



Universidade do Minho

Escola de Engenharia

Bruno Miguel Vasconcelos da Silva, a88289

Diogo Miguel Cunha Fernandes, a88262

Duarte Miguel Novo Rodrigues, a88259

Francisco Lopes Salgado, a88231

João Pedro Dias Miranda, a88237

José Tomás Lima de Abreu, a88218

DWR-19

Digital Waiter Robot

Projeto integrador

Laboratórios e Práticas Integradas

Trabalho realizado sob a orientação do

Professor Luís Barros

22 junho 2021

ÍNDICE

Lista de Figuras	v
Lista de Tabelas	vii
Acrónimos e Siglas	ix
Capítulo 1 Introdução	11
1.1 Introdução	11
1.2 Enquadramento	11
1.3 Especificações Previstas.....	12
1.3.1Especificações funcionais.....	12
1.3.2Especificações técnicas.....	13
1.4 Planeamento	15
Capítulo 2 Arquitetura e Módulos Utilizados	17
2.1 Introdução	17
2.2 Sensores	18
2.2.1Array de Sensores de reflexão	18
2.2.2Sensor de obstáculos.....	19
2.3 Driver	19
2.4 Módulos de comunicação sem fios	20
2.4.1Radio Frequency Identification (RFID)	20
2.4.2Bluetooth.....	21
2.5 Microcontrolador e <i>Shield</i>	22
2.6 Circuito de Alimentação	23
Capítulo 3 Implementação em Software	25
3.1 Introdução	25
3.2 Periféricos	26
3.2.1Direct Memory Access (DMA)	27
3.2.2Analog to Digital Converter (ADC)	27
3.2.3TIMER (TIM)	29
3.2.4Serial Peripheral Interface (SPI).....	30
3.2.5Universal Asynchronous Receiver/Transmitter (UART)	31
3.2.6Mapeamento dos periféricos	32
3.3 Descrição de <i>Software</i> e Módulos Criados	34
3.3.1Módulos Criados.....	34
3.3.2Aplicação de Interface.....	41
3.3.3Circuito de Controlo	42
3.3.3.1 O que é um Controlador?.....	42
3.3.3.2 Análise do sistema de controlo.....	43
3.3.3.3 Controlador implementado	45
Capítulo 4 Lista de Componentes	49
Capítulo 5 Circuito Mecânico Implementado	55
Capítulo 6 Resultados Experimentais.....	59
Capítulo 7 Análise do Produto	63
7.1 Introdução	63
7.2 Fiabilidade	63
7.3 Segurança	64
7.4 Certificação.....	65

Capítulo 8 Conclusões	69
8.1 Conclusão.....	69
8.2 Sugestões de Trabalho Futuro	70
Referências	73

Lista de Figuras

Figura 1.1 - Marcas de identificação de (a) Quarto; (b) Quarto no fim de um corredor sem saída; (c) Cruzamento.	13
Figura 1.2 - Diagrama de <i>Gantt</i> do planeamento inicial.	15
Figura 1.3 - Diagrama de <i>Gantt</i> do planeamento revisto.	15
Figura 2.1 - Diagrama geral das principais interações no sistema.	17
Figura 2.2 - <i>Array</i> de sensores QTR-8A.	19
Figura 2.3 - Módulo Driver L298N.	20
Figura 2.4 – Sistemas RFID utilizados: (a) Módulo RFID MFR522; (b) Etiqueta RFID de ID único.	21
Figura 2.5 - Módulo <i>Bluetooth</i> HC-05.	22
Figura 2.6 - Diagrama da comunicação <i>Bluetooth</i>	22
Figura 2.7 - <i>Shield</i> desenvolvida: (a) <i>Layout</i> da PCB desenvolvida em KiCad; (b) Imagem 3D KiCad (c) PCB já totalmente montada.	23
Figura 2.8 - Esquema de ligação das baterias e BMS.	24
Figura 3.1 - Diagrama da máquina de estados.	25
Figura 3.2 - Exemplo de transferência do DMA.	27
Figura 3.3 - Conversor por aproximação sucessiva.	28
Figura 3.4 - Modo de funcionamento independente (a) Single-channel, single conversion mode; (b) Multichannel, single conversion mode; (c) Single-channel, continuous conversion mode; (d) Multichannel, continuous conversion mode.	28
Figura 3.5 - Esquema de ligação entre <i>Master</i> e <i>Slave</i>	30
Figura 3.6 - Esquema de ligação entre 2 UARTs.	31
Figura 3.7 - Estrutura de um pacote de dados enviado por UART.	31
Figura 3.8 - Mapeamento dos periféricos em função dos conetores: (a) CN7; (b) CN8; (c) CN9; (d) CN10.	33
Figura 3.9 - Divisão do software criado nos vários módulos.	34
Figura 3.10 - Estrutura que definem as saídas que controlam o motor.	35
Figura 3.11 - Enumerado que representa os sensores do QTR utilizados.	35
Figura 3.12 - Estrutura que agrupa as variáveis de cálculo do algoritmo PID.	36
Figura 3.13 - Duração dos vários <i>timeouts</i> , em segundos, e definição das <i>flags</i> respetivas.	36
Figura 3.14 - Enumerado com os possíveis estados de movimento.	37
Figura 3.15 - Módulo RFID: a) Estrutura que define um cartão RFID; b) Estado do leitor RFID.	37
Figura 3.16 - Estado do módulo <i>Bluetooth</i>	38
Figura 3.17 - Definição de um callback de um comando e da estrutura que define um comando.	38
Figura 3.18 - Ações possíveis a realizar num <i>checkpoint</i> , e, definição de um <i>checkpoint</i>	38
Figura 3.19 - Definição de uma janela deslizante.	39
Figura 3.20 - Definição da estrutura <i>debounce</i>	39
Figura 3.21 - Estados da máquina de estados.	39

Figura 3.22 - Definição da máquina de estados (a) Array de funções de estado; (b) Execução da máquina de estados.	40
Figura 3.23 - Implementação do controlo remoto.	40
Figura 3.24 - Aplicação Interface (a) Ecrã inicial; (b) Menu principal; (c) Terminal de mensagens; (d) Ecrã controlo remoto.....	42
Figura 3.25 - Ações de controlo (a) ação proporcional; (b) ação integral; (c) ação derivativa.	43
Figura 3.26 - Diagrama de blocos do sistema de controlo.....	45
Figura 3.27 - Serviço de Rotina à Interrupção do algoritmo PID.	48
Figura 5.1 - Desenho (à esquerda) e imagem real (à direita) do DWR (a) vista superior; (b) vista inferior; (c) vista lateral direita; (d) vista dianteira; (e) vista traseira.	57
Figura 5.2 - Vista 3D do DWR.....	57
Figura 6.1 - Valor digital da saída do sensor de obstáculos em função da distância a um objeto.	60
Figura 7.1 - Marcação CE.	66
Figura 7.2 - Símbolos de perigo: (a) perigoso para o ambiente; (b) corrosivo; (c) comburente; (d) inflamável; (e) explosivo; (f) tóxico; (g) vários perigos; (h) <i>Electrostatic Sensitive Device - ESD</i>	67
Figura 7.3 - Símbolo WEEE.	67

Lista de Tabelas

Tabela 2.1 - Sensores e sua utilização.....	19
Tabela 2.2 - Tabelas de verdade do <i>driver</i> L298N (a) Controlo do Motor A; (b) Controlo do Motor B.	20
Tabela 2.3 - Gamas de frequência e alcance etiquetas RFID	21
Tabela 2.4 – <i>Pinout</i> do módulo <i>Bluetooth</i> HC-05.....	22
Tabela 3.1 - Mapeamento dos <i>timers</i>	29
Tabela 3.2 - Linhas lógicas para a transferência de dados do protocolo SPI.	30
Tabela 4.1 - Lista de componentes.	49
Tabela 7.1 - Tempo de vida dos componentes com maior probabilidade de falha, usados no AWR.	64
Tabela 8.1 - Número de horas investidas por elemento.	70

Acrónimos e Siglas

Acrónimo/Sigla	Significado
DWR	<i>Digital Waiter Robot</i> Robô Empregado de Mesa Digital
LED	<i>Light Emitting Diode</i> Díodo Emissor de Luz
PWM	<i>Pulse Width Modulation</i> Modulação de Largura de Pulso
BMS	<i>Battery Management System</i> Sistema de Manutenção das Baterias
PCB	<i>Printed Circuit Board</i> Placa de Circuito Impresso
ADC	<i>Analog to Digital Converters</i> Conversor Analógico Digital
DMA	<i>Direct Memory Access</i> Acesso Direto à Memória
RFID	<i>Radio Frequency Identification</i> Identificação por Radiofrequência
GPIO	<i>General Purpose Input/Output</i> Entrada e Saída de Propósito Geral
SPI	<i>Serial Peripheral Interface</i> Interface de Periféricos Série

CPU	<i>Central Process Unit</i> Unidade de Processamento Central
FIFO	<i>First in First out</i> Primeiro a entrar Primeiro a Sair
PID	<i>Proportional Integrative Derivative</i> Proporcional Integral Derivativo
ISR	<i>Interrupt Service Routine</i> Serviço de Rotina à Interrupção
DC	<i>Direct current</i> Corrente contínua
ISO	<i>Internacional Organization for Standardization</i> Organização Internacional de Normalização
ESD	<i>Electrostatic-Sensitive Device</i> Equipamento Sensível à Eletricidade Estática
WEEE	<i>Waste Electrical and Electronic Equipment</i> Resíduos de Equipamentos Elétricos e Eletrónicos
MTBF	<i>Mean Time Between Failures</i> Período Médio Entre Falhas
ID	<i>Identification</i> Identificação

Capítulo 1

Introdução

1.1 Introdução

Perante o atual panorama pandémico da Covid-19 [1] pretende-se, com a realização do Projeto Integrador da Unidade Curricular de Laboratórios e Práticas Integradas II (LPI II) do curso Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores, da Universidade do Minho, a implementação de uma ideia com o objetivo de diminuir os contactos interpessoais que possam surgir no momento da entrega de bens a pessoas hospitalizadas ou em isolamento profilático. Esta situação pandémica é uma oportunidade para acelerar a transformação da área saúde.

Tendo em consideração que os hospitais tiveram um aumento considerável do número de internamentos [2], pretende desenvolver-se um produto que permita a entrega e recolha de bens essenciais de forma segura. De forma a facilitar a sua desinfeção e o seu manuseamento, o robô deverá ter superfícies lisas e uma interface simples.

A maioria das ideias nesta área tem como foco principal a saúde pública da população em geral, tais como, robôs de desinfeção [3], robôs que repõem o *stock* em hospitais [4] ou que medem a temperatura corporal através de câmaras [5]. Prevê-se que o *Digital Waiter Robot* (DWR) possa ser aplicado em contexto hospitalar. Na China construiu-se um robô, denominado de *little peanut*, com a mesma finalidade, que foi utilizado num hotel para entregar comida porta a porta a hóspedes com suspeita de infeção [6].

A versatilidade do sistema permitirá que o DWR possa auxiliar na distribuição de bens noutros contextos dependendo das funcionalidades requeridas, como, por exemplo, na indústria hoteleira ou em ambiente doméstico.

1.2 Enquadramento

O DWR é um robô seguidor de linha focado na automatização de um hospital. O DWR estará parado num local apropriado à espera do envio de um pedido e de uma rota por parte de um funcionário

responsável. Assim que este conclua a comunicação com o robô, deverá colocar no seu suporte os pedidos respetivos. O robô fará chegar cada pedido a cada paciente, parando apenas nos quartos solicitados. Depois de atender a todos os pedidos, o DWR voltará ao local de onde partiu.

Considere-se o exemplo de o DWR ser responsável pela distribuição de bens alimentares. O robô deve ser colocado próximo dos locais de confeção dos alimentos. Assim que o funcionário responsável envie a rota a percorrer e as refeições a distribuir, um dos responsáveis pela secção alimentar deverá colocar estes pedidos na base do robô, dar ordem de início de marcha, e o robô tratará de tudo o resto. O DWR pode assumir outras funções, como por exemplo, na distribuição de medicamentos.

1.3 Especificações Previstas

O DWR seguirá uma linha preta previamente colocada no piso do hospital, que define os locais acessíveis pelo robô. Como os hospitais possuem vários quartos, em vários corredores, o robô terá de ser capaz de percorrer um percurso com várias intercessões de corredores.

Como a alimentação do robô será por intermédio de baterias eletroquímicas, terá de ser ligado à rede elétrica para ser carregado. Assim, existirá uma estação de carregamento que estará presente num local denominado por base.

O tipo de desenvolvimento deste produto pode ser classificado como “ofensivo”. “O objetivo é colocar no mercado um produto com funcionalidades e características inovadoras ou com preço significativamente mais baixo do que produtos com funcionalidades e características equivalentes, de forma a obter para o produto, quota de mercado ou aumento da quota de mercado em relação a produtos antecessores [7]”.

1.3.1 Especificações funcionais

Cada quarto e cruzamento no percurso tem um identificador único e o DWR deve ser capaz de os detetar para que possa seguir a rota previamente estabelecida parando apenas nos quartos previstos. Na Figura 1.1 (a) é apresentada a marca que permite identificar um quarto. Verifica-se que, num quarto, o DWR não é capaz de mudar de direção, apenas pode seguir em frente ou voltar para trás. No entanto, num quarto que esteja no fim de um corredor sem saída, Figura 1.1 (b), o robô é obrigado a voltar para trás. Comparando com a Figura 1.1 (c), onde é apresentada a marca que permite identificar um cruzamento, verifica-se que, num cruzamento, o DWR pode seguir em frente ou mudar de direção, neste caso, somente à esquerda. É de notar que, o cartão RFID, identificador de quartos e cruzamentos, está

colocado sob o cruzamento das linhas pretas (cor branca saliente na Figura 1.1), protegendo-o de desgaste proveniente do ambiente em que se encontra.

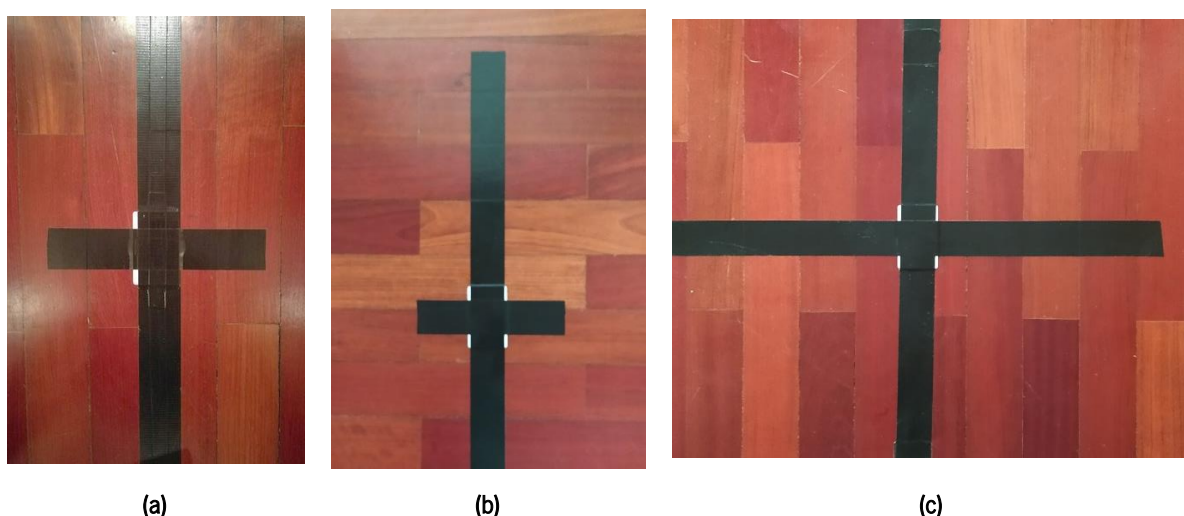


Figura 1.1 - Marcas de identificação de (a) Quarto; (b) Quarto no fim de um corredor sem saída; (c) Cruzamento.

Após o paciente efetuar a recolha dos bens a si destinados, poderá acionar o robô de forma que este reinicie o seguimento da linha. Assim que não existam mais pedidos pendentes, o DWR retorna à base.

Para evitar que o DWR colida, este tem um sistema de deteção de obstáculos, que, ao detetar um objeto no seu percurso, faz com que o pare, emitindo um sinal sonoro. Se o problema for resolvido, o DWR continua o seu trajeto. Se, pelo contrário, ao fim de um determinado intervalo de tempo, previamente estabelecido, a via se mantiver obstruída, o robô entrará num estado de erro e envia uma notificação para a aplicação de interface, alertando um funcionário responsável do sucedido. Além disso, sempre que o DWR se encontrar neste estado, pode ser acedida a funcionalidade de controlo remoto, permitindo ao funcionário movimentar o robô remotamente, repondo o estado normal de funcionamento do robô.

1.3.2 Especificações técnicas

Para implementação do sistema de controlo do DWR será usado um microcontrolador STM32F767ZI-NUCLEO [8], e o IDE STM32CubeIDE [9], que integra as ferramentas necessárias para a configuração de todos dos periféricos.

Para cumprir o objetivo de seguir de linha, usar-se-á um *array* de oito sensores de reflexão com saídas analógicas [10]. Ao contrário dos sensores digitais que apresentam apenas dois níveis nas suas saídas, alto ou baixo, este tipo de sensores possuem uma maior sensibilidade, permitindo que o sistema de seguidor de linha apresente menos oscilações.

A detecção e identificação de intercessões de corredores será feita por meio da tecnologia RFID [11]. O sistema de detecção de obstáculos será composto por um módulo de sensores de distância infravermelhos adequado para calcular com precisão a distância a objetos que possam aparecer na frente do DWR. O sistema de alerta sonoro fará uso de um *buzzer* ativo. A comunicação entre o robô e a unidade de controlo será implementada recorrendo a tecnologia *Bluetooth*.

1.4 Planeamento

Na Figura 1.2, mostra-se o diagrama de *Gantt* do planeamento inicial elaborado na etapa 0 deste projeto. Devido à elevada carga de trabalho desta e outras unidades curriculares, o planeamento inicial foi alterado para o apresentado na Figura 1.3.

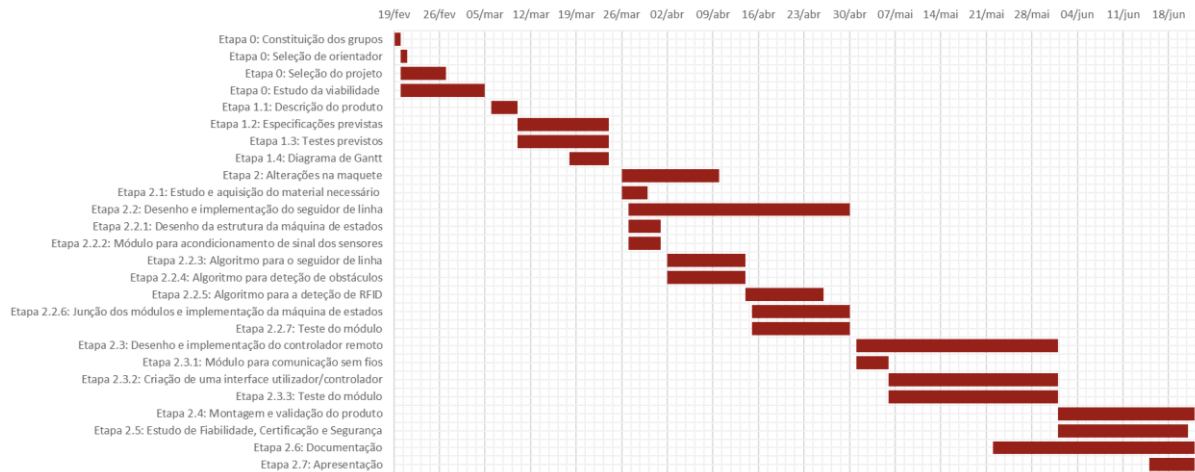


Figura 1.2 - Diagrama de *Gantt* do planeamento inicial.

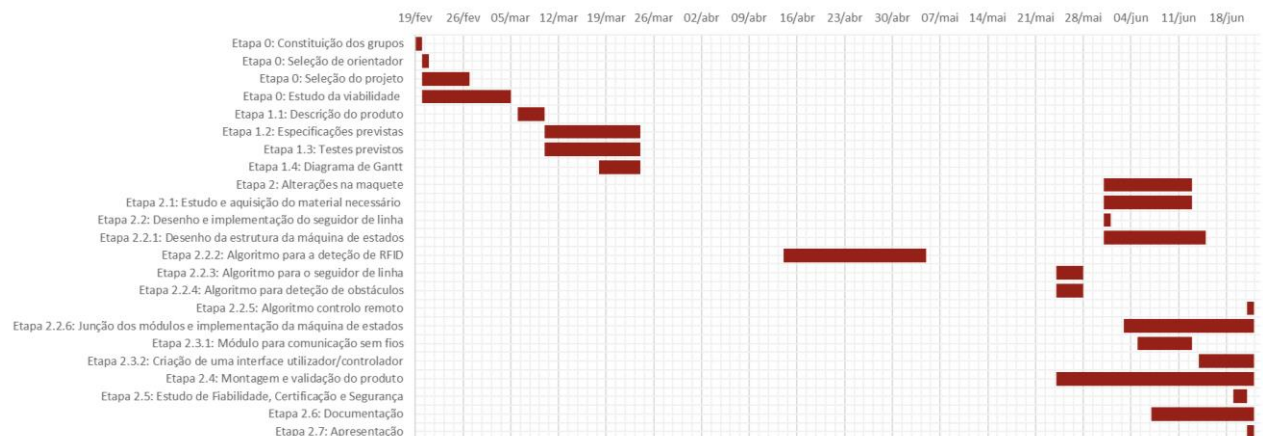


Figura 1.3 - Diagrama de *Gantt* do planeamento revisto.

Capítulo 2

Arquitetura e Módulos Utilizados

2.1 Introdução

O principal objetivo do DWR é auxiliar na distribuição de bens a várias pessoas, sendo controlado por um responsável. Na Figura 2.1, é apresentado o diagrama geral das principais interações do DWR. Considere-se o operador como o funcionário responsável pelo DWR e o utilizador como a pessoa à qual se destinam os bens.

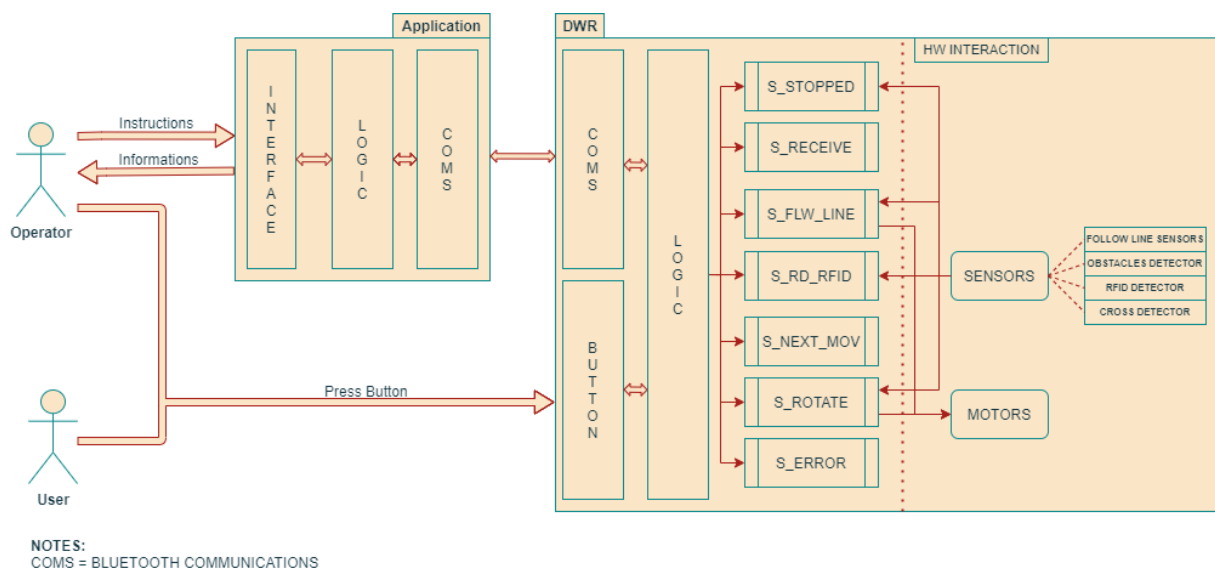


Figura 2.1 - Diagrama geral das principais interações no sistema.

O operador deve enviar instruções para o robô, via *Bluetooth*, através de uma aplicação, selecionando a rota que pretende que este efetue. Após isto, o operador pode iniciar o movimento do DWR, pressionando o botão de pressão presente na lateral deste. Este botão também é usado pelo utilizador, que, depois de levantar os bens a si relativos, pode premi-lo, permitindo ao robô continuar o seu percurso. Além disso, o DWR envia informações para o operador, de modo que este possa monitorizar o seu estado de funcionamento.

O DWR é estruturado em diferentes blocos. Os blocos de comunicação e botão permitem a interação entre o robô e o operador / utilizador. Estes blocos relacionam-se com a lógica desenvolvida para controlar os estados do robô, através de uma máquina de estados. Tal como apresentado na

Figura 2.1, os diferentes estados de funcionamento do robô são os seguintes: S_STOPPED, S_RECEIVE, S_FLW_LINE, S_RD_RFID, S_NEXT_MOV, S_ROTATE e S_ERROR. No estado S_STOPPED, o DWR está parado à espera de algum estímulo. O estado S_RECEIVE dedica-se à escolha de novas rotas, através da comunicação entre o operador e o robô. O estado S_FLW_LINE implementa o controlo do seguidor de linha. O estado S_RD_RFID é responsável pela leitura de um cartão RFID de identificação unívoca de cada quarto e cruzamento. O estado S_NEXT_MOV é um estado de decisão, responsável por fazer transitar o robô para um estado que esteja de acordo com o percurso a realizar. O estado S_ROTATE executa o controlo da mudança de direção do robô. O estado S_ERROR é o estado para o qual o robô transita aquando da ocorrência de um erro que comprometa o normal funcionamento do sistema, informando o operador do sucedido, permitindo o controlo do robô à distância.

A camada de interação com o hardware é composta por sensores e atuadores. As saídas dos sensores de obstáculos, de linha, de paragem e do leitor RFID são utilizadas nos estados S_STOPPED, S_FLW_LINE, S_RD_RFID e S_ROTATE. Os atuadores – motores – são controlados nos estados S_FLW_LINE e S_ROTATE.

2.2 Sensores

Um sensor é um dispositivo que responde a um estímulo do ambiente, físico ou químico, produzindo um sinal que pode ser transformado noutra grandeza física para fins de medição.

No DWR, foram usados sensores infravermelhos, muito utilizados em aplicações que envolvem leitura e deteção de proximidade. Estes usam a luz infravermelha, que é uma radiação eletromagnética, de baixa frequência, não visível ao olho humano. Este tipo de sensores fazem a sua leitura através da utilização de um emissor, normalmente, laser ou LED, e um recetor fotoelétrico, que contém um elemento optoeletrónico, como, por exemplo um fotodiodo ou um fototransistor. Este recetor fotoelétrico deteta a luz vinda do emissor e converte a intensidade da luz recebida num sinal elétrico em tensão. Perante as características acima enumeradas, os sensores infravermelhos podem ser usados para implementar o sensor de obstáculos e os sensores de linha.

2.2.1 *Array* de Sensores de reflexão

Com o objetivo do DWR seguir a linha escolheu-se o *array* de sensores QTR-8A [10], apresentado na Figura 2.2. Este pode ser alimentado com 3,3 V ou 5 V e tem oito sensores analógicos que apresentam na sua saída um valor de tensão compreendido entre 0 V e o valor de alimentação. Sobre uma superfície branca, os sensores medem uma tensão de, aproximadamente, 0 V. Já quando se encontram sobre uma

superfície preta, os sensores medem uma tensão de, sensivelmente, 3,3 V. Para as restantes cores, os sensores apresentam tensões entre estas duas gamas. No contexto deste projeto, é indispensável que a saída do *array* de sensores de reflexão apresente uma tensão máxima de 3,3 V, uma vez que este é o limite que os *Analog to Digital Converters* (ADCs) da STM32F767ZI-NUCLEO suportam.

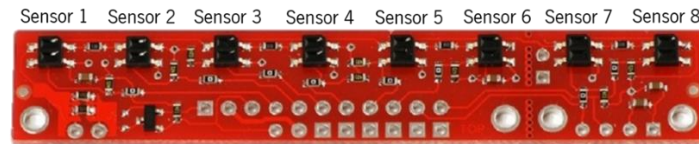


Figura 2.2 - *Array* de sensores QTR-8A.

A Tabela 2.1 apresenta a finalidade dos sensores do QTR-8A nos circuitos do DWR.

Tabela 2.1 - Sensores e sua utilização.

Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6	Sensor 7	Sensor 8
Deteção cruzamento /quarto	-	Seguidor de linha	Deteção linha preta	Deteção linha preta	Seguidor de linha	-	Deteção cruzamento /quarto

2.2.2 Sensor de obstáculos

Face à aplicação do DWR, torna-se necessário implementar um sistema de deteção de obstáculos, de forma a evitar a colisão com eventuais objetos que impeçam a sua passagem, evitando possíveis danos materiais e humanos. Para isso, utilizou-se o sensor de distância infravermelho de referência GP2Y0A21YK0F da fabricante SHARP [12]. Este sensor permite medir distâncias a objetos entre 10 e 80 cm, tendo uma saída do tipo analógico. Assim, para esta aplicação, este sensor será usado como sensor de proximidade. Este pode ser alimentado com tensões entre 4,5 V e 5,5 V, tendo sido usada uma tensão de alimentação de 5 V, sendo o valor máximo da sua saída 3,3 V.

2.3 Driver

Para controlo dos motores utilizou-se o módulo *driver* apresentado na Figura 2.3 [13]. Este *driver* possui seis entradas, três para cada motor: os sinais de *ENA* e *ENB* permitem ativar ou desativar os motores, controlando as suas velocidades de rotação, por intermédio de um sinal PWM. Por sua vez, os sinais de *IN1*, *IN2*, *IN3* e *IN4* definem o modo de rotação de cada motor. Na Tabela 2.2 encontra-se exemplificado o modo de operação do motor em função da combinação lógica dos sinais de entrada.

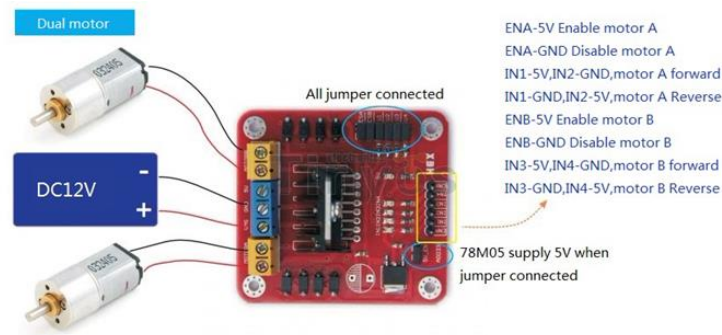


Figura 2.3 - Módulo Driver L298N.

Tabela 2.2 - Tabelas de verdade do *driver* L298N (a) Controlo do Motor A; (b) Controlo do Motor B.

(a)

ENA	IN1	IN2	Motor A
0 V	N/A	N/A	OFF
5 V	0 V	0 V	Trava
5 V	0 V	5 V	Sentido anti-horário
5 V	5 V	0 V	Sentido horário
5 V	5 V	5 V	Trava

(b)

ENB	IN3	IN4	Motor B
0 V	N/A	N/A	OFF
5 V	0 V	0 V	Trava
5 V	0 V	5 V	Sentido anti-horário
5 V	5 V	0 V	Sentido horário
5 V	5 V	5 V	Trava

2.4 Módulos de comunicação sem fios

Ao longo do percurso, o DWR tem de tomar várias decisões de modo a seguir a rota proposta fazendo as paragens necessárias para entrega de bens. Para isto, o DWR tem de ser capaz de identificar e distinguir pontos de referência como cruzamentos e/ou quartos existentes ao longo do percurso. Decidiu-se atribuir a cada ponto de referência uma etiqueta RFID com códigos de identificação únicos. De modo a ser possível comunicar com o robô à distância, decidiu-se implementar comunicação por *Bluetooth*.

2.4.1 *Radio Frequency Identification* (RFID)

A identificação por rádio frequência, na nomenclatura inglesa conhecido por *Radio Frequency Identification* (RFID), é uma tecnologia de leitura de dados sem contacto que usa ondas eletromagnéticas para ler o código de identificação de uma etiqueta RFID. Como cada etiqueta possui um código único, estas podem ser usadas para associar um ID a um objeto. Existem dois tipos de etiquetas RFID: passivas e ativas. Etiquetas passivas usam a energia fornecida pelas ondas eletromagnéticas para induzir uma corrente na antena, de modo a transmitir os dados da mesma. As etiquetas ativas possuem uma fonte

de alimentação própria, como uma bateria, para alimentar os circuitos necessários para a transmissão [14].

Outras características que influenciam a escolha das etiquetas e do módulo de leitura, são a frequência de comunicação, o alcance e o preço. Atualmente as etiquetas disponíveis no mercado operam em três gamas de frequência e têm diferentes alcances, como apresentado na. Tabela 2.3.

Tabela 2.3 - Gamas de frequência e alcance etiquetas RFID

	Gama de Frequência	Alcance	Tipo de Alimentação
<i>Low Frequency</i>	Entre 30 kHz e 300 kHz	Até 10 cm	Passiva
<i>High Frequency</i>	13,56 MHz	Até 30 cm	Passiva/Ativa
<i>Ultra High Frequency</i>	Entre 300 MHz e 3 GHz	Entre 20 m e 100 m	Ativa

Sabendo que, por norma, os módulos RFID que operam a frequências mais altas (*Ultra High Frequency*) têm preços mais elevados e que não existem muitos leitores RFID do tipo *Low Frequency* disponíveis no mercado, decidiu-se usar um modulo RFID do tipo *High Frequency*. Além do referido, as etiquetas não precisam de uma fonte de alimentação para o seu funcionamento e seu o alcance de deteção encontra-se na gama pretendida para o DWR. Escolheu-se, então, o módulo MFRC522 [15], representado na Figura 2.4 (a), uma vez que oferece as características pretendidas. Por sua vez, na Figura 2.4 (b), encontra-se representado um exemplo de uma etiqueta utilizada.

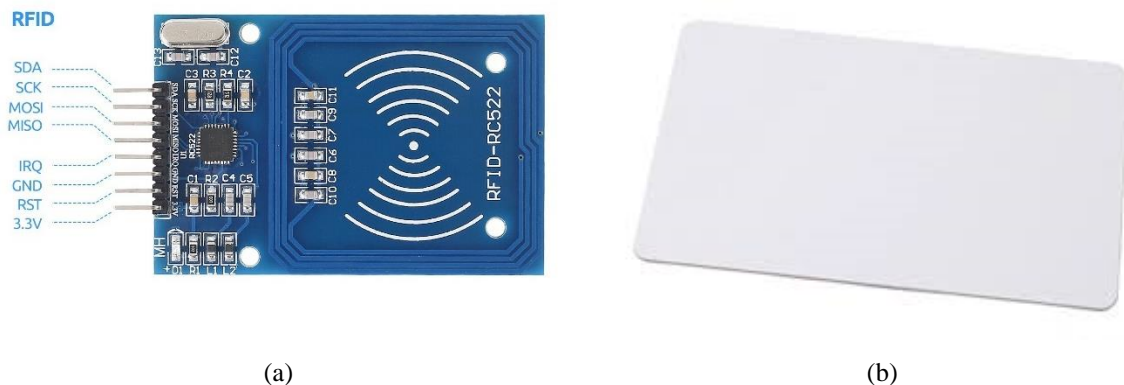


Figura 2.4 – Sistemas RFID utilizados: (a) Módulo RFID MFR522; (b) Etiqueta RFID de ID único.

2.4.2 Bluetooth

De forma a estabelecer-se uma comunicação à distância entre o DWR e um funcionário responsável, optou-se pela tecnologia *Bluetooth* por ser uma tecnologia fiável e de fácil implementação. Para a implementação desta funcionalidade, usou-se o módulo *Bluetooth* HC-05, [16] que se apresenta na Figura 2.5. Este módulo possui seis pinos com as especificações descritas na Tabela 2.4

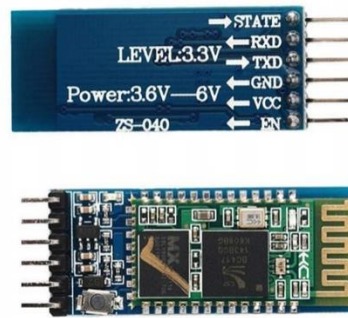


Figura 2.5 - Módulo *Bluetooth* HC-05.

Tabela 2.4 – *Pinout* do módulo *Bluetooth* HC-05.

Receive (Rx)	Receção de dados por comunicação série.
Transmite (Tx)	Transmissão de dados por comunicação série.
State	Indica que um dispositivo foi ligado ao módulo, tendo um LED associado.
Enable (EN)	É usado para alterar o modo de funcionamento do módulo para o modo de <i>AT Command</i> . Neste modo o HC-05 pode receber um conjunto de comandos, que são enviados através de comunicação série para modificar alguns parâmetros do módulo, como por exemplo o <i>baud rate</i> .
+5 V e GND	Alimentação do módulo.

Como se pode observar no diagrama da Figura 2.6, o utilizador interage com a camada de aplicação e os dados por si inseridos são enviados para o módulo HC-05 através da camada protocolar *Bluetooth*. A comunicação entre o módulo HC-05 e o microcontrolador é feita por comunicação série RS232 com *baud rate* de 9600 Bits/s.

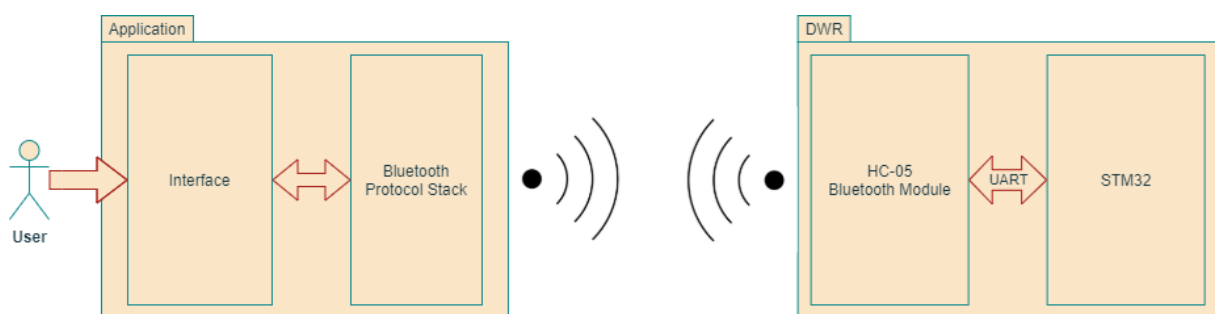


Figura 2.6 - Diagrama da comunicação *Bluetooth*.

2.5 Microcontrolador e *Shield*

O microcontrolador STM32F767ZI [8] oferece diversos periféricos e pinos de *General Purpose Input/Output* (GPIO) que são usados para diferentes finalidades. Em muitos casos, as entradas e saídas não estão organizadas de forma conveniente e existe uma enorme quantidade de ligações cruzadas na

parte superior da STM32. Além de não ser visualmente apelativo, esta desorganização pode levar, entre outros, a problemas de ruído e de maus contactos. Usando esta metodologia, sempre que se necessitasse de conectar novos módulos haveria uma grande dificuldade em fazê-lo. Estes problemas, levaram à necessidade da criação de uma *shield*, uma placa de circuito impresso (PCB), com o propósito de organizar e melhorar as conexões necessárias, recorrendo-se, para isso, ao *software* KiCad [17].

O *layout* da *shield* criada está apresentado na Figura 2.7. Foi projetada com o propósito único de servir os interesses do DWR, tendo em conta os periféricos, sensores e módulos utilizados, bem como a sua disposição no circuito mecânico, evitando que os fios atravessassem o microcontrolador. Além disto, integrou-se nesta *shield*, uma fonte [18] que permite a alimentação dos componentes, e o módulo *Bluetooth* [16].

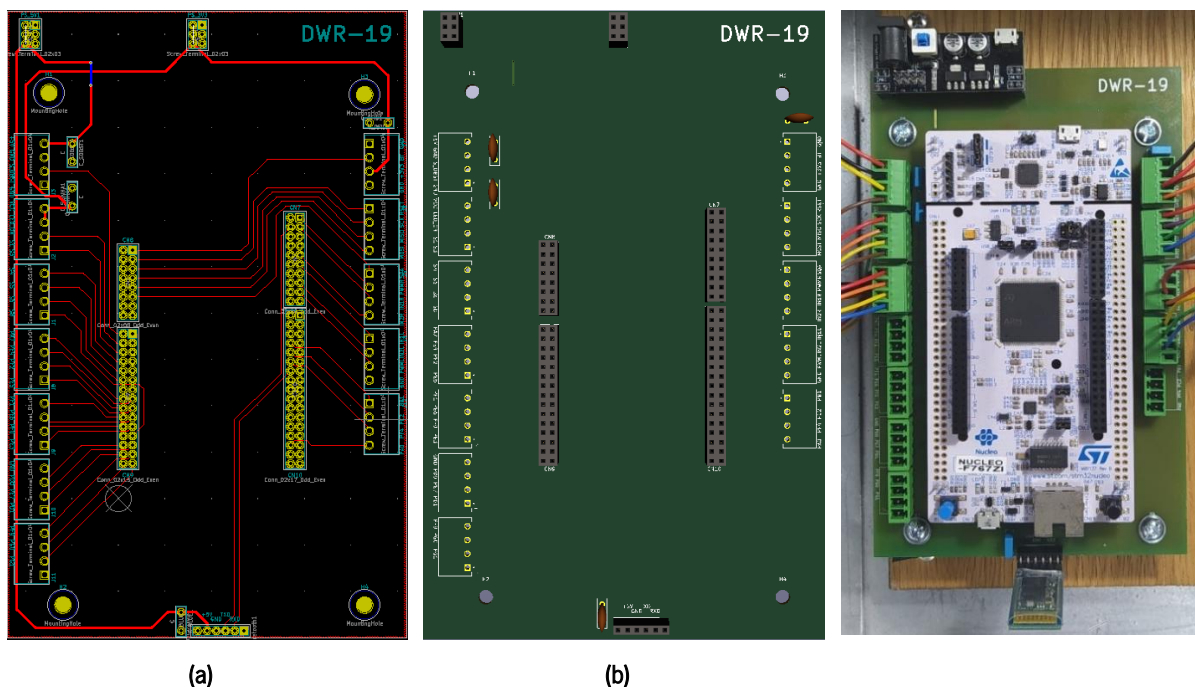


Figura 2.7 - *Shield* desenvolvida: (a) *Layout* da PCB desenvolvida em KiCad; (b) Imagem 3D KiCad (c) PCB já totalmente montada.

2.6 Circuito de Alimentação

Para alimentar os circuitos são necessárias tensões de 12 V, 5 V e 3,3 V. Para obter o primeiro valor de tensão, foram utilizadas três baterias recarregáveis LI-ION de 3,7 V [19] em série. De modo a aumentar a autonomia do DWR, foram colocadas mais três baterias, cada uma em paralelo com as três já existentes. Para proteção das baterias foi usado um dispositivo de BMS [20]. Este dispositivo controla a descarga das baterias não deixando que a sua tensão desça abaixo de um limite de segurança, evitando a sua degradação. Como as baterias necessitam de ser carregadas, utilizou-se um carregador [21] que se encontra conectado aos terminais da BMS. Para proteção de todos os componentes, foi usado um

fusível entre os terminais do carregador e a carga. O interruptor, *Power Switch*, é um botão ON/OFF que permite interromper a alimentação da carga. O esquema de ligação é apresentado na Figura 2.8.

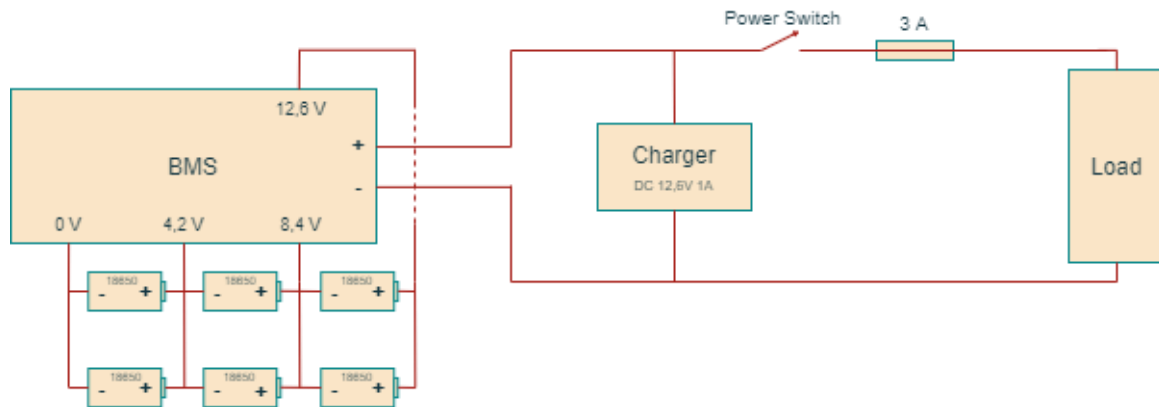


Figura 2.8 - Esquema de ligação das baterias e BMS.

Cada motor usado no DWR consome uma corrente de 580 mA [22], e como este possui dois motores, a corrente total consumida pelo par de motores será 1160 mA. Quanto aos sensores usados, o QTR-8A consome, sensivelmente, 100 mA, o sensor de distância 30 mA, o leitor de RFID 26 mA e o módulo Bluetooth 40 mA. O microcontrolador STM32F767ZI tem um consumo máximo de 258 mA. Somando as correntes consumidas por estes componentes, e assumindo que os restantes componentes têm consumos de corrente desprezáveis, conclui-se que consumo total do DWR será de, aproximadamente, 1614 mA. Assim, no circuito de alimentação, apresentado na Figura 2.8, o fusível deverá possuir um calibre superior ao consumo total de corrente esperado do robô, escolhendo-se um fusível de 3 A.

Visto que cada célula possui 2200 mAh e considerando que esta tem um rendimento de 80 %, então estão disponíveis 1760 mAh. Como cada célula está colocada em paralelo com uma outra, no total, estão disponíveis 3520 mAh para todos os circuitos. Assim, estimou-se uma autonomia máxima de 2 horas e 10 minutos. Caso se pretenda aumentar este valor, poder-se-á adicionar mais células em paralelo com as existentes.

De forma a obter tensões de 3,3 V e 5 V, necessárias para alimentação de sensores e módulos do DWR, usou-se uma fonte de alimentação para *breadboard* 3,3 V / 5 V DC [18]. Esta fonte é alimentada via micro USB. Para o efeito, usou-se um circuito abaixador para 5 V [23], com quatro portas USB, que tem como entrada os 12 V provenientes do circuito de alimentação. Optou-se por alimentar a STM usando este circuito abaixador. Atendendo que o *step-down* debita, no máximo, 8 A é suficiente para alimentar todos os componentes.

Capítulo 3

Implementação em Software

3.1 Introdução

O DWR tem de executar ações distintas durante toda a sua atividade. De forma a ser possível executar a ação correta em todos os momentos, é necessário um sistema que faça a gestão do estado atual do robô e das entradas que possam despoletar a alteração deste mesmo estado. Na Figura 3.1, está representado o diagrama que representa a máquina de estados implementada para este robô. Esta é composta por quatro estados principais: S_STOPPED, S_RECEIVE, MOVEMENT e S_ERROR. O estado de MOVEMENT pode ser subdividido em quatro estados secundários: S_FLW_LINE, S_RD_RFID, S_NEXT_MOV e S_ROTATE.

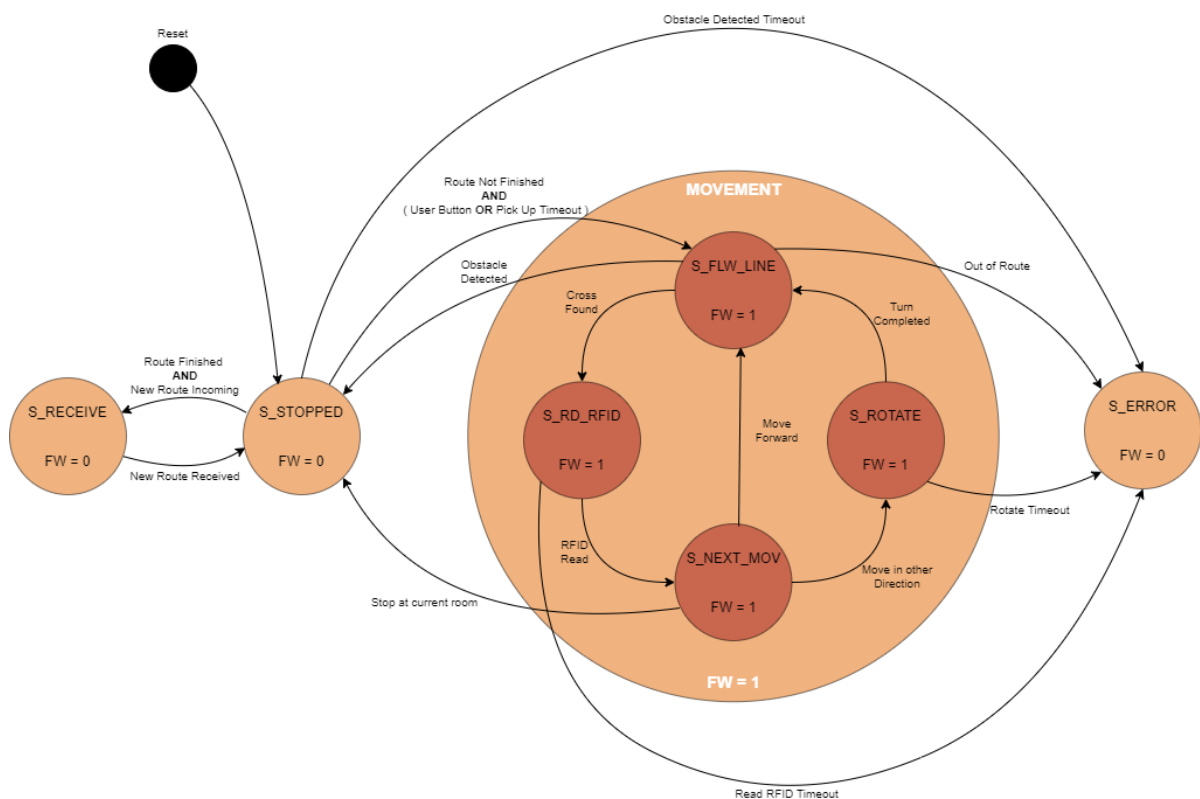


Figura 3.1 - Diagrama da máquina de estados.

Inicialmente, o DWR encontra-se no estado S_STOPPED. Quando o operador iniciar a transmissão de uma nova rota e não houver nenhuma rota em curso, o robô transita para o estado S_RECEIVE. Após

receber a rota selecionada pelo operador, o robô transita do estado `S_RECEIVE` para o estado `S_STOPPED`, onde espera que lhe seja dada a permissão para iniciar o seu movimento, através da pressão do botão presente na sua lateral. Assim que este botão seja pressionado, o sistema evolui para o estado de `S_FLW_LINE`. Neste estado o robô segue a linha até que aconteça algum dos seguintes imprevistos: presença de um obstáculo no percurso do robô, detecção de uma cruz de paragem (quarto ou cruzamento) ou ocorrência de falha no controlador do seguidor de linha. Caso o percurso esteja obstruído por um obstáculo, o DWR para, evitando a colisão com o mesmo, voltando para estado `S_STOPPED`. Se for detetada uma linha horizontal por ambos os sensores das extremidades do *array* de sensores (cruz de paragem), significa que o robô está na presença de um quarto ou cruzamento, sendo necessário efetuar a leitura do cartão RFID a este associado, no estado `S_RD_RFID`. Caso o cartão RFID seja lido com sucesso, o sistema evolui para o estado `S_NEXT_MOV`, onde determina qual o próximo estado do DWR, de acordo com a rota selecionada. No estado `S_NEXT_MOV`, o robô transita para o estado `S_FLW_LINE` se o RFID detetado for relativo a um quarto que não esteja marcado como local de paragem na rota ou a um cruzamento em que não seja necessário efetuar a mudança de direção. Pelo contrário, caso seja necessário efetuar paragem no quarto atual, dá-se uma transição para o estado `S_STOPPED`. No caso do RFID detetado ser relativo a um cruzamento e for necessário mudar de direção, o sistema transita para o estado `S_ROTATE`. Uma vez neste estado, o DWR roda na direção indicada pela rota, até estar orientado na direção pretendida, voltando ao estado `S_FLW_LINE`.

Todas estas ações têm *timeouts* associados, que são ativos quando a ação demora mais tempo a ser realizada do que o esperado, permitindo ter controlo sobre o robô em casos imprevisíveis, transitando para o estado `S_ERROR`, por exemplo, quando o DWR se encontrar parado à espera de que o percurso seja desobstruído por um longo período de tempo, quando ocorre um erro (cartão errado ou demasiado tempo) na leitura de um cartão RFID ou quando ocorre um erro na mudança de direção do robô. Além disso, o robô transita para o estado `S_ERROR` quando o robô sai totalmente da linha durante o seu percurso. Uma vez neste estado, o DWR pode ser controlado remotamente por um responsável até voltar ao seu estado de funcionamento.

3.2 Periféricos

De modo a cumprir com os objetivos proposto neste projeto, é necessário o uso de diferentes periféricos presentes no microcontrolador. Para gerir bases de tempo e gerar sinais modulados por largura de pulso (PWM) utilizam-se *timers*. Para a leitura do *array* de sensores de linha e do sensor de obstáculo faz-se uso do ADC em conjunto com o DMA. O módulo de RFID utiliza o protocolo SPI e a comunicação entre o microcontrolador e o módulo *Bluetooth* realiza-se através da UART.

3.2.1 *Direct Memory Access (DMA)*

Uma Unidade de Acesso Direto à Memória (DMA) pode ser usada em conjunto com o microprocessador para executar as operações de transferência de dados entre memória do microprocessador e os periféricos. Deste modo reduz-se, significativamente, a utilização da unidade central de processamento (CPU) do microprocessador. Os dispositivos compartilham o barramento de memória e os barramentos de periféricos com o CPU, tal como mostrado na Figura 3.2. No exemplo apresentado, o dispositivo DMA lê um valor de um periférico, a partir do barramento do periférico, e grava na memória do microprocessador, através do barramento de memória.

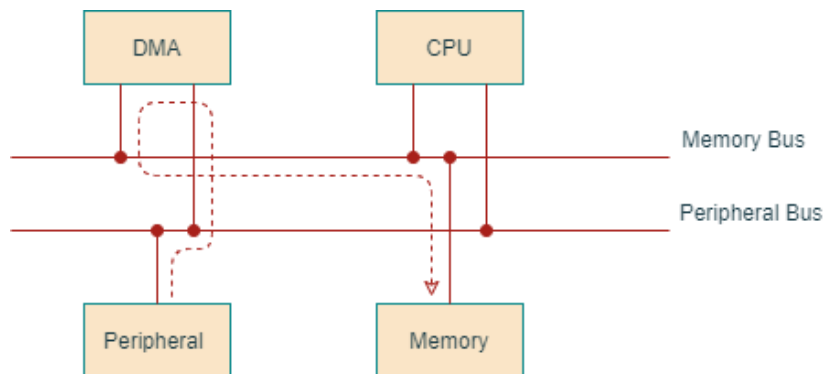


Figura 3.2 - Exemplo de transferência do DMA.

O DMA pode ser configurado em dois modos de transferência de dados: modo direto, em que a transferência de dados é feita de forma imediata, ou modo *First In First Out (FIFO)*, em que os dados são armazenados temporariamente antes de serem transmitidos para a memória. Outras potencialidades do DMA surgem, no facto, de os apontadores se auto incrementarem, possibilitando a escrita de várias posições de memória consecutivas e a sua reprogramação ser automática (modo circular).

3.2.2 *Analog to Digital Converter (ADC)*

Um ADC é um circuito eletrónico que converte uma entrada analógica num valor digital que representa um valor de tensão num código binário. A STM32F767ZI [8] possui ADCs de aproximação sucessiva (composto por um comparador e um *Digital to Analog Converter (DAC)* interno para aproximar, sucessivamente, o valor de saída do ADC ao valor de entrada, Figura 3.3). Um ADC apenas realiza leituras corretas para valores de entrada compreendidos entre 0 V e 3,3 V.

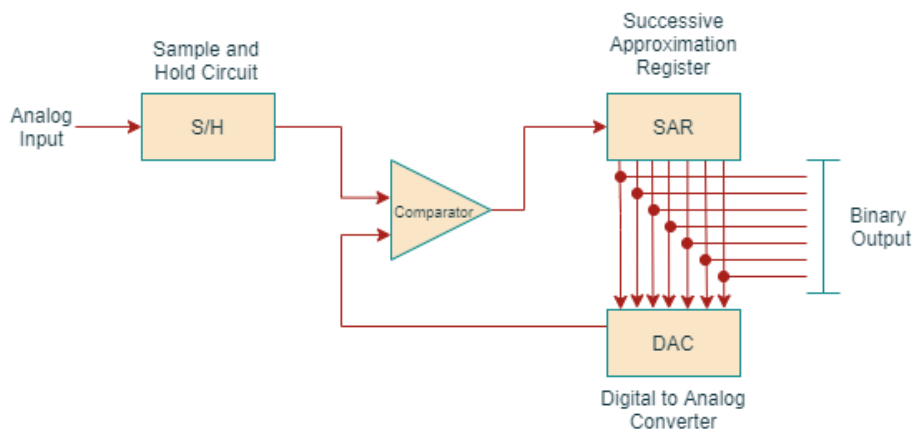


Figura 3.3 - Conversor por aproximação sucessiva.

O microcontrolador em uso possui três ADCs com múltiplos canais. Cada ADC pode ser configurado, no modo independente, como *Single-channel* ou *Multichannel*. No primeiro, apenas é lido um canal, ao passo que, no segundo, são lidos vários canais sucessivamente. Além disso, pode ser configurado nos modos de conversão simples ou contínua. No primeiro, o ADC realiza uma única conversão. No segundo, o ADC inicializa uma nova conversão logo que a conversão em curso termine. As possíveis combinações destes modos de operação estão apresentadas na Figura 3.4. É ainda possível configurar os ADCs em modo duplo/triplo e o DMA para guardar os dados.

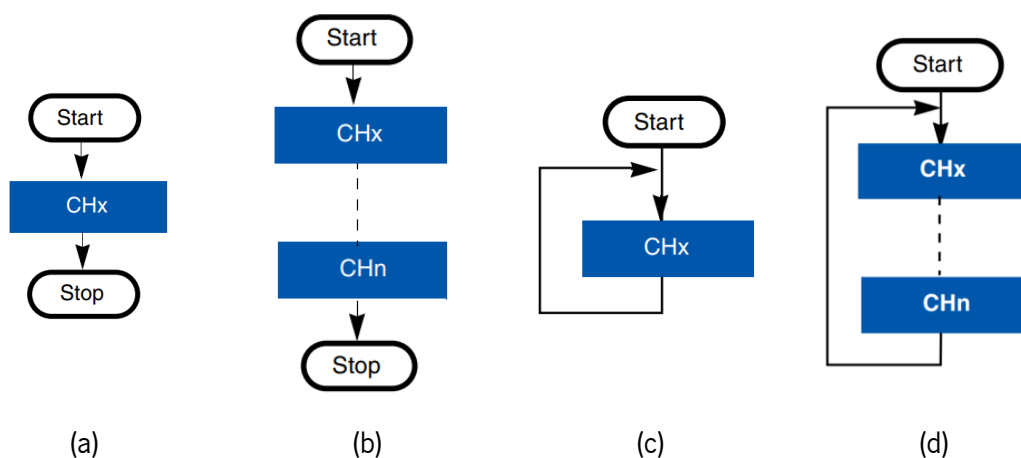


Figura 3.4 - Modo de funcionamento independente (a) Single-channel, single conversion mode; (b) Multichannel, single conversion mode; (c) Single-channel, continuous conversion mode; (d) Multichannel, continuous conversion mode.

Neste projeto, dado a natureza dos sensores adotados, foram utilizados dois ADCs com resolução de 12-bit e alinhamento dos dados à direita em conjunto com o DMA (DMA2), para transferência de dados entre os periféricos e a memória. O ADC2, em conjunto com a *stream2* do DMA, foi usado para a leitura do sensor de obstáculos, sendo apenas usado um dos seus canais, canal três (*single-channel*). O ADC3, em conjunto com a *stream1* do DMA, foi usado para efetuar a leitura dos sensores do QTR-8A, sendo, por isso, utilizados seis dos seus canais, três, seis, oito a dez e treze (*multichannel*). Como se pretende ler os vários sensores simultaneamente foi usado o modo de conversão contínua em ambos os ADCs.

3.2.3 TIMER (TIM)

Um *timer* é um dispositivo de hardware capaz de desencadear eventos a frequências conhecidas, gerar sinais a várias frequências, sinais de saída PWM e medir pulsos de entrada. Existem *timers* do tipo básico (TIM6 e TIM7), *general-purpose* (TIM2 ao TIM5 e TIM9 ao TIM14) e *advanced-control* (TIM1 e TIM8). Os *timers* básicos são os mais simples e são, normalmente, usados para gerar bases de tempo e gerar interrupções. Os *timers* do tipo *general-purpose* e *advanced-control* podem ser usados com vários propósitos, tais como medir o comprimento de pulsos de sinais de entrada e gerar ondas de saída. Na Tabela 3.1, pode ver-se o barramento a que cada *timer* pertence, bem como a frequência máxima do mesmo.

Tabela 3.1 - Mapeamento dos *timers*.

Timer	1	2	3	4	5	6	7	8	9	10	11	12	13	14
APB1(108 MHz)		x	x	x	x	x	x					x	x	x
APB2(216 MHz)	x							x	x	x	x			

Para configurar o *timer* com a frequência desejada é necessário configurar os valores de *prescaler* de *preload* fazendo uso da equação (3.1).

$$UpdateEvent = \frac{Timer_{clock}}{(Prescaler + 1)(Period + 1)} \quad (3.1)$$

Neste projeto, forem usados quatro *timers* para aplicações distintas. Para controlo dos *timeouts* que podem ocorrer durante o tempo de operação do DWR, do período de amostragem do algoritmo Proporcional Integrativo Derivativo (PID), sensor de obstáculos e do *debounce* apenas é necessário um *timer* que despolete uma interrupção ao fim de um determinado período de tempo, ou seja, um *timer* básico. Para o período de amostragem do algoritmo PID e do sensor de obstáculos utilizou-se o mesmo *timer*, TIM6, com uma base de tempo de 10 ms. Para o *debounce* do botão, usou-se o TIM7 com uma base de tempo de 50 ms. Como o microcontrolador apenas possui dois *timers* básicos, utilizou-se um *timer* genérico, TIM3, para gerir os *timeouts* com uma base de tempo de 1 segundo. Para gerar os sinais de PWM à saída do controlador PID é necessário um *timer* no modo PWM *Generation*. Sendo assim, seleccionou-se um *timer* genérico, TIM4, e utilizaram-se dois canais (*channel 3* e *channel 4*), de forma a gerar os sinais de PWM para cada motor.

3.2.4 Serial Peripheral Interface (SPI)

A *Serial Peripheral Interface* (SPI) é um protocolo de comunicação de curto alcance criado para a troca de dados entre microcontroladores ou entre microcontroladores e sensores. O SPI é síncrono e necessita que o transmissor e o recetor estejam sincronizados através de uma linha de *clock* partilhada. Este suporta transferências de dados em três modos: *full-duplex* (transferência de dados bidirecional simultânea), *half-duplex* (transferência de dados bidirecional não simultânea) ou *simplex* (transferência de dados unidirecional).

Os dispositivos conectados por SPI encontram-se numa relação *master-slave*. O *master* é, tipicamente, um microcontrolador que envia instruções ao *slave*, que, na maioria dos casos, se trata de um sensor, um *chip* de memória ou um display. De modo a proceder à comunicação *full-duplex* entre um dispositivo *master* e um dispositivo *slave*, as 4 linhas lógicas para a transferência de dados do protocolo SPI, apresentadas na Tabela 3.2, do *master* e do *slave* devem ser conectados segundo a configuração apresentada na Figura 3.5.

Tabela 3.2 - Linhas lógicas para a transferência de dados do protocolo SPI.

Serial Clock (SCLK)	Saída proveniente do master para sincronizar as transferências de dados nas linhas MISO e MOSI.
Master Out Slave In (MOSI)	Saída de dados do master com destino ao <i>slave</i> .
Master In Slave Out (MISO)	Saída de dados do <i>slave</i> com destino ao master.
Slave Select (SS)	Linha que permite seleccionar o dispositivo com o qual se pretende comunicar.

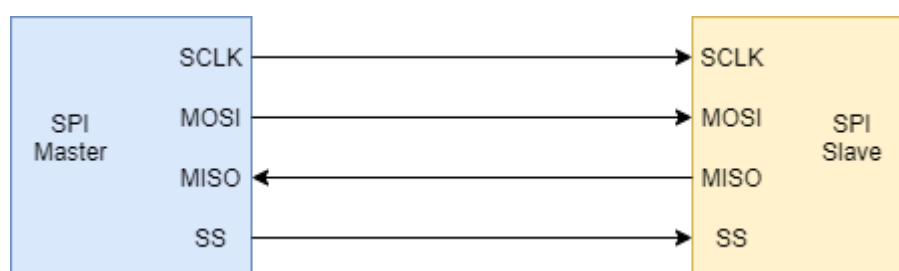


Figura 3.5 - Esquema de ligação entre *Master* e *Slave*.

Neste projeto, utilizou-se o protocolo SPI para comunicação com o chip MFRC522 [15]. Este atua como *slave* e suporta velocidades de transferência de dados até 10 Mbits/s. Para além disso, durante a receção e transmissão de dados, o módulo envia o bit mais significativo primeiro (MSB). Os dados transmitidos são estáveis durante a transição ascendente do *clock* e alterados durante a transição descendente.

Optou-se por utilizar a SPI3, dado que os pinos mapeados para esta interface não entram em conflito com pinos usados por outros periféricos. Foi configurada para funcionar com dados no formato Motorola e tamanho de 8-bit, em que o MSB é enviado primeiro. Configurou-se, ainda, o *clock* da interface para a transferência de dados ocorrer a 3,375 Mbits/s (sabendo que o *clock* do barramento APB1 é 54 MHz, o *prescaler* adquire o valor de 16), a polaridade do *clock* (CPOL) a *low* e a sua fase (CPHA) a zero. Deste modo, executa-se a captura dos dados provenientes do MFRC522 durante a transição ascendente.

3.2.5 Universal Asynchronous Receiver/Transmitter (UART)

A *universal asynchronous receiver transmitter* (UART) é uma comunicação série largamente usada para comunicar entre dois dispositivos, como sistemas embebidos, microcontroladores e computadores. Na comunicação UART, a comunicação direta entre dois dispositivos faz-se através de duas linhas de dados, a de transmissão (*TX*) e a de recessão (*RX*), como representado na Figura 3.6.

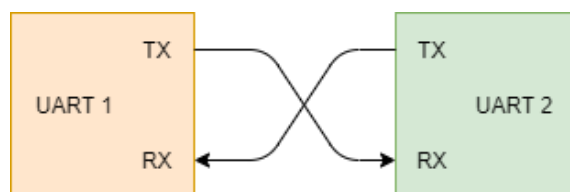


Figura 3.6 - Esquema de ligação entre 2 UARTs.

Este periférico transmite dados de forma assíncrona, ou seja, não necessita de um sinal de *clock* para se manter sincronizado. Em vez disso, o emissor acrescenta *bits* ao pacote a ser transmitido, sinalizando o início e o fim dos dados a ser transferidos, tal como mostrado na Figura 3.7.

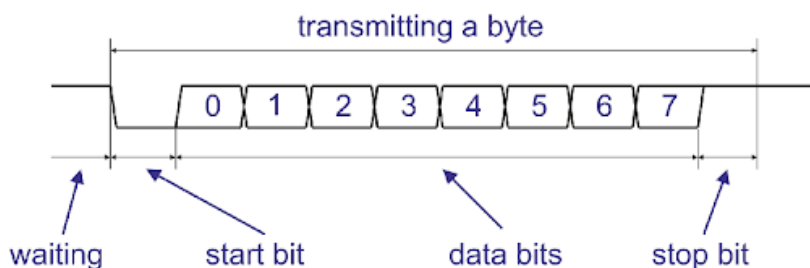


Figura 3.7 - Estrutura de um pacote de dados enviado por UART.

Quanto ao recetor, após detetar o *start bit*, começa a ler os bits que se seguem a uma determinada frequência, denominado *baud rate*, expresso em *bits* por segundo (bps). Para que a comunicação entre dois dispositivos possa ocorrer sem erros, ambas as UART têm de operar ao mesmo *baud rate*.

Na STM32F767ZI encontram-se disponíveis quatro módulos UART (UART4, UART5, UART7 e UART8) e quatro USART (USART1, USART2, USART3 e USART6). As últimas diferem das primeiras

apenas no facto de poderem ser configuradas no modo síncrono. Todos podem ser configurados em *full-duplex* (transferência de dados bidirecional simultânea) ou *half-duplex* (transferência de dados bidirecional não simultânea). A estrutura dos pacotes de dados pode ser modificadas, podendo escolher ter 7,8 ou 9 *bits* de dados, 1 ou 2 stop bits e uma definição da ordem da transmissão de dados.

Neste projeto, a UART possibilita a comunicação entre a STM e o módulo Bluetooth [16]. O formato da comunicação série deste módulo é de oito *bits* de dados, um stop *bit* e nenhum de paridade e opera a um *baud rate* de 9600 *bits/s*. Deste modo, configurou-se a USART1 com as características acima descritas e ativaram-se as interrupções para processamento dos dados

3.2.6 Mapeamento dos periféricos

Na Figura 3.8 apresenta-se o mapeamento dos periféricos usados no projeto.

CN7															
	UART6_T X PWM3/1 PC_6 PB_8 PWM4/3 UART5_R X I2C1_SCL CAN1_RX												PWM Right		
Bluetooth h	USB_OTG SPI2_MO UART4_C _HS_DP SI TS				UART1_R X	PWM1/3 N	PB_15	PB_9	PWM4/4	UART5_T X I2C1_SDA SPI2_CS CAN1_TX				IN1 Right	
	USB_OTG _HS_VBUS	USB_OTG _HS_ULPI D6	CAN2_TX	SPI2_SCK	UART3_C TS	UART5_T X	PWM1/1 N	PB_13	AVDD						
	USB_OTG _HS_ID	USB_OTG _HS_ULPI D5	CAN2_RX	SPI2_CS	UART5_R X			PB_12	GND						
	CAN3_TX SPI1_CS				UART4_R TS	UART7_T X	PWM2/1	PA_15	PA_5	PWM2/1	ADC1/5	DAC1/2	SPI1_SCK	USB_OTG _HS_ULPI _CK	IN2 Right
					UART6_R X	PWM3/2	PC_7	PA_6	PWM3/1	ACD1/6	SP1_MISO			IN1 Left	
	USB_OTG _HS_ULPI D7	CAN2_RX	SPI1_MO SI	UART5_R X		PWM3/2	PB_5	PA_7	PWM3/2	ADC1/7	SPI1_MO SI				
	CAN3_RX SPI1_SCK				UART7_R X	PWM2/2	PB_3	PD_14	PWM4/3	UART8_C TS				IN2 Left	
	USB_OTG _HS_SOF	SPI1_CS		DAC1/1	ADC1/4		PA_4	PD_15	PWM4/4	UART8_R TS				PWM Left	
	CAN3_TX		SPI1_MISO	SPI2_CS	UART7_T X	PWM3/1	PB_4	PF_12							

(a)

CN8										
			NC	PC_8	PWM3/3	UART5_R TS				Botão
			IOREF	PC_9	PWM3/4	UART5_C TS	I2C3_SDA	QSPI1_D0		RFID (RSET)
			NRST	PC_10	SPI3_SCK	UART4_T X		QSPI1_D1		RFID
			3V3	PC_11	SPI3_MIS O	UART4_R X				RFID
			5V	PC_12	SPI3_MO SI	UART5_T X				RFID
			GND	PD_2		UART5_R X				RFID (SDA)
			GND	PG_2						
			VIN	PG_3						

(b)

Sensor Obstatulos	USB_OTG_HS_ULPI_D0	ADC2/3	UART2_RX	PWM2/4	PA_3	PD_7	SPI1_MOSI		
Sensor 1	USB_OTG_HS_ULPI_STP	ADC3/10			PC_0	PD_6	SPI3_MOSI	UART2_RX	
Sensor 3	USB_OTG_HS_ULPI_NXT	ADC3/13		SPI2_MOSI	PC_3	PD_5	UART2_TX		
Sensor 4		ADC3/9			PF_3	PD_4	UART2_RSTS		
Sensor 5		ADC3/15			PF_5	PD_3	SPI2_SCK	UART2_CTS	
Sensor 6	QSPI_SCK	ADC3/8			PF_10	GND			
					NC	PE_2	SPI4_SCK	QSPI1_D2	
		ADC1/7	PWM1/1N	SPI1_MOSI	PA_7	PE_4	SPI4_CS		
					PF_2	PE_5	SPI4_MISO	PWM9/1	
		I2C2_SCL			PF_1	PE_6	SPI4_MOSI	PWM9/2	
		I2C2_SDA			PF_0	PE_3			
					GND	PF_8	SPI5_MISO	UART7_RSTS	
							ADC3/6	PWM13/1	QSPI1_D0
			UART4_RX	CAN1_RX	PD_0	PF_7	SPI5_SCK	UART7_TX	
							7	PWM11/1	QSPI1_D2
			UART4_TX	CAN1_TX	PD_1	PF_9	SPI5_MOSI	UART7_CTS	
							ADC3/7	PWM14/1	QSPI1_D1
					PG_0	PG_1			
CN9									

(c)

					AVDD	PF_13			
					AGND	PE_9	PWM1/1	UART7_RSTS	
Buffer					GND	PE_11	PWM1/2	SPI4_CS	
Buffer	USB_OTG_HS_ULPI_D2			ADC1/9	PWM1/3N	PB_1	PF_14	I2C4_SCL	
	USB_OTG_HS_ULP_DIR	SPI2_MISO		ADC1/12		PC_2	PE_13	PWM1/3	SPI4_MISO
				ADC3/14		PF_4	PF_15	I2C4_SDA	
Bluetooth	CAN2_RX	QSPI_CS	I2C_SCL	UART1_TX	PWM4/1	PB_6	PG_14	UART6_TX	SPI6_MOSI
		SPI3_MOSI		QSPI_SCK		PB_2	PG_9	UART6_RX	SPI1_MISO
					GND	PE_8	PWM1/1N	UART7_TX	
								UART7_RSTS	
				QSPI1_D3	I2C4_SDA	PWM4/2	PD_13	PE_7	UART7_TX
				QSPI1_D1	I2C4_SCL	PWM4/1	PD_12	GND	
				QSPI1_D0	UART3_RSTS		PD_11	PE_10	PWM1/2N
									UART7_CTS
				SPI4_SCK	QSPI1_D2	PE_2	PE_12	PWM1/3N	SPI4_SCK
						GND	PE_14	PWM1/4	SPI4_MOSI
				ADC1/10	UART2_CTS	UART4_TX	PWM2/1	PA_0	PE_15
	USB_OTG_HS_ULPI_D1			ADC1/8	UART4_CTS	PWM1/2N	PB_0	PB_10	PWM2/3
									QSPI1_CS
									IS2_SCL
									SPI1_SCK
									USB_OTG_HS_ULPI_D3
									USB_OTG_HS_ULPI_D4

(d)

Figura 3.8 - Mapeamento dos periféricos em função dos conectores: (a) CN7; (b) CN8; (c) CN9; (d) CN10.

3.3 Descrição de *Software* e Módulos Criados

3.3.1 Módulos Criados

Na criação de software torna-se indispensável a utilização de métodos de programação modular que consistem na divisão do código em diversos ficheiros, denominados de módulos. Esta abordagem permite uma estruturação clara e organizada do software, agrupando funções e variáveis relacionadas num mesmo ficheiro. Além disso, facilita a reutilização de funções e a manutenção de código. De notar que, para facilitar a tarefa de criação de *software* em equipa utilizou-se a plataforma GitHub [24], que proporciona a hospedagem de código fonte e utiliza a ferramenta de controlo de versões de ficheiros Git [25]. O código fonte e outros ficheiros gerados na criação deste projeto encontram-se num repositório público, acessível através do *link*: <https://github.com/LPI-2020/DWR-19>.

A Figura 3.9 representa a divisão do software criado nos vários módulos e as suas interações, em que, FSM significa *Finite State Machine* (Máquina de estados) e HAL significa *Hardware Abstraction Layer*. De notar que, tal como apresentado na legenda na Figura 3.9 os módulos estão agrupados por cores e as setas são usadas para representar as interações entre os módulos indicando a relação de dependência. Por exemplo, o módulo *FSM* depende do módulo *Motion*, mas *Motion* não depende do módulo *FSM*, em que *Motion* pertence ao grupo de módulos que controlam o movimento (*Movement Modules*).

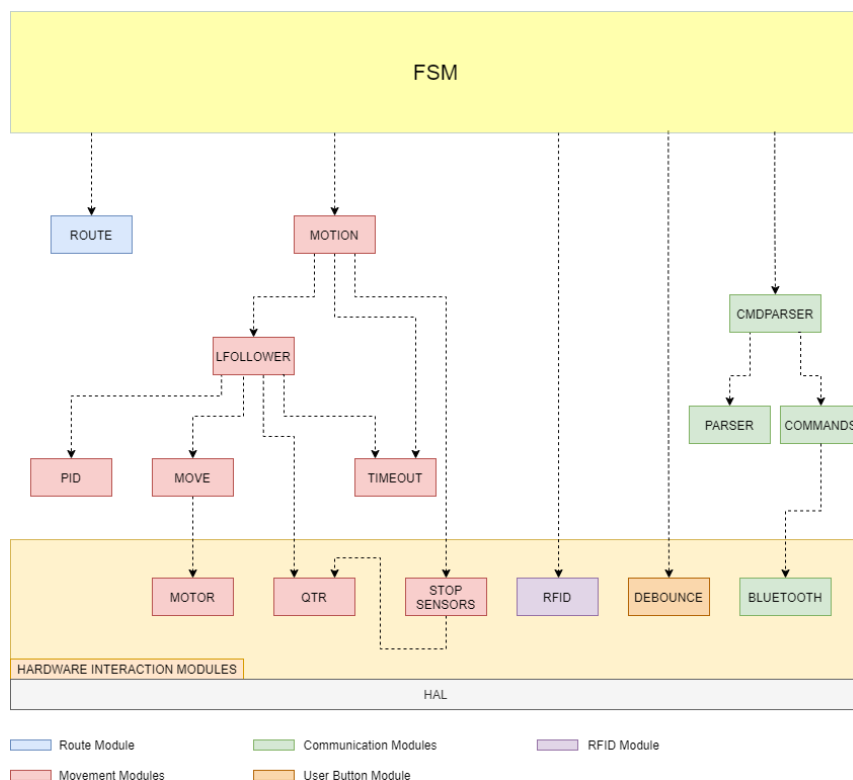


Figura 3.9 - Divisão do software criado nos vários módulos.

Começando pela descrição dos módulos que controlam o movimento (representados a cor avermelhada, na Figura 3.9), na camada de interação com o hardware existem três módulos: *Motor*, *QTR* e *Stop Sensors*. O módulo *Motor*, permite o controlo de um motor através de um *timer* com canal PWM, que efetua a variação da tensão de alimentação média do motor. O sentido de rotação do motor é definido por dois pinos de GPIO, o IN1 e o IN2, tal como apresentado na Tabela 2.2 do capítulo anterior. Na Figura 3.10 é apresentada a estrutura que define as saídas que controlam o motor.

```
typedef struct
{
    TIM_HandleTypeDef* htim;           // Motor PWM TIM instance
    uint8_t pwm_channel;               // Motor PWM channel

    GPIO_TypeDef *GPIO_port_IN1;       // Motor GPIO Port1
    uint16_t GPIO_pin_IN1;             // Motor GPIO Pin1

    GPIO_TypeDef *GPIO_port_IN2;       // Motor GPIO Port2
    uint16_t GPIO_pin_IN2;             // Motor GPIO Pin2
}motor_st;
```

Figura 3.10 - Estrutura que definem as saídas que controlam o motor.

O módulo QTR inicializa o *array* de sensores do seguidor de linha [10], possibilitando a obtenção do valor analógico e do valor lógico de cada um dos sensores, sendo o valor lógico alto definido a partir de 2,45 V. A leitura dos valores dos sensores do QTR é feita através do DMA. Na Figura 3.11, encontram-se representados, em forma de um enumerado, todos os sensores do *array* de sensores usados.

```
typedef enum {
    // QTR RIGHT side
    SENSOR1,
    SENSOR3,    // Line Follower RIGHT Sensor
    SENSOR4,
    // QTR Middle
    SENSOR5,
    SENSOR6,    // Line Follower LEFT Sensor
    SENSOR8
    // QTR LEFT side
} qtr_e;
```

Figura 3.11 - Enumerado que representa os sensores do QTR utilizados.

O módulo *Stop Sensors* permite detetar as marcas de paragem (cruzes no chão) e eventuais obstáculos que apareçam no percurso do DWR. Este módulo utiliza dois sensores do módulo QTR para fazer a deteção das marcas de paragem, *SENSOR1* e *SENSOR8*, e um sensor de obstáculos. Tal como no módulo *QTR*, a leitura dos valores do sensor de obstáculos é feita através do DMA. Definiu-se que o sensor de obstáculos sinaliza a presença de um objeto na trajetória do robô quando este se encontra a, aproximadamente, 15 cm de distância, correspondendo a um valor digital de 0x2000. Este módulo tem dois códigos de erro associados: *E_ST_CROSS_FOUND*, usado quando deteta uma marca de paragem, e *E_ST_OBS_FOUND*, usado quando deteta um obstáculo.

O módulo *Move* controla o movimento dos motores, definindo as suas velocidades e sentidos de rotação. Além disso, este módulo inicializa os dois motores a serem usados: motor direito e motor esquerdo.

O módulo PID implementa o algoritmo do controlador PID, tal como apresentado no Capítulo 3.3.2. Na Figura 3.12, é apresentada a estrutura que define as variáveis utilizadas para realizar o algoritmo PID.

```
typedef struct
{
    float kp_h;           // PROPORTIONAL constant with h (sampling time)
    float ki_h;           // INTEGRAL constant with h
    float kd_h;           // DERIVATIVE constant with h

    float error;          // current error
    float prev_error;     // previous error
    float sum_errors;     // sum of previous errors
    float sum_errors_bck; // backup of sum of previous errors

    float u_d;            // derivative component value
    float prev_u_d;       // previous derivative component value

    float u;              // command variable
    float u_sat_a;        // command variable saturation limit ABOVE
    float u_sat_b;        // command variable saturation limit BELOW
} pid_st;
```

Figura 3.12 - Estrutura que agrupa as variáveis de cálculo do algoritmo PID.

O módulo *Timeout* permite gerar quatro *timeouts*, com duração em segundos, como apresentado na Figura 3.13. Quando um *timeout* termina, a *flag* respetiva é ativa de forma a sinalizar o sucedido. Para isso, este módulo utiliza um *timer* que gera uma interrupção a cada segundo.

```
#define RFID_TIMEOUT      3 // RFID timeout time
#define ROTATE_TIMEOUT    4 // Rotate timeout time
#define PICK_UP_TIMEOUT   5 // Pick up timeout time
#define HOLD_TIMEOUT      5 // Obstacle presence (hold) timeout time

extern volatile uint8_t rfid_timeout; // RFID timeout flag
extern volatile uint8_t rotate_timeout; // Rotate timeout flag
extern volatile uint8_t pick_up_timeout; // Pick Up timeout flag
extern volatile uint8_t hold_timeout; // Hold timeout flag
```

Figura 3.13 - Duração dos vários *timeouts*, em segundos, e definição das *flags* respetivas.

O módulo *Lfollower* implementa o seguidor de linha, através dos sensores do QTR já mencionados, aplicando o algoritmo PID, utilizando o módulo *Move* para provocar uma alteração na velocidade de rotação dos motores. Além disso, este módulo implementa uma função que permite rodar o robô numa direção, direita ou esquerda, até que o sensor do QTR no lado correspondente à direção de rotação, *SENSOR1* e *SENSOR8*, respetivamente, detetem novamente a linha. Caso nenhum sensor detete a linha durante o movimento de rotação, este será parado ao fim de um *timeout* predefinido, *ROTATE_TIMEOUT* apresentado na Figura 3.13. Este módulo tem dois códigos de erro associados: *E_LF_OFF*, usado quando se tenta utilizar o seguidor de linha antes de o inicializar, e *E_LF_NO_LINE*, usado quando o seguidor de linha não encontra uma linha para seguir. Quando *ROTATE_TIMEOUT* termina, é utilizado o código de erro *E_TIMEOUT*.

O módulo *Motion* controla o movimento do robô utilizando o seguidor de linha, os sensores de paragem e o sensor de obstáculos. Na Figura 3.14, está representado um enumerado com os possíveis estados de movimento. Sempre que algum dos erros apresentados anteriormente acontece, é efetuada a mudança do estado do movimento para o estado respetivo.

```
typedef enum
{
    MOT_ON = 0,           // Motion ON (moving)
    MOT_OFF,             // Motion OFF (stopped)
    MOT_OK,              // Motion OK

    MOT_CROSS_FOUND,     // Motion: stopped at a cross
    MOT_HOLD,           // Motion in Hold (obstacle detected)

    MOT_ERR,             // Motion Error (Out of route)
    MOT_TIMEOUT          // Motion Timeout - too much time in MOTION_HOLD
} motion_status_e;
```

Figura 3.14 - Enumerado com os possíveis estados de movimento.

Este módulo utiliza um *timer* para provocar uma interrupção a cada 10 milissegundos, de forma a aplicar o seguidor de linha e verificar os sensores de paragem. Quando um obstáculo é detetado, o estado do movimento passa a ser MOT_HOLD, iniciando-se a contagem de um *timeout* com duração HOLD_TIMEOUT, tal como apresentado na Figura 3.13. Quando este *timeout* acaba, o estado do movimento passa para MOT_TIMEOUT. Quando não é detetada nenhuma linha durante o movimento, o estado do movimento passa para MOT_ERR.

O módulo RFID permite ler um cartão RFID, obtendo-se um CardID, a sua representação em *string* e o seu tipo, tal como apresentado na Figura 3.15 (a). O enumerado da Figura 3.15 (b) representa o estado do leitor RFID. Quando a leitura é bem-sucedida, o seu estado será MI_OK. Se houver um erro na leitura ou passar demasiado tempo após o início da leitura, o estado do leitor RFID será MI_ERR ou MI_TIMEOUT, respetivamente.

<pre>typedef struct { uint8_t CardID[4]; uint8_t type; char *CardID_str; } rfid_t;</pre> <p>(a)</p>	<pre>typedef enum { MI_OK = 0, // RFID good read MI_ERR, // RFID bad read MI_TIMEOUT, // RFID timeout } TM_MFRC522_Status_t;</pre> <p>(b)</p>
---	---

Figura 3.15 - Módulo RFID: a) Estrutura que define um cartão RFID; b) Estado do leitor RFID.

Relativamente aos módulos da comunicação, o módulo Bluetooth é responsável por receber e executar uma trama via UART, que está conectada a um dispositivo Bluetooth. Na Figura 3.16, está representado, em enumerado, o estado do módulo *Bluetooth*. Quando uma trama for recebida com sucesso o estado será BLUET_OK. Se uma trama estiver a ser recebida, o estado será BLUET_RECEIVING, enquanto quando o estiver pronto para receber, o estado será BLUET_READY.

```
typedef enum {
    BLUET_OK,           // Bluetooth received
    BLUET_READY,        // Bluetooth ready
    BLUET_RECEIVING,    // Bluetooth receiving
    BLUET_OFF           // Bluetooth not initialized
} bluet_state_t;
```

Figura 3.16 - Estado do módulo Bluetooth.

O módulo *Commands* define a lista de comandos válidos para esta aplicação. Na Figura 3.17 está representada a estrutura que define um comando, sendo composta pela *string* que define o comando, uma *string* com um texto de ajuda para o comando e a função que o executa (*callback*).

```
typedef char (*Command_cb)(uint8_t, char *[]);

typedef struct Command
{
    const char *cmd;           // the command string to match against
    const char *help;          // the help text associated with cmd
    Command_cb fn;             // the function to call when cmd is matched
} Command_t;
```

Figura 3.17 - Definição de um callback de um comando e da estrutura que define um comando.

O módulo *Parser* permite analisar uma trama, dividindo-a em diferentes *strings* a partir de um delimitador. Além disso, se a trama for um comando válido, este módulo executa a função relativa a este comando. O módulo *Cmd Parser* permite interpretar um comando, definido no módulo *Commands*, através do módulo *Parser*, e executá-lo.

O módulo *Route* permite criar uma rota com vários pontos de paragem (*checkpoints*). Assim, na Figura 3.18, está representada a estrutura que define um ponto de paragem numa rota, contendo uma *string* com o identificador do *checkpoint* (RFID) e a ação a executar nesse ponto de paragem (*action*).

```
typedef enum{
    ACT_RIGHT,           // turn right (at a cross)
    ACT_LEFT,            // turn left (at a cross)
    ACT_FORWARD,         // don't stop (at a room)
    ACT_STOP,            // stops (at a room)
    ACT_BACKWARD         // backwards
} action_e;

typedef struct{
    char *RFID;           // RFID: checkpoint unique identifier
    action_e action;      // Action to execute at the checkpoint
} checkpoint_t;
```

Figura 3.18 - Ações possíveis a realizar num *checkpoint*, e, definição de um *checkpoint*.

O módulo *Debounce* efetua o *debounce* de um botão de pressão associado a um pino GPIO, por software, através de um algoritmo janela deslizante, tal como mostra a Figura 3.19.

```
typedef struct
{
    uint8_t window;           // sliding window
    uint8_t count1s;         // count 1s in window
} slide_window_t;
```

Figura 3.19 - Definição de uma janela deslizando.

Uma janela deslizando é um algoritmo que permite armazenar um conjunto de valores, neste caso, um conjunto de *bits* numa janela de 8 *bits* (*window*), e também, saber quantos *bits* a nível lógico alto existem na janela. Como definido na estrutura apresentada na Figura 3.20, o módulo Debounce utiliza uma janela deslizando (*sw*) para que, quando o número de *bits* a nível lógico alto for superior a um certo valor, menor do que o tamanho da janela, coloque o *pin_output*, que representa o sinal resultante da aplicação do *debounce* ao pino GPIO, a nível lógico alto. Para isso, este módulo utiliza um *timer* que gera uma interrupção a cada 50 milissegundos.

```
typedef struct
{
    slide_window_t sw;        // Debounce Sliding Window

    GPIO_TypeDef* GPIOx;     // GPIO button port
    uint16_t GPIO_Pin;       // GPIO button pin

    uint8_t pin_output;      // Button output
} debounce_t;
```

Figura 3.20 - Definição da estrutura *debounce*.

O módulo FSM implementa a máquina de estados, descrita no Capítulo 3.1, que controla a evolução do estado de funcionamento do robô. O enumerado ilustrado na Figura 3.21 apresenta os possíveis estados de funcionamento do DWR.

```
typedef enum
{
    S_STOPPED = 0, // Robot stopped. Waiting for interaction
    S_RECEIVE,     // Robot is receiving a new route

    S_FLW_LINE,    // Robot is following line
    S_RD_RFID,     // Robot detects and reads RFID

    S_NEXT_MOV,    // Robot determines next movement to make
    S_ROTATE,      // Robot rotates in order to move in other direction

    S_ERROR        // Robot waiting for intervention
} state_e;
```

Figura 3.21 - Estados da máquina de estados.

Os estados são executados de forma contínua, até que um estímulo específico faça a máquina de estados transitar para outro estado. Para isso, criou-se um *array* de funções, *fsm_func_ptr*, com as funções que implementam os diferentes estados da máquina de estados, tal como mostra a Figura 3.22 (a). Este *array* de funções é acedido com base no valor do estado atual, *state*. Após a execução do estado, o valor do próximo estado, *nstate*, é atribuído ao estado atual, como apresentado na Figura 3.22 (b).

<pre>void (*fsm_func_ptr[])(void) = { s_stopped, s_receive, s_flw_line, s_rd_rfid, s_next_mov, s_rotate, s_error };</pre> <p style="text-align: center;">(a)</p>	<pre>while (1) { // execute state fsm_func_ptr[state](); // update current state state = nstate; }</pre> <p style="text-align: center;">(b)</p>
---	--

Figura 3.22 - Definição da máquina de estados (a) Array de funções de estado; (b) Execução da máquina de estados.

Quando a máquina de estados transita para o estado de erro, é aberta a possibilidade ao utilizador, de usar a aplicação de interface para aceder à funcionalidade de controlo remoto do DWR.

De forma a implementar o controlo remoto, o utilizador pode servir-se de quatro botões da aplicação que permitem controlar a direção do DWR, tal como será apresentado na Figura 3.24 (d). Na Figura 3.23, apresenta-se a lógica desenvolvida para implementar o controlo remoto. Quando o utilizador pressionar um botão, é enviada uma trama para o DWR. Consoante a trama recebida, o DWR executará o comando respetivo proporcionando o movimento do DWR no sentido desejado. Assim, quando o utilizador usa o botão que ordena o robô a andar em frente, a trama recebida é interpretada como ACT_FORWARD. Utilizando o botão que ordena o DWR a andar para trás, o robô recebe uma trama interpretada como ACT_BACKWARD. De igual forma, quando o utilizador ordenar o robô a mover-se para a esquerda ou direita, o utilizador pode utilizar os dois botões restantes, enviando tramas que serão interpretadas como ACT_LEFT e ACT_RIGHT, respetivamente.

```
switch(remote_ctrl_dir)
{
    case ACT_RIGHT:
        // move right
        move_rotate(MOVE_RIGHT, REMOTE_CTRL_SPEED);
        break;

    case ACT_LEFT:
        // move left
        move_rotate(MOVE_LEFT, REMOTE_CTRL_SPEED);
        break;

    case ACT_FORWARD:
        // move forward
        move_forward(REMOTE_CTRL_SPEED);
        break;

    case ACT_BACKWARD:
        // move backwards
        move_backwards(REMOTE_CTRL_SPEED);
        break;

    case ACT_STOP:
        // stop movement
        move_forward(0);
}
```

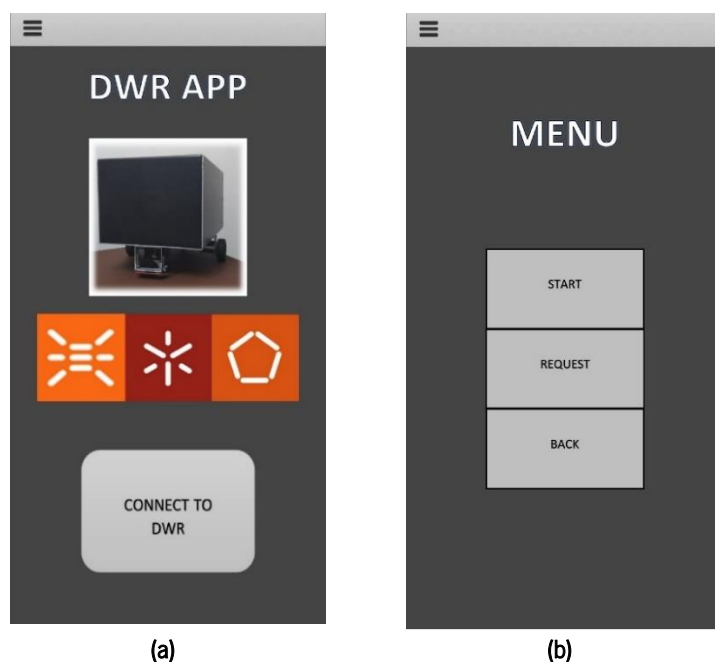
Figura 3.23 - Implementação do controlo remoto.

3.3.2 Aplicação de Interface

De modo a ser possível realizar as tarefas de inicialização do DWR e gestão dos percursos a efetuar decidiu-se criar uma aplicação de interface com o utilizador. Além destas tarefas, indispensáveis ao funcionamento do robô, como o controlo remoto, a aplicação servirá também o propósito de informação de estado do DWR, tais como, o último cartão de RFID lido e erros que possam ocorrer durante a sua atividade.

A aplicação foi implementada recorrendo à ferramenta *MIT App Inventor* [26], um software simples e útil para o desenvolvimento de aplicações para *smartphones*. O ecrã inicial da aplicação desenvolvida está representado na Figura 3.24 (a), no qual é pedido para o utilizador se conectar ao DWR. Para isso, o utilizador terá de ativar a funcionalidade *Bluetooth* do seu próprio dispositivo e emparelhá-lo com o dispositivo *Bluetooth* do DWR [16]. Depois de emparelhado, é necessário pressionar o botão “*Connect to DWR*”.

Com a conexão estabelecida, o funcionário terá acesso ao menu principal da aplicação, apresentado na Figura 3.24 (b). Para realizar um pedido, deve ser pressionado o botão “*Request*” e escolher, a partir da lista apresentada, qual a rota a efetuar. A utilização do botão “*Back*” permite voltar ao ecrã inicial da aplicação. O funcionário poderá iniciar a marcha do robô, pressionando o botão “*Start*”. Após pressionar este botão, o utilizador receberá as mensagens enviadas pelo DWR, tal como mostra a Figura 3.24 (c). Como apresentado em capítulos anteriores, se o funcionário desejar pode dar início à marcha do robô através do botão de pressão, presente na lateral do DWR, sem utilizar o botão “*Start*” na aplicação. Na Figura 3.24 (d), está representado o ecrã de interface para o controlo remoto.



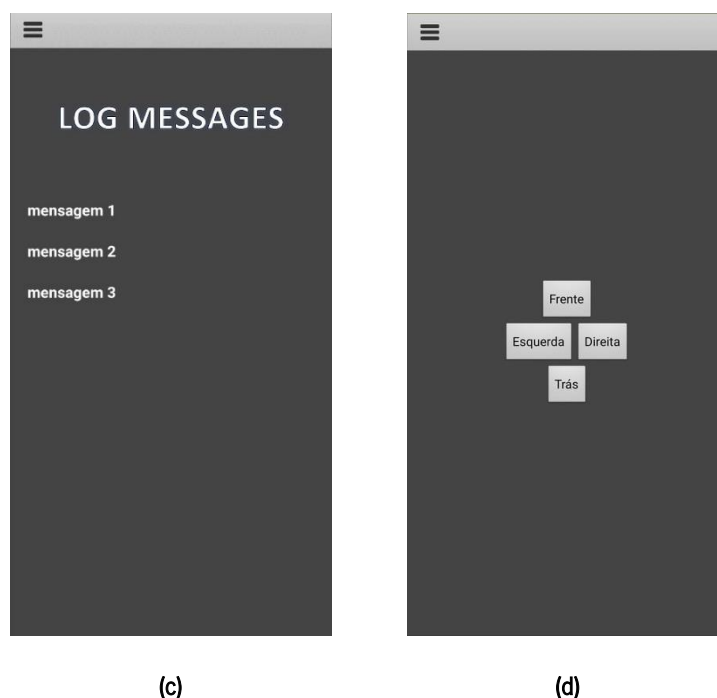


Figura 3.24 - Aplicação Interface (a) Ecrã inicial; (b) Menu principal; (c) Terminal de mensagens; (d) Ecrã controlo remoto

3.3.3 Circuito de Controlo

Para a implementação do módulo do seguidor de linha é necessário o desenvolvimento e implementação de um controlador que permita utilizar os valores lidos pelos sensores do QTR-8A de forma atuar sobre os motores.

3.3.3.1 O que é um Controlador?

Um controlador é responsável pelo controlo de processos através de algoritmos específicos. O seu principal objetivo consiste na monitorização, identificação e interpretação de processos, via modelos matemáticos, de forma a produzir uma ação de controlo conveniente. Existem três ações de controlo distintas: proporcional, integral e derivativa, que podem ser conjugadas entre si. A primeira tem uma ação imediata, proporcional ao valor atual do erro, acelera a resposta de um processo controlado, reduz o tempo de subida e o erro máximo. No entanto, aumenta o *overshoot*, o tempo de estabilização e produz um offset inversamente proporcional ao ganho. A ação integral produz uma ação de controlo gradual proporcional à integral do erro, respondendo, assim, ao passado do erro enquanto este for diferente de zero, elimina o offset e reduz o tempo de subida. Porém, aumenta o *overshoot*, o período de oscilação e tempo de estabilização, produzindo respostas lentas e oscilatórias. A ação derivativa produz uma ação antecipatória e proporcional à derivada do erro. É usada para acelerar e estabilizar a malha, reduzir o *overshoot*, o erro máximo e o período de oscilação. Contudo, não é indicada para processos com ruído.

Na Figura 3.25, estão presentes as três ações de controlo descritas. A Figura 3.25 (a) mostra a resposta de um sistema a uma ação proporcional. A saída deste corresponde à variável de erro multiplicada por uma dada constante. A Figura 3.25 (b) mostra a resposta de um sistema a uma ação integral, onde a saída deste corresponde à integral da variável de erro. Como a integral de uma constante é uma reta com declive não nulo, quando a entrada (variável de erro) é do tipo degrau, a resposta do sistema vai corresponder a uma rampa de declive igual à amplitude da variável de entrada multiplicada por uma constante. A Figura 3.25 (c) mostra a resposta de um sistema a uma ação derivativa, em que a saída deste corresponde à derivada da variável de erro. Como a derivada de uma reta é uma constante, quando a entrada é do tipo rampa, a saída do sistema corresponde a um degrau de amplitude igual à amplitude da rampa multiplicada por uma constante. Um controlador que conjugue as três ações é denominado por controlador proporcional PID.

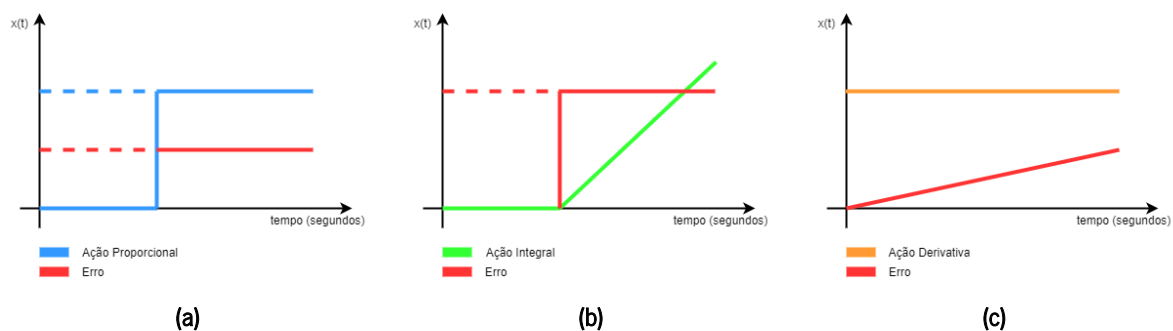


Figura 3.25 - Ações de controlo (a) ação proporcional; (b) ação integral; (c) ação derivativa.

3.3.3.2 Análise do sistema de controlo

O primeiro passo num projeto de controlo está relacionado com a compreensão qualitativa do sistema. É necessário compreender como o sistema a controlar funciona fisicamente, quais as variáveis de medida, a controlar, de atuação, distúrbio e comando, bem como o funcionamento dos atuadores, sensores e controlador.

O sistema a ser controlado assemelha-se a um paralelepípedo com uma roda posicionada na parte central de cada lateral do robô. Sendo o objetivo do sistema seguir uma linha, que pode conter trajetórias retas ou curvilíneas, a velocidade de rotação dos motores terá de variar de modo a ser possível ajustar as velocidades de translação e de rotação do centro de massa do robô. Se se pretender que o robô siga uma trajetória retilínea, os motores terão de rodar à mesma velocidade. Se se pretender que o robô efetue uma trajetória curvilínea, o motor do lado oposto ao que se pretende efetuar a trajetória terá de ter uma velocidade de rotação superior. Ou seja, quando a trajetória é uma curva à esquerda, o motor do lado direito terá de ter uma velocidade de rotação superior comparativamente com a do motor do lado esquerdo. Quando a trajetória é uma curva à direita, passa-se o oposto.

As variáveis de medida são as leituras efetuadas pelos dois sensores do seguidor de linha, o sensor 3 e o sensor 6, Figura 2.2. As variáveis a controlar são a velocidade de translação e velocidade de rotação do centro de massa do robô. As variáveis de atuação são os binários dos motores. As variáveis de comando são a fração de modulação do amplificador PWM de cada motor. A variável de perturbação será o atrito provocado pela superfície (binário de perturbação).

O atuador é composto pelo motor DC [22], a ponte H [13] e pelo microcontrolador [8]. O binário produzido pelo motor DC pode ser alterado através da variação da tensão de alimentação do motor (produzida por um amplificador PWM).

O sistema de controlo é responsável pela variação deste parâmetro, de maneira a produzir o binário adequado à velocidade requerida para que o DWR siga a linha corretamente. Em conjunto, o microcontrolador e a ponte H implementam o amplificador PWM. O algoritmo de controlo implementado no microcontrolador produzirá na saída a variável de comando, a fração de modulação do amplificador de PWM, sendo o ganho deste amplificador correspondente a tensão aplicada ao motor DC quando a fração de modulação é um.

O controlador é efetuado através do microcontrolador. A regra de controlo executada por este controlador está implementada numa rotina de serviço à interrupção (ISR), que será despoletada por um *timer* com período igual ao período de amostragem escolhido pelo sistema de controlo. Esta ISR terá como parâmetros de entrada os valores das leituras dos dois sensores a utilizar e como saída a fração de modulação de PWM, que servirá de entrada aos amplificadores PWM.

Na Figura 3.26, está ilustrado o sistema de controlo implementado. Sendo o objetivo principal do DWR o seguimento de uma linha, é necessário manter os sensores na parte exterior da mesma. Quando um dos sensores se aproximar da linha, o motor do lado oposto terá de compensar o desvio da trajetória. Pretende-se que a diferença entre as leituras dos dois sensores, ou seja, a variável de erro, seja nula, portanto, conclui-se que a variável de referência tem o valor zero.

Se o valor de erro for positivo, significa que o sensor esquerdo se encontra mais próximo da linha do que o sensor direito, implicando que o motor direito tenha uma velocidade de rotação superior. À variável de saída do bloco controlador PID soma-se um valor de *offset* que servirá de variável de comando ao atuador direito. Ao simétrico da variável de saída do bloco PID soma-se o mesmo valor de *offset*, que servirá de variável de comando ao atuador esquerdo. Deste modo, o valor da variável de comando do atuador direito será superior ao valor da variável de comando do atuador esquerdo.

Se o valor do erro for negativo, passa-se o oposto. Se a variável de saída do controlador PID for nula não é necessário fazer ajustes de direção e o DWR seguirá o percurso com velocidade de rotação constante em ambos os motores, com valor igual ao valor de *offset*.

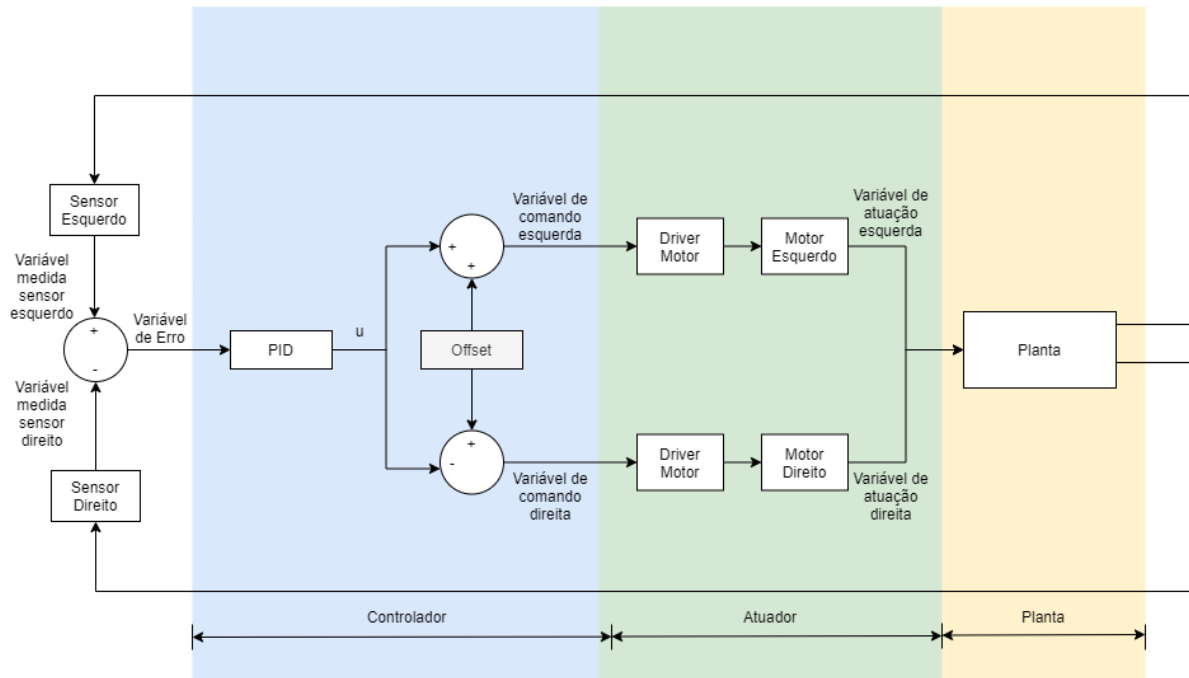


Figura 3.26 - Diagrama de blocos do sistema de controlo.

3.3.3.3 Controlador implementado

Um controlador PID contínuo é definido pela equação (3.2), em que k_p , k_i e k_d são referidos, respetivamente, como os ganhos proporcional, integral e derivativo, e $e(t)$ como a variável de erro ao longo do tempo. A variável de comando, $u(t)$, corresponde ao somatório das ações proporcional, integral e derivativa.

$$u(t) = k_p e(t) + k_i \int_0^t e(t') dt' + k_d \frac{de(t)}{dt} \quad (3.2)$$

Para uma implementação em microcontrolador é necessária uma versão digital do algoritmo de controlo. De entre as diversas famílias das regras PID implementou-se o algoritmo de posição. Esta versão consiste em substituir os termos integral e derivativo, da equação (3.2), pelos seus equivalentes discretos: soma e a diferença dividida de 1ª ordem, respetivamente. Assim, obtém-se a equação (3.3), em que $u[n]$ e $e[n]$ são, respetivamente, a ação de controlo e o erro no instante n . Em cada instante é calculado o valor real (de posição) do sinal de saída do controlador. No contexto do problema, a variável de erro corresponde à diferença entre as leituras dos dois sensores, aqui representados, simbolicamente, por IN_1 e IN_2 , equação (3.4).

$$u[n] = k_p e[n] + k_i \sum_{k=0}^{n-1} e[k] + k_d (e[n] - e[n-1]) \quad (3.3)$$

$$e[n] = IN_1 - IN_2 \quad (3.4)$$

Para efeitos de cálculo, o valor de $u[n]$ pode ser escrito de forma mais adequada, calculando o erro apenas uma vez e colocando em evidência as constantes na aproximação à integral e à derivada (equação (3.5)). Pode-se, agora, redefinir os ganhos proporcional, integral e derivativo, equações (3.6), (3.7) e (3.8), respectivamente, obtendo-se, por substituição, a equação (3.9).

$$u[n] = \frac{k_d}{h} (e[n] - e[n-1]) + k_p e[n] + k_i h \sum_{k=0}^{n-1} e[k] \quad (3.5)$$

$$k_{ph} = k_p \quad (3.6)$$

$$k_{ih} = k_i h \quad (3.7)$$

$$k_{dh} = \frac{k_d}{h} \quad (3.8)$$

$$u[n] = k_{dh} (e[n] - e[n-1]) + k_{ph} e[n] + k_{ih} \sum_{k=0}^{n-1} e[k] \quad (3.9)$$

Para calcular o somatório dos erros, presente na equação (3.9), define-se a variável $sum_e[n]$, equação (3.10), que pode ser escrita de forma recursiva, equação (3.11). Juntando todas as transformações efetuadas, obtém-se, a equação (3.12), uma nova versão da expressão inicial, equação (3.3).

$$sum_e[n] = \sum_{k=0}^{n-1} e[k] \quad (3.10)$$

$$sum_e[n] = sum_e[n-1] + e[n-1] \quad (3.11)$$

$$u[n] = k_{dh} (e[n] - e[n-1]) + k_{ph} e[n] + k_{ih} sum_e[n] \quad (3.12)$$

Uma forma de reduzir os problemas com erros de medida prende-se com a utilização de um filtro passa-baixo na ação derivativa. Na equação (3.13), apresenta-se a expressão desta ação. Calculando a sua transformada em \mathcal{Z} obtém-se a equação (3.14). A aproximação à derivada introduz um zero em $z = 1$ e um polo em $z = 0$, no plano \mathcal{Z} . Pode-se deslocar o polo para a direita, no plano \mathcal{Z} , com um filtro passa-baixo resultando as equações (3.15) a (3.17). A função de transferência tem um zero em $z = 1$, como no caso anterior, mas o polo situa-se em $z = a$, e não em $z = 0$. A implementação desta função de transferência está representada na equação (3.18). Pode-se então redefinir k_{dh} como apresentado na equação (3.19). Adicionando a componente do filtro passa-baixo à equação (3.12), chega-se à

equação (3.20). A segunda reduz-se à primeira se a for zero, ou seja, a ação derivativa sem filtro de passa-baixo é obtida com $a = 0$, o que implica que $k_{dh} = k_d/h$.

$$u_d[n] = \frac{k_d}{h} (e[n] - e[n-1]) \quad (3.13)$$

$$U_d(z) = k_d \frac{z-1}{h z} E(z) \quad (3.14)$$

$$U_d(z) = k_d \frac{(1-a)z}{z-a} \frac{z-1}{h z} E(z) \quad (3.15)$$

$$U_d(z) = \frac{k_d(1-a)}{h} \frac{z-1}{z-a} E(z) \quad (3.16)$$

$$U_d(z) = \frac{k_d(1-a)}{h} \frac{1-z^{-1}}{1-az^{-1}} E(z) \quad (3.17)$$

$$u_d[n] = a u_d(k-1) + \frac{k_d(1-a)}{h} [e[n] - e[n-1]] \quad (3.18)$$

$$k_{dh} = \frac{k_d(1-a)}{h} \quad (3.19)$$

$$u[n] = k_{ph} e[n] + k_{ih} \text{sum}_e[n] + a u_d(k-1) + k_{dh} [e[n] - e[n-1]] \quad (3.20)$$

Um dos problemas de algoritmo de posição está relacionado com a possível saturação do valor de u . Definem-se, então, limites de saturação superior (U_{sata}) e inferior (U_{satb}). Se o resultado do cálculo do valor de u for superior ao valor do limite de saturação superior, ou inferior ao valor do limite de comparação inferior, é necessário limitar o mesmo aos valores de saturação impostos. Assim, modifica-se a interrupção para detetar se o u transpôs os valores de saturação e, caso tal aconteça, fixa-se o valor de u nos mesmos. Além disso, a saturação do atuador pode dar origem a um aumento brusco dos valores dos somatórios dos erros levando a oscilações prejudiciais no valor da variável controlada.

Tendo em conta os aspetos mencionados acima e fazendo uso da equação (3.20), é possível desenhar o fluxograma que permite a implementação do algoritmo de controlo no microcontrolador, Figura 3.27. Como se pode verificar, a atualização do somatório dos erros só é efetuada quando a variável u se encontra dentro dos limites de saturação. Caso contrário, a atualização é anulada, o que equivale, em tempo contínuo, a parar a integração do erro, fixando-se o valor de u no valor de saturação respetivo.

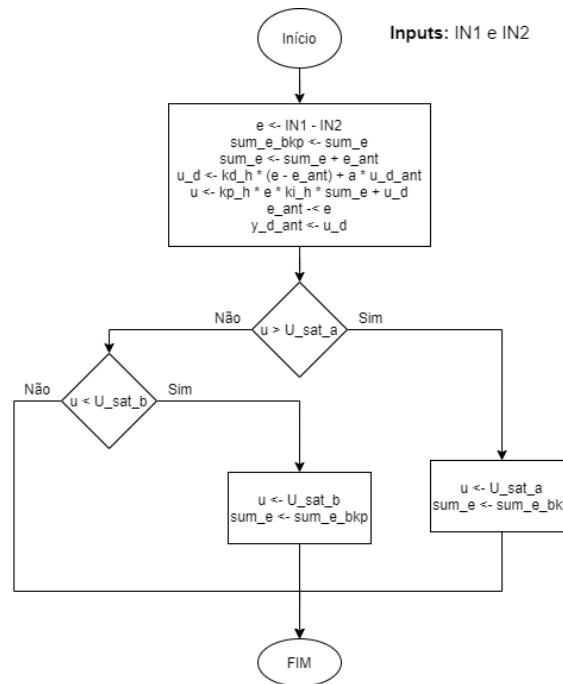


Figura 3.27 - Serviço de Rotina à Interrupção do algoritmo PID.








Tendo em conta que, para um valor da variável de comando baixo, os motores estão parados, torna-se necessária a soma de um *offset* a esta variável, de forma que, quando o erro for nulo, ou seja, quando $u = 0$, a variável de comando não seja nula. Por exemplo, para um valor de *offset* igual a 0,7 e $u = u_{sata}$, a variável de comando $V_{comandoR}$ será igual a $0,7 + 0,3 u_{sata}$ e a variável de comando $V_{comandoL}$ será igual a $0,7 - 0,3 u_{sata}$. Desta forma, garante-se que a variável de comando nunca será nula.

Capítulo 4



Lista de Componentes










Na Tabela 4.1, listam-se todos os componentes usados para o DWR, bem como a quantidade e o preço associado.








Tabela 4.1 - Lista de componentes.

	Material		Loja	Quantidade	Preço unitário	Preço total
1	Motor Bot'n Roll ONE 300 rpm		Bot'n Roll	2	17,50 €	35,00 €
2	Par de rodas Bot'n Roll ONE A		Bot'n Roll	1	11,50 €	11,50 €
3	Par de hubs Bot'n Roll ONE A		Bot'n Roll	1	4,50 €	4,50 €
4	Rodas livres (D25 mm)		Leroy Merlin	2	1,69 €	3,38 €
5	QTR-8A		Bot'n Roll	1	19,90 €	19,90 €
6	Sensor Sharp 2Y0A21YK		Bot'n Roll	1 ^(*)	11,95 €	11,95 €
7	Módulo RFID RC522		Bot'n Roll	1	4,90 €	4,90 €

	Material		Loja	Quantidade	Preço unitário	Preço total
8	Módulo Bluetooth HC-05		Bot'n Roll	1	6,80 €	6,80 €
9	Módulo <i>driver</i> de motores L298N		Bot'n Roll	1	12,90 €	12,90 €
10	Fonte de alimentação 3,3 V / 5 V		Bot'n Roll	1	6,5 €	6,5 €
11	Step-Down para USB 5 V		Bot'n Roll	1	9,70€	9,70€
12	Cabo USB A p/ micro USB B		Bot'n Roll	2	1,60 €	3,20 €
13	Suporte para uma pilha 18650 c/fios		Bot'n Roll	6	0,85 €	5,10 €
14	BMS para proteção baterias		Bot'n Roll	2	5,50 €	11,00 €
15	Pilha LI-ION 18650 3,7 V 2200 mAh		Bot'n Roll	6	3,90 €	23,40 €
16	Carregador baterias 18650		Bot'n Roll	1	8,90 €	8,90 €
17	Cabo DC com conector reto 9,5 mm		Bot'n Roll	1	1,70 €	1,70 €
18	Suporte Fusível Auto		AlfaElektor	1	1,22 €	1,22 €

	Material		Loja	Quantidade	Preço unitário	Preço total
19	Fusível Auto 3 A		AlfaElektor	1	0,10 €	0,10 €
20	STM32F767ZI		Bot'n Roll	1	30,90 €	30,90 €
21	PCB Shield STM32F767ZI		Guimo circuito	1	63,96 €	63,96 €
22	Conectores Buchanan 284517-4		TE Connectivity	12 (*)	1,19 €	14,26 €
23	Conectores Buchanan 284507-4		TE Connectivity	12 (*)	2,77 €	33,24 €
24	Barra (2x8) fêmea 2,54 mm PCB		Mauser	1	0,30 €	0,30 €
25	Barra (2x10) fêmea 2,54 mm PCB		Mauser	1	0,40 €	0,40 €
26	Barra (2x16) fêmea 2,54 mm PCB		Mauser	1	0,62 €	0,62 €
27	Barra (2x17) fêmea 2,54 mm PCB		Mauser	1	0,69 €	0,69 €
28	Barra (2x3) fêmea 2,54 mm PCB		Mauser	2	0,16 €	0,32 €
29	Barra (1x6) fêmea 2,54 mm PCB		Mauser	1	0,09 €	0,09 €

	Material		Loja	Quantidade	Preço unitário	Preço total
30	Bloco terminal 12 conetores		Gmlux	1	0,40 €	0,40 €
31	Condensador poliéster		AlfaElektor	4	0,10 €	0,40 €
32	Botão de painel 12 mm		Bot'n Roll	1	0,60 €	0,60 €
33	Interruptor de painel redondo		Bot'n Roll	1	1,20 €	1,20 €
34	Alumínio 2 mm		-	-	35,00 €	35,00 €
35	Placa madeira		-	1	5,00 €	5,00 €
36	Placa de acrílico policarbonato		-	1	10,00 €	10,00 €
37	Parafuso 4 mm		Ferritrofa	50	0,05 €	2,50 €
38	Parafuso 3 mm cabeça chata		Ferritrofa	13	0,05 €	0,65 €
39	Parafuso 6 mm		Ferritrofa	6	0,05 €	0,30 €
40	Parafuso 3 mm auto-roscante		Ferritrofa	6	0,04 €	0,24 €

	Material		Loja	Quantidade	Preço unitário	Preço total
41	Porca com orelhas 4 mm		Ferritrofa	8	0,10 €	0,80 €
42	Porca 4 mm		Ferritrofa	32	0,05 €	1,60 €
43	Porca 3 mm		Ferritrofa	13	0,05 €	1,15 €
44	Rebites		Ferritrofa	18	0,10 €	1,80 €
45	Anilhas		Ferritrofa	12	0,01 €	0,12 €
46	Kit Conectores 2,54 mm		Bot'n Roll	<i>pack</i>	10,95 €	10,95 €
47	Conjunto condutores flexíveis (1 m)		Bot'n Roll	4	1,00 €	4,00 €
Total:						368,14 €
Nota: (*) Empréstados ou arrançados como <i>samples</i> , sem custos pelos alunos.						

Capítulo 5

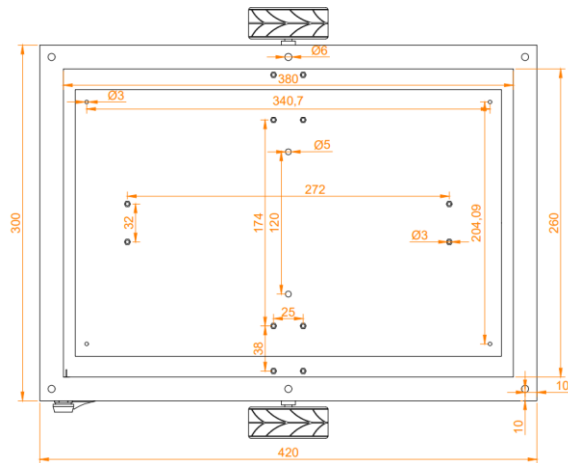
Circuito Mecânico Implementado

O objetivo do DWR é transportar um tabuleiro com alimentos e/ou medicamentos. Sendo a forma dos tabuleiros tradicionais retangular, optou-se por definir a forma da armadura do robô como um paralelepípedo, com dimensões aproximadas às dos tabuleiros: 300 mm de largura por 420 mm de comprimento. De forma a ser possível aceder e ver os circuitos no interior do DWR, usou-se, na parte superior da armadura, uma placa de acrílico policarbonato transparente e removível. Para a construção da restante armadura usou-se uma folha de alumínio de 2 mm. Para a fixação de todas as peças, usaram-se parafusos de 3 mm e 5 mm, porcas de 5 mm, porcas com asas de 5 mm, anilhas e rebites.

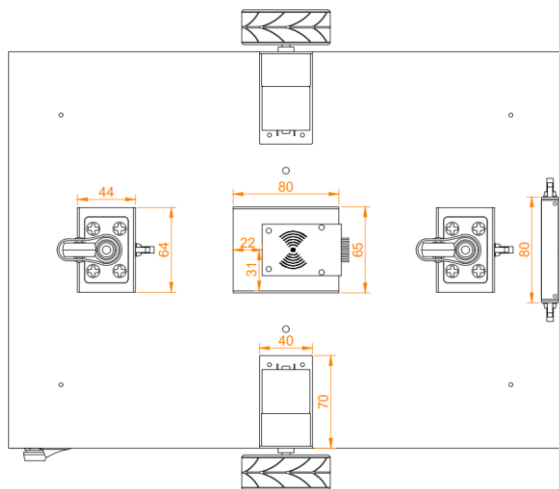
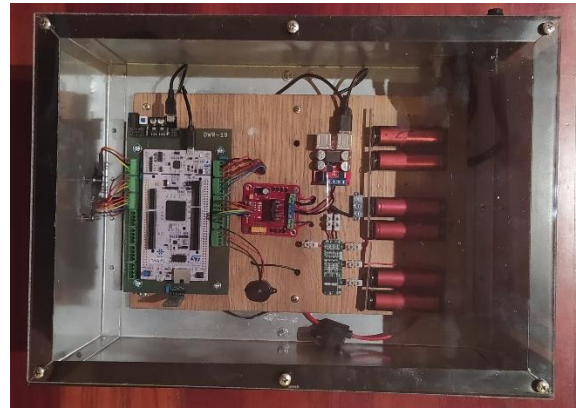
Relativamente às rodas, foram usadas duas rodas motoras e duas rodas livres. As rodas motoras, com 67 mm de diâmetro, são usadas nas laterais do robô e estão acopladas aos dois motores [22]. Os seus pneus são de borracha, com 26,5 mm de largura. As rodas livres (com 25 mm de diâmetro), tal como o próprio nome indica, rodam livremente em qualquer sentido e têm o propósito de dar estabilidade ao robô. Os suportes destas rodas foram colocados na dianteira e traseira da armadura, permitindo o ajuste da distância das rodas livres ao chão. Foi usado um suporte de altura ajustável para prender o *QTR-8A* [10] à armadura do robô. Como o *array* de sensores deve estar na parte inferior da dianteira do robô, o suporte deste foi fixado na dianteira da armadura. Na dianteira do DWR foi, também, colocado o sensor de obstáculos [12], permitindo detetar obstáculos que surjam à sua frente. O suporte, de altura ajustável, para o leitor RFID [15] está posicionado no centro da parte inferior da base do robô. O botão de interação com o utilizador, de 12 mm de diâmetro, está colocado na parte superior esquerda da lateral direita da armadura do robô. Na parte inferior da lateral direita, existe, ainda, a entrada do carregador das baterias [21] e um botão ON/OFF [27], com 20 mm de diâmetro, que permite desligar o DWR.

Na Figura 5.1 (a), é possível observar a vista superior do robô, onde a borda de 20 mm da armadura permite fixar a placa de acrílico através de parafusos. Na Figura 5.1 (b), mostra-se a fixação dos suportes das rodas, motores e sensor de linha e leitor RFID. Na Figura 5.1 (c), é possível observar o mecanismo de ajuste da altura das rodas livres, do sensor de linha e do leitor RFID. Além disso, vê-se

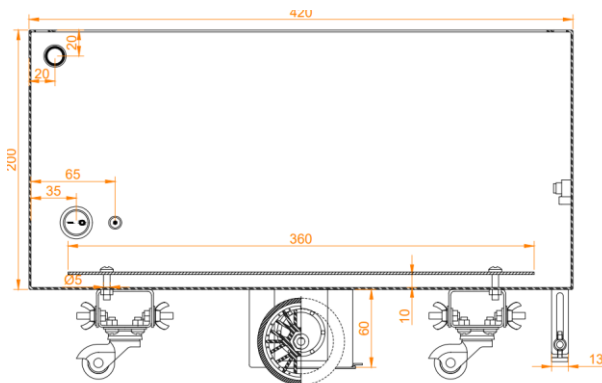
uma placa colocada no fundo da armadura com o propósito de suportar todos os circuitos e impedir contactos destes com a armadura. No canto superior esquerdo da armadura do DWR, está localizado o botão de interação com o utilizador e no canto inferior esquerdo estão localizados a entrada de carregamento e o botão *ON/ OFF*. Na Figura 5.1 (d) pode ver-se a vista dianteira do robô, onde se verifica a presença do sensor de obstáculos, e, na Figura 5.1 (e), a vista traseira do robô, respetivamente. Na Figura 5.2 apresenta-se uma imagem 3D do DWR.



(a)

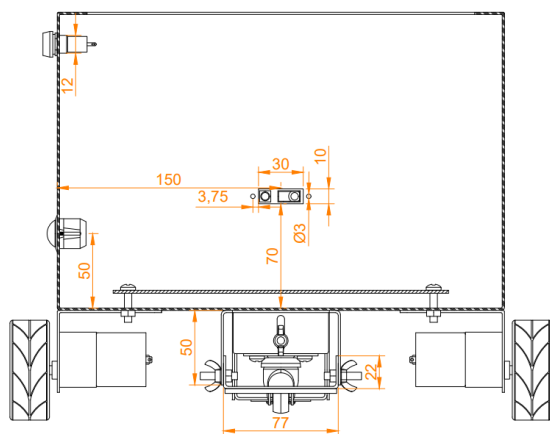


(b)

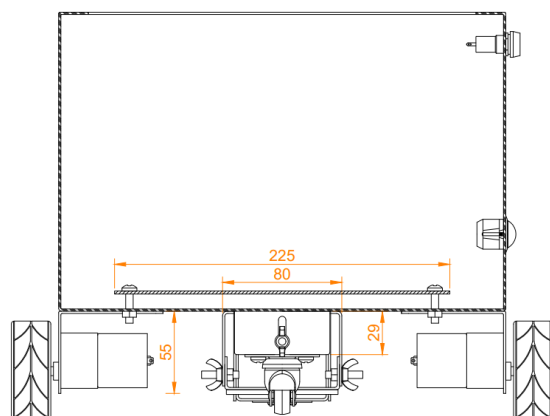


(c)





(d)



(e)

Figura 5.1 - Desenho (à esquerda) e imagem real (à direita) do DWR (a) vista superior; (b) vista inferior; (c) vista lateral direita; (d) vista dianteira; (e) vista traseira.

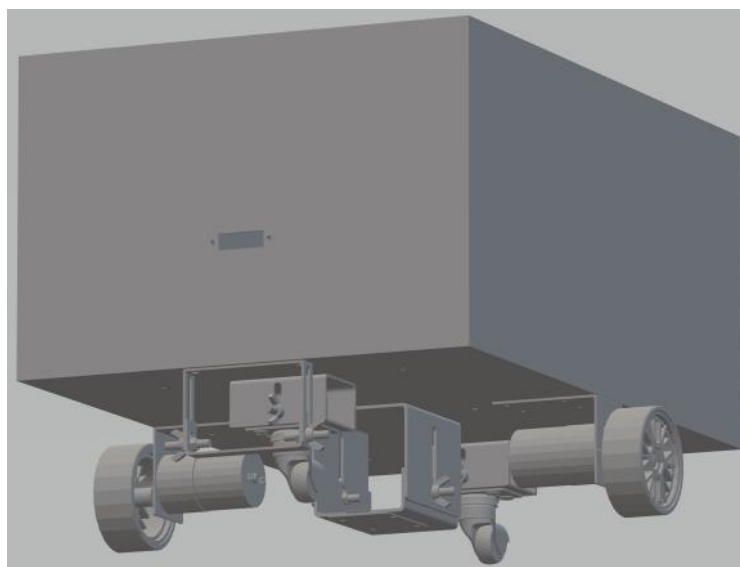


Figura 5.2 - Vista 3D do DWR

Capítulo 6

Resultados Experimentais

De forma a testar as especificações acima previstas, foram realizados ensaios experimentais simulando o ambiente no qual o robô irá operar. Começou-se por testar o sensor e verificou-se que as medidas dos sensores não atingem nem 0 V nem 3,3 V.

Para verificar qual a zona de funcionamento dos motores, apurou-se qual a gama de valores de *duty cycle* do sinal PWM necessários para garantir a operação dos mesmos. Concluiu-se que o motor apenas possui binário de arranque suficiente quando o *duty cycle* do sinal PWM ronda os 65 %. Já com o motor em funcionamento, o *duty cycle* pode atingir um mínimo de 50 % sem que os motores parem. Estes valores permitiram ajustar o bloco de offset do controlador do módulo seguidor de linha, estabelecendo-se um valor de 70 %.

Os parâmetros do controlador PID foram obtidos através de métodos heurísticos. Verificou-se que aplicando um ganho proporcional muito superior a um, a variável de atuação do controlador aproxima-se da saturação, impedindo a realização de trajetórias curvas. Os ganhos integral e derivativo têm valores muito menores ao valor do ganho proporcional. O ganho integral é importante, principalmente, na realização de curvas mais acentuadas, uma vez que tem em conta o valor dos erros anteriores. Ao longo do tempo, a velocidade de rotação do motor que tem de fazer a compensação da trajetória aumenta e a do motor contrário reduz, quando o DWR se encontra fora da trajetória ideal. No entanto, um valor elevado no ganho integral provoca a saturação do controlador, impedindo a realização de trajetórias curvas. Definiu-se uma gama de valores válida para o ganho integrativo entre 0,5 e 1. O ganho derivativo também é importante na realização de curvas mais acentuadas, uma vez que faz com que o sistema responda mais cedo à variação e assim manter-se na trajetória. Estabeleceu-se o valor máximo para o ganho derivativo como 0,1, sendo que, para valores superiores a este, o DWR apresenta instabilidade no movimento, mesmo em linha reta. É de referir que se usou um período de amostragem de 10 ms. Na prática, verificou-se que o ajuste da altura das rodas livres e dos sensores do DWR em relação ao chão influenciam os resultados obtidos, relativamente ao seguidor de linha. Estas variações fazem com que

os parâmetros do controlador PID sejam invalidados, pelo que as rodas livres e os sensores devem estar a uma distância constante ao chão.

O sensor de obstáculos utilizado permite a deteção de objetos a distâncias compreendidas entre os 10 e 80 cm. De forma a ser possível definir uma distância mínima para o sensor de obstáculos sinalizar a deteção de um obstáculo, traçou-se o gráfico da Figura 6.1, colocando uma folha de papel branca a diferentes distâncias do sensor de obstáculos, registando-se o valor digital adquirido através de um ADC do microcontrolador, STM32.

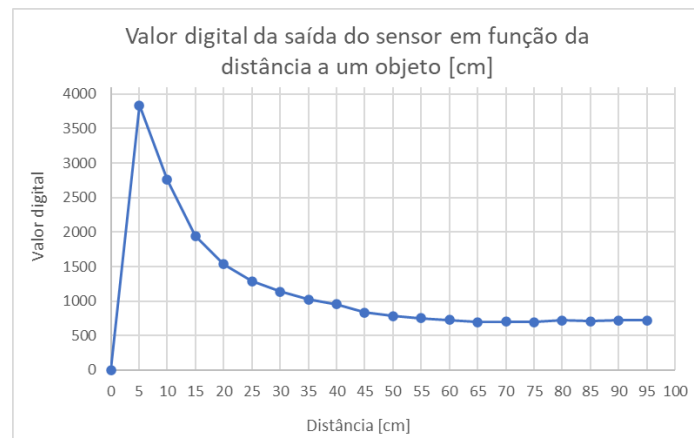


Figura 6.1 - Valor digital da saída do sensor de obstáculos em função da distância a um objeto.

Considerando que a curva característica obtida indica que o sensor apresenta uma maior gama de valores de saída entre os 5 e 25 cm, definiu-se assim, o valor da distância mínima para o sensor de obstáculos sinalizar a deteção de um obstáculo, o valor digital 0x2000, o equivalente à distância 15 cm, aproximadamente.

De forma a simular o ambiente no qual o DWR será utilizado, realizaram-se diferentes testes, dois dos quais registados em vídeo. No primeiro ([DWR-19 | Rotas Programadas](#)), executou-se uma sequência de duas rotas. Após a seleção da primeira rota e autorização de início de marcha, o DWR começa o seu trajeto. Ao encontrar o primeiro quarto, para, retomando a marcha quando se pressiona o botão lateral. Quando chega ao primeiro cruzamento, roda à esquerda e segue a linha até encontrar o próximo quarto. Neste ponto, o robô para e volta a arrancar quando acabar o tempo máximo de recolha do pedido, percorrendo o percurso até à origem. Assim, demonstra-se que ambos os métodos de arranque após paragem num quarto funcionam. Ao chegar ao ponto de partida, é selecionada uma nova rota. Neste trajeto, o DWR não para no primeiro quarto e no cruzamento segue em frente. Apenas para ao chegar ao último quarto no corredor, regressando, de seguida, ao ponto inicial. No segundo ([DWR-19 | Teste de perturbação](#)), efetuou-se testes mais severos de forma a observar o comportamento controlador do módulo seguidor de linha. Começou-se por aplicar uma força constante, podendo observar-se que o

controlador a conseguiu compensar, não produzindo grande oscilações. De seguida, aplicou-se forças não constantes, mas de maior intensidade, verificando-se que numa fase inicial as compensou. No entanto, a partir de certa intensidade, acaba por sair da sua rota, parando devido à não deteção da linha. Comprova-se, assim, que em caso de falha do seguidor de linha, garante-se que o DWR não provocará danos catastróficos, pois é enviado para o estado de erro. No terceiro ([DWR-19 | Teste de perturbação](#)), demonstra-se a funcionalidade de controlo remoto a ser usada em caso de erro.

Capítulo 7

Análise do Produto

7.1 Introdução

Em todos os projetos práticos é necessária uma análise do produto, em áreas como a fiabilidade, segurança e certificação do sistema. Esta análise permite identificar pontos de falha do equipamento e os potenciais perigos a estes associados, de forma a informar o utilizador.

7.2 Fiabilidade

Um sistema ou equipamento diz-se *“fiável quando está livre de erros catastróficos, é capaz de recuperar de erros e apresenta resultados previsíveis (determinismo)”* [28]. Apesar de, neste caso, não ser possível fazer uma avaliação do tipo quantitativa, realizou-se uma avaliação do tipo qualitativa por meio do estudo dos modos de falha e as suas consequências para o sistema e, também, influência do ambiente e do tempo.

Relativamente aos modos de falha, o desenho do sistema deve ter em conta as condições em que o robô se desvia da linha preta. Assim, este deverá ter uma velocidade adequada, uma vez que o sistema pode não responder suficientemente rápido a uma variação. Para diminuir a probabilidade de erros no seguimento da linha, deve ser favorecido o uso de linhas retas em todo o percurso. Em acrescento, a grossura da linha deve ser ligeiramente inferior à distância entre os dois sensores usados para seguir a linha, ou seja, 5 cm no máximo.

Do ponto de vista do ambiente que envolve o robô, devem ter-se em conta as condições climáticas, interações com o utilizador, condições do equipamento e obstrução da via de passagem do robô. O percurso não deverá ser obstruído com objetos, sob pena de danificar tanto o robô como o objeto que o obstrui ou pessoas que se encontrem próximas. Apesar de o DWR ter um sensor de obstáculos, este não cobre toda a área frontal do robô, não garantindo, por isso, total fiabilidade neste quesito. Além disso, sendo o robô um equipamento eletrónico, este não deverá estar sujeito a um ambiente húmido, tal como água ou neve, sujidade e temperaturas extremas. As condições anteriores podem provocar um

mau funcionamento dos circuitos constituintes do robô, podendo, em situações extremas, danificar permanentemente o equipamento.

Em relação ao tempo de operação, sabe-se que os componentes têm um tempo médio de vida previsto pelos fabricantes. Visto que não existe informação suficiente para calcular o consumo dos circuitos do DWR, torna-se difícil fazer uma previsão do tempo de vida dos seus componentes com exatidão. Num circuito eletrónico, os componentes com maior probabilidade de falha são os condensadores, semicondutores, baterias e motores. Para quantificar o número de horas previstos de funcionamento médio de um componente, existe um parâmetro denominado *Mean Time Between Failures* – MTBF. Os componentes enunciados têm os MTBFs apresentados na Tabela 7.1. Anormalidades nos circuitos poderão diminuir estes tempos consideravelmente, exigindo uma manutenção ou reparação precoce do robô.

Tabela 7.1 - Tempo de vida dos componentes com maior probabilidade de falha, usados no AWR.

Componente	MTBF
Baterias 18650	300 – 500 ciclos
Semicondutores	10 anos
Condensadores Poliéster	11,4 anos
Motores	1 000 a 3 000 horas

Como calculado no Capítulo 2.6, a autonomia prevista do robô é de duas horas e dez minutos. Se o DWR for usado, por exemplo, na distribuição de alimentos, assumindo uma hora de funcionamento por período de refeição e seis períodos por dia, a bateria deverá ser recarregada, aproximadamente a cada dois períodos de refeição, ou seja, três vezes por dia. Calcula-se, assim, que as baterias terão de ser substituídas ao fim de, aproximadamente, três meses podendo durar, no máximo, até cinco meses e meio.

Conclui-se que as baterias deverão ser os elementos que requerem maior atenção, pois, além das deficiências acima apresentadas, com o aumento do tempo de utilização, estas descarregam, conduzindo a alterações dos resultados práticos.

7.3 Segurança

A interação do robô com o utilizador deve ser realizada em segurança. Os sistemas elétricos e fios condutores não devem estar em contacto com a armadura do robô que, sendo metálica, é condutora, evitando curto-circuitos ou choques elétricos que podem ser fatais ao sistema e prejudiciais à saúde do

utilizador. Para garantir o isolamento da armadura do robô, usou-se, por questões económicas e para validação de conceitos, uma placa de madeira para suportar todos os circuitos do robô. Os motores, o *array* de sensores e leitor RFID, que ficam no exterior da armadura, estão expostos ao utilizador. De forma a isolar os motores eletricamente, usou-se fita isoladora envolvendo todos os terminais destes. Quanto ao *array* de sensores e ao leitor RFID, estes são ligados por conectores do tipo *Dupont* [29], garantindo o isolamento. Portanto, todos os componentes do robô encontram-se encapsulados e isolados do utilizador. Todavia, sabe-se que este equipamento é um dispositivo sensível à eletricidade estática (frequentemente abreviado como ESD – *Electrostatic-Sensitive Device*), o que significa que possui componentes que podem ser danificados por cargas elétricas estáticas que se acumulam em pessoas, ferramentas e outros materiais não condutores [30]. Se o utilizador violar o encapsulamento do produto, poderá representar perigos para este ou uma avaria no produto (Figura 7.2 (h)).

Deverá ter-se em conta que o robô possui elementos que apresentam perigos para o utilizador. A madeira usada para isolar os circuitos eletrónicos da armadura do robô é considerada um material comburente (Figura 7.2 (c)) e inflamável (Figura 7.2 (d)). As baterias usadas nos circuitos do DWR, são componentes inflamáveis (Figura 7.2 (d)), explosivos (Figura 7.2 (e)), tóxicos (Figura 7.2 (f)), perigosos para o ambiente (Figura 7.2 (a)), corrosivos (Figura 7.2 (b)) e comburentes (Figura 7.2 (c)). Assim, o robô não deverá estar exposto a qualquer condição ambiental extrema, tal como referido no subcapítulo anterior relacionado com a fiabilidade. Devido aos perigos acima descritos, deve ser vigiado o manuseamento do robô por crianças.

7.4 Certificação

Nenhum produto poderá ser exposto ao comércio antes de ser certificado. A certificação é o modo pelo qual uma entidade competente dá uma garantia escrita de que um produto está em conformidade com os requisitos especificados.

A marcação CE, apresentada na Figura 7.1, é um indicativo de conformidade obrigatória para diversos produtos comercializados no Espaço Económico Europeu. Esta marca indica que um produto respeita a legislação da União Europeia em requisitos como segurança, higiene e proteção ambiental, estando, desta forma, credenciado a circular por todo Espaço Económico Europeu [31].



Figura 7.1 - Marcação CE.

De acordo com DIRETIVA 2014/35/UE [32] ANEXO I, que informa os principais elementos dos objetivos de segurança para o material elétrico usado no equipamento, devem ser previstas medidas de ordem técnica a fim de que:

- a) As pessoas e os animais domésticos fiquem protegidos de forma adequada contra os riscos de ferimentos ou de outros acidentes resultantes de contactos diretos ou indiretos;
- b) Não se produzam temperaturas, descargas ou radiações que possam provocar perigo;
- c) As pessoas, os animais domésticos e os bens sejam protegidos de forma adequada contra os riscos de natureza não elétrica provenientes do material elétrico que a experiência venha a revelar;
- d) O isolamento seja adequado aos condicionamentos previstos.

Como já foi apresentado, o robô cumpre com os pontos a), b) e d). Uma vez que este projeto serve fins académicos, ou seja, tem o propósito de validar conceitos, apenas foi implementado um sensor de obstáculos na parte dianteira, não garantindo que não embata no objeto, visto que não cobre todo o raio de ação do DWR. Assim, o ponto c) não é cumprido, pelo que o certificado CE não poderá ser atribuído ao DWR.

Tratando-se o DWR de um veículo autónomo, este deve cumprir os requisitos de segurança ditados pelas normas europeias EN ISO-12100 [33] e ISO-3691-4:2020 [34], referentes a veículos autónomos. Estes documentos definem vários requerimentos, tais como o nível de segurança do produto, estudo dos riscos e perigos, condições de operação, documentação necessária para a sua utilização e integração, bem como as funcionalidades do sistema de controlo. Dado que o DWR não cumpre algumas destas normas, este não pode obter o certificado europeu de máquinas autónomas.

Devido aos perigos enunciados no Capítulo 7.2, relacionado com a segurança, deverão ser apresentados vários símbolos ao utilizador.

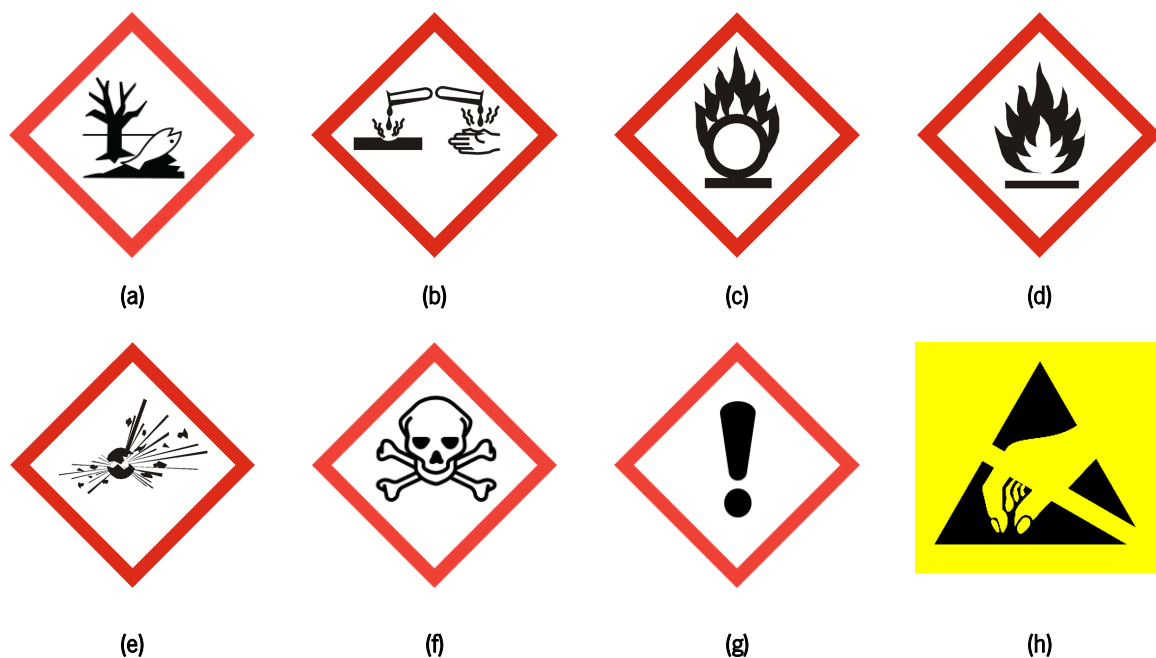


Figura 7.2 - Símbolos de perigo: (a) perigoso para o ambiente; (b) corrosivo; (c) comburente; (d) inflamável; (e) explosivo; (f) tóxico; (g) vários perigos; (h) *Eletrostatic Sensitive Device - ESD*.

O robô será constituído por circuitos e componentes eletrónicos que não podem ser enviados para o lixo doméstico. Estes devem ser encaminhados para locais próprios de reciclagem (Figura 7.3).

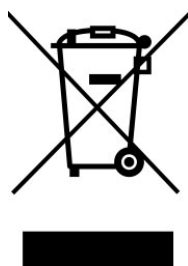


Figura 7.3 - Símbolo WEEE.

Capítulo 8

Conclusões

8.1 Conclusão

O projeto integrador da unidade curricular de LPI II pôs à prova algumas competências adquiridas ao longo do curso. Aplicaram-se conhecimentos das unidades curriculares das áreas de controlo, eletrónica, instrumentação e sensores, entre outras. Foi um projeto ímpar comparativamente a projetos já realizados porque envolveu um maior conhecimento de várias áreas e requereu muito trabalho autónomo.

Ao longo da conceção do projeto foram sugeridas várias ideias. Algumas delas, descartadas numa fase inicial, ao passo que outras apenas foram postas em causa já durante o desenvolvimento do projeto. Num primeiro momento, pensou-se em fazer a distinção dos quartos e cruzamentos através de um sensor de cores. No entanto, além de ser uma abordagem que poderia trazer resultados pouco precisos, era muito limitadora em relação ao número de combinações únicas. Pensou-se, ainda, em usar a tecnologia RFID apenas nos cruzamentos. As marcas de deteção de quartos seriam detetadas por um dos sensores do *array* do QTR-8A, reduzindo o número de etiquetas RFID que seriam necessárias adquirir. Com base no número de quartos detetados saber-se-ia o quarto em que seria necessário fazer paragem. Porém, chegou-se à conclusão que era uma abordagem que acrescentaria alguma complexidade ao sistema e era mais passível de falhas de leitura ao longo do período de operação do DWR, muito difíceis de identificar e controlar. Para evitar estes problemas, estabeleceu-se que se iria usar a tecnologia RFID em todos os cruzamentos e quartos, permitindo uma identificação precisa e um número de combinações únicas praticamente infinitas.

A criação da PCB *Shield* possibilitou a validação da máquina de estados, eliminando o ruído proveniente de maus contactos existentes nas ligações dos sensores e leitores à STM32. A criação desta foi tardia, despendendo-se muito tempo no tratamento dos valores lidos pelos sensores, que se verificou serem a maior fonte de maus contactos e ruído. Assim, torna-se importante que a PCB *Shield* seja criada na fase inicial de um projeto. Além disso, a utilização do *step-down* demonstrou-se útil pois permitiu a

utilização de um cabo USB para alimentar a STM32 durante o tempo de operação do robô, visto que este cabo já seria usado para transferir dados e programar a STM32.

Quanto ao desenvolvimento de software, revelou-se importante a realização de testes unitários a cada módulo desenvolvido, permitindo validar cada um destes antes da sua integração num projeto final, agilizando a identificação de erros. Este projeto permitiu, ainda, abordar tecnologias de comunicação sem fios ainda não estudadas, como o Bluetooth e o RFID.

Relativamente ao tempo investido no desenvolvimento deste projeto, visto que todo o grupo se juntou para trabalhar em horários definidos, todos os elementos do grupo trabalharam o mesmo número de horas. Após o desenho e conceção que foi desenvolvido em conjunto, o grupo foi dividido em três equipas de trabalho, sendo que uma se dedicou à implementação do controlo dos motores, outra ao desenvolvimento da máquina de estados que controla o estado de funcionamento do robô e outra ao desenvolvimento das comunicações sem fios. Assim, foram contabilizadas, individualmente, um total de 200 horas, como mostrado na Tabela 8.1. Em suma, este projeto foi desafiante, revelando a sua importância na formação enquanto futuros engenheiros.

Tabela 8.1 - Número de horas investidas por elemento.

Nome	Número de Horas
Bruno Silva	200
Diogo Fernandes	200
Duarte Rodrigues	200
Francisco Salgado	200
João Miranda	200
José Abreu	200

8.2 Sugestões de Trabalho Futuro

Durante a realização deste projeto, foram feitas opções que ditaram um rumo. Poder-se-iam ter tomado outras opções que modificariam os resultados obtidos. Este projeto não representa o fim de uma ideia, é, apenas, uma implementação de um conceito. Assim, neste subcapítulo são feitas algumas sugestões de, não só, melhorias à implementação desenvolvida, mas também de novas abordagens para esta ideia.

Uma das possíveis alterações seria usar células de carga que detetam a colocação de um tabuleiro sobre o DWR ao invés de usar de um botão de pressão para iniciar a marcha. Estas detetariam a colocação e/ou remoção de produtos no DWR enviando sinais ao sistema que agiria em conformidade.

Um dos pontos que poderia ser melhorado, prende-se com o algoritmo de controlo do módulo seguidor de linha. Quando o DWR se encontra numa trajetória reta, a velocidade de rotação dos motores está longe de ser a máxima permitida, mas, com o controlador atual, a velocidade reduzida é necessária para ser possível fazer a compensação nas trajetórias curvas. Num trabalho futuro, seria possível aumentar a velocidade de rotação dos motores quando os últimos valores do erro fossem próximo de nulo para valores mais próximos da velocidade máxima, e reduzir para valores que permitem o ajuste da trajetória quando os últimos valores do erro não fossem próximos de zero. Outras melhorias seriam substituir a tecnologia *Bluetooth* por tecnologia WI-FI, permitindo um maior alcance nas comunicações, e adicionar mais sensores de obstáculos de modo a cobrir uma maior área de deteção.

Além de modificações e melhorias poder-se-iam acrescentar novas funcionalidades ao DWR. Uma delas seria de uma unidade de controlo que faria a ligação entre um funcionário responsável e um DWR. Esta unidade de controlo poderia calcular rotas para o DWR efetuar, com base nos locais de paragem definidos pelo funcionário responsável, e transmiti-las para o DWR. Além disso, esta unidade de controlo poderia ser capaz de atender a pedidos de vários funcionários responsáveis, controlar vários DWR, sendo capaz de verificar quais os robôs prontos para efetuar um novo pedido, gerir o carregamento dos DWR, entre outros. Numa aplicação mais avançada, a unidade de controlo poderia ter acesso aos dados dos pacientes, gerindo de forma autónoma toda a distribuição de bens no hospital, tendo os dados dos quartos ocupados e dos pacientes aí hospitalizados, gerindo as suas necessidades, como, por exemplo, qual a medicação prescrita, bem como o horário a que deve ser tomada.

Poder-se-ia, também, ter adicionado uma câmara ao robô, sendo a imagem transmitida para a aplicação de um funcionário responsável, ficando esta transmissão ao encargo do DWR e da unidade de controlo, permitindo controlar o robô sem a necessidade de contacto direto entre o operador e o DWR.

Referências

- [1] SIC Notícias, “Pico da terceira vaga da pandemia atingido a 29 de janeiro,” 9 fevereiro 2021. [Online]. Available: <https://sicnoticias.pt/especiais/coronavirus/2021-02-09-Pico-da-terceira-vaga-da-pandemia-atingido-a-29-de-janeiro>. [Acedido em 20 junho 2021].
- [2] SIC Notícias, “Coronavírus - Hospitais de Lisboa quase a esgotar capacidade,” 14 outubro 2020. [Online]. Available: <https://sicnoticias.pt/especiais/coronavirus/2020-10-14-Hospitais-de-Lisboa-quase-a-esgotar-capacidade>. [Acedido em 24 março 2021]
- [3] “Conheça Jaci: o robô de desinfecção que auxilia no combate a Covid-19,” Tecnopuc, 29 abril 2020. [Online]. Available: <https://www.pucrs.br/tecnopuc/2020/04/29/conheca-jaci-o-robo-de-desinfeccao-que-auxilia-no-combate-covid-19/>. [Acedido em 15 março 2021]
- [4] E. L. Brand Talk, “Pandemic and the Smarter World: A Future of Robots?,” 5 maio 2020. [Online]. Available: <https://www8.gsb.columbia.edu/articles/brand-talk/pandemic-and-smarter-world-future-robots>. [Acedido em 19 fevereiro 2021].
- [5] R. K. Erico Guizzo, “How Robots Became Essential Workers in the COVID-19 Response,” IEEE SPECTRUM, 30 setembro 2020. [Online]. Available: <https://spectrum.ieee.org/robotics/medical-robots/how-robots-became-essential-workers-in-the-covid19-response>. [Acedido em 10 março 2021].
- [6] J. D'Onfro, “Robots To The Rescue: How High-Tech Machines Are Being Used To Contain The Wuhan Coronavirus,” 2 fevereiro 2020. [Online]. Available: <https://www.forbes.com/sites/jilliandonfro/2020/02/02/robots-to-the-rescue-how-high-tech-machines-are-being-used-to-contain-the-wuhan-coronavirus/?sh=73364f201779>. [Acedido em 2 março 2021].
- [7] P. Garrido, “Apresentação PI,” 2020. [Online]. Available: https://elearning.uminho.pt/bbcswebdav/pid-1045855-dt-content-rid-3987827_1/courses/2021.930504_1/ProjetoIntegrador_LPI1_2021.pptx%281%29.pdf.
- [8] STMicroelectronics, “NUCLEO-F767ZI - STM32 Nucleo-144 development board with STM32F767ZI MCU, supports Arduino, ST Zio and morpho connectivity,” [Online]. Available: <https://www.st.com/en/evaluation-tools/nucleo-f767zi.html>. [Acedido em 24 março 2021].
- [9] STMicroelectronics, “STM32CubeMX - STM32Cube initialization code generator,” [Online]. Available: <https://www.st.com/en/development-tools/stm32cubemx.html>. [Acedido em 24 março 2021].
- [10] P. Corporation, “pololu.com,” 2001-2014. [Online]. Available: <https://www.pololu.com/docs/pdf/OJ12/QTR-8x.pdf>. [Acedido em 3 fevereiro 2021].
- [11] R. Want, “IEEE Pervasive Computing,” An Introduction to RFID Technology, p. IEEE Computer Society Digital Library, janeiro - março 2006.
- [12] Sharp, “Distance Measuring Sensor Unit”. Patente GP2Y0A21YK0F.
- [13] STMicroelectronics, “DUAL FULL-BRIDGE DRIVER”. Patente L298 datasheet, janeiro 2000.
- [14] Botnroll, “Cartão RFID Mifare,” [Online]. Available: <https://www.botnroll.com/pt/rf-lora/3320-cart-o-rfid-mifare-1k-s50-compat-vel-tag-13-56mhz.html>. [Acedido em 21 junho 2021].
- [15] NXP, “Standard performance MIFARE and NTAG frontend”. Patente MFRC522, 27 abril 2016.
- [16] ITeadStudio, “Bluetooth to Serial Port Module”. Patente HC-05, 6 junho 2010.
- [17] Kicad, “Kicad EDA,” [Online]. Available: <https://www.kicad.org/>. [Acedido em 20 junho 2021].

- [18] Botnroll, “Fonte de alimentação para breadboard V2 5V/3.3V,” [Online]. Available: <https://www.botnroll.com/pt/alimentadores-acdc-5v/796-fonte-de-alimentacao-para-breadboard-tol123d3p.html>. [Acedido em 20 junho 2021].
- [19] T. Corporation, “TENENERGY 18650 2200 mAh Li-Ion Cell”. Patente Tenergy 18650.
- [20] “BMS PARA PROTEÇÃO BATERIAS 18650 3S 12,6V 20A,” [Online]. Available: <https://www.botnroll.com/pt/acessorios/2558-bms-para-protec-o-baterias-18650-3s-12-6v-20a.html>. [Acedido em 10 fevereiro 2021].
- [21] Botnroll, “Carregador 3 baterias 18650 em série - AC 100~240V DC 12,6V 1A,” [Online]. Available: <https://www.botnroll.com/pt/alimentadores-acdc-12v/2557-carregador-3-baterias-18650-em-s-rie-ac-100-240v-dc-12-6v-1a.html>. [Acedido em 20 junho 2021].
- [22] Z. Electromotor. Patente ZGB37RG.
- [23] Botnroll, “Step-down para 5V até 8A 40W com 4 portas USB entrada 8V~35V,” [Online]. Available: <https://www.botnroll.com/pt/conversores-dcdc/3650-step-down-para-5v-at-8a-40w-com-4-portas-usb-entrada-8v-35v.html>. [Acedido em 20 junho 2021].
- [24] “Github,” [Online]. Available: <https://github.com/>. [Acedido em 20 4 2021].
- [25] “Git - local branching on the cheap,” [Online]. Available: <https://git-scm.com/>. [Acedido em 20 4 2021].
- [26] MIT APP INVENTOR, [Online]. Available: <https://appinventor.mit.edu/>. [Acedido em 20 junho 2021].
- [27] Botnroll, “Interruptor painel redondo,” [Online]. Available: <https://www.botnroll.com/pt/interruptores-botoes/535-switch-de-painel-spst-redondo.html>. [Acedido em 20 junho 2021].
- [28] P. Carvalho, “Fiabilidade e boas práticas de projeto,” 2014. [Online]. Available: https://elearning.uminho.pt/bbcswebdav/pid-1045855-dt-content-rid-3987823_1/courses/2021.930504_1/FiabilidadeBoasPraticasProjeto_PCarvalho.pdf. [Acedido em 3 fevereiro 2021].
- [29] Botnroll, “Kit conectores 2.54mm 620PCS,” [Online]. Available: <https://www.botnroll.com/pt/cabo/3464-kit-conectores-2-54mm-620pcs.html>. [Acedido em 20 junho 2021].
- [30] “Electrostatic-sensitive device,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Electrostatic-sensitive_device. [Acedido em 18 junho 2021].
- [31] TUR CERT, “O que é o certificado CE?,” [Online]. Available: <https://www.ceisaret.com/pt/ce-certifikasi-nedir/>. [Acedido em 19 junho 2021].
- [32] Jornal Oficial da União Europeia, “DIRETIVA 2014/35/UE DO PARLAMENTO EUROPEU E DO CONSELHO,” 29 março 2014. [Online]. Available: <https://eur-lex.europa.eu/legal-content/PT/TXT/PDF/?uri=CELEX:32014L0035&from=EN>. [Acedido em 15 junho 2021].
- [33] ISO, “ISO 12100:2010,” novembro 2010. [Online]. Available: <https://www.iso.org/standard/51528.html>. [Acedido em 20 junho 2021].
- [34] ISO, “ISO 3691-4:2020,” outubro 2020. [Online]. Available: <https://www.iso.org/standard/70660.html>. [Acedido em 20 junho 2021].
- [35] “How to Prolong the Life of an 18650 Battery,” instructables circuits, [Online]. Available: <https://www.instructables.com/How-to-Prolong-the-Life-of-an-18650-Battery/>. [Acedido em 11 fevereiro 2021].

- [36] “How can I estimate the life of ceramic capacitors?,” Taiyo Yuden, [Online]. Available: <https://www.yuden.co.jp/eu/product/support/faq/q020.html>. [Acedido em 11 fevereiro 2021].
- [37] A. Perzan, “Brushed vs. brushless DC motors,” drive.tech, [Online]. Available: <https://drive.tech/en/stream-content/brushed-vs-brushless-dc-motors>. [Acedido em 11 fevereiro 2021].
- [38] T. Instruments, “Calculating Useful Lifetimes of Embedded Processors,” 2014. [Online]. Available: <https://www.ti.com/lit/an/sprabx4b/sprabx4b.pdf?ts=1612984192026>. [Acedido em 11 fevereiro 2021].
- [39] A. G. e. I. Cardoso, “Fecha tudo. Escolas e universidades em casa a partir de sexta-feira.,” Jornal de Notícias, 21 janeiro 2021. [Online]. Available: <https://www.jn.pt/nacional/fecha-tudo-escolas-e-universidade-em-casa-a-partir-de-sexta-feira-13256762.html>. [Acedido em 3 fevereiro 2021].