2018. Disponível em: https://www.cobb-vantress.com/assets/Cobb-Files/df5655a7e9/Broiler-Guide-2019-POR-WEB.pdf. Acesso em: 16 jan. 2021.

COELHO, M. H. *et al.* Application of fuzzy control in embedded sensing systems for beekeeping monitoring. In: LATIN AMERICAN COMPUTING CONFERENCE, 44., São Paulo, 2018. **Anais** [...]. São Paulo: IEEE, 2018.

CREDER, H. **Instalações elétricas**. 16 ed. Rio de Janeiro: LTC, 2016.

DALL'AZEN, F.; WEISE, A. D. Barreiras técnicas para as exportações: um estudo de caso do abate halal. **Organização sistêmica**, v. 5, n. 3, p. 56-75, jan./jun. 2014.

DE LA VEGA, A. S. **Tutorial sobre sistema de média móvel para fundamentos de processamento digital de sinais**. Niterói: UFF/TCE/TET, 2018. Disponível em: http://www.telecom.uff.br/~delavega/public/DSP/tutorial_SMM_dsp.pdf. Acesso em: 16 jan. 2021.

DIYI0T. **Arduino Uno Tutorial [Pinout]**. Disponível em: https://diyi0t.com/arduino-uno-tutorial/. Acesso em: 4 maio 2021a.

DIYI0T. **I2C Tutorial for Arduino**, **ESP8266 and ESP32**. Disponível em: https://diyi0t.com/i2c-tutorial-for-arduino-and-esp8266/. Acesso em: 4 abr. 2021b.

ECOLUME. **Lâmpada Ecologena**. Disponível em: http://ecolumeweb.com.br/?product=lampada-halogena-a55. Acesso em: 11 ago. 2021.

EFÁCIL. **Multímetro Thompson**. Disponível em: https://www.efacil.com.br/loja/produto/multimetro-thompson-digital-503632/. Acesso em: 20 abr. 2021.

ELETROBRAS. Programa Nacional de Conservação de Energia – Procel. **Relatório de resultados do Procel 2019**: Ano base 2018. Rio de Janeiro, 2019. p. 65. Disponível em: http://www.procelinfo.com.br/resultadosprocel2019/Procel_rel_2019_web.pdf. Acesso em: 15 jan. 2022.

EMBRAPA. **Embrapa em números**. Brasília, 2019. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/1121938/embrapa-em-numeros. Acesso em: 16 jan. 2022.

EMBRAPA. **Embrapa suínos e aves**: Central de inteligência de aves e suínos – estatísticas. Brasília, 5 maio 2021. Disponível em: https://www.embrapa.br/suinos-e-aves/cias/estatisticas/frangos/mundo. Acesso em: 1 fev. 2021.

ESPINDOLA, C. E. Trajetórias do progresso técnico na cadeia produtiva de carne de frango do Brasil. **Geosul**, Florianópolis, v. 27, p. 88-113, jan. 2012.

EXTENSÃO da Tarifa Rural Noturna traz alento à produção no Paraná. **Boletim informativo – Sistema FAEP**, Curitiba, ano 33, n. 1.460, p. 6-9, dez. 2018. Disponível em: https://www.sistemafaep.org.br/wp-content/uploads/2018/12/BI_1460-Ajustado-18_12.pdf. Acesso em: 16 jan. 2022.

FERON, B. T.; FRICK, G. Implementação de filtragem de média móvel para cálculo de temperatura com alarme na plataforma Arduino Uno. *In*: SEMANA DO CONHECIMENTO, 3., Passo Fundo, 2015. **Anais** [...]. Passo Fundo, 2015. p. 54-67.

FILIPEFLOP. **Sensor de Luz BH1750FVI Lux**. Disponível em: https://www.filipeflop.com/produto/sensor-de-luz-bh1750fvi-lux/. Acesso em: 12 fev. 2021.

FOXLUX. **Iluminação**: lâmpada fluorescente compacta – tipo espiral. Disponível em: https://www.foxlux.com.br/produto/lampada-fluorescente-compacta-tipo-espiral/. Acesso em: 5 jan. 2021.

FUTEMA, F. Crise já prejudica desempregados; cai em 41% ofertas de emprego. **Folha de S.Paulo**, São Paulo, 31 maio 2001. Editoria Mercado. Disponível em: https://www1.folha.uol.com.br/folha/dinheiro/ult91u23065.shtml. Acesso em: 10 maio 2021.

GALAXYLED. **Produtos**. 2021. Disponível em: http://galaxyled.com.br/produto_detalhe/bulbo-a60-dimerizavel. Acesso em: 15 maio 2021.

GERHARDT, T. E.; SILVEIRA, D. T. (org.). **Métodos de pesquisa**. Porto Alegre: Editora da UFRGS, 2009. Disponível em: https://lume.ufrgs.br/bitstream/handle/10183/52806/000728684.pdf?sequence=1&isAllowed=y. Acesso em: 16 jan. 2021.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2002.

GUEDES, F. A. C.; REIS, R. A.; JOUCOSKI, E. Contribuições da educação ambiental para o desenvolvimento territorial sustentável. *In*: SIMPÓSIO BRASILEIRO DE DESENVOLVIMENTO TERRITORIAL SUSTENTÁVEL, 1., Matinhos, 2015. **Anais** [...]. Matinhos: UFPR, 2015. p. 82-87.

INSTRUBRAS. **ITR-3000**: luxímetro digital para LED com certificado de calibração. 2021. Disponível em: https://www.instrubras.com.br/seguranca-do-trabalho/luximetros/luximetro-digital-para-led-com-certificado-de-calibracao. Acesso em: 22 mar. 2021.

KASABOV, N. K. **Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering**. 2. ed. Massachusetts: MIT Press, 1998.

LAGO, B. Ameaça de colapso. **Revista Negócios PE**, Recife, 18 jan. 2020. Disponível em: http://www.revistanegociospe.com.br/materia/Ameaca-de-colapso?search=Encontre+o+conte%C3%BAdo. Acesso em: 5 jan. 2021.

LENHARD, J. C. Aviário Modernos é inaugurado em Estrela. **Independente**, Lajeado, 24 out. 2017. Disponível em: https://independente.com.br/aviario-moderno-e-inaugurado-em-estrela. Acesso em: 1 fev. 2021.

LITTELL, J. H.; CORCORAN, J.; PILLAI, V. **Systematic reviews and metaanalysis**. New York: Oxford University Press, 2008.

LOPES, M. C.; TAQUES, F. H. O desafio da energia sutentável no Brasil. **Revista Cadernos de Economia**, Chapecó, v. 20, n. 36, p. 71-96, 2016.

MALVINO, A.; BATES, D. **Eletrônica**. 8. ed. Porto Alegre: AMGH, 2016. v. 1.

MAMEDE, J. F. **Instalações elétricas industriais**. 9. ed. Rio de Janeiro: LTC, 2017.

MAPELLI, Y. R.; LARANJA, A. C.; ALVAREZ, C. E. de. Avaliação de desempenho entre as tipologias de aberturas zenital e lateral no quesito iluminação natural de ambientes internos. **Cadernos PROARQ**, Rio de Janeiro, n. 31, p. 83-99, dez. 2018.

MARKUS, O. **Circuitos elétricos**: corrente contínua e corrente alternada. 9. ed. São Paulo: Érica, 2011.

NASCIMENTO, R. S. do; ALVES, G. M. Fontes alternativas e renováveis de energia no Brasil: métodos e benefícios ambientais. **Revista Univap**, São José dos Campos, v. 22, n. 40, edição especial, 2016. Disponível em: http://revista.univap.br/index.php/revistaunivap/article/view/713/640. Acesso em: 16 jan. 2022.

OLIVEIRA, I. O que foi o apagão de 2001? A conta de luz subiu? Pode acontecer de novo? **UOL**, 10 jun. 2021. Seção Economia. Disponível em: https://economia.uol.com.br/faq/o-que-foi-o-apagao-de-2001-risco-racionamento-energia-eletrica.htm. Acesso em: 15 jun. 2021.

OLIVEIRA, R. F. **Inteligência artificial**. Londrina: Educacional S. A., 2018.

OSAKI, C. M. N. **Luminotécnica**. Londrina: Educacional S. A., 2018.

PATHAK, H. *et al.* Visible light communication, networking, and sensing: a survey, potential and challenges. **IEEE Communications**: Surveys & Tutorials, v. 17, n. 4, 4th quarter, p. 2.047-2.077, 2015.

PECHE, G. A. Pollo de carne: su visión del entorno. **AviNews**, Barcelona, n. 23, p. 7-15, feb. 2017. Disponível em: https://issuu.com/avinews/docs/avinews_feb17. Acesso em: 16 jan. 2022.

PEREIRA, A. A. K. Tutorial utilizando o novo Dimmer Shield pelos botões. **Laboratórido de Garagem**, 27 jun. 2016. Disponível em: https://labdegaragem.com/profiles/blogs/tutorial-para-utilizar-o-novo-dimmer-shield?xg_source=activity. Acesso em: 2 mar. 2021.

PÉREZ, A. Conhecendo o cerne do microcontrolador Atmega328p Arduino Uno. (MIC165). **Instituto NCB**, 19 abr. 2018. Disponível em: https://www.newtoncbraga.com.br/index.php/microcontrolador/138-atmel/14863-conhecendo-o-cerne-do-microcontrolador-atmega328p-arduino-uno-mic165. Acesso em: 10 jan. 2021.

RASHID, H. **Eletrônica de potência**: dispositivos, circuitos e aplicações. 4. ed. São Paulo: Pearson Education do Brasil, 2014.

REUTERS. Exportação de carne de frango do Brasil cai 9,5% em março, diz ABPA. **G1**, São Paulo, 8 abr. 2019. Edioria Agronegócios. Disponível em: https://g1.globo.com/economia/agronegocios/noticia/2019/04/08/exportacao-de-carne-de-frango-do-brasil-cai-95percent-em-marco-diz-abpa.ghtml. Acesso em: 1 fev. 2021.

ROBOCORE. **Placa BlackBoard UNO R3**. Disponível em: https://www.robocore.net/placa-robocore/arduino-blackboard Acesso em: 5 jan. 2021.

ROMANZOTI, N. É assim que os pássaros veem o mundo em comparação aos seres humanos. **Hypescience**, 16 out. 2019. Disponível em: https://hypescience.com/e-assim-que-os-passaros-veem-o-mundo-em-comparacao-aos-seres-humanos/. Acesso em: 5 jan. 2021.

SANDY, A. M. *et al.* Eficiência energética: mercado de créditos de carbono. *In*: SILVEIRA, J. H. P. (org.). **Sustentabilidade e responsabilidade social**. Belo Horizonte: Poisson, 2018. v. 8. p. 42-53.

SCHWEAN-LARDNER, K.; CLASSEN, H. **Programa de luz para frangos de corte**. campinas: aviagen, 2010. 46 p.

SILVA, E. L. da; MENEZES, E. M. **Metodologia da pesquisa e elaboração de dissertação**. 3. ed. Florianópolis: Laboratório de Ensino a Distância da UFSC, 2001.

STEVAN JUNIOR, S. L.; SILVA, R. A. **Automação e instrumentação Industrial com Arduino**: teoria e projetos. São Paulo: Érica, 2015.

TANSCHEIT, R. **Sistemas *fuzzy***. Rio de Janeiro: Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, 2004.

TOLMASQUIM, M. As origens da crise energética brasileira. **Ambiente & sociedade**, Campinas, p. 179-183, jun. 2000. Disponível em: http://www.scielo.br/pdf/asoc/n6-7/20435.pdf. Acesso em: 15 jan. 2021.

VERCELLINO, R. A. **Efeito de diferentes sistemas de vedação de aviários no comportamento e bem-estar de frango de corte**. 2012. 154 f. Dissertação (Mestrado em Engenharia Agrícola) – Faculdade de Engenharia Agrícola, Universidade Estadual de Campinas, Campinas, 2012.

VIANA, A. N. C. *et al.* **Eficiência energética**: fundamentos e aplicações. Campinas: Elektro/Unifei/Excen/Fupai, 2012.

WATANABE, G. E. **O desenvolvimento da avicultura no Brasil e as tendências para os próximos anos**. 2016. 49 f. Trabalho de Conclusão do Curso (Especialização em Gestão do Agronegócio) – Departamento de Economia Rural e Extensão, Setor de Ciências Agrárias, Universidade Federal do Paraná, Curitiba, 2016. Disponível em: https://acervodigital.ufpr.br/bitstream/handle/1884/50816/R%20-%20E%20-%20GUSTAVO%20EIJI%20WATANABE.pdf?sequence=1. Acesso em: 16 jan. 2022.

WHAT is Arduino? **Arduino**, 5 fev. 2020. Disponível em: https://www.arduino.cc/en/Guide/Introduction?setlang=en. Acesso em: 10 jan. 2021.

ZEN, S. de *et al.* Evolução da avicultura no Brasil. **Informativo Cepea**, Piracicaba, v. 1, n. 1, out./dez. 2014. Disponível em: cepea.esalq.usp.br/upload/revista/pdf/0969140001468869743.pdf. Acesso em: 16 jan. 2022.

APÊNDICE A – IMPLEMENTAÇÃO COMPUTACIONAL DO ALGORITMO DO CONTROLE DO MÓDULO GY-302 E DO PROGRAMA DE LUZ

```
Const int ledPin =  LED_BUILTIN;
int ledState = LOW;
int dia=8;
int hora=14;
int minuto=53;
int C=0;
int BT=10;
int BT5=9;
int BT3=8;
int BT2=7;
int ALT=0;
int Set_point=0;
float lux3;
unsigned long previousMillis = 0;
unsigned long previousMillis_h = 0;
unsigned long previousMillis_m = 0;
int buttonState_12 = 0;
int buttonState_5 = 0;
int buttonState_3 = 0;
int buttonState_2 = 0;
const long interval = 86400000;
const long interval_h = 3600000;
const long interval_m =60000;
#include <Wire.h>
#include <BH1750.h>
#define N 10
int foto_res = 0;
int loop_read[N];
// programa do luximetro
BH1750 lightMeter(0x23);
int media;
int cont=0;
const int analogInPin = A0;
const int analogOutPin = 9;
int sensorValue = 0;
int outputValue = 0;
void setup(){
```

```
  pinMode(BT, INPUT);
  pinMode(BT5, INPUT);
  pinMode(BT3, INPUT);
  pinMode(BT2, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
  Wire.begin();
  if (lightMeter.begin(BH1750::CONTINUOUS_HIGH_RES_MODE)) {
  }
  else {
  }
}
void loop()
{
buttonState_12=digitalRead(BT);
buttonState_5=digitalRead(BT5);
buttonState_3=digitalRead(BT3);
buttonState_2=digitalRead(BT2);
if (buttonState_12==1)
{
ALT=1;
}
if (buttonState_5==1)
{
ALT=5;
dia=5;
}
if (buttonState_3==1)
{
ALT=3;
dia=3;
}
if (buttonState_2==1)
{
ALT=2;
dia=2;
}
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis_m >= interval_m)
    {
```

```
      minuto=minuto+1;
      if(minuto>=60)// para zerar o minuto quando chegar a 60 e acrescentar em hora
      {
        minuto=0;
        hora=hora+1;
       }
      if(hora>=24)
      {
        if(ALT!=5&&ALT!=3&&ALT!=2)
          {
          dia=dia+1;
          }
        if(ALT==5||ALT==3||ALT==2)
          {
          dia=dia-1;
          }
        hora=0;
       }
      previousMillis_m = currentMillis;
if (dia==0)
{
 // Serial.println("    ");
      //Serial.println("25 Lux, 0 hora de escuro");
 }
 if (dia>=1&&dia<7&&ALT==0)
 {
      //Serial.println("    ");
      //Serial.println("25 Lux, 1 hora de escuro");
      if (hora>=23)
      {
        //Serial.println("Setpoint=0 das 23:00 as 0:00");
        Set_point=1;
       }
        if (hora>=0&&hora<23)
        {
          //Serial.println("Setpoint=25 lux das 0:00 as 23:00");
           Set_point=33;
          }
  }
```

```
 if((dia>=7||ALT==1)&&dia<22)
 {
   ALT=1;
   //Serial.println(" ");
   //Serial.println("10 lux, 10 horas de escuro");
if (hora>=20)
    {
      //Serial.println("Setpoint=0 das 20:00 as 6:00");
      Set_point=1;
      }
      if (hora>=0 && hora<6)
    {
      //Serial.println("Setpoint=0 das 20:00 as 6:00");
      Set_point=1;
      }
      if (hora>=6 && hora<20)
      {
        //Serial.println("Setpoint=10 lux das 06:00 as 20:00");
        Set_point=10;
        }
   }
 if(dia>=22&&dia<28)
 {
   ALT=1;
   //Serial.println(" ");
   //Serial.println("10 lux");
   //Serial.println("9 horas de escuro");
if (hora>=21)
    {
      //Serial.println("Setpoint=0 das 21:00 as 6:00");
      Set_point=1;
      }
      if (hora>=0 && hora<6)
    {
      //Serial.println("Setpoint=0 das 21:00 as 6:00");
      Set_point=1;
      }
      if (hora>=6 && hora<21)
      {
        //Serial.println("Setpoint=10 lux das 06:00 as 21:00");
```

```
        Set_point=10;
        }
   }
if(dia>=28&&dia<35)
  {
   ALT=1;
   //Serial.println(" ");
   //Serial.println("10 lux, 8 horas de escuro");
  if (hora>=22)
     {
      //Serial.println("Setpoint=0 das 22:00 as 6:00");
      Set_point=1;
      }
      if (hora>=0 && hora<6)
     {
      // Serial.println("Setpoint=0 das 22:00 as 6:00");
      Set_point=1;
      }
      if (hora>=6 && hora<22)
      {
        //Serial.println("Setpoint=10 lux das 06:00 as 22:00");
        Set_point=10;
        }
   }
if(dia>=35 && dia<42)
  {
   ALT=1;
   //Serial.println(" ");
   //Serial.println("10 lux, 7 horas de escuro");
 if (hora>=23)
     {
      //Serial.println("Setpoint=0 das 23:00 as 6:00");
      Set_point=1;
      }
      if (hora>=0 && hora<6)
     {
      //Serial.println("Setpoint=0 das 23:00 as 6:00");
      Set_point=1;
      }
```

```
        if (hora>=6 && hora<23)
        {
          //Serial.println("Setpoint=10 lux das 06:00 as 23:00");
          Set_point=10;
          }
   }
if(dia>=42 && dia<49)
  {
    ALT=1;
    //Serial.println(" ");
    //Serial.println("10 lux, 6 horas de escuro");
        if (hora>=0 && hora<6)
       {
        //Serial.println("Setpoint=0 das 0:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 0:00");
          Set_point=10;
          }
   }
if(ALT==5&&dia==5)
  {
    //Serial.println(" ");
    //Serial.println("10 lux, 5 horas de escuro");
  if (hora>=1 && hora<6)
     {
        //Serial.println("Setpoint=0 das 1:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6 || hora<1)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 1:00");
          Set_point=10;
          }
   }
if(ALT==5&&dia==4)
  {
```

```
    //Serial.println(" ");
    //Serial.println("10 lux, 4 horas de escuro");
    if (hora>=2 && hora<6)
      {
        //Serial.println("Setpoint=0 das 2:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6 || hora<2)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 2:00");
          Set_point=10;
          }
   }
if((ALT==5||ALT==3)&&dia==3)
  {
    //Serial.println(" ");
   // Serial.println("10 lux, 3 horas de escuro");
    if (hora>=3 && hora<6)
      {
        //Serial.println("Setpoint=0 das 3:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6 || hora<3)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 3:00");
          Set_point=10;
          }
   }
if((ALT==5||ALT==3||ALT==2)&&dia==2)
  {
    //Serial.println(" ");
    //Serial.println("10 lux, 2 horas de escuro");
    if (hora>=4 && hora<6)
      {
        //Serial.println("Setpoint=0 das 4:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6 || hora<4)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 4:00");
```

```
          Set_point=10;
          }
   }
if((ALT==5||ALT==3||ALT==2)&&dia==1)
   {
    //Serial.println(" ");
    //Serial.println("10 lux, 2 horas de escuro");
    if (hora>=5 && hora<6)
      {
        //Serial.println("Setpoint=0 das 5:00 as 6:00");
        Set_point=1;
        }
        if (hora>=6 || hora<5)
        {
          //Serial.println("Setpoint=10 lux das 6:00 as 5:00");
          Set_point=10;
          }
   }
 }
if(Serial.available()>0)
 {
       char txt=Serial.read();// converte a leitura por txt
       txt=txt;
     if (txt=='D'||txt=='d')
     {
     Serial.println("   ");
     Serial.print("Dia ");
     Serial.print(dia);
     Serial.print("       Horario ");
     Serial.print(hora);
     Serial.print(":");
    if (minuto<10)//para colocar o "0" a esquerda de números menores que 10
    {
      Serial.print("0");
    }
          Serial.println(minuto);
     Serial.print("Setpoint");
     Serial.print("     ");
     Serial.println(Set_point);
      }
```

```
   }
  {// inico do BH1750*********************
  uint16_t lux = lightMeter.readLightLevel();
  //Serial.print("Light: ");
  float read_res = lux;
for (int i = N-1; i > 0; i--)
{
loop_read[i] = loop_read[i-1];
}
loop_read[0] = read_res;
long soma = 0;
for (int i = 0; i < N; i++)
{
soma = soma + loop_read[i];
}
media = soma/N;
int lux2=0.7777*lux-0.4562;//Equação obtida pelo Excel//lux sem média móvel
  if (Set_point==0 || Set_point==1)
  {
    lux3=100;
  }
  else
  {
  lux3=(lux2*100/Set_point)-100; //SETPOINT  em 25 Lux
  }
 //Serial.println(lux_2);
//Serial.print(" ");
  //Serial.println( lux3 );
  //Serial.println( lux2 );
  //Serial.println( media );
 // Serial.println(" lx");
  delay(200);
  if(Serial.available()>0)
  {
        char txt=Serial.read();// converte a leitura por txt
        txt=txt;
        if (txt=='C' || txt=='c')// exclusivos para testes
        {
        Serial.println("***********************************************");
        Serial.println("Sinal Original");
```

```
          Serial.println(lux);
          Serial.println(" ");
          Serial.println("Sinal Filtrado");
          Serial.println(media);
          Serial.println(" ");
          Serial.println("Sinal da equação");
          Serial.println(lux2);
          Serial.println(" ");
        }
if (txt=='B' || txt=='b')// exclusivo para testes
        {
          Serial.println(lux2);
               }
if (txt=='D' || txt=='d')// exclusivos para testes
        {
        Serial.println("***********************************************");
        Serial.println("   ");
        Serial.print("Dia ");
        Serial.print(dia);
        Serial.print("   ");
        Serial.print(hora);
        Serial.print(":");
        if (minuto<10)
        {
        Serial.print("0");
        }
        Serial.println(minuto);
        Serial.print("Setpoint");
        Serial.print("   ");
        Serial.println(Set_point);
        }
              if (txt=='A' || txt=='a')
        {
             if (lux3>=100)
        {
          lux3=100;
        }
        if (lux3<=-100)
        {
          lux3=-100;
```

```
        }
// mudança
if (Set_point==33)
{
        if (lux3>=-5 && lux3<=5)
        {
          lux3=0;
          }
}
if (Set_point==10)
{
if (lux3>=-9 && lux3<=9)
        {
          lux3=0;
          }
}
 //Serial.println( "**********************************" );
          //Serial.println( "valor original" );
        //Serial.println( lux);//valor original
        //Serial.println( " " );
         //Serial.println( "correcaum e media" );
        //Serial.println( media ); //valor map
        //Serial.println( " " );
          //Serial.println( "valor da equação" );
        //Serial.println( lux2 ); //valor map
        //Serial.println( " " );
        //Serial.println( "porcentagem para 25 Lux");
       Serial.println( lux3 ); //envia porcentagem do erro para 25 Lux
       //Serial.println( " " );
       //Serial.println( "porcentagem para 10 Lux");
       //Serial.println( lux4 ); //envia porcentagem do erro para 25 Lux
   }
}
  }//final do BH1750*************************************************
   }//Fechamento do loop
```

APÊNDICE B – IMPLEMENTAÇÃO COMPUTACIONAL DO ALGORITMO DO CONTROLE DO DIMMER SHIELD

```
int PIN_3 = 3;
volatile float POT =100;
volatile float POT2=0;
volatile float tv=0;
int BOT_A0 = A0, BOT_A3 = A3;
void setup()
{
  Serial.begin(9600);
pinMode(PIN_3, OUTPUT);
attachInterrupt(0, Passagem_por_ZERO, RISING);
pinMode(BOT_A0, INPUT);
pinMode(BOT_A3, INPUT);
digitalWrite(BOT_A0, LOW);
digitalWrite(BOT_A3, LOW);
}
void loop()
{
if(Serial.available()>0)
  {
    POT2=Serial.parseFloat();
    if (POT2>15)//limitação de erro
      {
      POT2=0;
      }
      if (POT2<-15)
      {
      POT2=0;
      }
      POT=POT+POT2;
    if (POT>=100)//limitação de potência
      {
      POT=100;
      }
     if (POT<=0)//limitação de potência
      {
      POT=0;
      }
//Serial.println(POT);
```

```
volatile float angulo=(83.33 * (100 - POT))*180/8333;
//Serial.println(angulo);
 volatile float POTtime_2 = (83.33 * (100 - POT))/1000;
volatile float Vef=296.98*sqrt(0.5-(angulo/360)+(sin(2*angulo)/(4*PI)));
//volatile float seno=sqrt(0.5-(angulo/360)+(sin(2*angulo)/(4*PI)));
/*Serial.println("**********************************");
Serial.println("Tempo para disparo");
Serial.println(POTtime_2);
Serial.println("   ");
Serial.println("Angulo");
Serial.println(angulo);
Serial.println("   ");
Serial.println("Tensao eficaz");
Serial.println(Vef);
Serial.println("   ");
Serial.println("   ");*/
Serial.println(angulo);
//Serial.println(POT);
  }
}//******************************** FIM DO LOOP
//**********************************INTERRUPÇÃO
void Passagem_por_ZERO()
{
float POTtime = (83.33 * (100 - POT));
if (POTtime <= 820)
{
digitalWrite(PIN_3, HIGH);
}
else if (POTtime >= 8000)
{
digitalWrite(PIN_3, LOW);
}
else if ((POTtime > 820) && (POTtime < 8000))
{
delayMicroseconds(POTtime);
digitalWrite(PIN_3, HIGH);
delayMicroseconds(8);
digitalWrite(PIN_3, LOW);
}
}// **********************************FIM da INTERRUPÇÃO
```

APÊNDICE C – IMPLEMENTAÇÃO COMPUTACIONAL DO ALGORITMO DA INFERÊNCIA FUZZY

```
clc, clear all, close all,
delete(instrfind({'Port'},{'COM3'}))
delete(instrfind({'Port'},{'COM4'}))
ard=serial('COM4','BAUD', 9600);%UNO
ard2=serial('COM3','BAUD', 9600);%MEGA
fopen(ard);
fopen(ard2);
pause(1)
fprintf(ard,'0');%potência do triac
 stp=25;
 disp ('Inicio do programa');
%%power=70;
pause(4);
passo = 0.01;
ind_s=0;
i=1;
j=1;
z=1;
w=1;
%data=1;
%data_vo=1;
data_stp=1;
 while ind_s<=150;
 format bank;
%% inserção via teclado
%ind_s=input('Informe o erro da leitura: ', 's');
%e0=str2double(ind_s);
 %% comunicação Arduino
 fprintf(ard2,'A');
%pause(1);
%fprintf(ard2,'A');%% para dar tempo do arduino enviar o valor correto (2
%%ciclos)
%disp('valor enviado do arduino:');
ind_s=fscanf(ard2,'%f');
disp('Valor corrigido');
disp(ind_s);
data(i)=ind_s;
```

```
subplot (3,1,2), plot(data,'b','LineWidth',3);
set(gca,'FontSize',12), %legend('LB','LM','LA')
xlabel('Ciclos'), ylabel('PE')
grid on;
i=i+1;
disp('erro');
%e0=(ind_s*100/stp)-100;%% fazer o calculo do erro
e0=ind_s;% erro já enviado pelo arduino
if e0>100
   e0=100;
end
if e0<-100
   e0=-100;
end
 %e0=30;
%% Erro de posição
e= -100:passo:100;
EB= trapmf(e,[-100 -100 -50 0]);
EM=trimf(e,[-50 0 50]);
EA=trapmf(e,[0 50 100 100]);
%% figura fuzzy
%subplot (3,1,1), plot(e,EB,'b',e,EM,'y',e,EA,'r','LineWidth',3)
%set(gca,'FontSize',12), legend('LB','LM','LA')
%xlabel('ERRO %'), ylabel('\mu(E)')
%grid on;
%axis([-50 50 0 1.1])%define o tamanho dos eixos x e y
%% Ação de controle
passo=0.1;
v=-15:passo:15;
SB= trapmf(v,[-15 -15 -9 0]);
SM=trimf(v,[-9 0 9]);
SA=trapmf(v,[0 9 15 15]);
%% figura fuzzy
%subplot (3,1,2), plot(v,SB,'b',v,SM,'y',v,SA,'r','LineWidth',3)
%set(gca,'FontSize',12), legend('TB','TM','TA')
%xlabel('Correção %'), ylabel('\mu(S)')
%axis([-15 15 0 1.1])
%grid on;
%% Leitura do erro
n=find(e==e0);
```

```
%subplot(3,1,1), hold on, plot(e0,EB(n),'*',e0,EM(n),'*',e0,EA(n),'*','LineWidth',3), hold off
%% Fuzzificar e Inferência de Mamdani
 %%if ind_s <= -11 ||  ind_s>=11
B1=min(SB,EA(n));
B2=min(SM,EM(n));
B3=min(SA,EB(n));
B=max(B1,max(B2,B3));
%% figura fuzzy
%subplot(3,1,3),plot(v,B,'LineWidth',3)
%set(gca,'FontSize',12), legend('V')
%axis([-15 15 0 1.1])
%grid on;
%% desfuzzificação centro de área
vo=defuzz(v,B,'centroid');
%% figura fuzzy
%hold on, plot(vo*ones(1,3),[0 0.5 2], 'r', 'LineWidth',3)
%disp('Centro de área:');
%title(vo);
%xlabel('Saída Correção'), ylabel('\mu(S)')
%% Saída do sinal
if e0==0
    vo=0;
end
disp(vo);
if vo<=2
fprintf(ard,'%f',vo);%para enviar o numero float para o arduino uno
end
if vo>2
fprintf(ard,'%d',vo);%para enviar o numero int para o arduino uno
end
 data_vo(j)=vo;
subplot (3,1,3), plot(data_vo,'b', 'LineWidth',3);
set(gca,'FontSize',12), %legend('LB','LM','LA')
xlabel('Ciclos'), ylabel('CE')
grid on;
 j=j+1;
 fprintf(ard2,'B');
%pause(1);
%fprintf(ard2,'A');%% para dar tempo do arduino enviar o valor correto (2
%%ciclos)
```

```
%disp('valor enviado do arduino:');
ind_x=fscanf(ard2,'%f');
                                    %ind_x=10;
disp('Valor Lux');
disp(ind_x);
data_x(z)=ind_x;
subplot (3,1,1), plot(data_x,'b', 'LineWidth',3);
set(gca,'FontSize',12), %legend('LB','LM','LA')
xlabel('Ciclos'), ylabel('Lux %')
grid on;
 z=z+1;
Vef_x=fscanf(ard,'%f');
disp('power');
disp(Vef_x);
%data_w(w)=Vef_x;
%subplot (4,1,4), plot(data_w,'b', 'LineWidth',3);
%set(gca,'FontSize',12), %legend('LB','LM','LA')
%xlabel('Ciclos'), ylabel('Vef')
%grid on;
 %w=w+1;
pause(3);
%% Envio de resultados para o arduino através da porta serial
 end
 delete(instrfind({'Port'},{'COM3'}))
delete(instrfind({'Port'},{'COM4'}))
```