INTELIGÊNCIA ARTIFICIAL APLICADA A SISTEMAS SUPERVISÓRIOS DE AUTOMAÇÃO RESIDENCIAL COM INTERNET DAS COISAS[1]

Giovanni Cardozo – giocardozo99@gmail.com

Prof. Ms. Ivair Reis Neves Abreu (Orientador) – ivair.abreu@mackenzie.com

RESUMO

A automação está cada vez mais sendo utilizada de forma simples e acessível devido a evolução das tecnologias de microcontroladores e sensoriamento, barateando o custo e democratizando o uso. A automação residencial, também chamada de domótica, surgiu em virtude desses fatores, possibilitando o monitoramento dos principais processos em uma residência, principalmente do consumo de energia. O alto trafego de dados nesses sistemas faz com que seja necessário um algoritmo de Inteligência Artificial para processá-los e, assim como os sistemas de automação, evoluíram muito e atualmente são bem mais acessíveis e simples. A Internet das Coisas (*Internet of Things*) é o resultado obtido ao associar os elementos citados acima com a conexão à Internet. Este trabalho implementa um estudo de caso construído com componentes simples e accessíveis capaz de realizar a captura de informações de um certo ambiente com sensores, transmitir via internet para um ambiente *on-line* dedicado e tratar essas informações por meio de um algoritmo de lógica *fuzzy*.

Palavras-chave: Logica *Fuzzy*. Arduino. Internet das Coisas. Domótica

ARTIFICIAL INTELLIGENCE APLIED TO HOME AUTOMATION USING INTERNET OF THINGS

ABSTRACT

Automation is increasingly being used in a simple and accessible way due to the evolution of microcontroller and sensing technologies, lowering costs and democratizing use. Home automation, which emerged as a result of the aforementioned factors, is a system that is capable of monitoring the main aspects of a house and providing an overview of energy consumption. The high data traffic in these systems makes the use of an algorithm based in Artificial Intelligence necessary, which, like automation, have evolved a lot and are currently much more accessible and simple. The Internet of Things is the result that is obtained by associating said elements with Internet connection. A case study was built with simple and accessible components capable of capturing information from a certain environment with sensors, transmitting it via internet to a dedicated online website and treating this information through a fuzzy logic algorithm.

[1] Artigo do Trabalho de Conclusão de Curso, Graduação em Engenharia Elétrica, EE, UPM, São Paulo, 2021.

Keywords: Fuzzy Logic. Arduino. Internet of Things. Home Automation

1 INTRODUÇÃO

Com o avanço e redução dos preços da tecnologia sensorial e dos microprocessadores de monitoramento, a automação residencial tornou-se acessível. Ações como controlar a iluminação de uma residência à distância, monitorar o gasto de energia de equipamentos e detectar situações prejudiciais a saúde dos ocupantes de uma residência, com a tecnologia atual, são processos rápidos, simples e eficientes.

A inteligência artificial tornou-se também uma tecnologia complementar para as novas soluções de automação. Com o avanço nos estudos de *Machine Learning* e Redes Neurais, os algoritmos de tomada de decisão passaram de ser baseados em uma lógica binária para serem mais próximos do raciocínio humano. Técnicas de modelagem de problemas, como a lógica *fuzzy*, auxiliam o entendimento e a concretização de soluções cujo entendimento, anteriormente, seria demasiado complicado.

O texto tem o propósito de estudar os aspectos fundamentais de uma aplicação IoT (*Internet of Things*), da Inteligência Artificial, sua integração e aplicação. Durante os capítulos desse trabalho serão aprofundados os conceitos de sensoriamento, inteligência artificial, lógica *fuzzy*, comunicação de redes e o funcionamento interno da placa de desenvolvimento Arduino.

O estudo de caso proposto para aplicar os conceitos estudados é baseido em uma placa de desenvolviento Arduino Mega (utilizando o microcontrolador ATmega 2560) controladora de sensores de luz, gás e presença, além de um módulo WI-FI (ESP-01) que realiza a comunicação sem fio. O monitoramento a distância do módulo será feito através de um ambiente *on-line*, onde o mesmo irá conter a interface de controle dos sensores.

O módulo desenvolvido será capaz de analisar os dados recebidos dos sensores através de um algoritmo de Inteligência Artificial, baseado na lógica *fuzzy*. Os dados serão tratados e, de acordo com os resultados da análise *fuzzy*, o módulo irá atuar sobre o ambiente em que está implementado.

2 REVISÃO DA LITERATURA

2.1 INTERNET DAS COISAS COM PLACA ESP8266, ARDUINO E RASPBERRY

O conceito por trás do termo Internet das Coisas (IoT) não é novidade e é discutido e desenvolvido desde o surgimento dos primeiros computadores. Segundo o autor Sérgio de Oliveira (OLIVEIRA, 2017) o processo que serve como alicerce para a evolução das tecnologias de comunicação é a troca de dados entre sistemas por uma rede sem fio. As primeiras tecnologias desenvolvidas para aplicar tal processo foi o RFID (*Radio Frequency Identification* – Identificação por Radiofrequência) em 1940 utilizado em aviões na Segunda Guerra Mundial.

A rede de comunicação entre os sensores e atuadores em aplicações IoT utiliza o conceito de Internet. Segundo Oliveira (2017), existem dois elementos fundamentais na arquitetura da comunicação via Internet: Servidor e Cliente.

- Servidor: é um *software* com porta serial (de comunicação) aberta à espera do pedido do cliente. Os servidores têm como objetivo armazenar as diversas ações e dados que o cliente possa precisar, e ao ser "acionado" tem a capacidade de fornecer os dados pedidos ou armazenar novas informações trazidas pelo cliente. Como exemplo têm-se os servidores WEB, de arquivos, de banco de dados, de mensagem e *e-mails* entre outros;
- Cliente: trata-se também de um software acionado normalmente pelo usuário. Navegadores WEB são um exemplo de cliente. O *software* tem a função de iniciar a comunicação com o servidor, seja por ação direta do usuário, seja uma resposta automática à um evento ou ação externa;

De acordo com o material estudado, ambos estes conceitos podem ser aplicados em dispositivos IoT, sendo que sensores podem ser representados como clientes e dispositivos podem ser associados aos servidores, mesmo com o *hardware* limitado.

O conceito de cliente e servidor abre espaço para o detalhamento de diversos protocolos e padrões importantes para o IoT, como: TCI/IP, RFID, XBee e o Wi-Fi.

Dentre os modelos de rede normalmente utilizados em aplicações de IoT, têm-se o modelo TCP/IP, o *Bluetooth* e o *ZigBee*. No modelo TCP/IP a camada de Internet é responsável pelo roteamento de pacotes de dados e a camada de acesso à rede determina o modo que os dados são trafegados. O *Bluetooth*, tecnologia sem fio, utiliza frequências na faixa de 2.4GHz com baixo consumo de energia, sendo ideal para aplicações de automação. O *ZigBee*, por fim, tem como características o baixo consumo de energia, flexibilidade e padronização na comunicação, podendo ser utilizada com três tipos de topologia: estrela, malha e árvore (SOUZA, 2016).

Para atender as diversas especificações de um dispositivo IoT a melhor opção são as placas de sistemas embarcados (OLIVEIRA, 2017). Nas placas de controle existem os microcontroladores, cuja construção é composta interfaces de entrada e saída (I/O), memória RAM, memória *Flash* ROM, memória EEPROM, circuito oscilador (*clock*), interfaces de comunicação (serial e USB) e mais recentemente interfaces de rede, Ethernet, Wi-Fi e *Bluetooth*.

Os principais sistemas embarcados utilizados para IoT são os módulos Arduino, Raspberry PI e ESP 8266. Uma das características fundamentais de um sistema embarcado é seu consumo de energia. Todos os sistemas citados acima têm um consumo médio na ordem de microWatts (μW) (OLIVEIRA, 2017) possibilitando o uso de baterias e pequenas placas solares para atuar como fonte de alimentação.

O sistema Arduino é como um "*open-source*" de hardware, isto é, permite a interação entre ambientes, dando suporte de leitura a sensores como: som, luz, temperatura, gás, entre outros e dar apoio a acionamento de LEDs, motores, *displays*, autofalantes e outros dispositivos (CUNHA, 2018).

A computação "em nuvem" (*cloud computing*) também é um tema importante, essencial para o funcionamento efetivo e de baixo custo dos dispositivos IoT (OLIVEIRA, 2017). Por conta da limitação de processamento e memória de dispositivos usados em IoT, é usual estes componentes enviarem as informações para um elemento centralizador com melhor processamento e disponibilidade. Se estes elementos se encontrarem no ambiente da nuvem, os projetos ganham com flexibilidade e escalabilidade (OLIVEIRA, 2017). Segundo o autor, as principais plataformas comerciais de nuvem são *Amazon AWS, IBM Watson* e o *Google Cloud Platform.*

2.2 DOMÓTICA DE BAIXO CUSTO USANDO PRINCÍPIOS DE IOT

O termo Domótica resulta de duas palavras de origem romana: domus, que significa casa, e a palavra robótica (STEVAN JUNIOR; FARINELLI, 2019). O termo resultante é a definição do processo de automatização do ambiente residencial. No contexto de domótica, segundo o autor, o termo pode abranger sistemas de controle, centralizados ou não, de segurança, iluminação, climatização, audiovisuais e telecomunicação, com o objetivo de proporcionar maior comodidade, conforto para o usuário e eficiência energética para o sistema.

Para automatizar uma residência, o ideal é que isto seja previsto na fase de projeto, antes da construção para possibilitar o projeto de cabeamento e instalações. Segundo TEZA (2002), esse sistema pode custar entre 1% e 7% do custo total da obra dependendo da abrangência.

Na automação residencial se utilizam sensores como medidores de temperatura, luminosidade, umidade, peso, gás, presença, entre outros. Eles têm a finalidade de converter uma grandeza física em grandeza elétrica para que o controlador possa realizar a conversão analógico/digital e processar essa informação acionando atuadores caso necessário (SILVA; MIRANDA, 2018). Os atuadores, como válvulas, motores, luzes, comandos sonoros, entre outros tem o objetivo de interferir no ambiente.

Os sensores de presença são sensores responsáveis por identificar a presença de alguém ou alguma coisa, em algum lugar ou região. A identificação pode ser de pessoas ou animais em um ambiente (com sensores infravermelhos passivos) ou de um objeto (como uma *tag* de um sensor RFID em um controle de acesso). As principais tecnologias utilizadas em seu funcionamento são sensores fotoelétricos, por ultrassom, por chave magnética, fim de curso e os sensores PIR (*passive infrared*) (STEVAN JUNIOR; FARINELLI, 2019).

O sensor de luminosidade mais comum e utilizado é o LDR *(Light Dependent Resistor*). Este sensor tem uma resistência variante, dependendo da quantidade de luz incidente sobre o mesmo

(STEVAN JUNIOR; FARINELLI, 2019). Por fim, segundo o autor, o sensor de gás mais utilizado é o MQ-2, por conta de seu baixo custo e da variedade de *Shields* (interfaces) do ambiente Arduino disponíveis. A aplicação típica é identificação de concentrações de gás combustível (GLP, metano, propano, butano, hidrogênio, álcool, gás natural e outros inflamáveis) e também de fumaça.

Em uma residência inteligente (*smart house*) o perfil de controlador passa a ser gerenciador e o sistema deixa de ser um controle remoto e passa a ser um supervisório. Os sensores podem apresentar uma conexão de mão dupla, ou seja, enviar e receber dados de um dispositivo da ponta do processo (TEZA, 2002).

A Domótica vem reduzindo seu custo de implantação aliada ao crescente número de pessoas com acesso à Internet. Segundo Castro (2012) *apud* Souza (2016), as políticas de inclusão digital foram fomentadas por países que adquiriram, mais rapidamente, o manejo dos aparatos tecnológicas para assegurar o desenvolvimento sustentável, o combate à pobreza e às desigualdades sociais no mundo.

Embora a automação residencial ainda seja recente no país, com o mercado em crescimento, o desenvolvimento deste sistema pode ajudar na popularização deste segmento. A automação residencial permite vários estudos e implementações na Engenharia. Diminuição de custos, facilidade, conforto e segurança, são fatores passiveis de serem explorados para melhorar o bem-estar do usuário. Com a placa de desenvolvimento Arduino é possível criar métodos para automatizar os mais diversos equipamentos de uma residência, visando alcançar maior comodidade nas tarefas do cotidiano (SARTORI; MOLINA; LIMA, 2015).

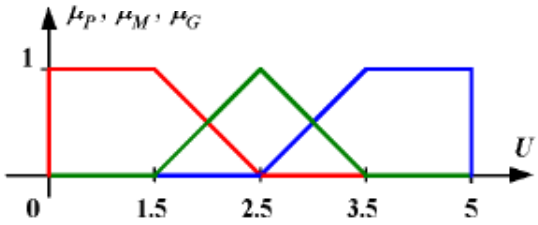
2.3 SISTEMAS “FUZZY”

O termo *fuzzy* é originado do inglês e geralmente pode ser definido como vago, indistinto ou incerto. Na engenharia o termo é geralmente traduzido para nebuloso ou difuso (ALMEIDA; EVSUKOFF, 2005). A lógica que leva esse nome pode ser aplicada para a representação e processamento de conhecimento de máquina, a fim de auxiliar a montagem e a modelagem de algoritmos de inteligência artificial.

A lógica *fuzzy* é uma maneira de quantificar e analisar os universos em que as variáveis contidas não podem ser descritas através de uma lógica booleana. Os objetos de uma mesma classe com características semelhantes são agrupados em conjuntos, podendo assumir mais que dois valores distintos. Na área de Engenharia existem situações com resultados vagos, incertos ou imprecisos e, desta forma, impossíveis de se caracterizar através da lógica clássica binária. A lógica *fuzzy* pode ser vista como uma extensão da teoria clássica de conjuntos, criada para tratar graus de pertinência intermediários entre a pertinência total e a não pertinência de elementos de um universo de discurso com relação a um dado conjunto (ALMEIDA; EVSUKOFF, 2005).

Em uma análise *fuzzy*, o grau de pertinência de um determinado objeto em relação ao conjunto pertence ao intervalo [0,1] onde “0” significa que o elemento não pertence ao conjunto e “1” representa a total pertinência do objeto ao conjunto. No exemplo a seguir (Gráfico 1), cada cor representa um conjunto *fuzzy*, enquanto que o eixo y representa a pertinência do objeto em relação aos conjuntos apresentados.

Gráfico 1 - Conjuntos “fuzzy”



Fonte: (ALMEIDA; EVSUKOFF, 2005).

Os conjuntos *fuzzy*, portanto, podem ser definidos como objetos matemáticos que representam a imprecisão, uma característica natural ao se analisar o pensamento humano (Nguyen e Walker, 2006 *apud* Oliveira, 2018). Os conjuntos *fuzzy* têm seus elementos definidos por uma função de pertinência, nome dado a relação que define o grau de similaridade do objeto em relação a um conjunto difuso. Uma função de pertinência é, portanto, uma relação que informa quanto um elemento, a depender de seu valor, pertence a um conjunto *fuzzy* e são usadas para determinar o grau de pertinência de um valor qualquer (OLIVEIRA, 2018).

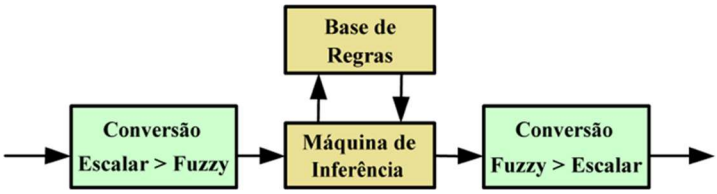
A segunda etapa do processo de implementar a lógica *fuzzy* é o processo que determina o grau (medido em percentagem em relação ao total) um valor qualquer da variável em questão pertence a alguma das funções de pertinência.

Após esse processo, inicia-se a inferência em sistemas de lógica nebulosa, ou seja, a análise dos dados recebidos da etapa anterior em relação à um conjunto de regras estabelecido. As regras em análises *fuzzy* são geralmente da natureza “Se X é verdade, então Y é verdade “ (RUSSELL; NORVIG, 2013 apud OLIVEIRA, 2018).

A condição da regra pode ser única ou composta por meio de operadores lógicos (OU, E), mas em todos os casos as regras devem apresentar resultados da Lógica Booleana, ou seja, os valores podem ser apenas VERDADEIRO ou FALSO (NGUYEN; WALKER, 2006 apud OLIVEIRA, 2018).

O passo final da resolução de problemas por meio de Lógica Nebulosa é o processo de regressão da lógica *fuzzy*. Nesta etapa são produzidos resultados quantificados (numéricos) a partir das regras criadas para as funções de pertinência de um modelo nebuloso. A Figura 1 a seguir demonstra uma visão geral do processo descrito acima.

Figura 1 - Diagrama típico de um modelo de inferência



Fonte: ALMEIDA; EVSUKOFF, 2005

Adaptado para uso em sistemas de automação residencial essa técnica pode contribuir para a maior eficiência e durabilidade, dependendo da base de regras utilizadas, de sistemas de iluminação, segurança e temperatura (CONGRESSO BRASILEIRO DE AUTOMÁTICA, 2014).

2.4 PROGRAMAÇÃO COM SISTEMA ARDUINO

O sistema Arduino é uma plataforma de programação de microcontroladores cuja facilidade de uso e sua natureza aberta (*open source*) fazem dele uma ótima opção para a implementação de projetos eletrônicos (MONK, 2017). Um dos componentes do sistema Arduino é o microcontrolador com os itens básicos de um computador dentro de um Circuito Integrado (CPU + Memória + Periféricos). Além de conter alguns recursos normalmente presentes em um computador, um microcontrolador apresenta outros itens dedicados: Processador Central (CPU), Memória RAM (*Random Access Memory*) para guardar dados, memória EPROM (*Erasable Programmable Read Only Memory*) ou Memória *Flash* para armazenar os programas e ainda interface paralela, como pinos de entrada e saída (I/O).

Uma funcionalidade importante nos módulos Arduino é o uso de Interrupção pois permite realizar mais de uma tarefa em um mesmo processo. Assim, cada um desses pinos pode atuar como uma entrada que, ao receber um sinal que obedeça a certas especificações, fará o microcontrolador interromper a sequência do processamento e realizar uma chamada para uma função específica a fim de atender a esse evento (MONK, 2017).

Um ponto importante para entender o funcionamento e a programação do ambiente Arduino é detalhar qual o seu microcontrolador. O ATmega2560 é o microcontrolador utilizado na placa Arduino Mega produzido pela empresa Atmel (MONK, 2015) e utilizado no estudo de caso deste trabalho.

O microcontrolador da família ATMega contém uma série de registradores que permitem configurar o hardware interno do microcontrolador. Os *GPIOs* (Pinos PB, PD e PC) podem ser configurados de acordo com o valor escrito nos seus registradores de controle correspondentes: DDRx, PINx e PORTx. Os registradores DDR configuram se o pino será uma entrada (0) ou saída

(1), os registradores PIN são responsáveis por armazenar a informação lida dos pinos e os PORT têm a função de escrever 0 ou 1 em pinos declarados como saída (ATMEL CORPORATION, 2014).

Programas mais robustos desenvolvidos no módulo Arduino podem necessitar do uso de interrupções de *timers* para seu funcionamento mais eficaz. As interrupções, tratadas no programa por meio da função ISR (*Interrupt Service Routine*), podem ser disparadas pelos próprios GPIOs utilizando duas opções: *Pin Change* (qualquer alteração nos pinos de entrada) ou INT (interrupções externas), que permitem a configurar a borda de disparo. Outra forma de interrupção do processamento são os *Timers* 0, 1 e 2 internos ao microcontrolador ATMega.

A programação do sistema Arduino voltada para microcontroladores da família ATMega apresenta vantagens na sua utilização. Uma vez que é possível controlar internamente os processos do microprocessador do Arduino o código fica mais econômico, personalizável e simples de alterar e detectar problemas além de ser mais prático aplicar melhorias.

3 METODOLOGIA

Neste tópico será abordado todas as práticas utilizadas para compor o projeto final além da apresentação dos materiais e o desenvolvimento até a obtenção dos resultados.

O estudo de caso deste trabalho será o desenvolvimento de um módulo de IoT residencial. O módulo tem como principal elemento uma placa Arduino Mega (utilizando o microcontrolador ATmega 2560) cuja função é controlar sensores de luz, gás e presença.

O módulo irá contar com dois sensores digitais e um sensor analógico. Os sensores digitais são o sensor de presença PIR (*Passive Infrared*) e o sensor de gás MQ-2, voltados para monitorar aspectos de segurança dos usuários de uma residência. O sensor analógico utilizado no projeto é o LDR (*Light Dependent Resistor*), cuja leitura servirá de base para um algoritmo *fuzzy* que controlará um LED. A lógica *fuzzy* será modelada conforme o modelo de Mandami utilizando o método de centro de gravidade.

O controle do módulo poderá ser realizado localmente ou através de um ambiente utilizando servidor externo (“nuvem”). A conexão com a Internet será feita através do servidor externo dedicado às soluções IoT chamado Cayenne. O site deste servidor irá conter a interface de controle de todos os sensores e as informações obtidas pelos mesmos possibilitando, assim, monitoramento completo pelo usuário em tempo real.

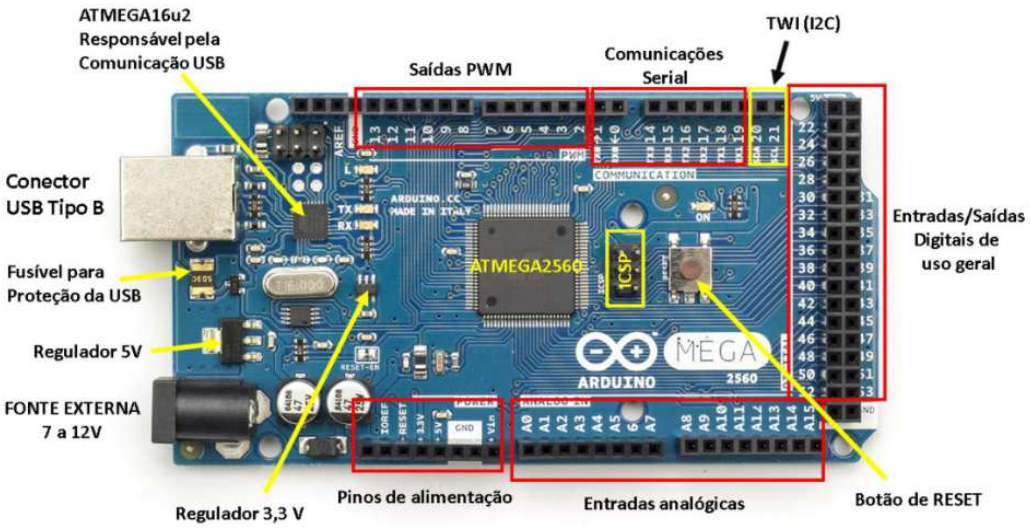
3.1 ARDUINO MEGA E ESP - 01

O programa do ambiente Arduino será programado através do *software* de desenvolvimento Arduino IDE (*Arduino Integrated Development Environment*). Nele serão atribuídas diversas funções para as entradas digitais, analógicas e PWM. As funções padrão do *software* de desenvolvimento

serão escritas na linguagem C++ e a configuração das funções internas, como interrupção e *timers*, do ATMega serão escritas na linguagem Assembly.

Serão utilizadas as entradas e saídas digitais, analógicas, saídas PWM e pinos de comunicação serial. A placa Arduino Mega permite a conexão de diversos sensores e atuadores, além da possibilidade de utilizar diferentes portas seriais para diferentes tipos de comunicação, devido a grande quantidade de portas presentes na placa como mostrado na Figura 2.

Figura 2 - Detalhes placa Mega



Fonte: SOUZA. 2014

O módulo WI-FI (ESP-01), mostrado na Figura 3, será responsável por realizar a comunicação sem fio. O mesmo consiste em um microprocessador ARM de 32 bits com suporte embutido à rede WiFi e com memória flash integrada, utilizada neste projeto junto a placa Arduino para agregar conexão sem fio a mesma, tendo suporte às redes WiFi 802.11 b/g/n e criptografia WPA e WPA2 (TECHNOLOGY).

Figura 3 - ESP-01

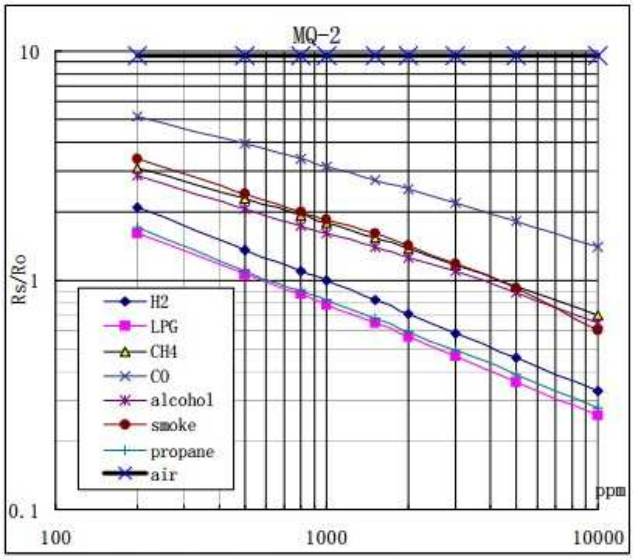
Fonte: FILIPEFLOP

3.2 O SENSOR DE GÁS MQ - 02

O material sensível do sensor de gás MQ-2 é o SnO2 (Dióxido de Estanho), responsável por detectar a existência de gases no ambiente. A condutividade do sensor aumenta diretamente proporcional com o aumento da concentração do gás no ambiente (COMPANY, 2009). A relação da

resistência do sensor, portanto, é inversamente proporcional ao aumento da concentração dos gases presentes conforme Figura 4.

Figura 4 - Comportamento da resistência do sensor na presença de gás

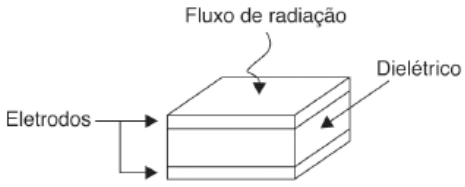


Fonte: COMPANY, 2009

3.3 O SENSOR DE PRESENÇA PIR

O sensor PIR HC-SR501 utilizado possuí, dentro de sua lente, um material piroelétrico de alta sensibilidade capaz de detectar a radiação infravermelha emitida por corpos. Os piroelétricos são formados por um filme de material cerâmico ferroelétrico dispostos como um capacitor. Dois eletrodos são posicionados neste filme a fim de criar as conexões do sensor (BALBINOT; BRUSAMARELLO, 2019). A Figura 5 a seguir demonstra um esboço de um sensor piroelétrico.

Figura 5 - Modelo de um sensor piroelétrico



Fonte: BALBINOT; BRUSAMARELLO, 2019

O funcionamento do sensor é baseado no efeito piroelétrico, onde a radiação incidente oriunda de um corpo faz aumentar a temperatura do material cerâmico. Isso provoca uma alteração na sua polarização interna que, por sua vez, ocasiona uma redução da carga superficial do material e um excesso de carga induzida nos eletrodos. Assim, se faz necessário um circuito para medir a carga induzida. Um dos métodos para medir a resposta do sensor é o modo tensão, onde a tensão v_o de saída é calculada conforme a Equação 1:

$$v_o = \omega\alpha\tau_F \Phi A p \frac{1}{\delta} R \frac{1}{\sqrt{1+(\omega\tau_T)^2}} \frac{1}{\sqrt{1+(\omega\tau_E)^2}} \quad (1)$$

Sendo ω a frequência da fonte de excitação, α, o coeficiente de absorção de radiação, Φ, o fluxo de radiação, A, a área efetiva do sensor exposta à radiação térmica, p, o coeficiente piroelétrico do material que constitui o sensor, δ, a condutância térmica, τ_F, a transmissão do filtro IR, τ_E, a constante de tempo elétrica e τ_T, a constante de tempo térmica que é determinada pela Equação 2:

$$\tau_T = H_p / \delta \quad (2)$$

Em que Hp representa a capacidade térmica do material piroelétrico e δ a condutância (condutividade) térmica.

3.4 O SENSOR DE LUMINOSIDADE LDR

O LDR é um fotoressitor, ou seja, são resistores variáveis controlados pela luz. Os fotorresistores são compostos de cristais semicondutores, como o CdS (sulfeto de cádmio) e o PbS (sulfeto de chumbo).

A análise do sensor LDR será feita através de inteligência artificial, utilizando os princípios da lógica *fuzzy*. Serão coletados dados de leitura de luminosidade do sensor LDR e medida a saída PWM a ser controlada. Diagramas *fuzzy* de entrada e saída serão desenvolvidos e, com o auxílio de ferramentas de cálculo como o *SciLab*, o modelo *fuzzy* será implementado.

3.5 DESENVOLVIMENTO DO CÓDIGO

O código desenvolvido teve como principal objetivo realizar as rotinas necessárias da forma mais eficiente possível, buscando o máximo de paralelismo no Arduino. O código deve controlar quatro rotinas essenciais: a leitura dos botões e sensores, o acionamento dos LEDs de indicação, a comunicação com o ambiente WEB e o tratamento *fuzzy* do LDR, sendo estas rotinas escritas utilizando o software de simulação e implementação no ambiente Arduino IDE.

O código utilizado no projeto pode ser acessado no servidor:
https://docs.google.com/document/d/1KXqRxeyVHqd1I2PGxbWpF0XbHLDDwCTg/edit?usp=sharing&ouid=104686207497224090986&rtpof=true&sd=true

3.5.1 Leitura Botões e Sensores

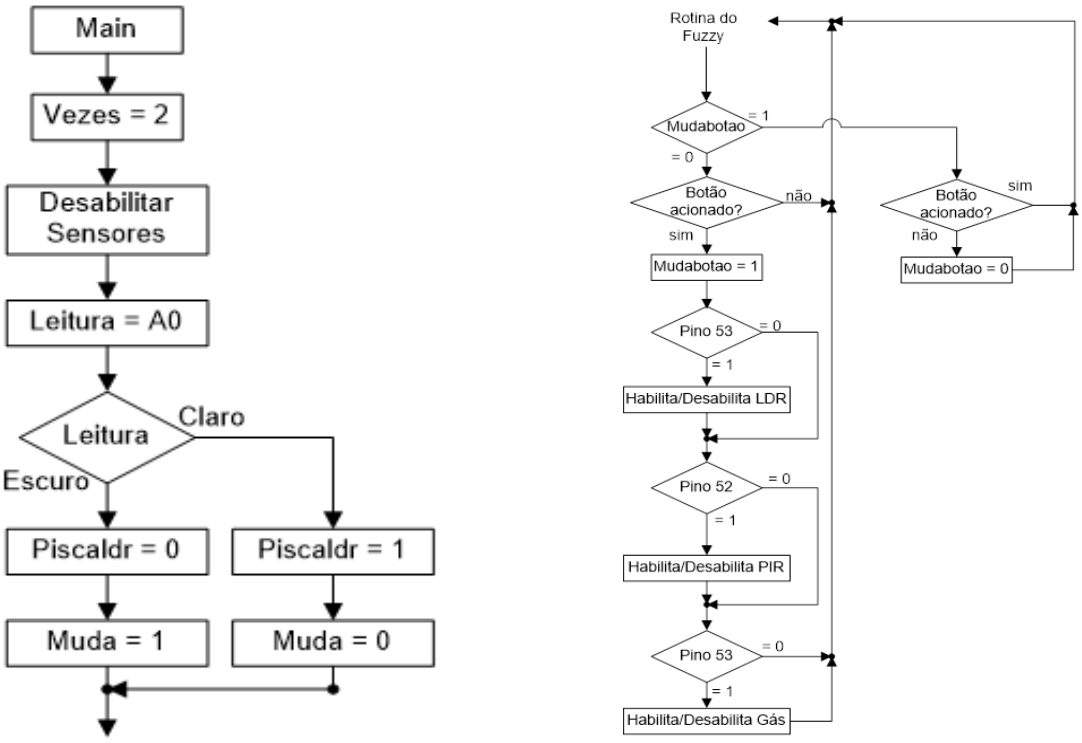
Essa rotina realiza a leitura dos botões presentes na placa e, caso este seja acionado, é habilitada a leitura dos sensores correspondentes para acionamento dos LEDs sinalizadores. A figura 6 apresenta o fluxograma de inicialização das variáveis que habilitam ou desabilitam o monitoramento dos sensores e a inicialização da variável de leitura do sensor de luminosidade (LDR) ligado a entrada analógica.

A figura 6 também apresenta o fluxograma sobre o tratamento das funcionalidades dos botões. Esse trecho do projeto foi desenvolvido de maneira a não ficar em um laço eterno, para permitir o tratamento de outras funções em paralelo como a rotina do *fuzzy* e as funções do servidor externo "Cayenne".

Essa é a função da *flag* "MudaBotão" que, quando é igual à zero, trata o acionamento dos botões e quando é igual a 1, trata a espera pelo usuário soltar os botões. Isso foi feito devido ao ruido elétrico ("bounce") existente em botões mecânicos.

Da mesma forma que ocorre no comando remoto via WiFi, os botões têm a função de habilitar ou desabilitar o monitoramento dos sensores de maneira local (situação "off line").

Figura 6 - Inicialização das variáveis e tratamento dos botões



Fonte: Autoria própria (2021)

3.5.2 Acionamento Leds

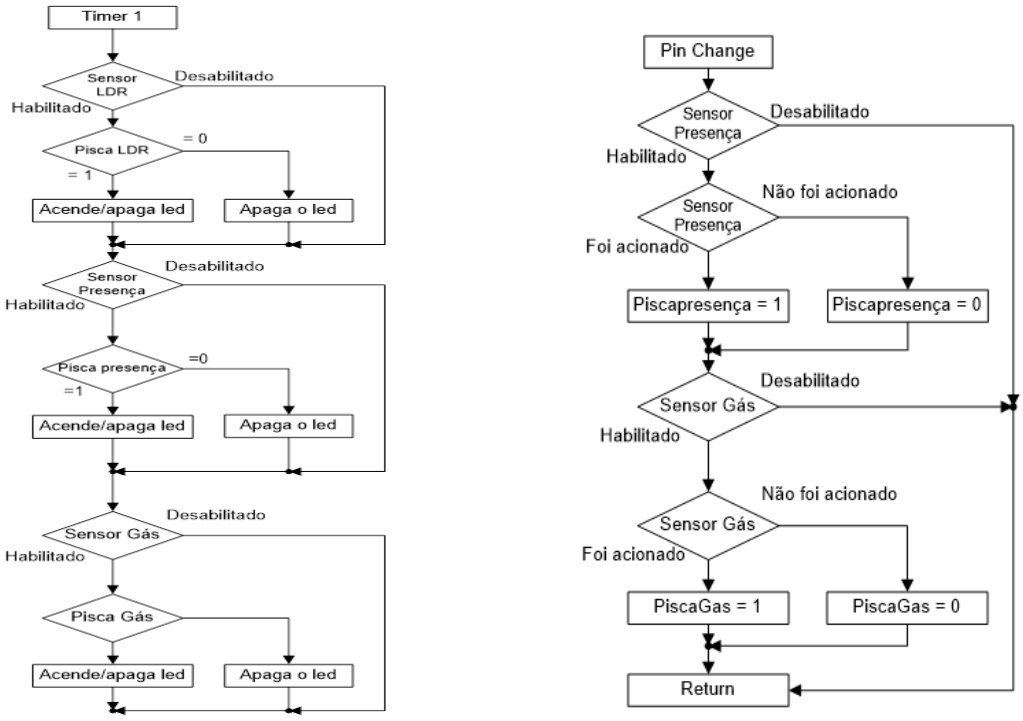
O tratamento do acionamento dos LEDs é feito através de rotinas de interrupção de *timer*. A rotina de interrupção externa (*Pin Change*) é detalhada na figura 7 onde observa-se que ao ocorrer uma alteração em um dos pinos de interrupção o programa principal será interrompido e esse trecho de código será executado.

Caso o monitoramento do sensor de presença ou do sensor de gás tenha sido habilitado, localmente ou remotamente, a leitura do estado do sensor definirá se um LED piscará para indicar que há presença de pessoas ou objeto no ambiente ou se existe um vazamento de gás providências a serem tomadas. Ao final a rotina retorna ao programa principal para dar sequência ao funcionamento do módulo.

O tratamento da interrupção de temporização (“Timer 1”) é ser dividida em duas etapas distintas e com uma base de tempo ajustada para T_1 = 250ms. Assim, ocorre um *overflow* do timer (indicação de tempo finalizado), a rotina principal é interrompida e o programa segue a execução do código inserido no vetor do “Timer 1”. Esta análise determina se os LEDs devem ser alternados entre acesos e apagados, dando um efeito visual de pisca-pisca com frequência de 2Hz, ou seja, 250ms apagado e 250ms aceso.

Contudo, nesse trecho de código (Figura 7) as *flags*, que foram alteradas pelos sensores, serão testadas a fim de decidir se os LEDs devem ser alternados ou não. Caso nenhum sensor tenha detectado alguma irregularidade a ser sinalizada, o programa apaga o LED e o mantém assim até receber uma atualização de estado.

Figura 7 - Rotina da interrupção e timer



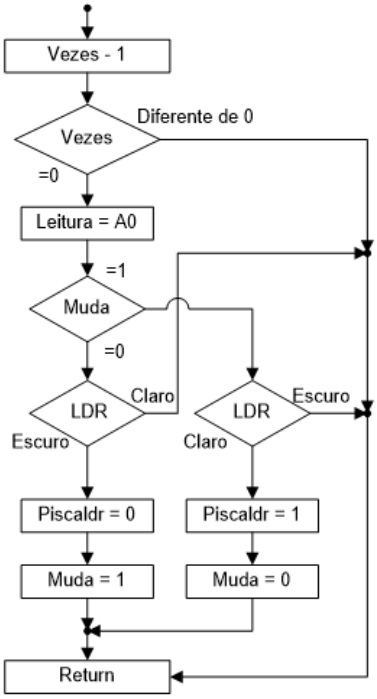
Fonte: Autoria própria (2021)

A fim de economizar energia e recurso de processamento, a mesma interrupção de timer de 16 bits (Timer 1) foi utilizada para realizar a leitura do sensor de luminosidade LDR (*Light Dependent Resistor*) através de uma entrada analógica.

Uma variável “Vezes” foi criada para garantir que a leitura do sensor aconteça apenas a cada 500ms ou seja, com uma frequência de 2Hz. A leitura da entrada analógica passa por dois trechos distintos do programa dentro da interrupção: um trecho verifica quando o ambiente apresenta uma luminosidade equivalente a uma lâmpada acesa e o outro trecho verifica quando o ambiente está escuro determinando assim a condição da *flag* que habilita piscar o LED de aviso de lâmpada acesa.

Ao finalizar do tratamento da interrupção a programação retorna ao programa principal (*main*), como pode ser visto na Figura 8.

Figura 8 - Rotina do Timer e LDR



Fonte: Autoria própria (2021)

3.5.3 Comunicação *Wi-fi*

A comunicação WiFi foi feita com o auxílio de uma ferramenta chamada “Cayenne”. Esta ferramenta possibilita criar uma página na WEB contendo objetos como botões e painéis de leitura *on-line*. Para realizar tal função, foi utilizada a biblioteca “*CayenneMQTTESP8266Shield.h*” cujas principais funções são:

a) *#define EspSerial Serial1*: Define qual interface serial será utilizada na comunicação com o shield ESP. Nesse caso, será utilizada a serial 1 (pinos 18 e 19).

b) *Cayenne.begin(username, password, clientID, wifi, ssid, wifiPassword)*: Realiza a comunicação do módulo ESP dada as informações fornecidas pelo *site*.

c) *Cayenne.loop()*: Função mais importante. Quando ocorre qualquer mudança no estado dos objetos criados, a função correspondente ao mesmo é chamada.

A ferramenta *Cayenne* informada ao usuário um endereço de um servidor próprio e a porta de acesso ao mesmo para o trafego dos dados dos sensores. Como o *Cayenne* utiliza o protocolo MQTT (*Message Queuing Telemetry Transport*) para realizar essa comunicação, o servidor representa o *broker* do sistema, comunicando os sensores ao cliente (nesse caso sendo o ambiente web desenvolvido).

Existem diversos tipos de objetos que podem ser criados na ferramenta, mas para o projeto, foram desenvolvidos os botões de acionamento dos sensores e um painel de leitura do LDR, como mostrado na figura 9.

Figura 9 - Página do módulo, criado através da ferramenta *Cayenne*

Fonte: Autoria própria (2021)

3.5.4 Lógica *Fuzzy*

Os parâmetros iniciais a serem definidos ao se implementar uma lógica *fuzzy* em um projeto são, o modelo de inferência (fuzzyficação) e o método de defuzzificação. Para o projeto foi utilizado o modelo de inferência Mamdani, onde tanto as regras antecedentes quanto as precedentes do modelo podem possuir relações *fuzzy*. O modelo escolhido utiliza operações de união e intersecção dos elementos através dos operadores máximo e mínimo, respectivamente (ALMEIDA; EVSUKOFF, 2005).

O modelo de defuzzificação (transformar de quantitativo para qualitativo) escolhido, por conta de ser utilizado na maioria dos *softwares*, foi o centro de gravidade. O método calcula, para o conjunto fuzzy de saída resultante da aplicação da base de regras, o valor (baseado nos parâmetros de saída pré-estabelecidos) do ponto de centro de massa correspondente, e o utiliza como valor escalar de saída (ALMEIDA; EVSUKOFF, 2005). A expressão analítica da implementação deste método é mostrada na Equação 3.

$$z^* = \frac{\int z \times \mu_c(z) dz}{\int \mu_c(z) dz} \quad (3)$$

Onde z^* é o valor final, z é o valor da abcissa no conjunto de saída e μ é o valor de pertinência correspondente ao valor z.

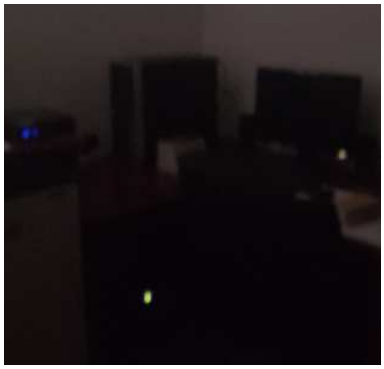
A aplicação da lógica *Fuzzy* no tratamento da leitura do LDR foi feito a partir da biblioteca eFLL (*Embedded Fuzzy Logic Library*). Esta biblioteca permite o usuário criar objetos de todas as etapas de uma lógica *Fuzzy*: variáveis de entrada / saída, o conjunto de regras e os gráficos de pertinência.

Para visualizar os modelos das variáveis de entrada e saída foi utilizado o software SciLab e um pacote de funções chamado "*Fuzzy Toolbox*". A primeira etapa da construção do modelo foi definir quais serão os conjuntos *Fuzzy* de entrada para a variável iluminação.

Os nomes escolhidos foram "escuro", "normal" e "iluminado". Fotos do ambiente foram obtidas para análise e os valores de luminosidade correspondentes obtidos de acordo com o coletado na leitura recebida do LDR no ambiente online.

Uma pesquisa foi elaborada com o intuito de se obter os melhores limites estre esses conjuntos de acordo com as situações apresentadas nas fotos. A cada foto era pedido ao participante determinar em qual dos conjuntos (escuro, normal ou claro) a situação era melhor representada. Ao final, foram coletadas as respostas de 50 pessoas, retornando o grau de pertinência de cada faixa de luminosidade aos conjuntos pré-estabelecidos. A Figura 10 demonstra uma das situações apresentadas na pesquisa, bem como o grau de pertinência obtido.

Figura 10 - Luminosidade 870 – 1023, Escuro -100%

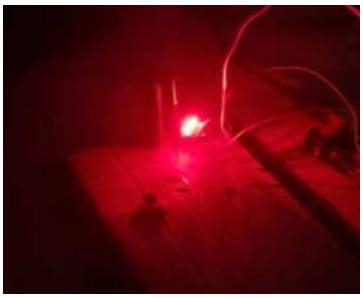


Fonte: Autoria própria (2021)

A variável de saída, denominada de brilho, foi construída de maneira semelhante. Os conjuntos *fuzzy* foram denominados "baixo", "médio" e "elevado" e diversas fotos foram tiradas. O brilho foi obtido experimentalmente através de leds controlados por saídas PWM do microcontrolador da placa ATmega.

A pesquisa para determinar os limites dos conjuntos da variável de saída foi semelhante ao apresentado anteriormente. A Figura 11 demonstra uma das situações apresentadas, seu respectivo valor e o retorno da pesquisa está apresentado a seguir:

Figura 11 - Brilho 30 – 70, Baixo = 4%;
Médio = 28%; Elevado = 68%



Fonte: Autoria própria (2021)

A próxima etapa a ser concluída foi aplicar os modelos demonstrados acima no programa do ambiente Arduino utilizando as funções fornecidas pela biblioteca eFLL. Para criar a aplicação *fuzzy* foram utilizadas as seguintes funções:

a) *FuzzyInput *iluminação:* criação da variável de entrada onde serão armazenadas, com o auxílio de outras funções, as informações dos conjuntos, seus valores e formatos (triangular, trapezoidal, unitário, etc)

b) *FuzzyOutput *brilho:* semelhante ao anterior, para a variável de saída.

c) *FuzzyRule *fuzzyRuleN:* criação da variável que representa as *N* regras a serem aplicadas na análise. Nessa variável, com o auxílio de outras funções, serão armazenadas informações como o modo de comparação entre as variáveis (simples, OU e E), e quais conjuntos devem ser comparados.

A programação implementada pode ser acessada no link do código já disponibilizado.

4 RESULTADOS E DISCUSSÃO

4.1 MONTAGEM DO MÓDULO IoT

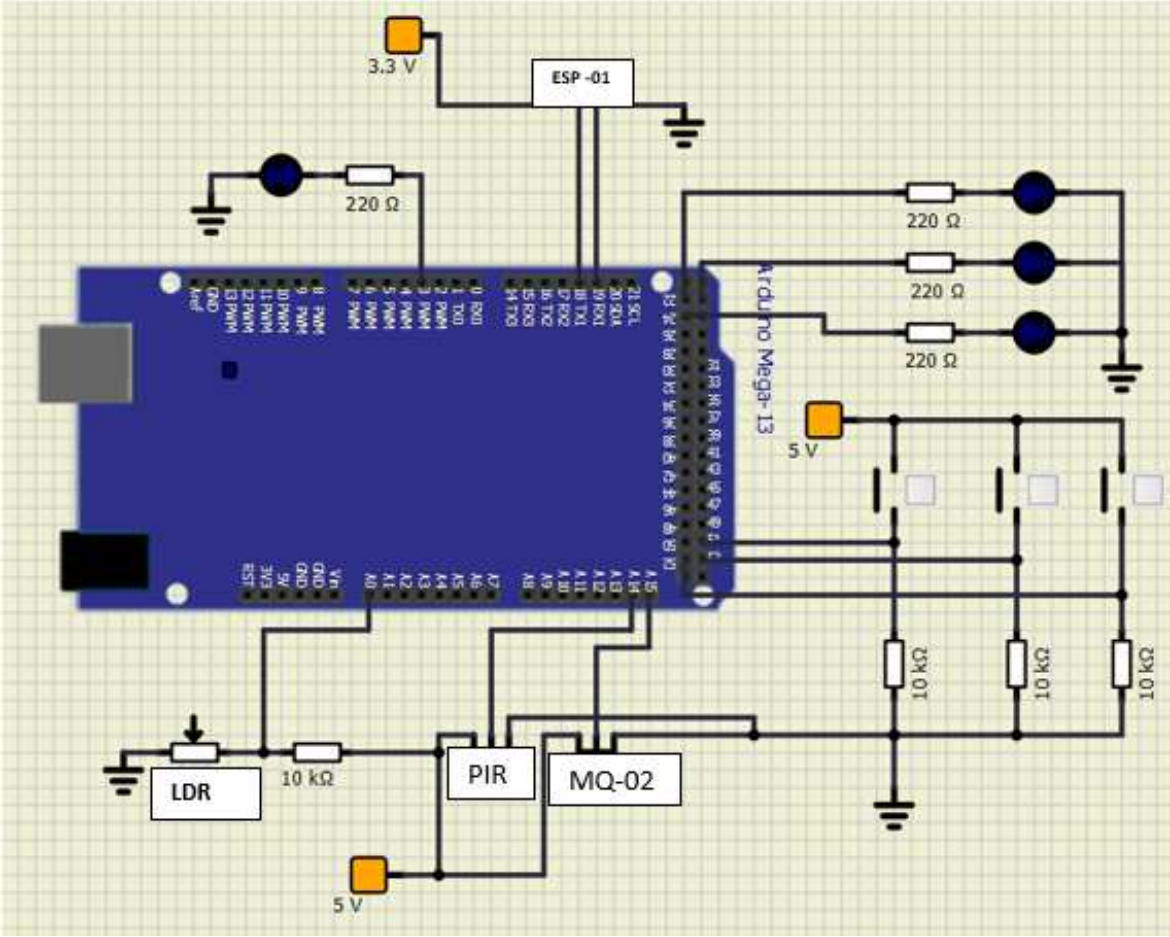
O circuito foi projetado para o usuário habilitar ou desabilitar os sensores manualmente (*in loco*) sem a necessidade da rede WiFi por meio de botões. Além disso, um módulo WiFi (ESP-01) foi ligado à interface serial 1 (pinos 19, RXD, e 18, TXD) do microcontrolador para atender a necessidade de controlar o módulo remotamente.

Dois sensores digitais foram ligados aos pinos de interrupção externa (Pin Change) de números A15 e A14 e o sensor analógico foi ligado ao pino de entrada analógica (A0) do microcontrolador. Os sensores ligados aos pinos de interrupção externa (Pin Change) são: sensor de

presença PIR (*Passive Infrared Sensor*) e sensor de gás MQ2. O sensor de luminosidade ligado a entrada analógica foi um LDR (*Light Dependent Resistor*).

Os LEDs de indicação foram conectados às saídas digitais 22, 23 e 24 e o LED controlado por inteligência artificial foi conectado à saída PWM 3. Os botões *push butons* para o controle local foram conectados às entradas digitais 51, 52 e 53. A seguir, na Figura 12, é possível visualizar o esquema elétrico.

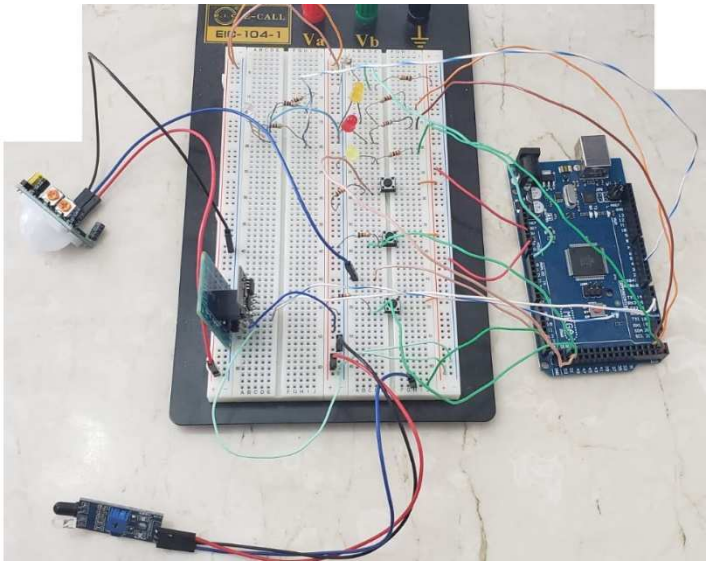
Figura 12 - Esquema elétrico e protótipo montado



Fonte: Autoria própria (2021)

Após o desenvolvimento do esquma elétrico teórico, foi feita a montagem do protótipo de testes. Foi utilizado um protoboard para auxiliar as conexões entre sensores e atuadores. Para facilitar os testes, foi utilizado um sensor de obstáculos para simular o funcionamento do sensor de gás, uma vez que ambos são digitais e ativados por nível lógico baixo. A Figura 13 demonstra a montagem do protótipo finalizada.

Figura 13 - Protótipo montado e finalizado



Fonte: Acervo pessoal (2021)

4.2 MODELAGEM *FUZZY* E O AMBIENTE WEB

A etapa de modelagem *fuzzy* foi concluída com o término da pesquisa que englobou o período de 2 semanas e envolveu 50 pessoas. Os resultados, em forma de tabelas, permitiram obter valores mais coesos e dentro da percepção geral do público para os limites de cada conjunto *fuzzy*. A seguir, nas tabelas 1 e 2, são apresentados os resultados obtidos, para a variável de entrada e saída, respectivamente.

Tabela 1 - Resultado da Pesquisa Sobre a Luminosidade

	Muito Claro	Claro	Normal	Escuro
LUMINOSIDADE DE 0 - 80	26%	46%	16%	12%
LUMINOSIDADE DE 80 - 340	0%	22%	52%	26%
LUMINOSIDADE DE 340 - 870	0%	0%	12%	88%
LUMINOSIDADE DE 870 - 1023	0%	0%	0%	100%

Fonte: Autoria própria (2021)

Tabela 2 - Resultado da Pesquisa de Brilho

	Baixo	Médio	Elevado
BRILHO DO LED 0 - 20	62%	30%	8%
BRILHO DO LED 20 - 30	14%	69%	16%
BRILHO DO LED 30 - 70	4%	28%	68%
BRILHO DO LED 70 - 125	4%	10%	86%
BRILHO DO LED 125 - 255	0%	0%	100%

Fonte: Autoria própria (2021)

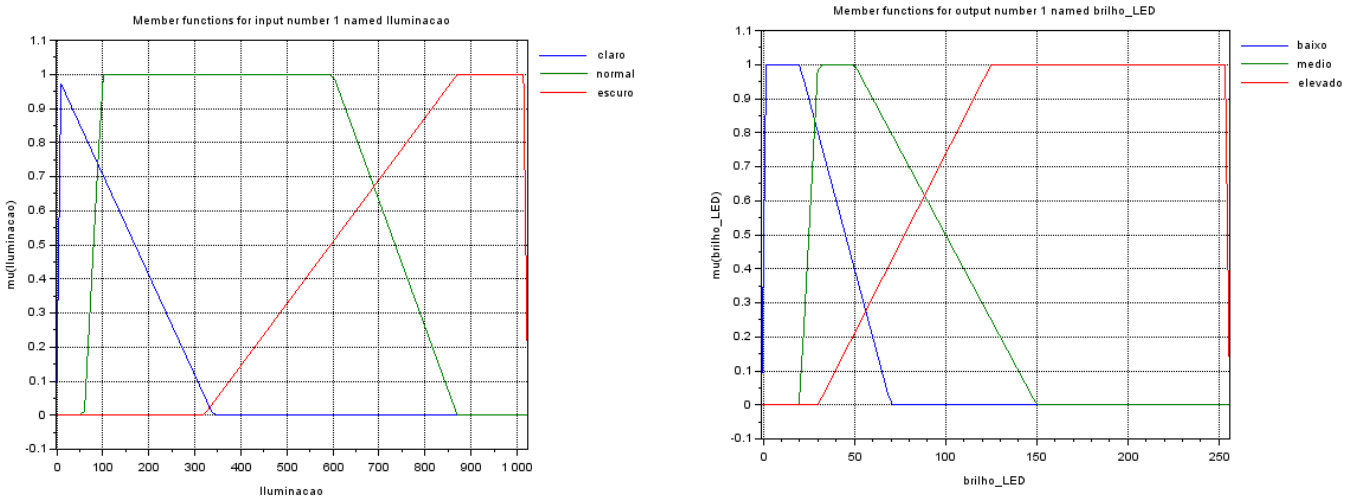
Os valores obtidos na pesquisa possibilitaram modelar as variáveis de entrada e saída no código do Arduino, utilizando as funções disponíveis na biblioteca eFLL. A aplicação no Arduino não fornece uma visualização gráfica da modelagem *fuzzy*, para isso foi necessário utilizar o SciLab.

Os limites dos conjuntos foram determinados pelas porcentagens apresentadas na pesquisa. Para a luminosidade, a categoria “Muito Claro” foi descartada, pois seu impacto no resultado não é

significativo. Desta forma esta variável foi considerada como “Claro” o que facilitou a montagem dos gráficos e a elaboração da base de regras.

No caso da variável de brilho do LED todas as categorias foram incluídas, pois todas apresentam impacto significativo para o resultado final. A montagem das tabelas de resultados possibilitou o entendimento do comportamento do cliente e suas preferências, a fim de elaborar o melhor modelo *fuzzy* possível como demonstrado na Figura 14.

Figura 14 - Modelos *fuzzy* de entrada e saída, respectivamente



Fonte: Autoria própria (2021)

Por fim, foram implementados os modelos detalhados acima na programado do ambiente Arduino utilizando a biblioteca eFLL. Em funcionamento todas as funções previstas são atendidas e funcionam em paralelo. A plena operação do módulo foi possível devido a utilização dos “timers” e interrupções do microcontrolador e das estruturas baseadas em funções de ambas as bibliotecas de comunicação com a WEB e de lógica *fuzzy*. Segue o *link* para o vídeo demonstração do funcionamento do protótipo desenvolvido: https://youtu.be/Rua9egbhdoY.

5 CONSIDERAÇÕES FINAIS

O projeto foi finalizado atendendo todos os aspectos propostos operacionais. O módulo é capaz de realizar o monitoramento da luminosidade do ambiente, da presença de gases e da presença de pessoas ou animais de forma simples e eficiente, podendo ser controlado localmente e via internet.

O ambiente *on-line* desenvolvido permite ao usuário visualizar a luminosidade do ambiente em tempo real, além de proporcionar o controle dos sensores. A página do aplicativo de visualização foi desenvolvida em um serviço gratuito e acessível, a fim de facilitar a personalização da página, de acordo com as preferências do usuário. A segurança dos dados é garantida por um *login* único para o usuário, sendo que não é possível acessar as informações do módulo sem as credencias necessárias.

O controle de luminosidade do ambiente utilizando um algoritmo de modelagem *fuzzy* também demostrou resultado satisfatório. O objetivo principal era implementar uma forma de economia de energia baseados nos dados históricos do ambiente. Os testes e o protótipo final controlam a luminosidade de um LED, uma forma simples encontrada para validar a hipótese inicial.

As principais melhorias possíveis no sistema é sua implementação em *dimmers* e a medição de energia consumida. Os dados obtidos na medição podem ser utilizados para realimentar o modelo desenvolvido e aprimorá-lo. Além disso, a prototipação do sistema desenvolvido em uma placa de circuito impresso possibilita que o mesmo possa ser implementado de uma forma mais prática.

REFERÊNCIAS

ALMEIDA, P. E. M., EVSUKOFF, A. G., **Sistemas Inteligentes: Fundamentos e Aplicações**, cap. Sistemas “fuzzy”, Manole, Barueri, São Paulo, 2005.

BALBINOT, Alexandre; BRUSAMARELLO, Valner João. **Instrumentação e fundamentos de medidas:** Volume 2. 3. ed. Rio de Janeiro: LTC, 2019. 515 p.

COMPANY, Hanwai Electronics. **TECHNICAL DATA MQ-2 GAS SENSOR**. Henan, China, 2009

CONGRESSO BRASILEIRO DE AUTOMÁTICA, 20., 20242014, Belo Horizonte. **Anais do XX Congresso Brasileiro de Automática**. Belo Horizonte: Escola de Engenharia Ufmg, 2014. 4313 p. Disponível em: http://www.swge.inf.br/CBA2014/anais/PDF/1569929521.pdf. Acesso em: 11 abr. 2021.

CORPORATION, Atmel. **ATMEGA 2560 DATASHEET**. San Jose, Ca: 0, 2014. 426 p.

CUNHA, Welliton Sousa da. ESTUDO DA INTELIGÊNCIA ARTIFICIAL APLICADA EM INTERNET DAS COISAS, VOLTADA NA AUTOMAÇÃO RESIDENCIAL. **Semana Acadêmica**, Fortaleza, v. 01, n. 000121, p. 1-29, 02 abr. 2018. ISSN 2236-6717. Disponível em: http://www.semanaacademica.org.br/. Acesso em: 05 abr. 2021.

FILIPEFLOP. **Módulo WiFi ESP8266 ESP-01**. s/d. Disponível em: https://www.filipeflop.com/produto/modulo-wifi-esp8266-esp-01/#tab-description. Acesso em: 22 set. 2021.

MONK, Simon. **Programação com Arduino**: começando com sketches. 2. ed. Porto Alegre: Bookman, 2017. 173 p.

MONK, Simon. **Programação com Arduino II**: passos avançados com sketches. Porto Alegre: Bookman, 2015. 247 p.

OLIVEIRA, Sérgio de. **Internet das Coisas**: com esp8266, arduino e raspberry pi. São Paulo: Novatec, 2017. 236 p.

OLIVEIRA, Ruy Flávio de. **Inteligência Artificial**. Londrina: Editora e Distribuidora Educacional S.A., 2018. 224 p.

OLIVEIRA, C, C; OLIVEIRA, C, D; GONÇALVES, C, J; KUNIWAKE, T, J. **Pratical Introduction to Internet of Things**: Practice using Arduino and Node. Teresina: IFTM, 2016

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 3. ed. Rio de Janeiro: Elsevier Editora Ltda, 2013. 935 p. Tradução de: Regina Célia Simille de Macedo.

SARTORI, Guilherme; MOLINA, Leandro Ariel; LIMA, Willian Cezar Gonçalves de. **DESENVOLVIMENTO DE UM SISTEMA MICROCONTROLADO DE BAIXO CUSTO UTILIZANDO SMARTPHONE PARA APLICAÇÕES DE AUTOMAÇÃO RESIDENCIAL**. 2015. 86 f. TCC (Graduação) - Curso de Engenharia Industrial Elétrica – Ênfase Eletrotécnica, Departamento Acadêmico de Eletrotécnica, Universidade Tecnológica Federal do Paraná, Curitiba, 2015.

SILVA, Caio Alexandre da; MIRANDA, Vanderlei Luiz Daneluz. AUTOMAÇÃO RESIDENCIAL COM INTELIGENCIA ARTIFICIAL: residential automation with artificial intelligence. **Revista Inovação, Tecnologia e Sustentabilidade na Engenharia Elétrica**, São Paulo, v. 1, n. 1, p. 48-70, 17 dez. 2018. Disponível em: https://www.unifafibe.com.br/revistaeletrica/?pagina=principal&edicao=69. Acesso em: 18 abr. 2021.

SOUZA, Marcelo Varela de. **Domótica de baixo custo usando princípios de IoT**. 2016. 49 f. Dissertação (Mestrado) - Curso de Engenharia de Software, Instituto Metrópole Digital, Universidade Federal do Rio Grande do Norte, Natal, 2016.

SOUZA, Fábio. **Arduino MEGA 2560**, 2014. Disponível em: https://www.embarcados.com.br/arduino-mega-2560/. Acesso em: 22 set. 2021.

STEVAN JUNIOR, Sergio Luiz; FARINELLI, Felipe Adalberto. **DOMÓTICA**: automação residencial e casas inteligentes com arduino e esp8266. São Paulo: Érica, 2019. 296 p.

TECHNOLOGY, Shenzhen Anxinke. **ESP-01 WIFI MODULE**. Shenzhen, China, 19 p. Disponível em https://pdf1.alldatasheet.com/datasheet-pdf/view/1179098/ETC2/ESP-01.html. Acesso em 11 nov. 2021.

TEZA, Vanderlei Rabelo. **ALGUNS ASPECTOS SOBRE A AUTOMAÇÃO RESIDENCIAL - DOMÓTICA**. 2002. 106 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Universidade Federal de Santa Catarina, Florianópolis, 2002.

TOCCI, Ronald J.; WIDMER, Neal S.; MOSS, Greogory L.. **SISTEMAS DIGITAIS**: princípios e aplicações. 11. ed. São Paulo: Pearson Prentice Hall, 2011. 817 p. TRADUÇÃO DE JORGE RITTER.