Tomás Marinho Brito dos Santos

Licenciado em Ciências da Engenharia Eletrotécnica e de Computadores

Gestão energética de iluminação pública centrada no utilizador

Dissertação para obtenção do Grau de Mestre em
Engenharia Eletrotécnica e de Computadores

Orientador: Rui Manuel Tavares, Professor Auxiliar,
Universidade Nova de Lisboa

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2021

Cognitive User-Aware Public Light Management

À minha família e amigos

Agradecimentos

Quero desde já agradecer não só à faculdade como ao professor Rui Tavares por me ter proporcionado esta oportunidade tão importante na minha formação enquanto futuro engenheiro. Ajudou-me certamente a compreender que para atingir os nossos objetivos vamos muitas vezes falhar.

Quero agradecer a todos os meus amigos mais próximos por todo o apoio dado durante estes anos de curso e boas memórias que certamente levarei comigo para a vida.

Por fim, agradeço a toda a minha família, em especial aos meus pais e irmã, por todo o apoio que me deram durante o meu percurso académico, proporcionando-me os melhores tempos de sempre.

A todos vós, Muito Obrigado!

Resumo

No contexto económico atual, a subida dos preços energéticos é uma realidade, assim como o impacto ambiental provocado por esse consumo.

A população está cada vez mais sensibilizada para esta pro*ble*mática, resultando numa alteração de comportamento perante o consumo energético.

O objetivo deste trabalho é fazer um estudo de um sistema de controlo inteligente para a iluminação, baseado num sistema de aquisição de dados que recolherá informações sobre a posição de indivíduos numa determinada área, para consequentemente interagir com os equipamentos de iluminação.

Depois de estudados vários documentos e artigos desta área, verifica-se que algumas cidades mundiais têm desenvolvido sistemas para diminuir o desperdício de energia associado à iluminação.

Palavras-chave: controlador; *Machine Learning*; *MQTT*; *ESP-32;* Iluminação inteligente; Internet das Coisas; *Bluetooth Low Energy*

Abstract

In the current financial context, the rise in energy prices is a reality, as well as the environmental impact caused by this consumption.

The population is increasingly aware of this problem, resulting in a change in behaviour towards energy consumption.

The aim of this work is to carry out a study of an intelligent control system for lighting, based on a data acquisition system that will collect information about the position of individuals in a certain area, in order to consequently interact with the lighting equipment.

After studying several documents and articles in this area, it appears that some cities in the world have developed systems to reduce energy waste associated with lighting.

Keywords: controller; Machine Learning; *MQTT*; ESP-32*;* Smart Lighting; Internet of Things; *Bluetooth* Low Energy;

Índice

Lista de Figuras

Índice de tabelas

Acrónimos

AC	*Alternated Current*
BLE	*Bluetooth Low Energy*
DC	*Direct Current*
GPIO	*General Purpose In and Out*
HDMI	*High Definition Multimedia Interface*
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	*Internet of Things*
IP	*Internet Protocol*
LED	*Light-emitting Diode*
LDR	*Light Dependent Resistor*
MAC	*Media Access Control*
ML	*Machine Learning*
MOSFET	*Metal Oxide Semiconductor Field Effect Transistor*
MQTT	*Message Queuing Telemetry Transport*
PIR	*Passive Infrared*
PWM	*Pulse Width Modulation*
P2P	*Peer to Peer*

RAM	*Random Access Memory*
RFID	*Radio Frequency Identification*
RSSI	*Received signal strength indicator*
SD	*Secure Digital Card*
SSID	*Service Set Identifier*
USB	*Universal Serial Bus*

1. Introdução

A iluminação é um elemento extremamente relevante para a sociedade, permitindo o prolongamento das horas do "dia", alterando assim os hábitos sociais.

A história demonstra que o homem foi desenvolvendo ao longo dos séculos as formas mais variadas de obter luz artificial, desde a simples fogueira, passando pelas velas de matéria gorda animal, lamparinas, até aos alimentados a energia elétrica da atualidade, que se foram desenvolvendo com o progresso da tecnologia e ciência, dando resposta às necessidades cada vez maiores das populações.

Tal desenvolvimento provocou maior consumo de energia e consequentemente um aumento da poluição luminosa. A poluição luminosa é uma consequência do mau aproveitamento da iluminação que se encontra associada à deficiente direção da luz. [1].

Urge diminuir a poluição luminosa de forma a reduzir o consumo dos recursos naturais, resultando numa redução do impacto energético e ambiental e consequentemente numa maior qualidade de vida dos cidadãos.

Nos Estados Unidos da América, por exemplo, estima-se que 30% da iluminação exterior é desperdiçada, levando a um consumo adicional de 3,3 mil milhões de dólares a que corresponde a libertação de 21 milhões de toneladas de CO_2 em apenas um ano [2].

A necessidade de redução do consumo energético do planeta e dos recursos naturais disponíveis leva à procura de fontes luminosas cada vez mais eficientes nomeadamente, a mais recente conhecida por *LEDs* (*Light Emitting Diode*) substituindo a anterior geração de lâmpadas fluorescentes. Com a iluminação *LED* os custos de energia exterior diminuíram entre 50%-70%, em comparação com as lâmpadas fluorescentes. A eficiência dos *LEDs* continua a melhorar, resultando num consumo inferior assim como o seu custo, dada a massificação da tecnologia [3].

Todavia não existem aparelhos que tenham rendimentos de 100%, isto é, existe sempre degradação de energia. Uma parte desta degradação na iluminação ocorre do facto de existirem iluminações desnecessárias, nomeadamente no mau aproveitamento da mesma. Tal poderia evitar-se usando mecanismos de alteração de intensidade luminosa, em situações onde não existam indivíduos, o que acontece com muita frequência em vias de circulação pública ou privada.

A redução de desperdícios e de um desenvolvimento sustentável está relacionada com o conceito emergente de *Society* 5.0, conceito este que começa a ser desenvolvido pelos japoneses.

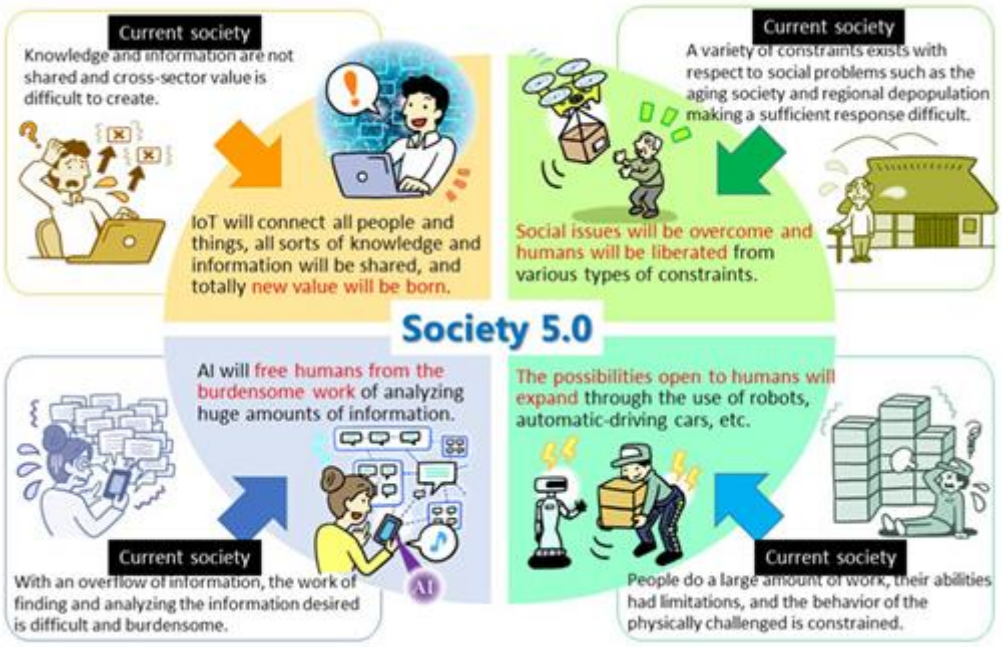

Figura 1.1 - Sociedade atual vs *Society* 5.0 [4]

Este conceito, permitirá intervir nas diversas áreas que afetam o tão preocupante aquecimento global e nas causas que o têm provocado, como por exemplo, a agricultura, a indústria, produção de energia entre outras.

Figura 1.2 - Facilidades da *Society* 5.0 [4]

Na *Society* 5.0 uma grande quantidade de informação é obtida por sensores e é acumulada no cyberspace, onde a informação é analisada por inteligência artificial e devolvida no espaço físico em inúmeras formas diferentes [4]. Para a presente *Society* 4.0 (sociedade da informação) a maneira mais comum de obter informações é através da internet, e essa informação é analisada por humanos, enquanto que na nova sociedade as pessoas, objetos e sistemas estão todos interligados no cyberspace. Tendo em conta o estudo em [5] podemos perceber como se processa toda esta troca de informações, para que seja um conceito viável.

Observando a Figura 1.2 entende-se que a aplicação deste conceito pode levar a diversos benefícios na sociedade nomeadamente redução de custos e desperdícios com a automatização dos sistemas. Comparando o trabalho aqui desenvolvido e o conceito de *Society* 5.0, entende-se que estão relacionados, pois é do interesse destes que os custos relacionados com a energia sejam diminuídos.

1.1. Objetivos

O objetivo desta dissertação é desenvolver um sistema de controlo inteligente para a iluminação, baseado num microcontrolador que recolherá informações sobre a posição de indivíduos numa determinada área, para consequentemente interagir com os equipamentos de iluminação.

O sistema a desenvolver teve como suporte original o diagrama de blocos que se encontra descrito na Figura 1.3, onde se "liga" um modulo de comunicação, de deteção de movimento e uma fonte de luz a um controlador.

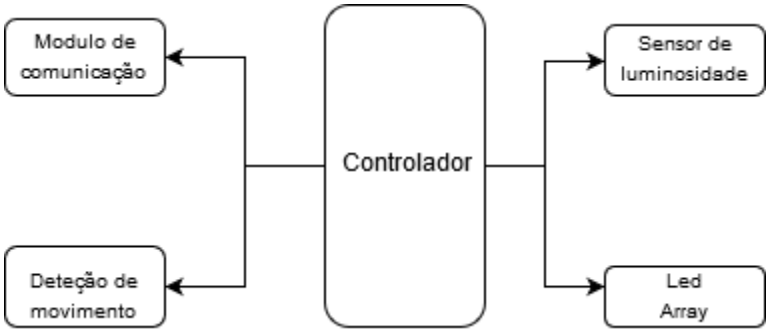

Figura 1.3 - Diagrama de blocos do sistema

Este sistema inteligente permitirá diminuir a quantidade de energia desperdiçada pela iluminação durante a noite, economizando o seu consumo, contribuindo assim para um meio ambiente mais sustentável.

A esta solução, vamos ainda adicionar a oportunidade de o utilizador, através de uma aplicação, escolher o seu tipo de luz preferido, seja este um tipo de cor mais frio ou mais quente.

1.2. Motivações

No contexto económico atual, a subida dos preços energéticos é uma realidade, assim como o impacto ambiental provocado por esse consumo. A população está cada vez mais sensibilizada para esta problemática, resultando numa alteração de comportamento perante o consumo energético.

O consumo de energia elétrica é um fator que nós enquanto cidadãos do mundo podemos controlar facilmente, seja em casa ou no trabalho. Da mesma forma, a iluminação é uma área que com o devido controlo, pode contribuir para uma redução desse consumo através de um sistema inteligente, capaz de controlar a iluminação necessária para a atividade de qualquer cidadã

2. Estado de Arte

Entende-se por iluminação o conjunto de fontes de emissão luminosa que se instala num determinado lugar com a intenção de o iluminar.

Nesta secção será realizada uma análise às tecnologias adotadas para a implementação do sistema, assim como a alguns estudos já realizados, que contribuíram para o desenvolvimento desta dissertação.

2.1. Iluminação inteligente

A iluminação inteligente (*Smart Lighting*) é um sistema avançado de controlo que pode ser utilizado em casas ou nas vias públicas. Estes sistemas contêm *softwares* que estão ligados a mecanismos de controlo que permitem usar de forma mais eficiente a iluminação de um espaço, possibilitando um uso energético mais sustentável.

Algumas cidades já iniciaram a implementação deste tipo de iluminação. Em Barcelona estão a ser instaladas cerca de 10.000 lâmpadas *LED* por toda a cidade. A cada luminária está associado um detetor de movimento, de forma a minimizar a energia quando ninguém está na área de deteção. Estas luminárias estão também equipadas com detetores de poluição, humidade e temperatura e servem de pontos de ligação *Wi-Fi* gratuitos para a população da cidade [3] [6].

Soluções parecidas com a cidade de Barcelona, foram também implementadas nas cidades de Copenhaga, Chicago e Londres [3].

Em Eindhoven, na Holanda, um sistema inteligente de iluminação está a ser desenvolvido pela *Heijmans* em parceria com a *Philips*. Neste sistema de controlo, para além de ser possível detetar a presença de um indivíduo na rua, permite também que os residentes escolham o seu *mood* de cor [7] [8].

2.1.1. *ZigBee Based Remote Control Automatic Street Light System* [9]

A função deste sistema de controlo inteligente, é semelhante à que aqui se apresenta, baseado na combinação de um microcontrolador monitorizando sensores cuja comunicação é feita com tecnologia *ZigBee*.

Apresenta-se um sistema adequado e sustentável uma vez que permite a segurança dos cidadãos nas diversas vias, reduzindo os custos e o consumo de energia, bem como diminui a emissão de CO_2 na atmosfera.

Este protótipo de funcionamento do sistema (Figura 2.1) é composto por:

- *Arduino* (controlador do sistema);
- *PIR* (*Passive Infrared*) sensor (para movimento de peões);
- Protocolo de comunicação *ZigBee*;
- LDR (*Light Dependent Resistor*) sensor (detetor de luz);
- Painel solar e bateria;
- *LEDs*.

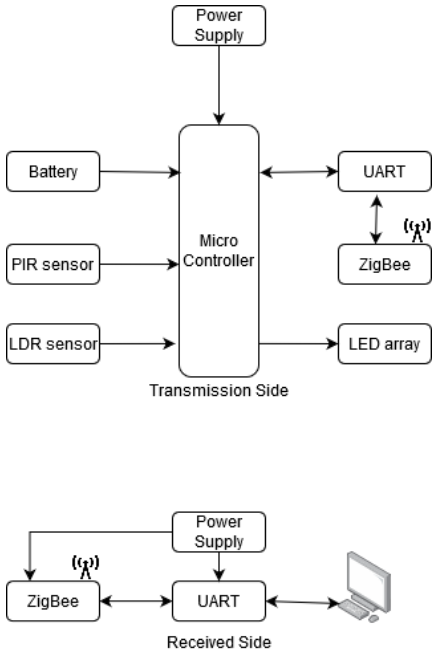

Figura 2.1 - Diagrama de blocos do sistema [9]

Depois do protótipo ser montado de acordo com o diagrama de blocos apresentado anteriormente, o seu funcionamento percorre os seguintes passos:

A energia das luminárias da estrada é fornecida pelos painéis solares. Os sensores recolhem a informação que é enviada para o microcontrolador.

O microcontrolador recebe os sinais e percorre-os pelo software para analisar o sistema.

Inicialmente os sensores de movimento só ativam o microcontrolador quando deteta peões e de seguida, este ativa o sensor de luminosidade.

O sensor de luminosidade deteta se a iluminação existente é suficiente (comparando com um *threshold* definido previamente) de modo a ligar ou desligar as luzes.

O sistema *ZigBee* está pronto para receber e transferir todos os dados fornecidos pelos sensores e simultaneamente comunica ponto a ponto para que consiga detetar luzes defeituosas.

Quando o sensor de movimento detetava a presença de um objeto a luz associada a esse sensor ligava-se, enquanto as restantes estavam ainda apagadas. Com o movimento do objeto pela estrada, as luzes iam ligando e desligando de acordo com a sua posição [9].

Embora este sistema seja uma solução para o nosso problema, não é a mais adequada, uma vez que não há a diminuição gradual da luz ao longo de uma avenida, sendo impossível para o pedestre ter uma visão clara do que o rodeia.

2.1.2. *A traffic-aware street lighting scheme for Smart Cities using autonomous networked sensors*

Neste artigo percebe-se que para uma circulação adequada é preciso que o pedestre consiga ter noção do que o rodeia, permitindo assim uma melhor navegação, a deteção de obstáculos com antecedência, a identificação de outros pedestres e a sensação de segurança [10] [11]. Estima-se que um pedestre evita uma colisão com outro a uma distância de 9 metros e uma colisão com um obstáculo a uma distância de 7 metros [12].

É preciso então haver iluminação no campo de visão do pedestre. Existem três tipos diferentes de distribuição de iluminação: convencional, em que numa dada avenida, todas as luminárias têm igual intensidade de iluminação; descendente, em que a intensidade de iluminação de cada luminária reduz gradualmente, dependendo da posição do pedestre, e ascendente, que é o oposto da descendente [13].

Considerando o caso de estudo de Haans e Kort [13], que se focaram na iluminação descendente, sendo esta a mais viável, temos que:

$$U_{ped} = \frac{1}{300}\int_{-150}^{150} z(x,t)dx \tag{2}$$

$$z(x,t) = \begin{cases} \varepsilon(x,t), & \varepsilon(x,t) \leq 1 \\ 1, & \varepsilon(x,t) > 1 \end{cases} \quad (3)$$

$$\varepsilon(x,t) = \frac{\gamma(x,t)}{1 - 0.2\left[\frac{|x|}{30}\right]}, -150 \leq x \leq 150 \quad (4)$$

Considerando as equações anteriores, em que x representa a distância de um pedestre no tempo t e $z(x,t)$ representa o rácio de iluminação necessária a x metros do pedestre. U_{ped} representa a utilidade da iluminação para a segurança pública e $\gamma(x,t)$ representa a iluminação mínima no ambiente em que o pedestre se encontra.

Este modelo considera uma distância de 150 m do pedestre, tanto para a retaguarda como para a vanguarda da sua posição. A avenida é dividida em segmentos de 30 m, que é aproximadamente a cobertura de iluminação de uma luminária exterior. Cada zona requer um diferente nível de iluminação que tem uma redução gradual de 20%, de acordo com a distância ao pedestre. É possível observar esta redução nos valores apresentados na Tabela **2.1**.

Tabela 2.1 - Relação do output de iluminação com a distância do pedestre à luminária [14]

Distancia de uma luminária, d	**Output de iluminação** (%)
$\mathbf{0 \leq d < 30}$	100
$\mathbf{30 \leq d < 60}$	80
$\mathbf{60 \leq d < 90}$	60
$\mathbf{90 \leq d < 120}$	40
$\mathbf{120 \leq d < 150}$	20
$\mathbf{d > 150}$	0

Baseando-se no caso de estudo anterior, Lau, Merrett, Weddell e White [14] projetaram e implementaram um sistema de controlo inteligente capaz de controlar a iluminação numa avenida, de acordo com a necessidade de um indivíduo, designando-o por projeto *TALiSMaN*.

O *TALiSMaN* baseia-se então na deteção de um pedestre numa avenida utilizando sensores locais e pelas informações transmitidas pelos nós vizinhos. Quatro operações de estado são definidas no *TALiSMaN*:

- *Lamp on by sensor*;
- *Lamp on by neighbour*;
- *Lamp on by delay*;

- *Lamp off*.

Na Figura 2.2 podemos observar o funcionamento do controlador, verificando quando se transita de um estado para o outro.

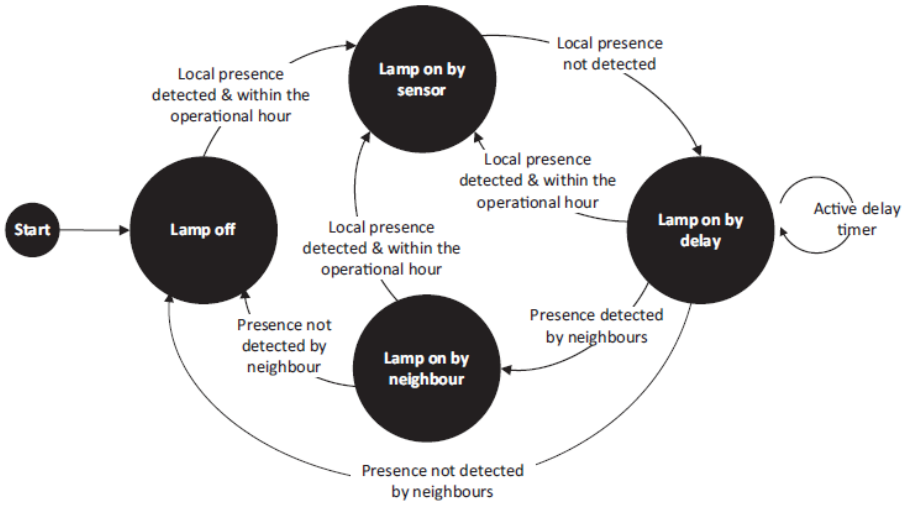

Figura 2.2 - Máquina de estados do funcionamento do controlador [14]

Um exemplo de algoritmo para o controlo de cada fonte de iluminação é descrito por:

if operation state is 'Lamp off' **then**

$$L_{ped} = 0$$

if operation state is 'Lamp on by sensor' **or** 'by neighbour' **then**

$$L_{ped} = 1 - 0.2\, Z_{ped}(d_{aprox})$$

if operation state is 'Lamp on by delay' **then**

current $L_{ped}(d_{aprox})$ remains

Em que L_{ped} define o nível de iluminação da luminária, baseando-se na distância relativa de um pedestre detetado. Z_{ped} e d_{aprox} determinam a zona de iluminação da luminária, de acordo com d_{aprox}.

Na Figura 2.3 pode-se observar o funcionamento do *TALiSMaN* numa avenida. Em s_2 é detetada a presença de um pedestre, fazendo com que s_1, s_2 e s_3 apresentem um *output* de iluminação de 100%. Nas restantes luminárias, a iluminação é continuamente reduzida num fator de 0.2.

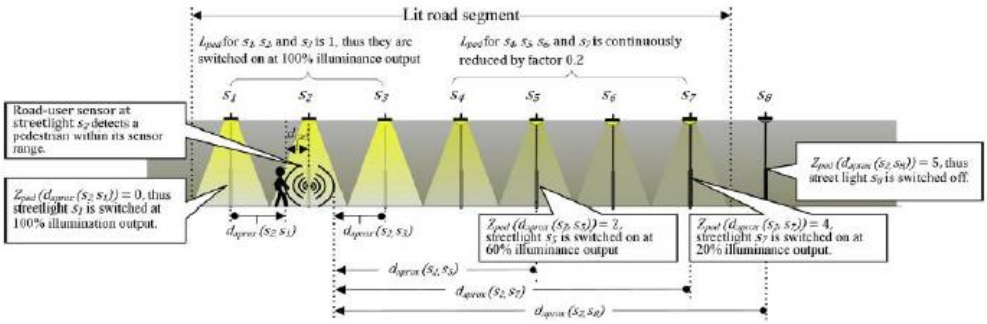

Figura 2.3 - *TALiSMaN* em funcionamento numa avenida [14]

Depois de implementado o sistema de controlo descrito anteriormente os autores quiseram testar o seu funcionamento. Para tal, utilizaram um simulador (*StreetlightSim* [15]) que permitiu projetar um ambiente para avaliar a eficiência do *TALiSMaN*, em que as luminárias exteriores estejam ligadas em rede.

Para controlar todas as luminárias existentes, é desenvolvido um controlador que comunica por *wireless*. Este recebe todos os dados detetados por cada luminária permitindo a localização de pedestres nas proximidades. Caso seja detetado um pedestre, essa informação é partilhada com o controlador, que faz o controlo de todas as luminárias, para que se adaptem à necessidade do peão.

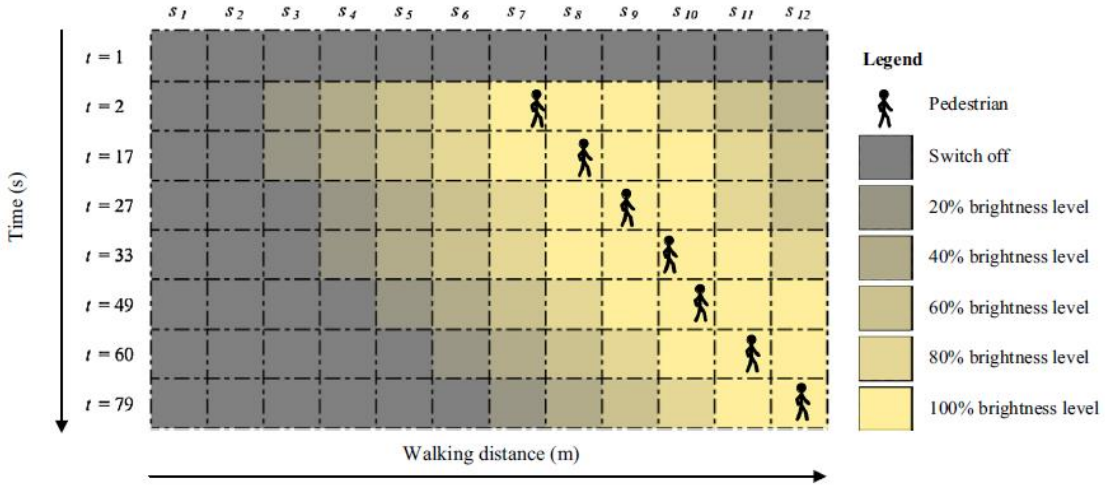

Figura 2.4 - Variação da intensidade de luz de acordo com a posição do pedestre [14]

Ao analisar a Figura 2.4 verifica-se que à medida que o pedestre se desloca as luminárias mais próximas são as que apresentam maior intensidade luminosa. A iluminação na sua posição e nas duas adjacentes está com máxima iluminação, enquanto as restantes têm uma redução gradual de 20%.

As luminárias comunicam também o seu consumo com o controlador, de forma a permitir a análise da diferença de consumos desta implementação em comparação com a iluminação convencional.

Tendo em conta todas as simulações e informações recebidas no controlador, conclui-se que a utilização do *TALiSMaN* traria uma melhoria nos consumos de energia, minimizando os seus custos, em comparação com a iluminação convencional existente. Dependendo do tráfico existente numa avenida, é possível economizar entre 45% e 98% de energia com esta aplicação.

Com estes resultados considera-se que é uma solução viável ao problema, sendo uma implementação prática exequível.

2.1.3. *Automatic Street Light Control System Using Microcontroller* [16]

Analisando a implementação de iluminação inteligente descrita em [16] verificam-se diferenças das soluções descritas anteriormente.

Neste estudo são utilizados os seguintes componentes:

- Sensor *LDR*
- Sensor Fotoelétrico
- Regulador de alimentação
- Relés
- *PIC*16F877A *Microcontroller*

O microcontrolador é responsável por receber a informação recebida pelo *LDR* e pelos sensores fotoelétricos, e decidir como vai operar todas as luminárias. O sensor *LDR*, medindo a intensidade luminosa na área, decide então quando o sistema deve estar em funcionamento. É decidido previamente um valor de intensidade luminosa para o início do sistema. Os sensores fotoelétricos são responsáveis pela deteção de movimento perto de uma luminária.

Os relés permitem controlar a corrente sendo muito utilizados como interruptores de controlo remoto.

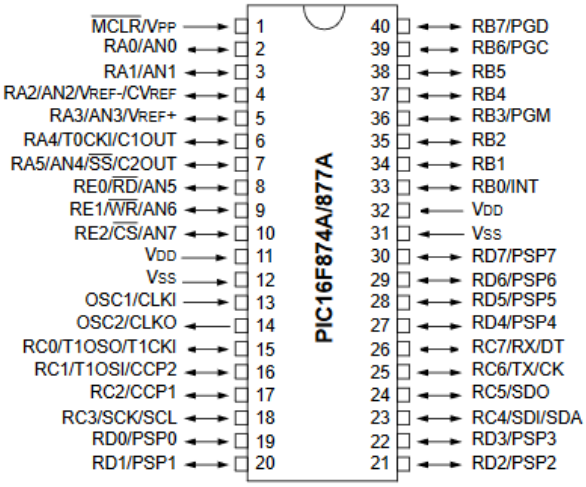

Figura 2.5 - *PIC*16F877A

Na imagem da Figura 2.5 podemos observar qual foi o microcontrolador utilizado pelos autores e os seus respetivos G*PIO* (*General Purpose In and Out*). Para que esta escolha de *PIC* fosse a mais adequada, os autores levaram em consideração toda a tecnologia utilizada nesta aplicação, verificando assim se este iria conter os requisitos do sistema.

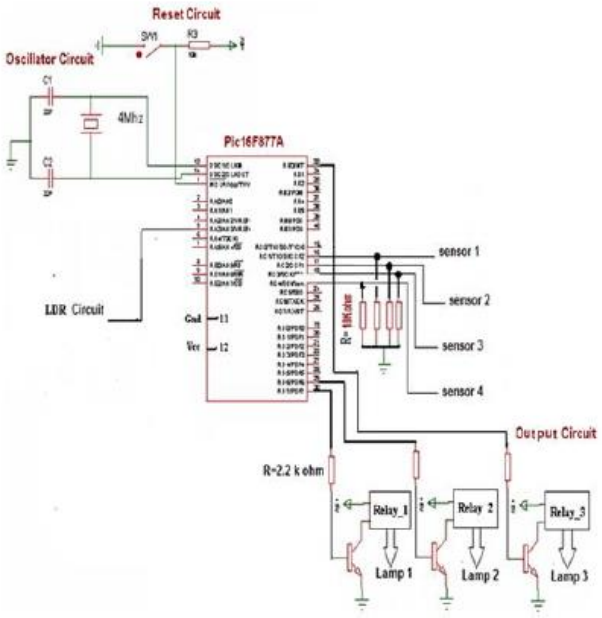

Figura 2.6 - Ligação dos Componentes

A Figura 2.6 permite então ver de que forma todos os componentes do sistema foram ligados.

O microcontrolador foi programado para que o sistema se comporte da forma pretendida. O sistema só entra em execução quando o sensor de luminosidade deteta uma intensidade luminosa inferior à pré-definida. Quando o microcontrolador recebe esta informação, começa então o funcionamento do sistema. Todas as luminárias estão *OFF* se não houver deteção de objetos. Os sensores fotoelétricos, quando detetam um objeto, comunicam com o microcontrolador para que sejam tomadas as devidas ações. A luminária associada a esse sensor fotoelétrico é então colocada *ON*. De seguida, quando a luminária seguinte detetar um objeto, é imediatamente colocada em *ON*. O sensor da luminária anterior, não detetando um objeto, permanece *ON* por alguns instantes, passando para *OFF* passados esses instantes. O sistema comporta-se sempre de maneira idêntica para todas as outras luminárias.

Foi produzida uma maquete para se testar esta implementação. Para luminárias foram utilizadas lâmpadas.

Depois de implementado e testado o funcionamento deste sistema de controlo na maquete construída, os autores concluíram que os resultados foram os esperados. Quando um sensor fotoelétrico deteta um objeto, a lâmpada associada a esse sensor fica *ON*, enquanto as outras ficam *OFF*.

2.2. *Machine Learning*

Com os desenvolvimentos feitos até à data, as áreas da eletrónica e de *Machine Learning* começaram também a desenvolver as suas opções de iluminação.

Machine Learning (*ML*) é a ciência que se foca na aprendizagem de um computador para que este se comporte tal como um humano, aperfeiçoando o seu conhecimento autonomamente, recorrendo à informação recebida da interação com o mundo real [17].

Esta tecnologia tem dominado todo o mercado onde é preciso gerir e extrair dados em grande número [18]. Sem este desenvolvimento, seria impossível gerir tal quantidade de dados por uma pessoa ou mesmo grupos de pessoas devido à complexidade dos padrões que têm de ser detetados. Com ele podemos melhorar o desempenho ou mesmo fazer previsões em qualquer área tecnológica [19].

Sem darmos por isso, estamos rodeados por esta tecnologia [18]: em toda a publicidade digital, onde recebemos os produtos ou atividades relacionadas com os nossos gostos pessoais; na deteção de fraudes nas transações de cartões de crédito; na deteção facial e de voz nos *smartphones*; nos serviços de *streaming* e áudio como *Netflix* e *Spotify* que utilizam um algoritmo semelhante ao utilizado pela publicidade digital, para que a aplicação nos sugira filmes ou músicas relacionadas com os gostos pessoais.

Nos dias que decorrem, a linguagem de programação *Python* é a mais popular para estas tarefas de *ML*, levando a uma vasta coleção de bibliotecas para a implementação de *Machine Learning* reduzindo assim o tempo de desenvolvimento desses algoritmos [20] [21].

Um programador utilizando estas bibliotecas poupa tempo na implementação, e ainda consegue utilizar certos comandos para o seu auxílio. Na Figura 2.7 encontram-se algumas das razões que justificam o uso desta linguagem de programação para *ML*.

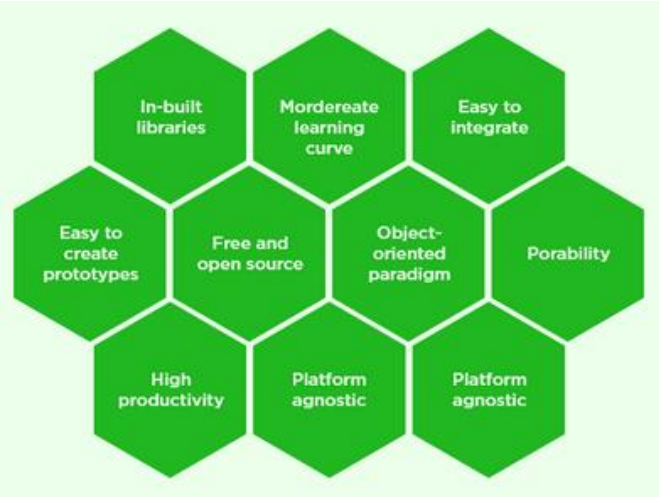

Figura 2.7 - Razões para escolher *Python* para *ML* [21]

As melhores bibliotecas de *Python* para *Machine Learning* são:

- *TensorFlow*

- *Keras*
- *Theano*
- *Scikit-learn*
- *PyTorch*
- *NumPy*
- *Pandas*
- *Seaborn*

2.3. Plataformas de *Hardware* de microcontroladores

Para o controlo deste sistema é necessário um microcontrolador capaz de comunicar com todas as luminárias presentes, de modo a processar toda a informação recebida pelas luminárias. Após a receção e tratamento da informação, o microcontrolador comanda o funcionamento de cada luminária.

Podemos considerar vários tipos de microcontroladores:

- *Arduino*
- *Raspberry Pi*
- *PIC Microcontroller*

2.3.1. *Arduino*

O *Arduino* é uma plataforma eletrónica composta por um microcontrolador. Tem a sua própria linguagem de programação e o seu próprio *IDE*. Esta tecnologia permite fazer programas interativos, mesmo para iniciantes na implementação de algoritmos. Para além de ser uma tecnologia muito versátil e acessível a novos utilizadores, tem flexibilidade para desenvolver projetos mais complexos, sendo assim utilizado por empresas para desenvolver protótipos [22].

As placas de *Arduino* [23] são capazes de ler entradas (*inputs*), tais como sensores, botões de pressão, câmaras e até mesmo mensagens de uma rede social e processá-los em forma de saída (*output*), tais como ativar um alarme, acender um *LED* ou produzir a imagem captada por uma câmara.

Todo o *software* deste sistema é aberto a qualquer utilizador, sendo apenas preciso adquirir a placa de *Arduino* que possibilita a ligação do *software* com todo o *hardware*.

O *software* deste sistema tem várias bibliotecas já desenvolvidas pelos fabricantes desta tecnologia. O *hardware* que se aplica a este sistema é também muito vasto, desde sensores, *SD card* (*Secure Digital Card*), *RFID* (*Radio Frequency Identification*), entre muitos outros.

Com a utilização de *hardware* extra, desenvolvido pelos fabricantes, é possível o *Arduino* comunicar com a internet.

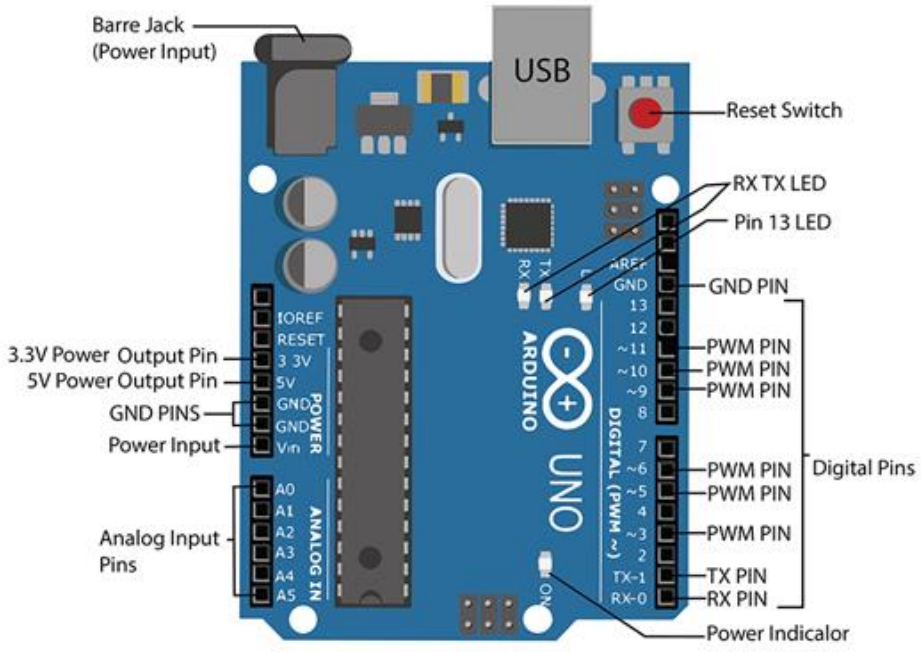

Figura 2.8 - Placa de *Arduino*

Na Figura 2.8 podemos observar a composição de uma placa de Arduíno que é constituída por: 6 *GPIO* analógicos e 14 digitais, *Pi*ns de alimentação (para alimentar circuitos auxiliares ou periféricos, por exemplo), *power input*, entrada *USB* (*Universal Serial Bus*) (apenas input), *LEDs RX* e *TX* (indicando quando o *Arduíno* está a receber ou partilhar informação com outras entidades) e botão de *Reset*.

Com todas estas especificações, o uso do *Arduino* na indústria não é economicamente sustentável, devido à quantidade de tecnologia inserida na placa. As placas de Arduíno geralmente ocupam uma grande área o que impossibilita a conceção de pequenos circuitos controladores.

2.3.2. *Raspberry Pi*

O *Raspberry Pi* é um computador de baixo custo, composto apenas por uma placa do tamanho de um cartão bancário. É possível conectar periféricos ao *Raspberry Pi*, nomeadamente monitores, teclados e ratos para monitorizar uma tarefa ou aplicação [24].

Pode ser usado por utilizadores que queiram começar os seus primeiros algoritmos ou por aqueles já familiarizadas com a implementação de algoritmos.

O *Pi* é muito utilizado para controlo de sistemas, como por exemplo correr um website (desde que não tenha muitos usuários), uma pequena base de dados, um emulador de jogos digitais e muitas outras tarefas que se poderia fazer com um computador [25].

O valor de cada placa *Raspberry Pi* depende da tecnologia inserida em cada modelo. Quanto mais poderosa for a placa, mais dispendiosa será.

Depois de adquirida a placa, todo o software é de livre acesso, nomeadamente o seu sistema operativo principal, o *Raspbian* assim como *Linux*.

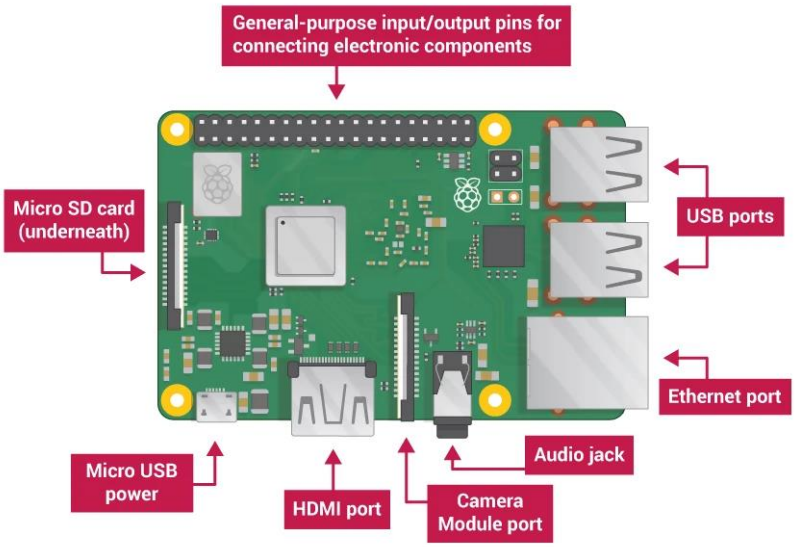

Figura 2.9 - Placa *Raspberry Pi*

Na imagem da Figura 2.9 pode-se observar a constituição de uma placa *Raspberry Pi*. Esta placa é constituída por: uma porta *Ethernet*, possibilitando o acesso direto com a internet; porta HDMI *(High Definition Multimedia Interface*), entrada para *SD card*, e *audio jack*; quarenta *GPIO's*, todos digitais; entradas *USB* para a conexão de periféricos e uma entrada micro USB para alimentação.

Ao comparar esta placa com a do *Arduino* verifica-se que contém mais elementos.

O *Pi* tem incorporado um processador e uma memória *RAM* (*Random Access Memory*) o que leva a possibilidade de controlar vários programas ao mesmo tempo.

É de salientar que o *Raspberry Pi*, embora poderoso e com grande capacidade de processamento, não substitui um computador pessoal, nunca sendo possível alcançar as suas velocidades de processamento e a qualidade de ligação *Ethernet*.

À semelhança do Arduíno, o *Pi* não é a melhor opção para a indústria devido ao seu custo por placa.

2.3.3. *PIC Microcontroller*

Os microcontroladores *PIC* (*Programmable Interface Controllers*) são muito populares pois têm uma fácil programação e conexão com outros dispositivos ou periféricos, sendo de baixo custo e com capacidade para controlar várias tarefas.

São muito utilizados na indústria devido ao seu baixo consumo de energia, elevado desempenho, facilidade de aquisição (devido ao vasto mercado associado a estes microcontroladores) e uma vasta disponibilidade das ferramentas de *hardware* e *software* associados tais como com*pi*ladores e simuladores, totalmente gratuitos [26]. Estão presentes na maioria dos dispositivos eletrónicos, desde os sistemas de alarmes, telemóveis, computadores e periféricos, entre outros [27].

É uma tecnologia que tem estado em constante desenvolvimento, devido à sua elevada procura. Na Figura 2.10 podemos observar a evolução desta tecnologia nos últimos anos.

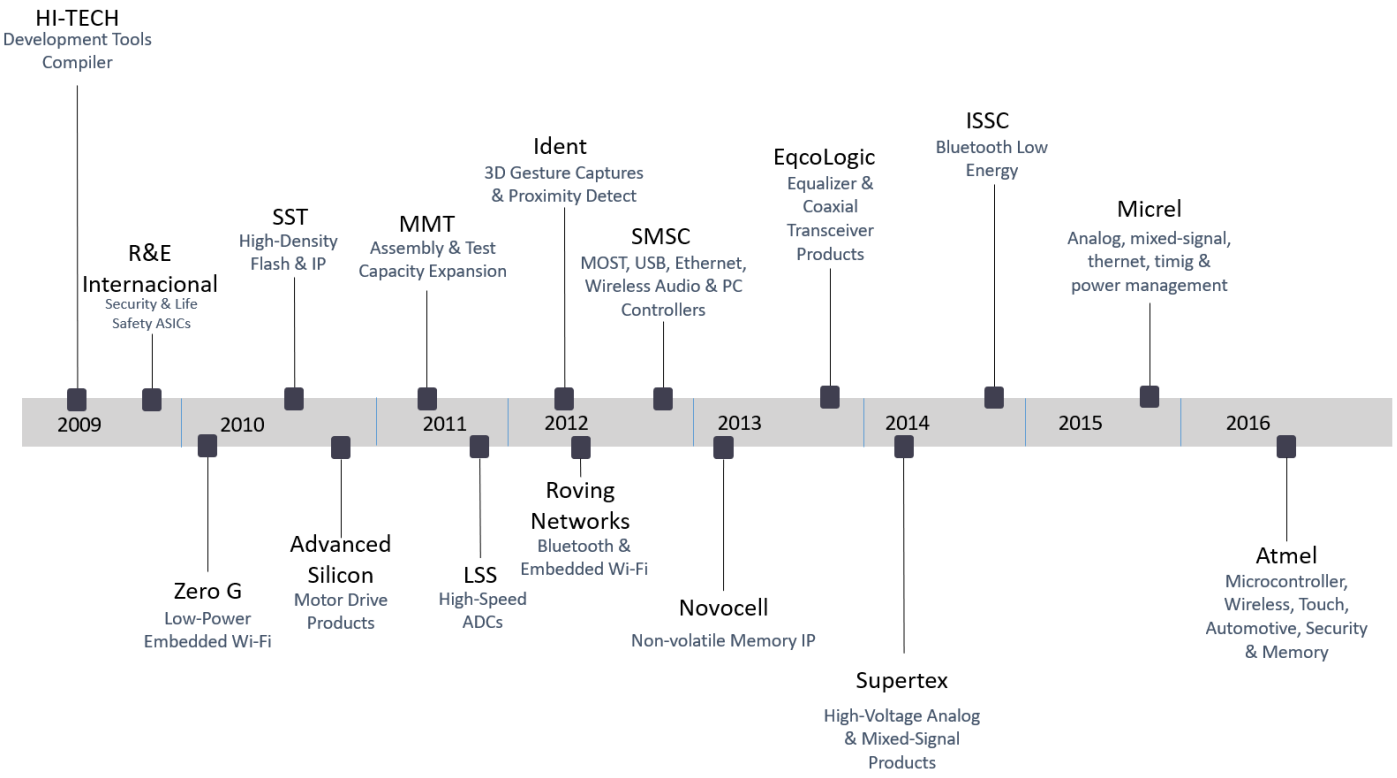

Figura 2.10 - Evolução da tecnologia dos *PIC microcontrollers*

O *PIC* tem de ser previamente escolhido de acordo com a implementação desejada, considerando o tipo de tarefas que vai operar e o número de operações a efetuar. Operando desta maneira, o programador irá escolher o *PIC* que mais se adequa à sua implementação, de modo a minimizar custos e tempo.

2.3.4. Comparação das Tecnologias

Tabela 2.2 - Comparação de Controladores

	Arduino	*PIC* (microcontrolador)	*Raspberry Pi*
Preço Base ($)	30	2/10	35
Consumo	~50mA @ 5V	~20mA @ 5V	~3A @ 5V
Principal aplicação	*Hardware*	*Hardware*	*Software*
Número de I/O *Pi*ns	14 Digitais, 6 Analógicos	[5,28] digitais e [5, 14] analógicos	8 digitais, 0 analógicos
Periféricos	Não	Não	Sim
Internet	Com *Shield*	Com *Shield*	Yes

De acordo com a Tabela 2.2, construída a partir da informação disponível em [28], e a solução descrita nesta dissertação verifica-se que o controlo das luminárias será mais eficiente utilizando um *Raspberry Pi*. Nesta utilização cada luminária terá um microcontrolador incorporado que será responsável pelo controlo da deteção de um individuo e pela comunicação desta informação com o servidor principal (*Raspberry Pi*).

A escolha desta opção de controlador foi baseada nas especificações de todos os controladores aqui descritos. Devido à necessidade de várias luminárias num local, o microcontrolador é a opção mais económica, pois se se colocasse um *Pi* ou um *Arduino* para esta função, a solução não seria economicamente sustentável. O microcontrolador tem tecnologia suficiente para a comunicação com o servidor, para o controlo da luminária e para o controlo da deteção de pessoas.

Para o controlo de todas as luminárias, o *Pi* foi o escolhido devido à sua capacidade de processamento e de gerir várias tarefas em simultâneo. Controlar todas a luminárias com um *Arduino* ou com um microcontrolador poderia ter problemas de implementação devido à falta de processamento e de memória.

No Capítulo 3 será indicado o microcontrolador escolhido para este projeto.

2.4. *PWM*

O *Pulse Width Modulation* (*PWM*) é um método para gerar um sinal analógico utilizando um sinal digital. O comportamento de um sinal *PWM* é definido por dois principais parâmetros: *duty cycle* e frequência.

O duty cycle é o período de tempo em que o sinal *PWM* está ativo. Se o sinal estiver sempre *ON* o *duty cycle* é de 100% e se estiver sempre *OFF* o *duty cycle* é de 0%. A frequência determina a velocidade em que o sinal demora a dar um período [29] [30].

$$duty\ cycle = 100 * \frac{Turn\ ON\ time}{Turn\ ON\ time + Turn\ OFF\ time} \quad (4)$$

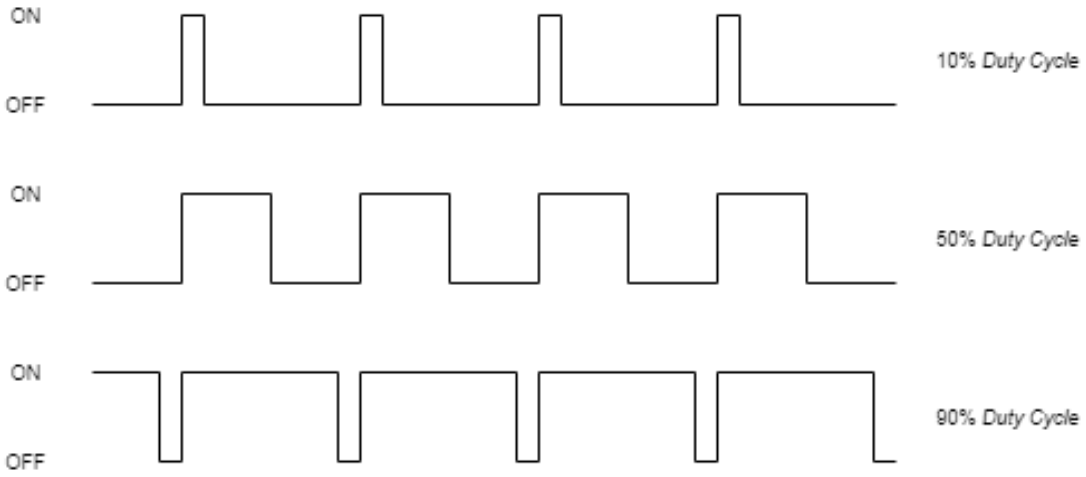

Figura 2.11 - Onda do sinal variando *duty cycle* [31]

Alternando este sinal entre *ON* e *OFF* com a velocidade correta e com um determinado *duty cycle*, o output comportar-se-á como uma tensão analógica quando estiver a alimentar outros elementos. Por exemplo, para criar um sinal de 3V a partir de um sinal digital , que tanto pode estar *ON* (5V) ou (0V), utilizando um sinal *PWM* com um *duty cycle* de 50%, o output será 5V (cinquenta por cento do tempo), de acordo com a Figura 2.11. Se o sinal digital percorrer cada período rapidamente (frequência suficiente) e se o *OFF* digital for de 0V (valor normal), a tensão média de saída pode ser calculada por $5V * 50\% = 2.5V$. Utilizando um *duty cycle* de 80% a tensão de saída seria de 4V, e assim sucessivamente [29].

Este método é muito utilizado no controlo de iluminação de *LED*. Uma vez que a tensão dos *pins* de saída dos controladores é geralmente 5V, para ajustar a intensidade luminosa dos *LEDs* é utilizado um *PWM*. Podemos perceber melhor a relação entre um *PWM* e um *LED* em [32]. A quantidade de corrente conduzida num *LED* é o que define a intensidade de iluminação desse *LED*: quanto maior a corrente, maior a intensidade de iluminação. À semelhança da equação 4, a corrente também se relaciona com o *duty cycle*:

$$I_{avg} = I_{max} * (duty\ cycle)\% \quad (5)$$

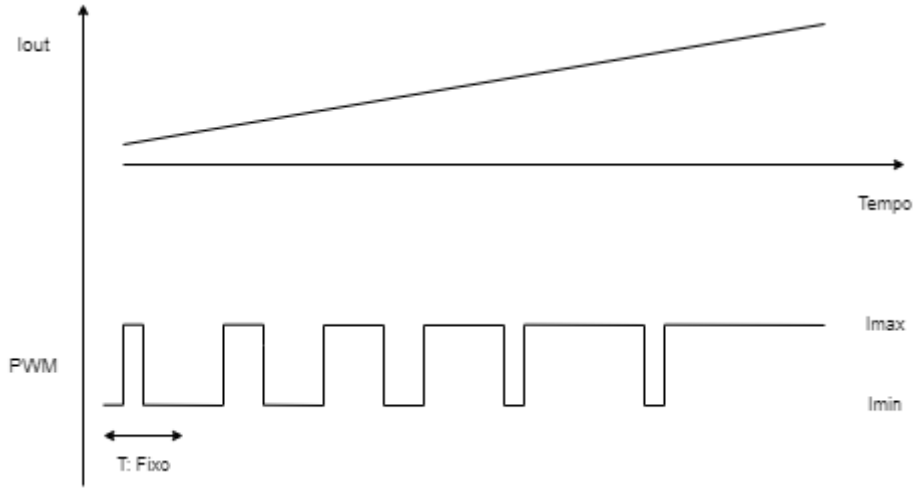

Figura 2.12 - Comparação da variação do *duty cycle* com corrente [32]

Na Figura 2.12 podemos observar como a variação do *duty cycle* está relacionado com a corrente e, consequentemente, com a intensidade luminosa do *LED*.

Como para esta solução, é necessário ajustar esta intensidade de iluminação, será utilizado um *PWM* para controlar esse parâmetro.

2.5. Comunicações *Wireless*

Nesta secção será feita uma explicação das várias comunicações sem fios estudadas para a conceção desde sistema inteligente. Foram então estudadas as seguintes comunicações *wireless*:

- *Bluetooth*
- *Wi-Fi*
- *ZigBee*

2.5.1. *Bluetooth*

O *Bluetooth* [33] é a comunicação *wireless* mais utilizada para ligações indoor e de curta distância devido à simples conexão entre dispositivos. A distância máxima para este tipo de ligação é de 10m. Esta tecnologia está presente em todos os computadores, *smartphones*, ta*ble*ts entre outros dispositivos móveis. O *Bluetooth* é utilizado, por exemplo, na ligação do telemóvel com o carro, ligação de periféricos a um computador e/ou telemóveis (rato, teclado, auscultadores, aparelhagens etc.). O *Bluetooth* é baseado no protocolo *IEEE* (*Institute of Electrical and Electronics Engineers*) 802.15.1.

Com os desenvolvimentos desta tecnologia, o *Bluetooth* foi igualmente progredindo até termos o *Bluetooth* 4.0. Pode ser também chamado de *Bluetooth Low Energy* (*BLE*) [34]. Este tipo de tecnologia traz a possibilidade de as ligações operarem a uma menor potência. Ao contrário do *Bluetooth* simples, o *BLE* opera em modo de repouso, exceto quanto uma ligação é estabelecida. A conexão entre dispositivos é também mais rápida.

2.5.2. *Wi-Fi*

O *Wi-Fi*, baseado no protocolo *IEEE* 802.11, é uma comunicação *wireless* bastante versátil e de fácil utilização [35]. Consegue ser usada tanto em longas como em curtas distâncias. Este tipo de comunicação encontra-se em rede com um router ou um outro *host*, o que permite então a conexão de dispositivos com a internet. Tem um alcance de aproximadamente 100m em espaço aberto.

O *Wi-Fi* também permite a ligação entre dispositivos, não se focando apenas nas ligações com a internet. Esta ligação entre dispositivos que foi desenvolvida pela *Wi-Fi Alliance* denomina-se *Wi-Fi Direct* que é responsável por ligações *P2P* (*peer to peer*). Inicialmente esta ligação *wireless* não era utilizada devido à falta de eficiência na ligação, levando a um uso de energia não sustentável [36].

Com o constante desenvolvimento deste tipo de comunicação surgiu, então, o *Wi-Fi* 4G onde as velocidades de troca de informação e as latências têm melhorado substancialmente com este modelo mais atual. Permitiram também que as ligações *P2P* se implementassem de forma correta e sustentável [36] [37].

2.5.3. *ZigBee*

A comunicação *ZigBee*, baseada no modelo *IEEE* 802.15.4, é um tipo de ligação *wireless* simples e de fácil implementação à semelhança do *Bluetooth*. É muito versátil e pode ser utilizada em inúmeras tecnologias.

Este tipo de comunicação foca-se principalmente na área de controlo de *smart homes* e *smart buildings*, permitindo uma fácil implementação em todos os sensores necessários para este tipo de edifícios. Uma especificação desta tecnologia, é que cada módulo de *ZigBee* é repetidor de sinal de outro módulo, isto é, os módulos *ZigBee* vão comunicando entre eles, sendo possível então um controlador adquirir informação de um módulo, mesmo que este não se encontre no alcance do módulo [38].

Apesar de todos estes benefícios do *ZigBee*, este tipo de comunicação não é suportado por dispositivos móveis e tem também uma baixa taxa de sinal máxima, o que não permite transmitir muita informação de uma só vez. A tecnologia *ZigBee* precisa também de *gateways* para a sua implementação em controladores, o que leva a uma maior dificuldade de a colocar em funcionamento e um custo mais elevado.

2.5.4. Comparação de tecnologias

	DISTÂNCIA MÁXIMA	DADOS TRANSMITIDOS	FREQUÊNCIA DE BANDA	CONSUMO	SUPORTADO POR DISPOSITIVO MÓVEL
BLUETOOTH	10m	1Mb/s	2.4GHz	Médio/Baixo	Sim
BLUETOOTH LOW ENERGY	100m	2Mb/s	2.4GHz	Baixo	Sim
ZIGBEE	50-500m	250-500Kb/s	2.4GHz	Baixo	Não
WI-FI	100m	11-54Mb/s	2.4/5GHz	Alto	Sim
***WI-FI* 4G**	250m	600Mb/s	2.4GHz	Alto	Sim

Tabela 2.3 - Comparação tecnologias *wireless*

Tendo em conta a informação disponível em [39] e em [40] foi construída a Tabela 2.3. Considerando esta informação e todas as tecnologias *wireless* descritas anteriormente, para esta implementação foi escolhida: a tecnologia *Bluetooth Low Energy* para a comunicação entre o utilizador e o controlo do sistema e o *Wi-Fi* para a comunicação entre luminária e o controlador.

Dando foco à ligação utilizador-sistema, o *Wi-Fi* é a tecnologia que tem maior poder de transferência de dados, maior distância máxima de ligação e é suportado por todos os dispositivos móveis. Apesar destes benefícios o seu consumo é elevado.

O trabalho aqui desenvolvido não necessita de elevado número de transferência de dados, nem de uma distância superior a 5m pelo que o *Wi-Fi* foi descartado.

O *ZigBee* tem um nível de consumo baixo e é muito utilizado em ambientes de comunicação entre microcontroladores e sensores, no entanto, o número de dados transmitidos é baixo e não é uma tecnologia presente nos dispositivos móveis, o que impossibilita o funcionamento ideia inicial.

O *BLE* apresenta todas as especificações para a implementação. A distância de cobertura e o número de dados são aceitáveis e é uma tecnologia presente em todos os dispositivos móveis. É também uma ligação de curto alcance segura e de fácil acesso.

Com o *BLE* é possível saber que dispositivo móvel foi detetado pelo sistema, através do seu endereço *Bluetooth*. Sendo os dispostitos móveis instrumentos pessoais, é possível identificar que pessoa é detetada pelo sistema. Ainda considerando o estudo feito por [41], através dos valores de *RSSI* (*Received signal strength indicator*) e endereço *MAC* (*Media Access Control*) é possível aceder à localização da

pessoa. Com ambos os conceitos aplicados ao trabalho aqui proposto, é possível saber então a posição e a identidade de um individuo num determinado espaço.

Na comunicação entre a luminária e o controlador, foi escolhido o *Wi-Fi* pela sua vasta disponibilidade, fácil implementação e com maior velocidade de troca de dados comparado com o *Bluetooth* e o *ZigBee*. O *ZigBee*, para além disso, precisaria de *gateways* para ser possível esta ligação, o que levaria a uma implementação mais complexa e mais dispendiosa.

2.6. *Smart Cities*

A ideia base para o desenvolvimento das *smart cities* é melhorar a qualidade dos serviços oferecidos aos cidadãos, reduzindo os custos de administração pública [42]. São capazes de obter informação do mundo real, através de sensores, medidores, aparelhos, dispositivos pessoais, entre outros. A cidade está constantemente conectada, partilhando essa informação captada pelos sensores com os vários serviços da cidade. São capazes também de incluir serviços complexos de análise, otimização e visualização para tomar melhores decisões operacionais [43].

Nestes sensores e aparelhos que têm a capacidade de obter informação do ambiente que os rodeia está também presente um módulo *IoT* (*Internet of Things*) que possibilita obter essas informações via internet.

No entanto, as *smart cities* ainda estão a ser desenvolvidas, devido à complexidade da sua implementação. Estima-se que nesta década seja possível implementar uma cidade inteligente totalmente operacional. No âmbito destas cidades autónomas, a iluminação inteligente é a tecnologia que é mais fácil de implementar com os desenvolvimentos feitos até à data [3].

3. Trabalho Desenvolvido

Neste capítulo é apresentada a descrição do trabalho desenvolvido para este projeto o qual foi dividido em quatro pontos:

1. Uma síntese do sistema desenvolvido nesta Dissertação;
2. Protocolo de comunicação utilizado entre os sensores e o controlador;
3. O módulo deteção de dispositivos *BLE* (sensores);
4. O módulo controlador do sistema (controlador).

3.1. Sistema inteligente de iluminação

A solução proposta para este sistema inteligente, encontra-se aqui descrita. Na Figura 3.1 podemos observar o controlador a comunicar com todas as luminárias este receciona a informação detetada por cada luminária e decide como processar.

Como apresentado no capítulo anterior, o controlador escolhido para este projeto foi o *Raspberry Pi*. Esta placa de desenvolvimento tem a capacidade de controlar vários sinais em simultâneo e suporta todas as tecnologias utilizadas neste projeto. O *Pi* comunica com os sensores através do protocolo de comunicação *MQTT* (*Message Queuing Telemetry Transport*) (3.2). O *Pi* tem também acesso às informações dos utilizadores que se inscreveram na aplicação, nomeadamente, o nome da pessoa, o *MAC address* do dispositivo *Bluetooth* com que se inscreveu e o tipo de luz que prefere.

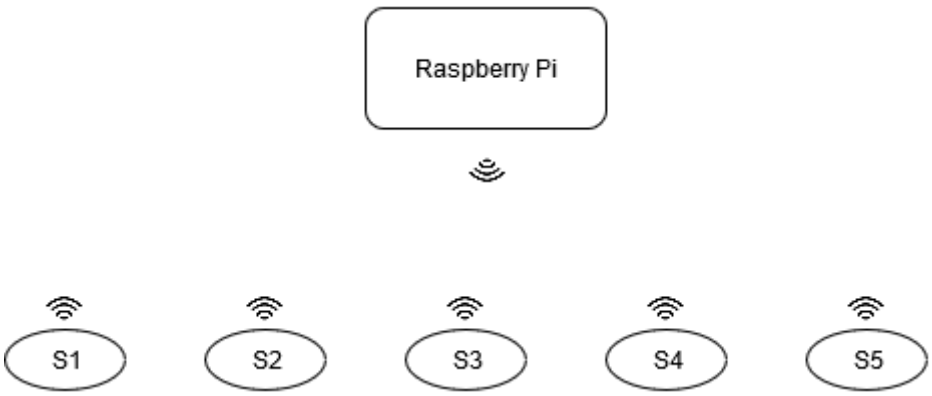

Figura 3.1 - Controlador e luminárias

Na Figura 3.2 podemos observar os componentes da luminária.

Foi indicado anteriormente também, que cada luminária teria um microcontrolador para ser possível fazer a medição e envio da informação detetada pela luminária, para esta ser enviada ao *Pi*. Foi então escolhido um dispositivo que contenha tanto um microcontrolador capaz de controlar sensores e capaz de fazer trocas de informação por *Wi-Fi*, dispositivo este detalhado no ponto 3.3 deste Capítulo.

O sensor escolhido para este projeto deteta e recebe a informação de que dispositivos *Bluetooth* estão por perto, e os seus respetivos valores de *MAC address* e *RSSI*. O sensor envia esta informação ao controlador principal por *MQTT*. O controlador, comparando o endereço *MAC* com a sua lista de endereços conhecidos, informa então como se deve comportar essa luminária.

A luminária mais próxima ao utilizador (se este for reconhecido pelo sistema) encontrar-se-á com 100% de intensidade luminosa, enquanto que as restantes, vão diminuindo gradualmente a sua intensidade luminosa (daí a necessidade de sinais *PWM*), de acordo com a sua distância ao utilizador.

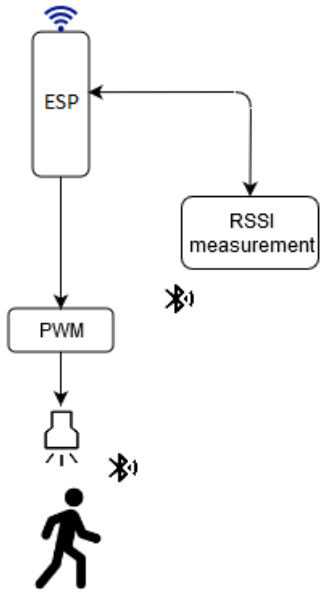

Figura 3.2 - Composição Luminária

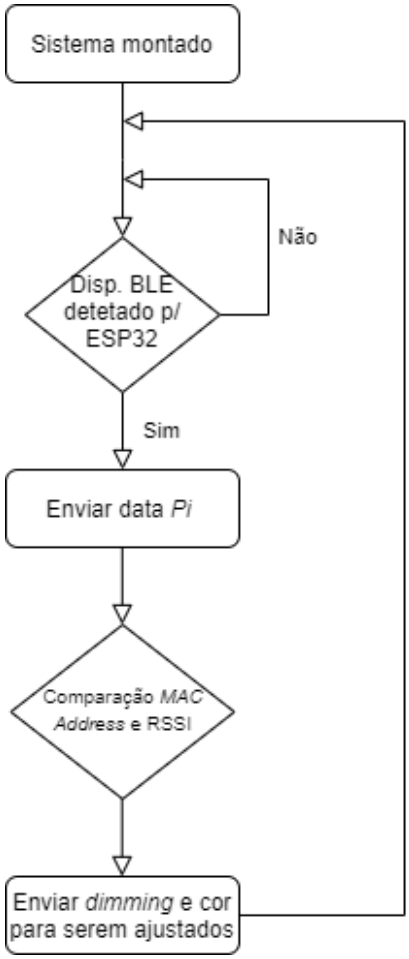

Figura 3.3 - Diagrama de Fluxo do sistema

Foi construído o Diagrama de Fluxo presente na Figura 3.3 para mais fácil compreensão do algoritmo a desenvolver em ambas as partes do controlo do sistema.

Os *ESP-32* detetam dispositivos *BLE* nas suas redondezas que se forem detetados enviam essa informação ao *Pi*.

O *Pi* recebendo esses valores, analisa o *MAC address* do dispositivo detetado que se for um conhecido do sistema, é enviado o ajuste de *dimming* para a luminária.

Foi prevista uma montagem teste, mais tarde implementada para efetuar testes. Podemos observar essa montagem na Figura 3.4. O sensor encontra-se numa *breadboard*, com um *LED* ligado ao *PIN* 32 através de uma resistência para limitar a corrente.

Podemos ver também a presença do *Pi*, para comunicar com o sensor por *MQTT*.

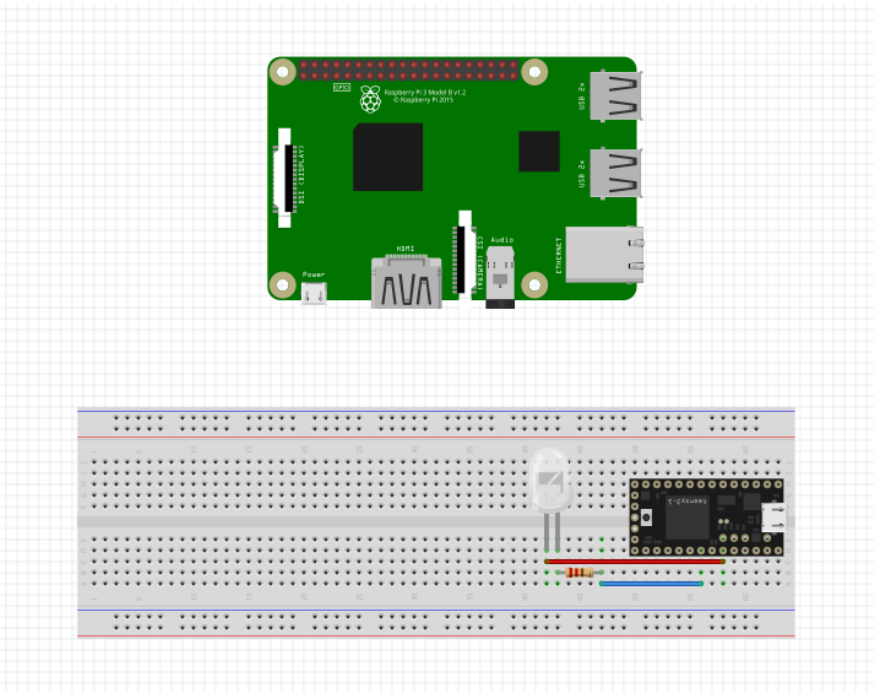

Figura 3.4 - Esquema da montagem-teste

3.2. Protocolo de comunicação *MQTT*

Foram utilizados protocolos de comunicação de modo que a informação recebida pelos sensores, fosse enviada para o controlador principal. Foi então utilizado o protocolo de comunicação *MQTT*.

Como referido anteriormente, a troca de informações entre os sensores (*ESP-32*) e o controlador (*Raspberry Pi*) através do *Wi-Fi* sendo necessário um protocolo de transporte de dados. O protocolo será o *MQTT* (*Message Queuing Telemetry Transport*) adequado a microcontroladores como o *ESP-32* e *Raspberry Pi* que têm *Wi-Fi* incorporado [44].

Um sistema de comunicação *MQTT* contém três diferentes componentes com distintas funções:

1. *Publisher* que gera e envia dados para o *MQTT broker*. O *Publisher* pode ser qualquer dispositivo capaz de medir presença de pessoas e enviar um sinal para o *broker;*
2. *Broker* que funciona como um *server* para receber as mensagens de todos os *publishers*, guardam essas mensagens e distribuem-na pelos *subscribers*;
3. *Subscriber* é o componente que subscreve a certo tipo de mensagens vindas dos *publishers*. Um *subscriber* pode ser um simples painel a assinalar a presença de pessoas.

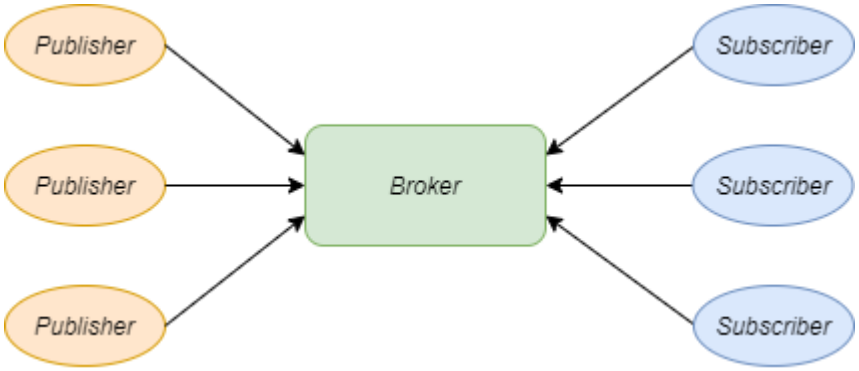

Figura 3.5 - Sistematização Protocolo *MQTT* [44]

Na Figura 3.5 pode-se observar graficamente como funciona este protocolo. Os *MQTT publishers* partilham informação com o *MQTT broker* e os *MQTT subscribers* vão buscar essa informação ao *broker*.

Nesta implementação, o *Raspberry Pi* servirá como broker, sendo responsável pela receção das mensagens vindas dos *publishers* (quem envia) para os *subscribers* (quem recebe).

Numa primeira instância, os nossos *ESP-32* que estão a funcionar como sensores de *RSSI* serão os nossos publishers, e o *Raspberry Pi* será também o nosso *subscriber*, uma vez que o controlador tem de receber todos os valores medidos pelos *ESP-32*. Nos próximos subcapítulos é explicado todo o algoritmo de publishers e *subscribers*.

3.3. Rede de sensores

Para a implementação deste sistema de controlo, a tecnologia escolhida para a deteção de presença e posição será o *Bluetooth* e a sua capacidade de medir os valores de *RSSI* (*Received signal strength indicator*). Para este efeito foram utilizados módulos *ESP-32* [45].

Figura 3.6 - Módulo *ESP-32* [45]

Cada módulo *ESP-32* presente na Figura 3.6 é responsável pela deteção de aparelhos *Bluetooth*, pela leitura do valor *RSSI* e endereço *MAC* entre esses aparelhos. Este dispositivo é programado de diversas maneiras.

Para este projeto, foi escolhido trabalhar com este módulo sobre o ambiente *Arduino IDE*, devido à sua versatilidade e vasta informação disponível nas várias utilizações possíveis.

Este ambiente de trabalho tem uma ampla rede de bibliotecas incluídas e de utilização ilimitada, incluindo bibliotecas de *Bluetooth* e *Bluetooth Low Energy* que serão bastante úteis para o algoritmo desenvolvido.

Para o desenvolvimento desta dissertação, foi realizada uma montagem teste com os componentes descritos anteriormente.

Resumindo o comportamento do sistema assenta nos seguintes passos:

1. O *ESP-32* deteta a presença de sinais *Bluetooth* nas proximidades e deteta os seus *MAC address* e *RSSI*;
2. Estas informações são enviadas para o *Pi* por *MQTT* para este ler o endereço *MAC* e *RSSI* medidos pelo sensor. Decide então, como o sistema vai operar segundo tais condições;
3. Depois de processada a informação, o *Raspberry Pi* envia para o *ESP-32* o *dimming* a que este deve acender o *LED* de acordo com o valor de *RSSI* medido;
4. O *ESP-32* recebe a informação do *Pi* e atualiza o estado do LED de acordo com o recebido.

3.3.1. Algoritmo medição *RSSI*

Na Figura 3.7 podemos observar o algoritmo desenvolvido, para receber o melhor valor de *RSSI* medido por um dispositivo.

Inicializamos o módulo *BLE* do *ESP-32* com a instrução BLEDevice::init("") e de seguida procura-se todos os aparelhos *Bluetooth* na proximidade.

Depois de detetados todos os dispositivos, o algoritmo desenvolvido recebe o melhor valor de *RSSI* medido por esse *ESP-32*.

```
void loop() {
  int rssi = 0;
  BLEDevice::init("");
  BLEScan *scan = BLEDevice::getScan();
  scan->setActiveScan(true);
  BLEScanResults results = scan->start(1);
  int best = CUTOFF;
  for (int i = 0; i < results.getCount(); i++) {
    BLEAdvertisedDevice device = results.getDevice(i);
    rssi = device.getRSSI();
    if (rssi > best) {
      best = rssi;
    }
  }
```

Figura 3.7 - Deteção de aparelho e medição de *RSSI*

Como explicado anteriormente, o controlador terá de receber esta informação para monitorização do sistema.

Resta ainda fazer a deteção da pessoa presente no sistema obtendo o endereço *MAC* do dispositivo presente. Mais tarde nesta implementação irá ser descrito todo este processo.

3.3.2. *MQTT Publisher*

Para o *MQTT Publisher* serão precisos o *SSID* (*Service Set Identifier*), a *password* da rede *Wi-Fi* a utilizar bem como o *IP* (*Internet Protocol*) do dispositivo que servirá de *broker*, que no presente trabalho será o *Raspberry Pi*. Estas informações estão indicadas na Figura **3.8**.

```
//SSID/Password combination
const char* ssid = "NOS-C78C";
const char* password = "3MJM5QJZ";

//MQTT Broker IP address:

const char* mqtt_server = "192.168.1.23";
```

Figura 3.8 - Informações *Wi-Fi* e *broker*

O *Arduino IDE*, tem também inúmeras bibliotecas de comunicações *MQTT*, todas livres de utilização.

Na Figura 3.9 e Figura 3.10 poderemos observar as inicializações feitas para esta comunicação ser efetuada com sucesso.

```
void setup() {
  Serial.begin(115200);

  BLEDevice::init("");
  delay(500);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

  pinMode(ledPin, OUTPUT);
}
```

Figura 3.9 - Inicialização *MQTT*

```
void setup_wifi() {
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

Figura 3.10 - Conexão ao *Wi-Fi*

Realizadas as inicializações anteriores, é necessário fazer a conexão e a troca de informação entre os dois pontos de comunicação.

No *loop* principal do algoritmo controlador, mostrado na Figura 3.11, começamos por rever a conexão com o *client*. O sistema fica nesta instrução até ser conectado com sucesso ao *client*. Enquanto esta conexão não for estabelecida, é corrida a função *reconnect*, mostrada na Figura 3.12.

Depois de ter estabelecido esta ligação com sucesso, o *ESP-32* irá realizar o algoritmo de medição de *RSSI*, explicado anteriormente. Por fim, comunica ao *MQTT subscriber* os valores de *RSSI* medido nas proximidades.

O *ESP-32* comporta-se assim indefinidamente, partilhando sempre com o controlador todos os valores medidos.

```
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  delay(500);

    BLEScan *scan = BLEDevice::getScan();
    scan->setActiveScan(true);
    BLEScanResults results = scan->start(1);
    for (int i = 0; i < results.getCount(); i++) {
      BLEAdvertisedDevice device = results.getDevice(i);
      rssi = device.getRSSI();
    }
    // Convert the value to a char array
    char tempString[8];
    dtostrf(rssi, 1, 2, tempString);
    Serial.print("RSSI: ");
    Serial.println(tempString);
    client.publish("esp32/RSSI", tempString);
    rssi = 0 ;
}
```

Figura 3.11 - Função *loop()*

```
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Subscribe
      client.subscribe("esp32/output");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Figura 3.12 - Função *reconnect()*

Tendo todo o *firmware* descrito anteriormente, temos então o nosso *ESP-32* pronto para medir valores de *RSSI* e mandar esta informação para o *MQTT broker* para trocar mensagens com o *Pi*.

Resta então configurar o *broker* e o *subscriber* para ter a comunicação implementada.

3.3.3. Algoritmo de *dimming*

O *ESP-32* é ainda responsável por todo o *dimming* da iluminação, de acordo com a informação recebida. Foi então desenvolvido um algoritmo, mostrado na Figura 3.13 para atuar sobre essa iluminação, baseado no conceito do *PWM* anteriormente explicado.

```
void pwm_ramp(int valor_a_chegar){
  int x = 0;
  if (dutyCycle != valor_a_chegar){
    if (dutyCycle < valor_a_chegar){
      for(x = dutyCycle; x <= valor_a_chegar; x++){
      // changing the LED brightness with PWM
        ledcWrite(ledChannel, x);
        delay(5);
      }
    }
    else {
      for(x = dutyCycle; x >= valor_a_chegar; x--){
      // changing the LED brightness with PWM
        ledcWrite(ledChannel, x);
        delay(5);
      }
    }
    dutyCycle = valor_a_chegar ;
    Serial.print("DUTY CYCLE: ");
    Serial.println(dutyCycle);
  }
  else
    ledcWrite(ledChannel, valor_a_chegar);

}
```

Figura 3.13 - Algoritmo de variação de *dimming*

3.4. Controlador do Sistema

Neste subcapítulo é explicada a implementação executada para fazer o controlo deste sistema. Foi utilizado o *Raspberry Pi*, como referido anteriormente para esta tarefa, e numa primeira instância é importante perceber o processo de toda a receção e envio de informação com os sensores.

3.4.1. *MQTT Broker*

O *Raspberry Pi*, enquanto broker, será responsável por coletar todas as informações enviadas pelos publishers e distribuí-las para os *subscriber*.

Foi instalada uma biblioteca *MQTT* no *Pi* para este ter a habilidade de utilizar o protocolo *MQTT*. Uma vez instaladas e executadas as tarefas, foram realizados testes de funcionamento. No Capítulo 4, apresenta-se a análise desses testes.

3.4.2. *MQTT Subscriber*

Como referido anteriormente, teremos o *Raspberry Pi* a funcionar como *subscriber que* será responsável por receber a informação proveniente dos *ESP-32*.

Na Figura 3.14 e Figura 3.15 podemos observar o código *Python* (Foi utilizado o *Thonny IDE*) para o nosso controlador. Este será responsável por:

- Subscrever ao tópico onde o *ESP-32* publicou;
- Processar a informação recebida;
- Enviar ao *ESP-32* informação sobre o *dimming* a utilizar.

Analisando o código podemos então observar os seguintes parâmetros mais relevantes:

- Importação das bibliotecas de *MQTT* em *Python*;
- O endereço do *broker* a que deve ir buscar a informação (semelhante ao IP do *Pi*);
- O tópico a que se deseja subscrever (para conseguir obter a informação vinda dos sensores)
- Função *on_message*: sempre que há uma nova mensagem no tópico que desejamos ler, esta função é corrida, tendo como função ler essa mensagem, e de acordo com o valor lido, deve publicar no *MQTT Broker* o *dimming* adequado;

```
29
30
31  ######################################################
32
33  broker_address = "192.168.1.24"
34  print("creating new instance")
35  client = mqtt.Client("P1-")
36  client.on_message=on_message
37  print("connecting to broker")
38  client.connect(broker_address)
39  client.loop_start()
40  print("Subscribing to topic ", "esp/RSSI")
41  client.subscribe(MQTT_TOPIC)
42  #client.publish("esp32/output", "on")
43  #time.sleep(4)
44  #client.loop_stop()
45
```

```
Shell
Python 3.7.3 (/usr/bin/python3)
>>>
```

Figura 3.14 - Função inicialização

```
New  Load  Save  Run  Debug  Over  Into  Out  Stop  Zoom  Quit  Switch to regular mode
dissertacao_controller.py *
1  import paho.mqtt.client as mqtt
2  import time
3  MQTT_TOPIC= 'esp/RSSI'
4  message_received = "-50"
5  message_topic= "0"
6  zero = "0.00"
7  ###############
8  def on_message(client, userdata, message):
9      global message_received
10     global message_topic
11
12     message_received = str(message.payload.decode("utf-8"))
13     message_topic = message.topic
```

Figura 3.15 - *MQTT* função *on_message*

O algoritmo presente na Figura 3.16 introduz então todo o tratamento de dados efetuado pelo controlador.

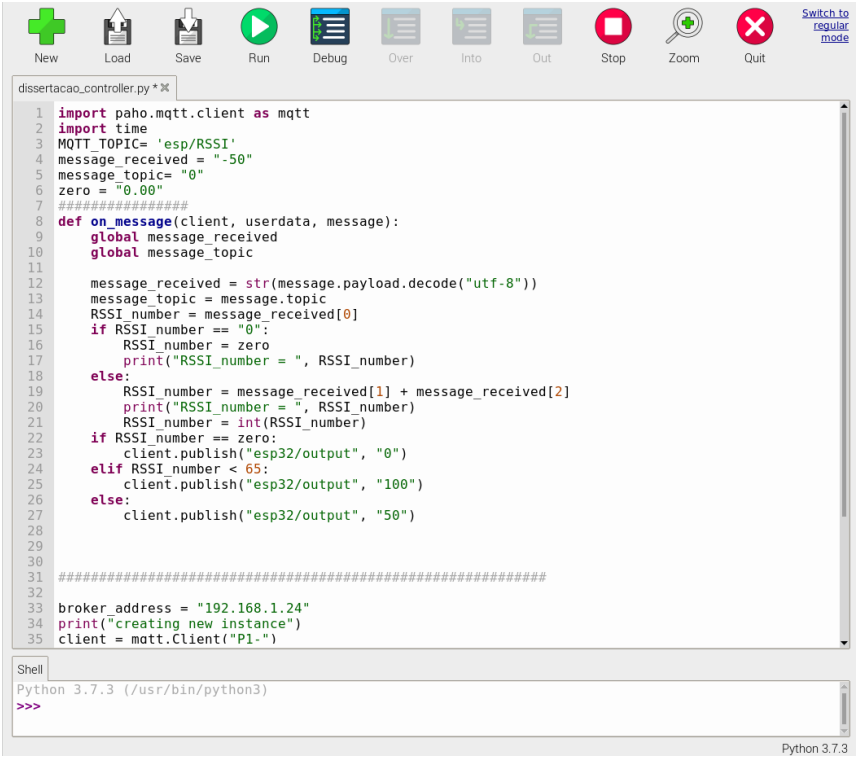

Figura 3.16 - Tratamento de dados

O valor *RSSI* recebido é passado para valor absoluto, para se poder comparar mais facilmente. Assumimos numa primeira instância que valores inferiores a sessenta e cinco (65) são considerados próximos do sensor, logo é enviado o valor de *dimming* máximo (100%). Acima deste valor é enviada a instrução para 50% de *dimming*. Se detetado 0, é enviada a instrução de *dimming* 0 (luminária desligada).

O sensor recebe esta informação e atua sobre o *LED* de acordo com a decisão do controlador.

Os resultados desta implementação podem ser observados no capítulo seguinte.

Uma vez implementados os algoritmos anteriores para a medição de *RSSI* resta então relacionar esse *RSSI* ao *MAC address* do dispositivo que emite esse sinal.

Foi acrescentada uma instrução (Figura 3.17) de leitura do endereço MAC do dispositivo detetado no *ESP-32*, de modo que o endereço seja enviado juntamente com o *RSSI* quando existir troca de informação entre o *ESP-32* e o *Pi*.

```
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  delay(10);

    BLEScan *scan = BLEDevice::getScan();
    scan->setActiveScan(true);
    BLEScanResults results = scan->start(1);
    for (int i = 0; i < results.getCount(); i++) {
      BLEAdvertisedDevice device = results.getDevice(i);
      rssi = device.getRSSI();
      mac_address = device.getAddress().toString().c_str();
     /* if (rssi > melhor_rssi){
      char tempString[8];
      dtostrf(rssi, 1, 2, tempString);
      strcpy(mqtt_buf, mac_address);
      strcat(mqtt_buf,tempString);

      //msg_mqtt = mac_address + tempString;
     // Serial.print("RSSI: ");
      Serial.println(mqtt_buf);
     // Serial.println(tempString);
      client.publish("esp/RSSI/ESP_1", mqtt_buf);
    }

    // Convert the value to a char array
    /*char tempString[8];
    melhor_rssi = -99;
    rssi = 99 ;
}
```

Figura 3.17 - Função *loop()* final

Assim que é detetado um dispositivo BLE é enviado por *MQTT* o endereço *MAC* juntamente com o valor do *RSSI*. Esta mensagem contém então o seguinte formato: yy:yy:cc:dd:ff:gg-xx, em que "yy:yy:cc:dd:ff:gg" representa o endereço MAC e "-xx" o valor de *RSSI* medido.

Sabendo a composição desta mensagem, no lado do controlador é realizada a sua interpretação.

Sendo uma mensagem válida (com a estrutura indicada) é então processado o endereço numa primeira instância, pois o sistema só atua para endereços MAC identificados. Se for um endereço conhecido, o sistema opera de acordo com o *RSSI* medido, como explicado anteriormente.

Com as implementações e testes nesta fase do sistema foi encontrado um problema, conforme a seguir descrito:

Para detetar a pessoa presente no sistema, utilizam-se bibliotecas do *Arduino* para se conseguir obter o *MAC address* do dispositivo presente.

Tendo uma base de dados com todos os *MAC address* já inscritos, e sendo este um endereço único para todos os dispositivos, é assim possível determinar a pessoa que está presente nas instalações.

A implementação, como explicado anteriormente, baseia-se tanto na posição da pessoa bem como com a identificação da mesma. Todo o sistema foi então pensado para usar o smartphone sendo este um dispositivo com *BLE* e de uso pessoal.

Fazer a medição de *RSSI* de um *smartphone* foi possível, assim como o seu rastreio, mas não a sua identificação. Os fabricantes de *smartphones* sejam estas a *Apple*, *Samsung* entre outras, como medida

de segurança fazem com que os seus dispositivos "emitam" endereços *MAC* sempre aleatórios, para que a identificação destes aparelhos se torne difícil.

Essa identificação de dispositivos é possível utilizando *beacons* desenvolvidos pelos mesmos fabricantes dos *smarphones* permitindo saber a sua localização e identificação.

Particularizando no *iBeacon*, equipamento este que emite um sinal *Bluetooth* detetado pelos nossos dispositivos (*smartphones*, *tablets*, *smart watches*) [46] permitindo então:

1. Monitorizar uma região para detetar a presença de um *iBeacon;*
2. Utilizar o alcance do *beacon* (*smartphone*) para determinar a proximidade do *iBeacon* detetado.

Analisando o estudo em [46] conclui-se que o rastreio deste tipo de aparelhos não é possível utilizando *hardware* e *software* não produzido pelos respetivos fabricantes. Foi então necessário arranjar uma alternativa na identificação de pessoas utilizando *ESP-32*.

Com estas variantes, optou-se por arranjar cartões que emitem sinais *BLE*, sempre constantes, possíveis de observar na Figura 3.18. Esses cartões são os *Card Beacon C7*, que são a nova geração de *beacons* incorporando a tecnologia *Bluetooth* 5.0 [47].

Figura 3.18 - *Multi-use Card Beacon C7*

Introduzindo estes *beacons* ao sistema implementado, conseguiu-se uma alternativa, não tão pratica, mas aplicável comparando com os *smartphones*. A leitura dos endereços *MAC* foi sempre constante, sendo possível associar este *Beacon* a um indivíduo em particular.

Perante tais dificuldades, colocou-se a necessidade de uma aplicação para *smartphone*, dado não ser possível fazer a identificação das pessoas pelo *smartphone* utilizando *ESP-32*.

3.5. Aplicação de interface com o utilizador

Como descrito anteriormente, para este projeto será necessário identificar que pessoas estão a utilizar este sistema de controlo de iluminação, de forma que o estilo de cor escolhido se adapte às suas preferências.

Para que esta funcionalidade fosse possível, foi desenvolvida uma aplicação, em que o utilizador irá inserir:

- Nome
- Definir o dispositivo *Bluetooth* a ser detetado
- Cor de luz preferida (mais quente ou mais fria)
- Uma palavra-passe para início de sessão

Depois destas informações serem todas inseridas, a aplicação irá comunicar com o sistema inteligente, para partilhar com este, todos os utilizadores deste sistema, de modo a que este se comporte da maneira desejada.

3.5.1. Aplicação Android

O utilizador terá então que fazer uma conta para utilizar este sistema de controlo.

Na Figura 3.19 podemos observar a primeira página que o utilizador encontra ao adquirir esta aplicação. Será necessário introduzir o seu nome de utilizador, e a senha, para que o seu dispositivo se prepare para comunicar com o sistema inteligente.

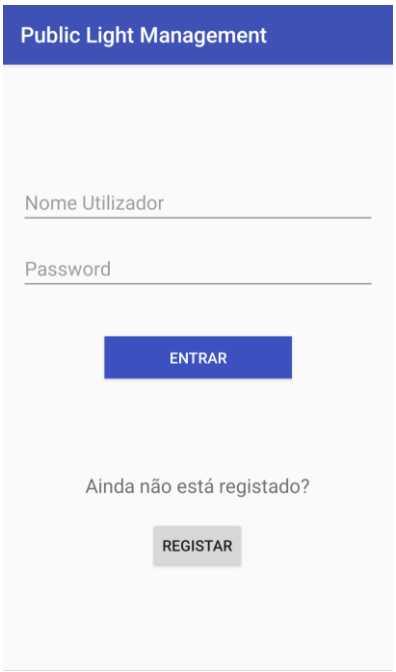

Figura 3.19 - Página Inicial Aplicação

Caso o utilizador ainda não tenha uma conta, terá a possibilidade de o fazer. Para isso, terá de selecionar o botão "Registar" para que carregue os seus dados pessoais. Na Figura 3.20 podemos ver então a janela da aplicação, onde os dados do utilizador são inseridos. Estes dados e ainda o *MAC address* serão adicionados a uma base de dados, partilhada com o sistema controlador para que este obtenha as informações de todos os utilizadores do sistema.

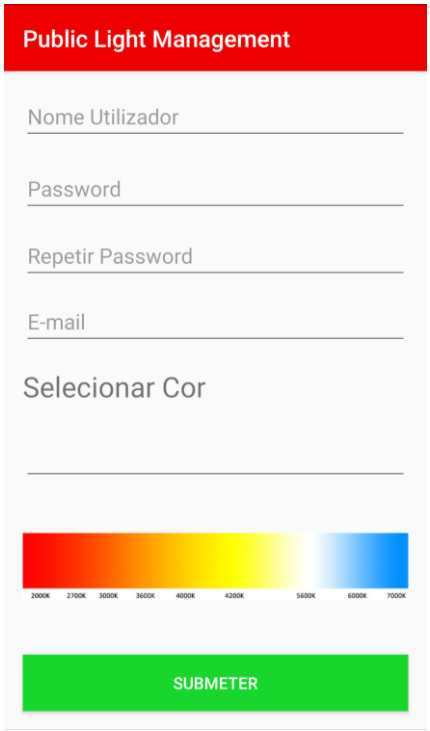

Figura 3.20 - Página de Registo

Como explicado anteriormente, não foi possível a identificação utilizando *smartphones*. A aplicação de interface foi assim concluída, sem nunca comunicar com o sistema controlador uma vez não ter aplicação prática para este projeto.

4. Testes e validações

Neste capítulo será abordado tudo o que é referente a testes e validações efetuadas à implementação deste sistema.

Foi também construída uma maquete final, com o sistema ligado a uma luminária alimentada por um driver *AC/DC* (*Alternated Current*/*Direct Current*) de 24V.

4.1. Implementação da medição de valores de *RSSI*

Foi implementado o código descrito no capítulo anterior num *ESP-32* para testar se a deteção de dispositivos *Bluetooth* era bem-sucedida e se essa informação era enviada para o *Raspberry Pi* via *MQTT*.

Numa primeira instância, ocorreu um erro no *upload* do *firmware* ao *ESP-32*. O mesmo consistia na falta de memória que o *ESP-32* apresenta, como podemos observar na Figura 4.1.

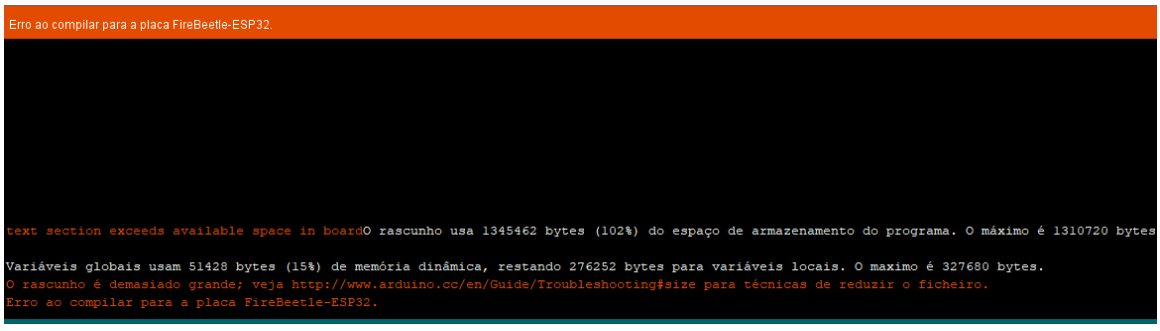

Figura 4.1 - Erro falta de memória dinâmica

Foi possível arranjar solução para este erro: o *ESP-32* tem parte da memória que não é possível aceder quando esta placa vem de fábrica. Mudando algumas definições e especificações na placa e no *Arduino IDE* foi então possível aceder a esse espaço de memória e carregar o código no microcontrolador.

Uma vez ultrapassado o erro anterior, foi possível observar resultados positivos desta implementação. Analisando os seguintes dados:

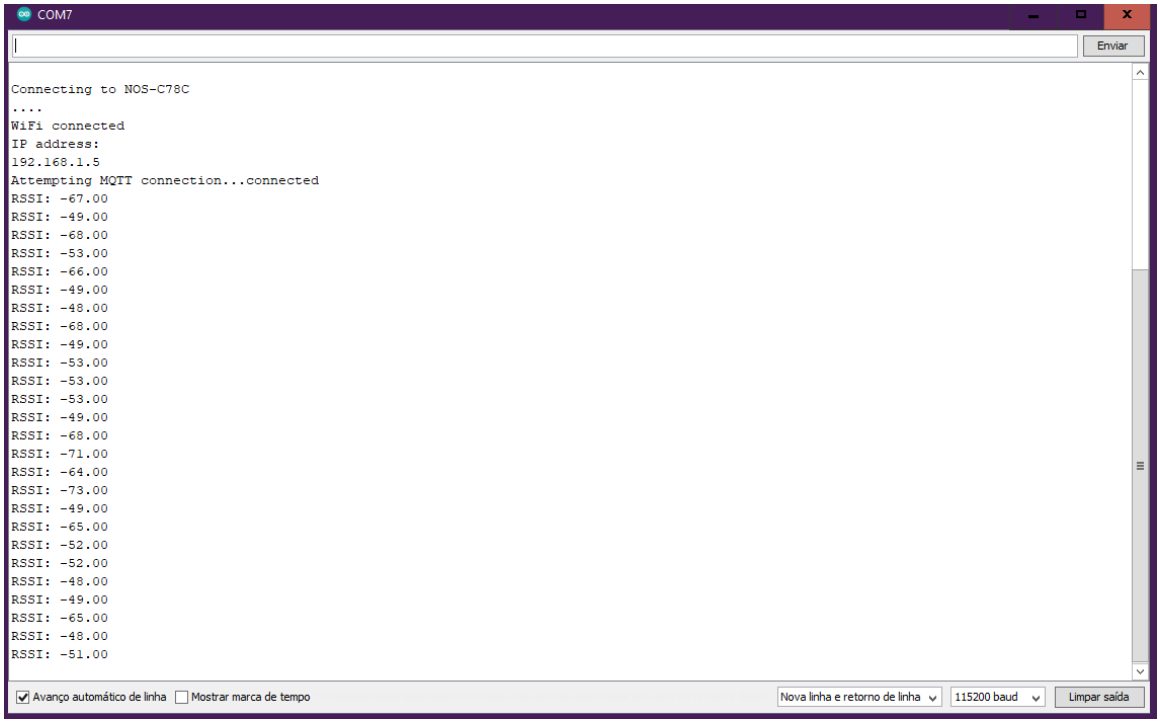

Figura 4.2 - Resultado *ESP-32*

Na Figura 4.2, é possível observar os vários passos dados pelo *firmware* implementado no *ESP-32*:

- Em "*Connecting* to *NOS-C78C*" o *ESP-32* tenta-se conectar ao *Wi-Fi* para ser possível a troca de mensagens por *MQTT*.
- Assim que a ligação ao *Wi-Fi* esteja estabelecida "*Wi-Fi connected*" existe a tentativa de conexão *MQTT* "*Attempting MQTT connection…connected*".
- Respeitando os passos anteriores, é então iniciada a medição de *RSSI* dos dispositivos *Bluetooth* próximos.

Analisando os resultados anteriores, podemos observar que a medição de *RSSI* de dispositivos *Bluetooth* está a ser feita, pois existe de facto um dispositivo *Bluetooth* próximo do sensor (*ESP-32*) do trabalho desenvolvido.

Os valores de *RSSI* são sempre negativos e com estes resultados, conclui-se que quanto mais afastado do sensor está o dispositivo a medir, maior o valor absoluto avaliado.

De seguida valida-se se a comunicação *ESP-32*/*Raspberry Pi* está a ser bem-sucedida.

4.2. Conexão *MQTT*

Observando a Figura 4.3, pelo comando "mosquito_sub -t "esp/*RSSI*" " é possível aceder às bibliotecas de *MQTT* previamente instaladas neste *Raspberry Pi* fazendo um *MQTT subscribe* ao tópico "*esp*/*RSSI*" para onde está o *ESP-32* a mandar a informação desejada.

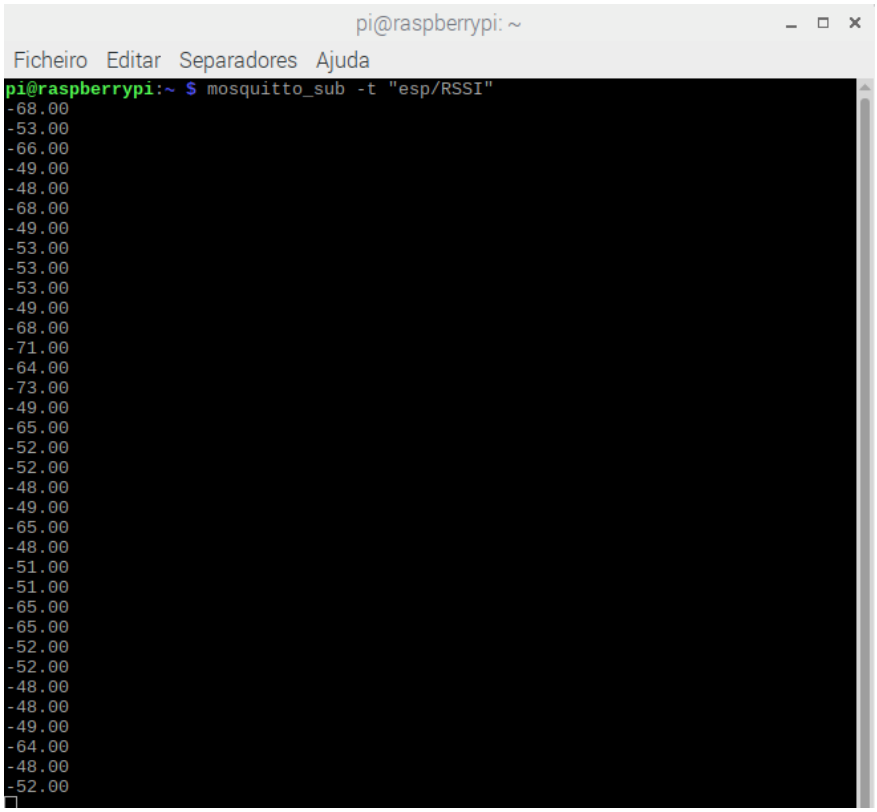

Figura 4.3 - Resultado *Raspberry Pi*

Foram medidos quinze valores consecutivos, para poder haver uma melhor comparação, tanto pelo *ESP-32* como pelo *Pi*, sendo estes tabelados a seguir:

Tabela 4.1 - Comparação de Valores

Valor *ESP-32*	Valor *Raspberry Pi*
-67	N/D
-49	N/D
-68	-68
-53	-53
-66	-66
-49	-49
-48	-48
-68	-68
-49	-49
-53	-53
-53	-53
-53	-53
-49	-49
-68	-68

Na Tabela 4.1 é possível verificar que os valores coincidem (admitindo que N/D deve-se ao facto do *ESP-32* iniciar o processamento de dados antes do *Pi*), uma vez que os valores medidos e apresentados pelo *ESP-32* são equivalentes aos apresentados pelo *Pi*.

Sendo as medições anteriores satisfatórias, foi realizado depois um outro teste: afastar do recetor de *RSSI* para verificar se os valores alteravam e de seguida desligar o *Bluetooth* do meu dispositivo. A análise destes dados é apresentada na Figura 4.4.

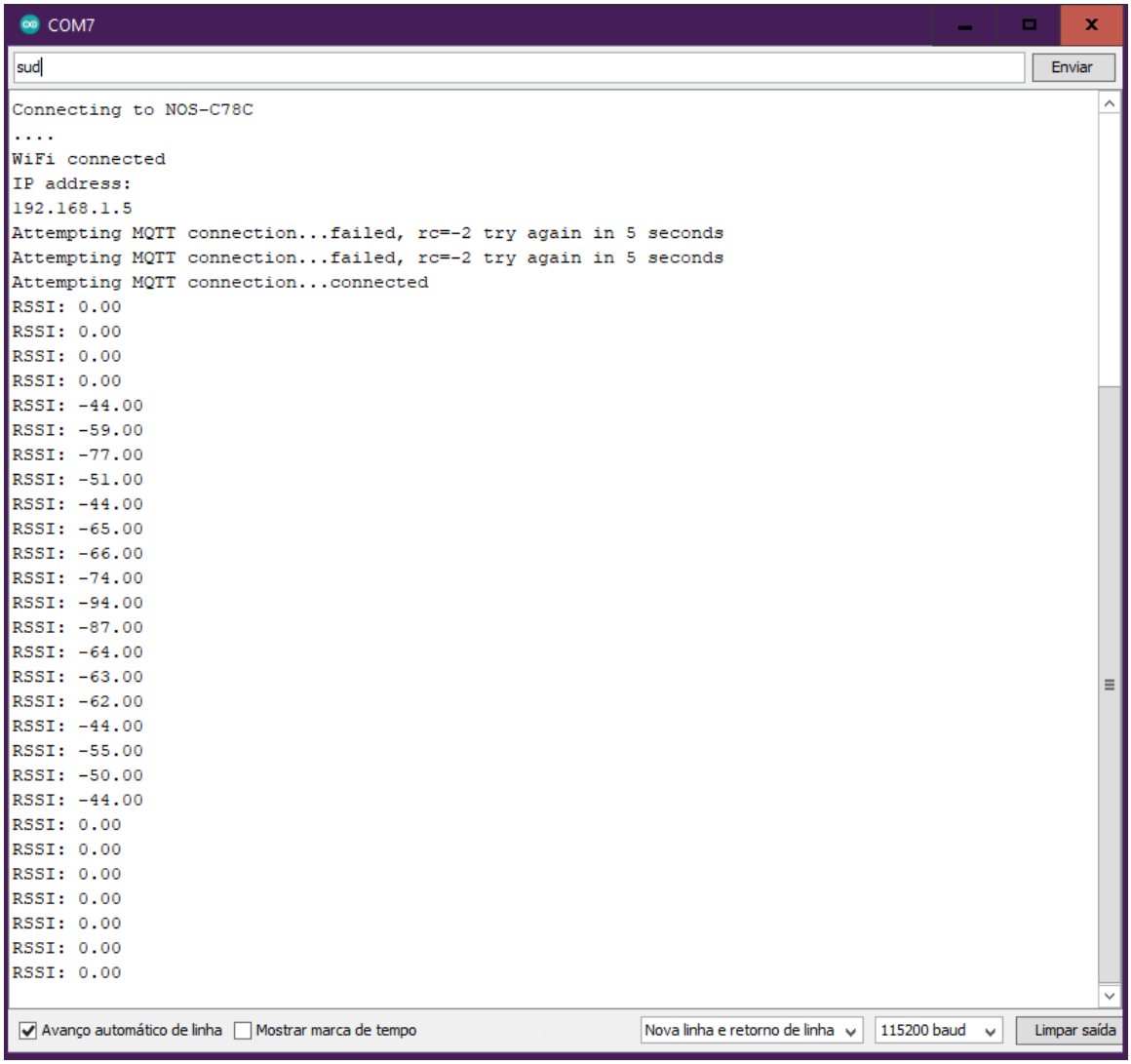

Figura 4.4 - Resultado 2 *ESP-32*

Observando a Figura 4.4 e o *output* nela apresentado verifica-se que apresenta algumas diferenças relativamente ao *output* da Figura 4.2, o que é expectável, pois o procedimento deste teste foi diferente.

À semelhança do resultado anterior, foi estabelecida a conexão com o *Wi-Fi*. Uma vez estabelecida, foi tentada a ligação com o *MQTT* broker, que de início não foi bem sucedida, uma vez que o *broker* ainda não estava ligado. Uma vez ligado, foi estabelecida a conexão.

O dispositivo *Bluetooth* não estava inicialmente ligado, daí aparecerem os resultados de *RSSI* de 0. Assim que foi ligado, os valores de força de sinal do *Bluetooth* começaram a ser medidos.

Neste teste foi-se afastando e aproximando o dispositivo *Bluetooth* do sensor.

Quanto maior o valor absoluto medido pelo *ESP-32* maior a distância a este. Com estes resultados, conclui-se que é possível desenvolver um controlador que obtendo essas medidas de *RSSI*, consiga prever a posição de determinado dispositivo *Bluetooth*, e por sua vez, a do individuo que a esteja a usar.

Vamos agora observar os resultados deste teste no *Raspberry Pi*

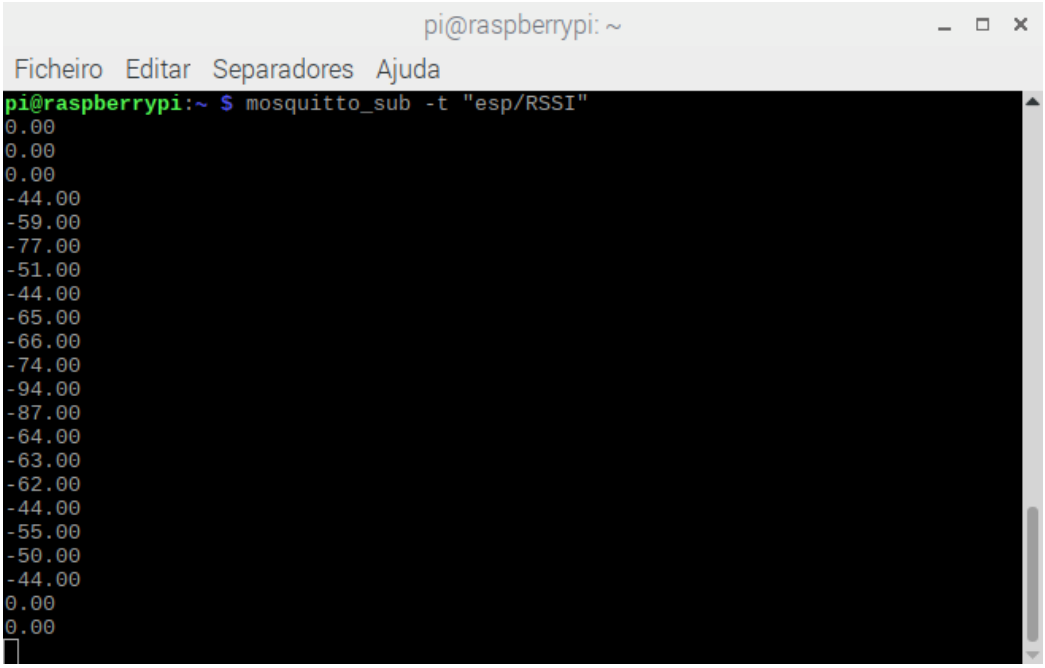

Figura 4.5 - Resultado 2 *Raspberry Pi*

Mais uma vez foram tabelados os valores das Figura 4.4 e Figura 4.5 para melhor análise de resultados.

Tabela 4.2 - Comparação de Valores Resultado 2

Valor *ESP-32*	Valor *Raspberry Pi*
0	N/D
0	0
0	0
0	0
-44	-44
-59	-59
-77	-77
-51	-51
-44	-44
-59	-59
-77	-77
-51	-51
-44	-44
-65	-65
-94	-94
-87	-87
-64	-64
-50	-50
0	0

Mais uma vez, os valores coincidem. Conclui-se assim que a comunicação *MQTT* funciona.

Como referido no capítulo 3.4.2, foi implementado um código *Python* para o *Pi* operar como *subscriber*. De seguida, apresentam-se os testes feitos a esse algoritmo.

Iniciámos então o *broker*, testado anteriormente, e foram implementados tanto o código para a deteção de dispositivos *Bluetooth* como o código *Python* para o controlo.

Os algoritmos foram implementados de forma a que os valores medidos fossem visíveis, para facilitar a veracidade dos resultados.

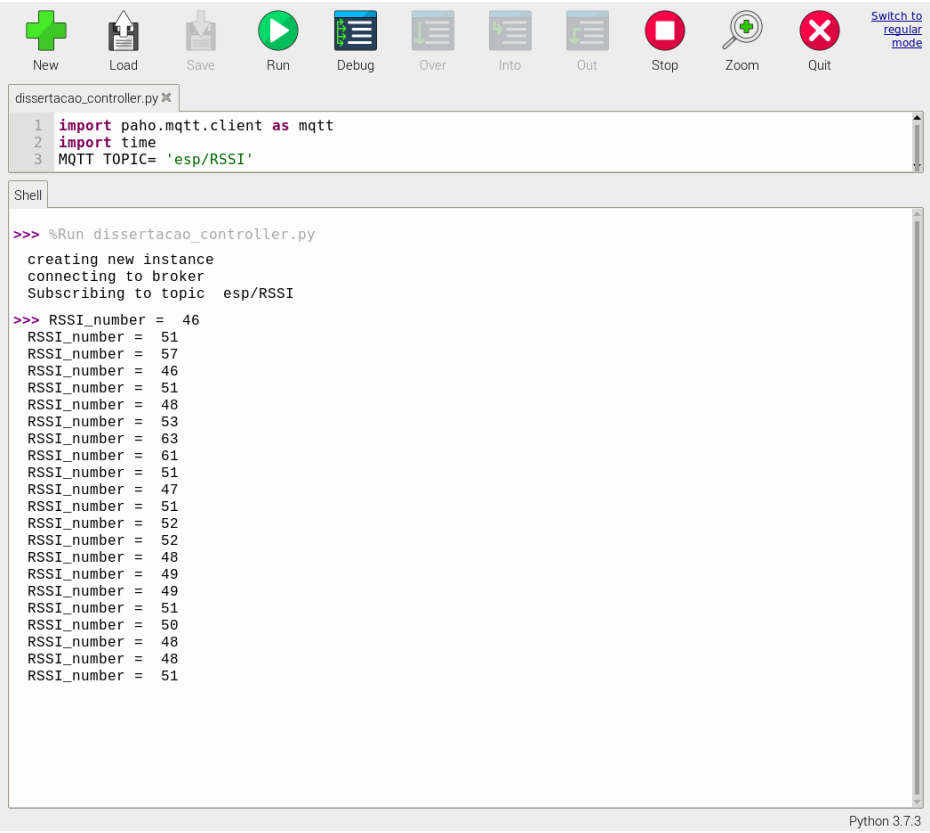

Figura 4.6 - *Output* consola *Thonny IDE*

Na Figura 4.6 é possível observar os valores de *RSSI* recebidos pelo controlador. De acordo com estes valores, o algoritmo presente no controlador envia informação ao *ESP-32* para este fazer o ajuste de *dimming* como é possível de observar na Figura 4.7

Figura 4.7 - *Output ESP-32* à mensagem recebida

De seguida é preciso então tratar da deteção do *MAC address* do dispositivo e enviá-lo igualmente, e o tratamento dessa informação.

Foram ensaiados o *software* de deteção do *MAC address* no *ESP-32*, o de envio desta informação para o controlador e o de receção e tratamento de dados pela parte do controlador.

Analisando a Figura 4.8 percebemos que o controlador está a receber a informação corretamente: a mensagem recebida tem o formato indicado no sub-capítulo 3.4.2.e o tratamento da mensagem está a ser feito da maneira indicada. O endereço *MAC* é conhecido pelo sistema, logo envia a mensagem de *output* para a luminária de acordo com o valor de *RSSI* medido.

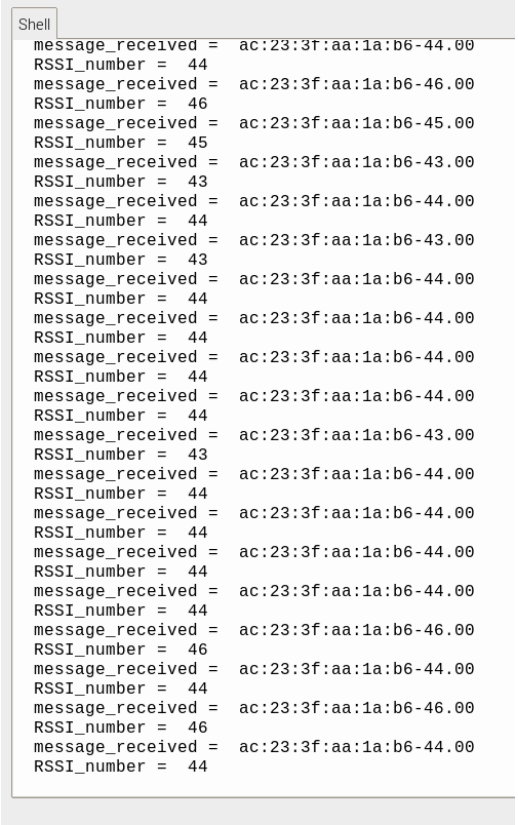

Figura 4.8 - Mensagem recebida no controlador

Com estes resultados satisfatórios foi construída uma montagem final, simulando uma luminária inteligente, controlada pelo sistema aqui descrito.

4.3. Montagem Final

Foram adquiridas três luminárias, presentes na Figura 4.9 para fazer um ensaio real desta implementação. Foi preciso adicionar eletrónica à já existente, para possibilitar fazer o *dimming* destas luminárias. Foi construída a Tabela 4.3 com os principais materiais e custos de cada luminária inteligente.

Tabela 4.3 - Tabela de preços Luminária

MATERIAL	CUSTO
LUMINÁRIA *LED*	30€
ALIMENTADOR LUMINÁRIA	21€
ESP-32	6€

Para além dos materiais indicados anteriormente, foram utilizados também um *MOSFET* (*Metal Oxide Semiconductor Field Effect Transistor*) e uma resistência, de preço desprezável, uma vez serem componentes que comprados em grande escala têm um preço unitário de cêntimos.

É também preciso considerar o preço do controlador. Cada *Raspberry Pi* tem um valor médio de 35€ dependendo da versão adquirida.

Figura 4.9 - Luminárias adquiridas

Foi então adicionado o *MOSFET NMOS* com o sinal *PWM* do *ESP-32* ligado à *gate* do transístor os 24V. Este circuito denomina-se um circuito aditivo de *PWM* pois queremos fazer o *dimming* de uma luminária a 24 V e o sinal *PWM* máximo gerado pelo *ESP-32* (3.3V) não é suficiente para este *dimming*.

É de alertar que este circuito, embora aditivo, inverte o sinal de *PWM* sendo que quando o *PWM* é máximo do lado do *ESP-32*, vai ter um mínimo no *drain* do *MOSFET*. Teve que ser feita então uma alteração no *firmware* do *PWM*, sendo que quando queremos um *dimming* máximo na luminária, é preciso agora gerar um *PWM* mínimo (0V).

Na Figura 4.10 e na Figura 4.11 é possível observar esta eletrónica montada numa *breadboard*. As conexões A, B e C são as necessárias para a alimentação da luminária adquirida, sendo:

- A: 24 V da fonte de alimentação da luminária;
- B: Sinal *PWM* para a luminária;
- C: *ground* da fonte, ligado ao *ground* do *ESP-32.*

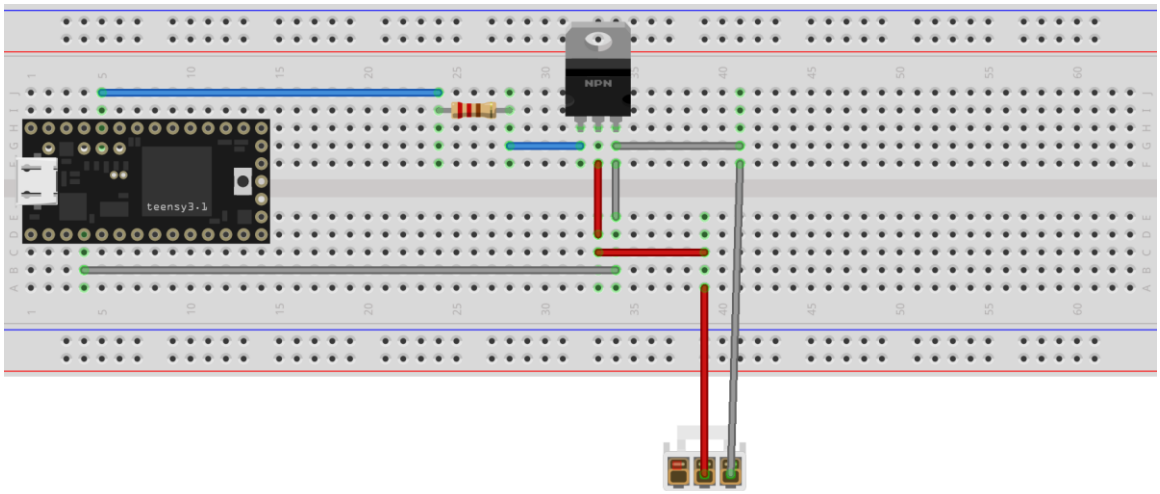

Figura 4.10 - Esquema da *breadboard*

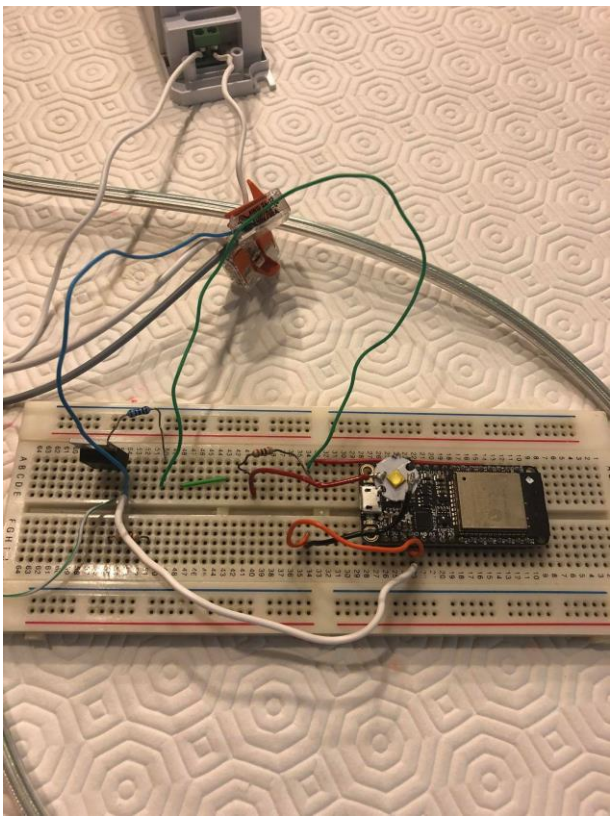

Figura 4.11 - Eletrónica *ESP-32*

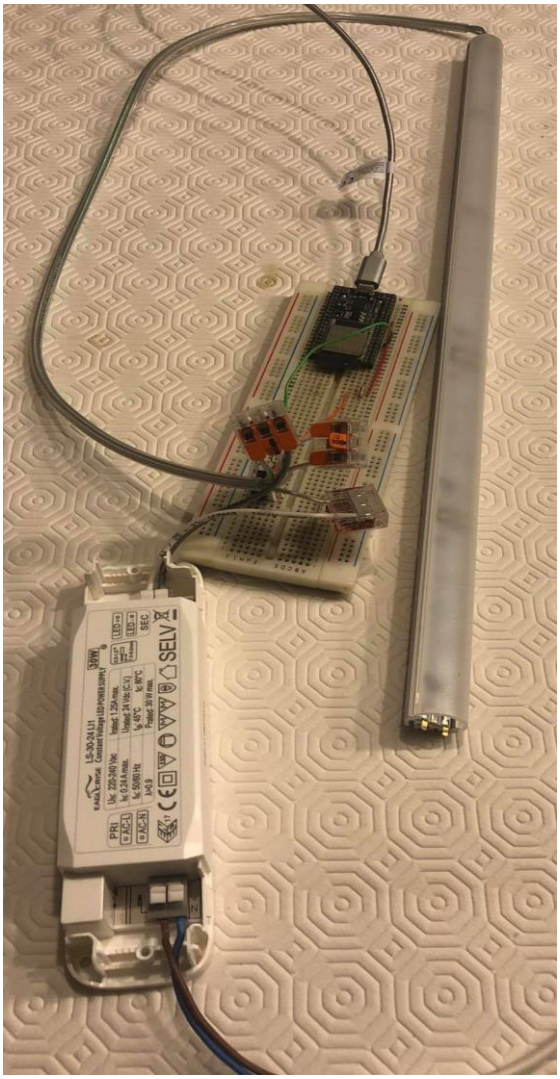

Figura 4.12 - Luminária inteligente

Foram construídas duas luminárias semelhantes, em que foi definido um “endereço” diferente para cada uma, para não existirem problemas de comunicação entre luminárias inteligentes e o controlador. Estas luminárias foram ligadas com uma distância de 20 m entre elas.

Implementando o algoritmo em cada luminária, e colocando o controlador em funcionamento foi então possível ver o sistema a funcionar. Utilizando o cartão *BLE* apresentado anteriormente, sendo este o conhecido do sistema, a luminária mais próxima foi a que teve um *dimming* de 100%, enquanto que a segunda luminária estava com um *output* de 50% da iluminação máxima.

Andando em direção à segunda luminária, ao aproximar-me desta, a sua iluminação mudou o seu *output* para os 100%, sendo esta agora a mais próxima do utilizador. A luminária mais afastada estava com um *output* de 50%.

Afastando-me de ambas as luminárias, estas desligaram-se quando foi impossível de detetar o dispositivo *BLE* conhecido.

Analisando uma das luminárias com a Figura 4.13 e Figura 4.14 é possível de observar o funcionamento anterior.

Em fotografia, o *dimming* de cada luminária pode não ser o mais percetível. Foi deixado o *LED* de teste inicialmente utilizado. Como explicado anteriormente, com o uso do circuito aditivo para o *PWM*, este tem um comportamento inverso, logo analisando as figuras, observando o *LED* presente na *breadboard* é possível de perceber o *output* da luminária. Na Figura 4.13 o *LED* e a luminária estão com um *output* de 50% e na Figura 4.14 o *LED* tem um output de 0% enquanto que a luminária tem um *output* de 100% (inverso ao *LED*).

Figura 4.13 - Luminária com 50% *dimming*

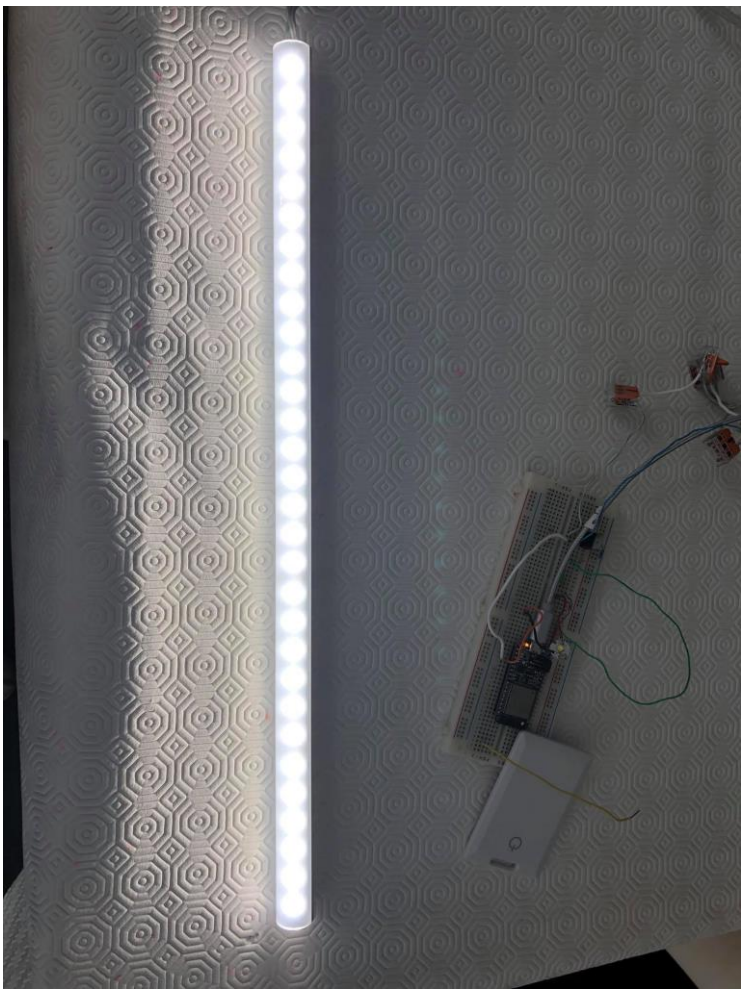

Figura 4.14 - Luminária com 100% *dimming*

Com esta montagem final é demonstrado que este tipo de controlo, sendo afinado e desenvolvido profissionalmente, tem uma aplicabilidade prática, sendo uma solução viável ao modelo de iluminação utilizado nos dias de hoje.

5. Conclusões e trabalho futuro

5.1 Conclusões

Com o trabalho desenvolvido foi possível demonstrar um sistema de iluminação inteligente de conceção exequível. Esta aplicação conduz a uma sociedade com um menor desperdício energético ligado à iluminação e consequentemente custos relacionados com estes, bem como a redução das emissões de CO_2.

A iluminação inteligente poderá ser implementada nas futuras cidades inteligentes, como o modelo a adotar para a iluminação pública e privada.

A solução aqui apresentada será mais indicada para interiores, uma vez ser baseada na tecnologia *Bluetooth* e *Wi-Fi* podendo não ser muito fiáveis em campo aberto. Conseguindo adaptar as luminárias atualmente instaladas para estas inteligentes e conectando a um controlador com métodos de inteligência artificial, como *Machine Learning* é possível então fazer um controlo autónomo da iluminação.

Os testes à implementação e a montagem-teste final aqui demonstrada permitem concluir que o tipo de tecnologia aqui utilizada tem resultados igualmente satisfatórios aos estudados no estado de arte desta dissertação. Foram estudadas várias soluções propostas já existentes bem como toda a tecnologia usada para este tipo de implementação.

Concluindo, esta dissertação ajudou a compreender a necessidade deste tipo de iluminação nos ambientes públicos e que é uma tecnologia pronta a implementar nos dias que decorrem.

5.2 Trabalho futuro

Como trabalho futuro seria apropriado tornar a deteção da pessoa mais prática, isto é, utilizar o seu *smartphone*. Com detetores *BLE* específicos, é possível fazer a deteção e identificação destes *smartphones*. Fazendo uma *app* de inscrição destes dispositivos e associá-los a um utilizador seria mais prático.

Após a aplicação desenvolvida seria necessário existir uma comunicação entre o controlador (*Raspberry Pi*) e uma base de dados que contenha a informação de todos os utilizadores inscritos na aplicação, associando um endereço *MAC* a um nome, bem como a sua temperatura de cor e quantidade de iluminação preferidos.

Bibliografia

[1] "Poluição luminosa: o desperdício inútil de recursos energéticos - Saber - Planetazul – O Portal de Ambiente e Sustentabilidade." [Online]. Available: http://www.planetazul.pt/edicoes1/planetazul/desenvArtigo.aspx?c=2252&a=18635&r=37. [Accessed: 26-Dec-2019].

[2] "Desperdício de energia – Dark Skies Rangers – Portugal." [Online]. Available: http://dsr.nuclio.pt/poluicao-luminosa/desperdicio-energia/. [Accessed: 26-Dec-2019].

[3] "Boosting smart cities with smart street lighting." [Online]. Available: https://www.smartcitylab.com/blog/urban-environment/lighting-the-road-to-smart-cities-and-sustainability/. [Accessed: 05-Jan-2020].

[4] "Society 5.0." [Online]. Available: https://www8.cao.go.jp/cstp/english/society5_0/index.html. [Accessed: 06-Feb-2020].

[5] X. Wang, L. Li, Y. Yuan, P. Ye, and F.-Y. Wang, "ACP-based social computing and parallel intelligence: Societies 5.0 and beyond," *CAAI Trans. Intell. Technol.*, vol. 1, no. 4, pp. 377–393, 2016, doi: 10.1016/j.trit.2016.11.005.

[6] "The Technologies That Turned Barcelona Into A Smart City - Barcinno." [Online]. Available: http://www.barcinno.com/barcelona-smart-city-technologies/. [Accessed: 10-Jan-2020].

[7] "Illuminating cities with sustainable smart lighting systems | Guardian Sustainable Business | The Guardian." [Online]. Available: https://www.theguardian.com/sustainable-business/sustainable-smart-lighting-systems-cities. [Accessed: 26-Dec-2019].

[8] "Heijmans and Philips start work on smart lighting in Eindhoven | Heijmans N.V." [Online]. Available: https://www.heijmans.nl/en/news/heijmans-and-philips-start-work-smart-lighting-eindhoven/. [Accessed: 26-Dec-2019].

[9] M. Srikanth and K. N. Sudhakar, “ZigBee Based Remote Control Automatic Street Light System,” *IJESC*, vol. June 2014, no. June, pp. 639–643, 2014, doi: 10.4010/2014.208.

[10] P. Raynham and I. Lewin, “An examination of the fundamentals of road lighting for pedestrians and drivers,” *Light. Res. Technol.*, vol. 36, no. 4, pp. 307–316, 2004, doi: 10.1191/1365782804li125oa.

[11] N. Davoudian and P. Raynham, “What do pedestrians look at at night?,” *Light. Res. Technol.*, vol. 44, no. 4, pp. 438–448, Dec. 2012, doi: 10.1177/1477153512437157.

[12] T. Fujiyama, C. Childs, D. Boampong, and N. Tyler, “Investigation of Lighting Levels for Pedestrians,” pp. 1–14.

[13] A. Haans and Y. A. W. de Kort, “Light distribution in dynamic street lighting: Two experimental studies on its effects on perceived safety, prospect, concealment, and escape,” *J. Environ. Psychol.*, vol. 32, no. 4, pp. 342–352, 2012, doi: 10.1016/j.jenvp.2012.05.006.

[14] S. P. Lau, G. V. Merrett, A. S. Weddell, and N. M. White, “A traffic-aware street lighting scheme for Smart Cities using autonomous networked sensors,” *Comput. Electr. Eng.*, vol. 45, pp. 192–207, Jul. 2015, doi: 10.1016/j.compeleceng.2015.06.011.

[15] S. P. Lau, G. V. Merrett, A. S. Weddell, and N. M. White, “StreetlightSim: A simulation environment to evaluate networked and adaptive street lighting,” *Proceedings, APWiMob 2014 IEEE Asia Pacific Conf. Wirel. Mob. 2014*, pp. 66–71, 2014, doi: 10.1109/APWiMob.2014.6920267.

[16] D. Sunehra and S. Rajasri, “Automatic street light control system using wireless sensor networks,” *IEEE Int. Conf. Power, Control. Signals Instrum. Eng. ICPCSI 2017*, no. October 2013, pp. 2915–2919, 2018, doi: 10.1109/ICPCSI.2017.8392257.

[17] “What is Machine Learning? | Emerj.” [Online]. Available: https://emerj.com/ai-glossary-terms/what-is-machine-learning/. [Accessed: 04-Jan-2020].

[18] P. Kulkarni, “Understanding machine learning opportunities,” *Intell. Syst. Ref. Libr.*, vol. 128, pp. 23–48, 2017, doi: 10.1007/978-3-319-55312-2_2.

[19] “Foundations of Machine Learning - Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar - Google Livros.” [Online]. Available: https://books.google.pt/books?hl=pt-PT&lr=&id=V2B9DwAAQBAJ&oi=fnd&pg=PR5&ots=3bYvMQZanv&sig=S5066RmGRUeh-WI5narZqQVeMNY&redir_esc=y#v=onepage&q&f=false. [Accessed: 04-Jan-2020].

[20] “Best Python libraries for Machine Learning - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/best-python-libraries-for-machine-learning/. [Accessed: 06-Feb-2020].

[21] “Top 8 Python Libraries for Machine Learning & Artificial Intelligence - By Supriya Rathi.”

[Online]. Available: https://hackernoon.com/top-8-python-libraries-for-machine-learning-and-artificial-intelligence-y08id3031. [Accessed: 06-Feb-2020].

[22] E. R. Melgar, C. C. Díez, P. Jaworski, and E. Ramos, “Arduino Basics,” *Arduino Kinect Proj.*, pp. 1–22, 2012, doi: 10.1007/978-1-4302-4168-3_1.

[23] “Arduino - Introduction.” [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Accessed: 20-Jan-2020].

[24] “What is a Raspberry Pi?” [Online]. Available: https://www.raspberrypi.org/help/what- is-a-raspberry-pi/. [Accessed: 28-Jan-2020].

[25] “Raspberry Pi Technology, Working and Its Applications.” [Online]. Available: https://www.watelectronics.com/know-all-about-raspberry-pi-board-technology/. [Accessed: 30-Jan-2020].

[26] “PIC Microcontroller : Architecture and Its Applications.” [Online]. Available: https://www.watelectronics.com/pic-microcontroller-architecture-and-applications/. [Accessed: 27-Jan-2020].

[27] “What is a PIC microcontroller? What can it do?” [Online]. Available: http://www.technologystudent.com/pics/picgen1.html. [Accessed: 27-Jan-2020].

[28] F. Leccese *et al.*, “A new acquisition and imaging system for environmental measurements: An experience on the Italian cultural heritage,” *Sensors (Switzerland)*, vol. 14, no. 5, pp. 9290–9312, 2014, doi: 10.3390/s140509290.

[29] “What is a Pulse Width Modulation (PWM) Signal and What is it Used For? - National Instruments.” [Online]. Available: https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019OkFSAU&l=pt-PT. [Accessed: 02-Feb-2020].

[30] “What is PWM: Pulse Width Modulation.” [Online]. Available: https://circuitdigest.com/tutorial/what-is-pwm-pulse-width-modulation. [Accessed: 02-Feb-2020].

[31] “MecaWeb - PWM - Modulação Por Largura de Pulso.” [Online]. Available: http://www.mecaweb.com.br/eletronica/content/e_pwm. [Accessed: 02-Feb-2020].

[32] M. W. Hung, C. J. Chen, C. L. Chang, and C. W. Hsu, “The impacts of high frequency pulse driving on the performance of LED light,” *Phys. Procedia*, vol. 19, pp. 336–343, 2011, doi: 10.1016/j.phpro.2011.06.171.

[33] “Bluetooth Technology Website.” [Online]. Available: https://www.bluetooth.com/. [Accessed: 14-Jan-2020].

[34] “Bluetooth Vs. Bluetooth Low Energy: What’s The Difference?” [Online]. Available:

https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy. [Accessed: 14-Jan-2020].

[35] “What Is Wi-Fi? Everything You Need to Know | Digital Trends.” [Online]. Available: https://www.digitaltrends.com/computing/what-is-wi-fi/. [Accessed: 14-Jan-2020].

[36] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, “Device-to-device communications with WiFi direct: Overview and experimentation,” *IEEE Wirel. Commun.*, vol. 20, no. 3, pp. 96–104, 2013, doi: 10.1109/MWC.2013.6549288.

[37] “Discover Wi-Fi | Wi-Fi Alliance.” [Online]. Available: https://www.wi-fi.org/discover-wi-fi. [Accessed: 14-Jan-2020].

[38] A. Mahmood, N. Javaid, and S. Razzaq, “A review of wireless communications for smart grid,” *Renew. Sustain. Energy Rev.*, vol. 41, pp. 248–260, 2015, doi: 10.1016/j.rser.2014.08.036.

[39] V. Bonaiuto, P. Boatto, N. Lanotte, C. Romagnoli, and G. Annino, “A Multiprotocol Wireless Sensor Network for High Performance Sport Applications,” *Appl. Syst. Innov.*, vol. 1, no. 4, p. 52, 2018, doi: 10.3390/asi1040052.

[40] N. Lethaby, “Wireless connectivity for the Internet of Things, one size does not fit all,” *Texas Instruments*, p. 16, 2017.

[41] M. Matteucci, “An adaptive indoor positioning system based on bluetooth low energy RSSI,” p. 147, 2013.

[42] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, 2014, doi: 10.1109/JIOT.2014.2306328.

[43] C. Harrison *et al.*, “Foundations for Smarter Cities,” *IBM J. Res. Dev.*, vol. 54, no. 4, pp. 1–16, 2010, doi: 10.1147/JRD.2010.2048257.

[44] “Send data from ESP8266 or ESP32 to Raspberry Pi via MQTT.” [Online]. Available: https://diyi0t.com/microcontroller-to-raspberry-pi-wifi-mqtt-communication/. [Accessed: 13-Aug-2021].

[45] Expressif Systems,"V2.9 Datasheet", “ESP32-WROOM-32 Datasheet,” 2019.

[46] “What is iBeacon? A definition by WhatIs.com.” [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/Apple-iBeacon. [Accessed: 17-Sep-2021].

[47] Minew, "V3.11. Datasheet", “Card Beacon,” May, 2019.