Instituto Politécnico de Tomar

Escola Superior de Tecnologia de Tomar

Bernardo Mourão Ferreira

Estágio de Investigação Aplicada no LINE.IPT/TAGUSVALLEY - Tecnopolo do Vale do Tejo

Relatório de Estágio

Orientado por:

Professor Doutor Manuel Fernando Martins Barros

Relatório de Estágio apresentado ao Instituto Politécnico de Tomar para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrotécnica - Especialização em Controlo e Eletrónica Industrial

RESUMO

No âmbito da unidade curricular de Estágio, do 2º ano do curso de Mestrado em Engenharia Eletrotécnica – Especialização em Controlo e Eletrónica Industrial, foi efetuado um estágio curricular, com a duração de 1458 horas, correspondendo a aproximadamente 9 meses.

O estágio foi realizado no parque tecnológico do Vale do Tejo, TAGUSVALLEY - Abrantes, no departamento do LINE.IPT – Laboratório de Inovação Industrial e Empresarial. Este centro de investigação desenvolve produtos, soluções e tecnologias essencialmente para as empresas, com especial foco para projetos na área da eletrotecnia, mecânica, informática e tecnologias de informação e comunicação.

No presente relatório, expõe-se o trabalho desenvolvido durante o período de estágio, bem como os fundamentos teóricos e a descrição detalhada de cada um dos projetos desenvolvidos nesse âmbito. Os projetos desenvolvidos relacionam-se sobretudo com o processamento de imagem ou visão por computador, no entanto foi ainda desenvolvido algum trabalho no âmbito da programação eletrónica de microcontroladores e microprocessadores.

O início de cada um dos projetos foi procedido por uma fase de pesquisa em processamento de imagem e visão por computador, com o objetivo de definir métodos e ferramentas a utilizar, e escolha do equipamento fotográfico e de iluminação. Tendo em vista a complexidade e especificidade técnica do trabalho a executar, foi necessário prever e frequentar um conjunto de formações em visão artificial e métodos avançados de visão artificial. As tarefas desenvolvidas nas áreas da eletrotecnia e informática, o trabalho em equipa, a colaboração com várias entidades assim como o contacto com vários projetos de investigação e desenvolvimento, foram uma experiência bastante enriquecedora, que contribuíram positivamente para a aquisição de novas competências e para a integração no mercado de trabalho e no mundo empresarial na área da eletrotecnia.

Palavras-chave: Visão Artificial; Processamento de Imagem; Eletrónica; Estimativa de Biomassa; Microprocessadores.

ABSTRACT

Regarding the course unit “Internship” of the 2nd year of the Master course in Electrical Engineering – Specialization in Control and Industrial Electronics, a curricular internship was achieved, with a total 1458-hour length, corresponding to approximately 9 months.

The internship took place in the technological park of Vale do Tejo, TAGUSVALLEY – Abrantes, in the LINE.IPT – Industrial and Business Innovation Laboratory department. This research centre develops products, solutions and technologies mainly to companies, focusing specially on projects concerning electrical, mechanical and computer engineering and information and communications technologies.

This report displays the work executed during the entire internship period, as well as the theoretical bases and detailed description of each one of the projects developed. The projects executed were mostly related to machine/computer vision, however, some microcontrollers and microprocessors electronic programming related work was also developed.

The beginning of each one of the projects was preceded by a research phase in image processing and vision by computer, whose goal was to define which methods and tools to utilize, and the choice of the photographic and lighting equipment. Bearing in mind the complexity and technical specificity of the work executed, it was necessary to attend various training courses in artificial vision and artificial vision advanced methods.

The numerous tasks performed in several electrical and computer engineering fields, the teamwork, the interaction with various entities as well as the direct contact with several research and development projects were a very enriching experience that allowed the trainee to acquire new competences and to connect with the labour market and the business world of electrical engineering.

Keywords: Artificial Vision; Image Processing; Electronics; Biomass Estimation; Microprocessors

AGRADECIMENTOS

Durante o estágio bem como o período de redação deste relatório contei com o apoio de familiares, amigos, professores, colegas e colaboradores a quem gostaria deixar um agradecimento.

Em primeiro lugar, gostaria de agradecer aos meus familiares e amigos, em especial ao meu pai e à Ana Carolina Lopes, que me ajudaram em algumas fases da elaboração deste documento.

Gostaria de agradecer ao meu orientador, Professor Doutor Manuel Barros, por toda a disponibilidade, amizade e apoio ao longo deste estágio bem como ao longo do meu percurso neste Mestrado.

A todos os colaboradores e instituições integrantes no projeto AQUATROPOLIS, em especial ao Professor Pedro Granchinho, Professor Carlos Ferreira e ao Engenheiro Pedro Neves por toda a colaboração e incentivo prestados ao longo deste projeto.

Ao Sérgio Saramago da Vestas Portugal pelo apoio prestado no projeto com esta empresa.

À TAGUSVALLEY S.A., nomeadamente ao departamento LINE.IPT, pelas oportunidades e desafios, bem como o bom ambiente profissional.

Aos colegas Hugo Magalhães, David Ferreira, Luís Prates e Nuno Cardoso pelo espírito de equipa e interajuda demonstrados durante o meu estágio no LINE.IPT.

Parte do trabalho descrito neste relatório teve o suporte financeiro do projeto AQUATROPOLIS- *Intelligent Management System for Sustainable Aquaculture*, Projeto de I&D em Co-promoção POCI - 01-0247-FEDER-017888 - Aviso no 33/SI/2015, Compete2020.

Índice

Índice de Figuras

Índice de Tabelas

Lista de Abreviaturas, Siglas e Símbolos

µm - micrómetro

ADC - *Analog to Digital Converter*

cm - centímetro

g/L - gramas por litro

GPIO - *General Purpose Input/Output*

I/O - *Input/Output*

I2C - *Inter-Integrated Circuit*

m - metro

mm - milímetro

mV - milivolt

nm - nanometros

PCB - *Printed Circuit Board*

PWM - *Pulse Width Modulation*

ROV - *Remote Operated Vehicle*

RSSF - Rede de Sensores Sem Fio

SOC - *System On Chip*

SOM - *System On Module*

Tupla - Estrutura de dados semelhante às listas, no entanto uma vez escrita é inalterável

UART - *Universal Asynchronous Receiver/Transmitter*

UCB - Unidade de Controlo de Biomassa

UMA - Unidade de Monitorização Ambiental

USB - *Universal Serial Bus*

V - Volt

1 Introdução

1.1 Enquadramento e Objetivos

No âmbito da unidade curricular de Estágio, do 2º ano do curso de Mestrado em Engenharia Eletrotécnica – Especialização em Controlo e Eletrónica Industrial, o aluno Bernardo Ferreira realizou um estágio curricular, com a duração de 1458 horas, correspondendo a aproximadamente 9 meses, no LINE.IPT – Laboratório de Inovação Industrial e Empresarial do Instituto Politécnico de Tomar.

O LINE.IPT é um departamento do parque tecnológico do Vale do Tejo, TAGUSVALLEY - Abrantes, e é administrado em parceria pelo Instituto Politécnico de Tomar, a Câmara Municipal de Abrantes, a TAGUSVALLEY e a NERSANT. Este laboratório identifica-se como um centro de investigação focado no desenvolvimento de novos produtos e tecnologias, trabalhando maioritariamente direcionado para as empresas/industrias.

Os principais projetos que tiveram uma intervenção direta no âmbito do trabalho de estágio foram o projeto "Aquatropolis"® - *Intelligent Management System for Sustainable Aquacultures* e o projeto para a Vestas Portugal. No projeto Aquatropolis foi desenvolvido um protótipo de um sistema de visão artificial que permite a execução automática das tarefas de contagem e monitorização do desenvolvimento da biomassa em tanques de aquacultura semi-intensiva (Unidade de Controlo da Biomassa - UCB). No âmbito deste projeto, foram ainda desenvolvidas tarefas relacionadas com a programação de microcontroladores e microprocessadores e desenvolvimento de circuitos eletrónicos, para a Unidade de Monitorização Ambiental (UMA) e para o veículo robotizado (ROV – *Remote Operated Vehicle*) desenvolvido também no âmbito deste projeto. A documentação do trabalho elaborado foi também uma fase de destaque deste projeto, em que o aluno participou ativamente redigindo vários relatórios intermédios e o relatório de conclusão do projeto, entregue à entidade responsável pelo mesmo.

No projeto desenvolvido para a Vestas Portugal, o aluno contribuiu para o desenvolvimento de uma ferramenta para as equipas de manutenção dos parques eólicos desta empresa. Esta ferramenta permite aferir o desgaste dos dentes das engrenagens (*yaw*

ring) de rotação da *nacelle* em relação à torre eólica, através de um sistema de visão por computador.

Para além destes dois projetos, o aluno efetuou tarefas na área de engenharia eletrotécnica no âmbito de outros projetos. Estas tarefas foram, nomeadamente, programação de microcontroladores e microprocessadores, desenho e fabrico de placas de circuito impresso.

É relevante frisar que as tarefas desenvolvidas neste estágio foram precedidas por trabalho já elaborado pelo aluno na instituição acolhedora, uma vez que o aluno desempenhava as funções de bolseiro de investigação no LINE.IPT desde setembro de 2017.

1.2 Instituição Acolhedora

1.2.1 TAGUSVALLEY

O TAGUSVALLEY - Tecnopolo do Vale do Tejo é um Parque de Ciência e Tecnologia, fundada a 7 de novembro de 2003, está instalada no concelho de Abrantes, nas antigas instalações da União Fabril do Azoto da CUF em Alferrarede. Foi criado como um projeto da Câmara Municipal de Abrantes (em 2000), com o intuito de estimular o empreendedorismo e a competitividade na região, tendo por base a inovação e a tecnologia. Os parceiros que colaboram neste projeto de promoção do Vale do Tejo são a Associação Empresarial da Região de Santarém (NERSANT) e o Instituto Politécnico de Tomar (IPT). O TAGUSVALLEY conta também com associados como o Instituto Politécnico de Santarém (IPS) e a Tejo Energia. [1]

O TAGUSVALLEY disponibiliza, no Parque Tecnológico do Vale do Tejo, um conjunto de serviços de suporte às empresas e aos empreendedores da região, dividindo-se em três setores distintos: o INOV.POINT, o INOV'LINEA e o LINE.IPT.

O INOV.POINT é uma infraestrutura de incubação e desenvolvimento de empresas vocacionada para o acolhimento e apoio ao arranque de iniciativas empresariais inovadoras e tecnológicas. [2]

O INOV'LINEA é um Centro de Transferência de Tecnologia Alimentar, uma estrutura de apoio à inovação, focado na aplicação de novas tecnologias, desenvolvimento de novos produtos e técnicas inovadoras no processamento e conservação de alimentos.[3]

O LINE.IPT surge da parceria entre o Instituto Politécnico de Tomar, a Câmara Municipal de Abrantes, o TAGUSVALLEY e a NERSANT, como catalisador da inovação e desenvolvimento tecnológico da região, promovendo a competitividade do tecido empresarial. [4]

Figura 1 - Edifício INOV.POINT TAGUSVALLEY [5]

1.2.2 LINE.IPT

O LINE.IPT é um departamento do TAGUSVALLEY, e é administrado em parceria pelo Instituto Politécnico de Tomar, a Câmara Municipal de Abrantes, o TAGUSVALLEY e a NERSANT.

Sendo um centro de investigação inteiramente direcionado para as empresas, o objetivo do LINE.IPT é o desenvolvimento de novos produtos, tecnologias e processos e/ou melhoria/reconversão de produtos ou processos já existentes, diretamente aplicáveis na indústria. Assume-se como catalisador da inovação e desenvolvimento tecnológico, promovendo a competitividade e nível de formação e especialização dos quadros técnicos das empresas. [4]

Este laboratório pretende assim:

- Alavancar o desenvolvimento de competências nas áreas das Engenharias e Desenvolvimento de Produtos;
- Fomentar a criação de empresas de âmbito tecnológico;

- Promover a cooperação científica e tecnológica entre empresas e instituições de I&DT regionais, nacionais e internacionais;
- Colaborar na incorporação de tecnologia e inovação pelas empresas.

Durante o estágio a que este relatório se refere, a equipa do LINE.IPT era composta por cinco elementos, incluindo o aluno autor deste relatório. Cada um destes elementos estava associado a uma área de trabalho específica, nomeadamente, Eletrotécnica, com três elementos, Informática e Mecânica, respetivamente com um elemento cada.

Recorrendo a esta equipa multidisciplinar, o LINE tem vido a desenvolver projetos relacionados com sistemas de aquisição de dados em tempo real, comunicação destes dados via *wireless* e respetiva apresentação numa plataforma WEB. Este tipo de sistema é utilizado sobretudo no desenvolvimento de soluções de monitorização de processos produtivos, onde são analisados valores como tensão, corrente e consumo de determinados equipamentos, temperatura e humidade ambiente, parâmetros de qualidade da água ou qualquer outro parâmetro dependendo das especificações do projeto e da área de aplicação em que se insere. São também desenvolvidos projetos relacionados com automação industrial, robótica móvel e visão artificial, área na qual o aluno desenvolveu grande parte do seu trabalho neste estágio. Nos vários projetos elaborados, são também desenvolvidas estruturas mecânicas e máquinas, são efetuadas otimizações de processos, mecanismos e peças para implementação nas mais diversas aplicações. Uma das ferramentas mais utilizadas nesta área é a modelação 3D que permite criar protótipos e estruturas para os projetos desenvolvidos.

1.2.3 Certificações

O TAGUSVALLEY possui algumas certificações, que são uma mais-valia no mundo empresarial. Possuir tais certificações, permite fornecer às empresas um conjunto de ferramentas para iniciarem e promoverem projetos recorrendo a fundos comparticipados e a um conjunto de competências essenciais para a gestão e desenvolvimento de projetos. De seguida, são enumeradas as certificações do TAGUSVALLEY, ordenadas pela ordem cronológica em que foram atribuídas à instituição:

- 2011 – EU BIC *Business Innovation Center* (EBN);
- 2015 – Vales de Inovação;
- 2015 – Vales de Empreendedorismo;
- 2015 – Vales I&D;
- 2016 – Vales de Incubação;
- 2017 – Vales de Indústria 4.0.

Além das certificações enunciadas anteriormente, o TAGUSVALLEY assegura também a Presidência da TECPARQUES, a Comissão Executiva RIERC (Rede de Incubadoras da Região Centro) e o Conselho Fiscal BIC'S (Rede Nacional do *Business Innovation Center*)

1.3 Organização do Documento

A organização do documento é efetuada tendo por suporte as normas base para elaboração de teses de mestrado do IPT. Neste primeiro capítulo, é apresentada a introdução do documento e a descrição da entidade acolhedora do estágio.

Uma vez que o trabalho elaborado pelo aluno no estágio confere algumas características típicas de um trabalho de investigação, é pertinente introduzir alguns conceitos teóricos que detalhem alguns pormenores relativos às tecnologias de visão artificial utilizadas. Assim, o segundo capítulo aborda o estado da arte das tecnologias de visão artificial e processamento de imagem utilizadas em aplicações de aquacultura. No terceiro capítulo, são descritos os conceitos teóricos das técnicas de visão por computador empregues no desenvolvimento da solução do Projeto Aquatropolis.

O capítulo 4 é composto por uma revisão do estado da arte de técnicas de triangulação laser e pela descrição do método de triangulação laser empregue nos trabalhos do projeto

Vestas. Os dois capítulos subsequentes, 5 e 6, descrevem respetivamente o trabalho elaborado no âmbito do Projeto Aquatropolis e no Projeto com a Vestas Portugal, doravante designado Projeto Vestas. Finalmente, são apresentadas as considerações finais e alguns resultados no capítulo da Conclusão.

É importante referir que as informações contidas neste relatório, especialmente as que são relativas a projetos ainda em curso, são de cariz confidencial, não podendo ser divulgadas na totalidade. Todos os projetos abordados ao longo deste relatório são fruto do trabalho do autor do documento e da equipa do LINE.IPT. No caso específico do Aquatropolis, é detalhado também algum trabalho desenvolvido pelas empresas ou instituições que constituem o conjunto de parceiros do projeto.

2 Estado da Arte em Tecnologias de Visão por Computador na Aquacultura

Este capítulo aborda conceitos introdutórios pertinentes para entender o trabalho desenvolvido pelo aluno no Projeto Aquatropolis - *Intelligent Management System for Sustainable Aquacultures*, no âmbito do estágio realizado no LINE.IPT. Nele é efetuada uma atualização do estado da arte relativa a métodos e tecnologias de visão por computador no universo da aquacultura, com especial foco nos métodos e tecnologias utilizados na solução desenvolvida pelo aluno no referido projeto.

2.1 Visão por Computador e Processamento de Imagem

O registo fotográfico é hoje em dia uma ferramenta imprescindível para o estudo e desenvolvimento científico. Esta ferramenta trouxe à ciência avanços consideráveis, uma vez que permite documentar vários detalhes que outrora o desenho manual, por exemplo, não possibilitava. Com a evolução da tecnologia, a determinada altura, houve necessidade de automatizar o processo de análise e observação das imagens obtidas, considerando que este processo recorria a métodos de avaliação manuais que consumiam bastante tempo de trabalho. Assim, foram-se desenvolvendo dispositivos optomecânicos, semi ou completamente automáticos para desempenhar tarefas de quantização, descrição e ilustração dos fenómenos observados. No entanto, devido à especificidade destes equipamentos, desenhados para tarefas em áreas como a astronomia ou a análise de partículas, não se estendeu imediatamente a outras aplicações.[6]

Com a evolução da tecnologia, os computadores e *hardware* de computação tornaram-se muito mais competentes, sendo capazes de processar volumes de informação inimagináveis aquando da criação dos primeiros computadores. Assim, o processamento de imagem evoluiu também, uma vez que, estas máquinas se tornaram suficientemente capazes de realizar tarefas desta natureza. A visão por computador e o processamento de imagem são, hoje em dia, ferramentas amplamente utilizadas em várias áreas de aplicação.[6]

A visão permite ao ser humano ter perceção do espaço que o rodeia, de cores, formas, texturas, contrastes e luminosidade, resultando do raciocínio cerebral, através dos

recetores sensoriais ativados pela luz. Pode considerar-se resumidamente que a visão é o processo de descobrir a partir da imagem o que está presente no mundo, e onde.[7]

Do ponto de vista das ciências biológicas, a visão por computador pretende fornecer modelos computacionais baseando-se no sistema de visão humano. Relativamente ao ponto de vista da engenharia, a visão por computador pretende levar à criação de sistemas autónomos capazes de desempenhar as tarefas que a visão humana efetua e, por vezes, efetuar tarefas ainda mais complexas.[8] A maioria destas tarefas estão relacionadas com a extração de informação útil de imagens. Por exemplo, é frequente utilizar visão por computador para obter a geometria 3D de um objeto representado numa imagem ou detetar e identificar expressões faciais.

Analisando os conceitos de visão por computador e processamento de imagem: o intuito da visão por computador é criar modelos e extrair informação de imagens, enquanto que o processamento de imagem está relacionado com a implementação de transformações a imagens e com a melhoria de contrastes de contornos, cores, redução de ruído, exposição de imagem, entre outros.[9], [10]

A grande maioria das aplicações de visão por computador recorre primeiramente ao processamento de imagem para, posteriormente, extrair a informação necessária. Assim, é recorrente referir-se que os algoritmos e operações de processamento de imagem têm como variável de entrada uma imagem e como saída irão ter igualmente uma imagem. Considera-se que o processamento de imagem está interligado a outras áreas de conhecimento científico, podendo ser associado a áreas como o reconhecimento de padrões e análise métrica, ou áreas mais abrangentes como as redes neuronais, inteligência artificial e perceção visual.[6]

Quando se refere visão por computador, usualmente, pretende abordar-se o conceito já apresentado de um sistema computadorizado que executa tarefas muito semelhantes ao sistema de visão humano. Já quando abordada a temática da visão artificial *(machine vision),* pretende referir-se o tipo de sistema utilizado em ambiente industrial para auxílio de processos de fabrico, onde um sistema de visão executa tarefas de medição e verificação da integridade de objetos no processo de fabrico.[6]

2.2 Aplicações de Visão por Computador em Aquacultura

Parte do trabalho desenvolvido pelo aluno neste estágio esteve relacionado com o projeto Aquatropolis, que será abordado posteriormente neste documento. Um dos objetivos deste projeto foi o desenvolvimento de uma Unidade de Controlo da Biomassa, um sistema que tem como função executar tarefas de monitorização da evolução da biomassa em tanques de aquacultura. Numa primeira fase, estas tarefas deveriam ser concretizadas recorrendo a duas tecnologias não intrusivas, câmaras e sonar. O sonar, que acabou por não ser utilizado, teria a finalidade de localizar o cardume, enquanto que, as câmaras seriam utilizadas para aferir as suas dimensões e estimar a biomassa total.

Neste capítulo, pretende apresentar-se uma atualização do estado da arte das soluções de visão na aquacultura, demonstrando os avanços tecnológicos e científicos nesta área. No caso particular do projeto Aquatropolis, é particularmente relevante descrever a evolução da tecnologia associada às técnicas de estimativa da biomassa presente em tanques de aquacultura.

Atualmente, existe uma grande quantidade de indústrias a beneficiar do uso das tecnologias de visão por computador. Usualmente, recorre-se a esta solução para tarefas relacionadas com a otimização de processos, seleção e classificação automatizada e processamento automatizado.[11] No entanto, estas tecnologias não estão a ser tão utilizadas na indústria da aquacultura, quando comparado com as restantes indústrias, devido a alguns obstáculos que têm dificultado a sua implementação, nomeadamente:

- Os objetos de estudo são sensíveis e facilmente stressáveis;
- Os objetos movem-se livremente e aleatoriamente no seu ambiente;
- Usualmente não é possível controlar luminosidade, visibilidade e estabilidade deste ambiente;
- Qualquer tipo de instrumento de medida ou sensores têm de trabalhar imersos em água. [12]

Apesar destas dificuldades, muitos cientistas afirmam que a utilização de visão por computador para executar tarefas de monitorização e controlo de qualidade na indústria da produção de peixe e da aquicultura é necessária e poderá trazer benefícios a nível financeiro e promover a sustentabilidade nesta indústria. Atualmente, muitas destas tarefas do processo produtivo são efetuadas manualmente por operadores humanos. Estes

métodos são dispendiosos, tanto em recursos humanos como em recursos financeiros, sendo ainda invasivos e demorados. Tendo em vista estes factos, a utilização de métodos rápidos, não invasivos e pouco dispendiosos, como os sensores óticos, câmaras e sistemas de visão, é importante e bastante desejável, sendo relevante a investigação de novas abordagens nesta indústria. [13]

No artigo *"The use of computer vision technologies in aquaculture – A review"* (Zion, 2012) [12], o autor documenta a evolução das tecnologias de visão por computador aplicadas à aquacultura, expondo pormenorizadamente a utilização deste tipo de sistema nas várias tarefas do ciclo produtivo desta indústria. Este documento aborda várias temáticas relacionadas com as necessidades desta indústria e, portanto, será analisado e citado, de forma a dar a conhecer a evolução dos sistemas de visão na aquacultura, focando particularmente soluções de medição de tamanho e estimativa de massa.

2.2.1 Métodos de Medição e Estimativa da Biomassa em Aquacultura

Ao longo de vários anos, a relação entre as características da forma e a massa do peixe tem sido estudada por vários intervenientes, figurando em vários artigos científicos. A expressão matemática mais utilizada para representa a relação entre o comprimento do peixe (L) e a sua massa (W) é o modelo exponencial $W=aL^b$, onde a e b são índices estruturais característicos de cada família e espécie. Na publicação "*Predicting biomass of Atlantic salmon from morphometric lateral measurements*" (Beddow e Ross 1996) [14], os autores explicam que as regressões matemáticas utilizadas por outros cientistas anteriormente para a predição da biomassa do salmão do atlântico e de outras espécies, que unicamente recorriam à relação entre comprimento e massa dos peixes, demonstraram ser pouco fiáveis e de baixa precisão. Assim, neste artigo, os autores criaram cinquenta e duas regressões matemáticas para prever com uma exatidão na ordem dos 98%, a massa de peixes individualmente usando medidas de forma convencional (b) da Figura 2) e através de redes de treliças (a) da Figura 2).

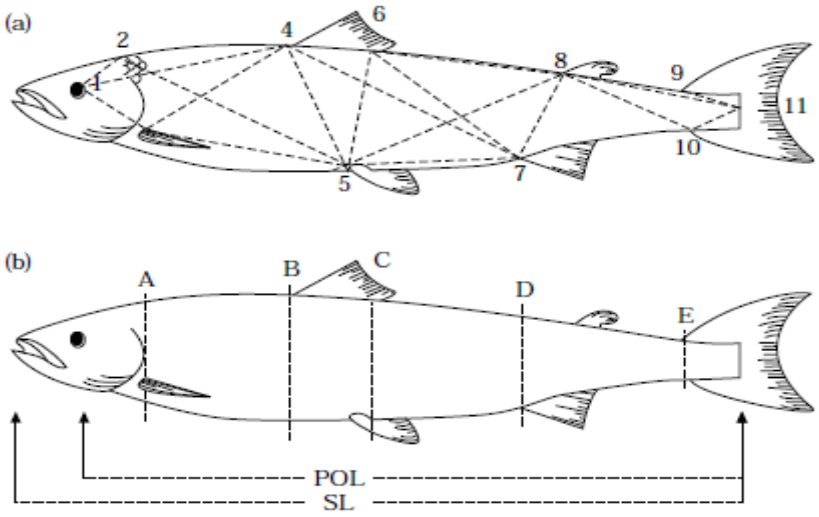


Figura 2 - Pontos de medição do perfil lateral de trutas [14]

Na revisão bibliográfica de tecnologias de visão e processamento de imagem, Boaz Zion apresenta alguns estudos de desenvolvimento de métodos e equipamentos capazes de efetuar a medição das dimensões de peixes e da sua massa corporal. No artigo *"Length measurement of fish by computer vision"*, Strachan (1993) [15], o autor descreve o desenvolvimento de um sistema que fotografa os peixes dentro de um caixa iluminada, apresentada na Figura 3, de forma a obter o seu comprimento. Obteve-se uma precisão de medição ± 3%, num método que se baseava numa linha central do corpo do peixe, na Figura 4, para determinar o seu comprimento.

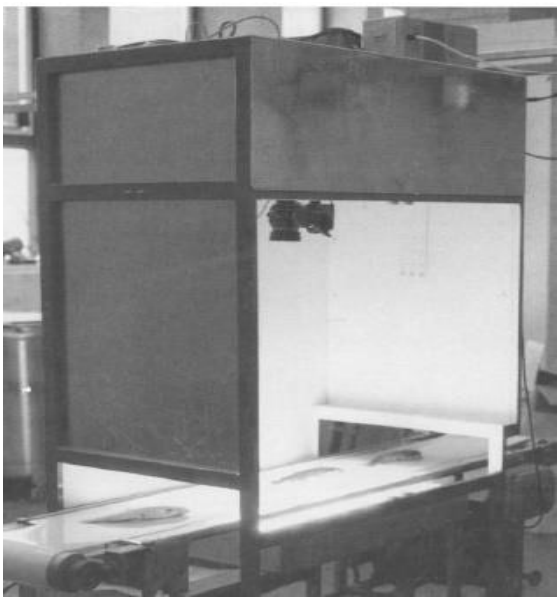


Figura 3 - Câmara iluminada utilizada para efetuar medição de peixes [15]

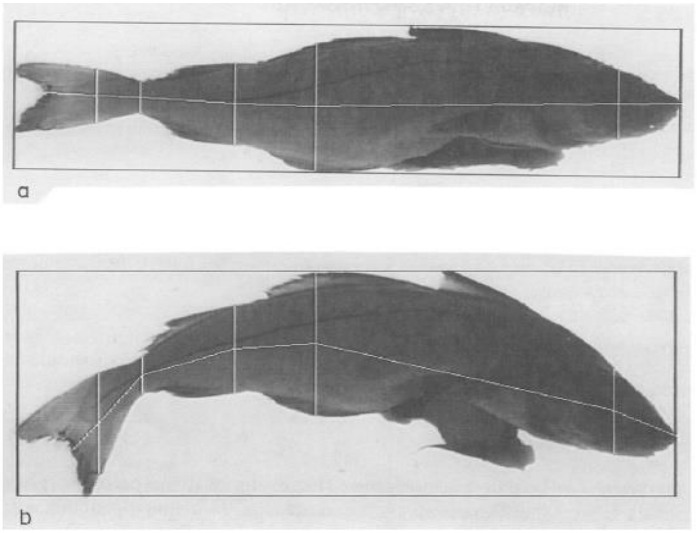


Figura 4 - Demonstração da definição dos pontos médios de medição [15]

Nos artigos *"Prediction of the Weight of Alaskan Pollock Using Image Analysis"* (Balaban, Chombeau, Cirban e Gümüş, 2010) [16], *"Using Image Analysis to Predict the Weight of Alaskan Salmon of Different Species"* (Balaban, Ünal Şengör, Soriano e Ruiz, 2010) [17] e *"Prediction of the Weight of Aquacultured Rainbow Trout (Oncorhynchus mykiss) by Image Analysis"* (Gümüş e Balaban, 2010) [18] é efetuado o estudo da correlação entre os modelos matemáticos exponenciais e a massa corporal de algumas espécies de peixes. Em todos os artigos foi possível concluir que a relação exponencial entre a área e a massa do peixe foi a que demonstrou melhores resultados.

Destacando o primeiro exemplo [16]: neste artigo, a população de estudo foram cento e sessenta exemplares do escamudo do Alasca (*Theragra chalcogramma*). Os peixes foram pesados e posteriormente fotografados numa caixa de luz, demonstrada na Figura 5. Ao lado de cada exemplar fotografado, foi colocado como referência um quadrado com uma área conhecida. Foram testadas várias equações que permitem relacionar a área observada do peixe com a respetiva massa. Neste cenário a relação exponencial entre as duas grandezas foi a que demonstrou melhores resultados.

Figura 5 - Câmara digital e caixa de luz utilizada para aquisição de imagens [16]

Em *"Automatic stress-free sorting of sturgeons inside culture tanks using image processing*" (Hufschmied, Fankhauser e Pugovkin, 2011) [19] foi construído um sistema subaquático que consistia num tubo dentro de um tanque de aquacultura de esturjão (*Acipenser baerii*), capaz de direcionar os peixes para várias saídas dependendo do seu peso. Os indivíduos em análise foram fotografados de topo, sendo utilizada uma regressão linear para estimar a massa do peixe com base na área da silhueta. Esta correlação foi capaz de estimar a massa corporal com um erro médio relativo de 5,5% (n = 50).

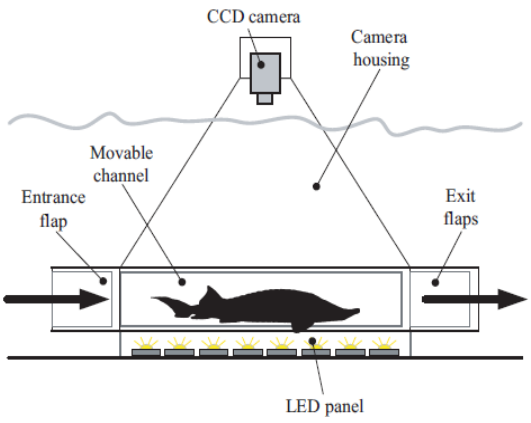


Figura 6 - Sistema utilizado para fotografar o esturjão [19]

Em algumas situações, não é possível nem conveniente condicionar ou forçar o movimento dos peixes. Nestas situações, caso se pretenda obter conclusões relativamente à massa ou dimensão, recorre-se a métodos de visão estéreo (duas câmaras), que permite ter a perceção da profundidade do campo de visão. Um exemplo de utilização desta técnica, é o artigo "*Fish Sizing and Monitoring Using a Stereo Image Analysis System Applied to Fish Farming*" (Ruff, Marchant, and Frost 1995) [20] em que foi elaborado um sistema em estéreo (duas câmaras), para determinar características dos peixes como o tamanho e a posição. O sistema foi testado num tanque com dois metros de diâmetro, com dois salmões com comprimento previamente conhecido. As câmaras foram colocadas suspensas dentro de água, ver Figura 7, montadas verticalmente acopladas a uma vara que mantinha a estabilidade da montagem, conectadas ao sistema de aquisição de vídeo montado à superfície. Numa sequência de imagens onde, figurava apenas um peixe a cerca de um metro de distância do par de câmaras, foram selecionados manualmente os pontos de referência (cabeça e cauda) e obtiveram-se resultados de medição com uma precisão de aproximadamente 96,5%.

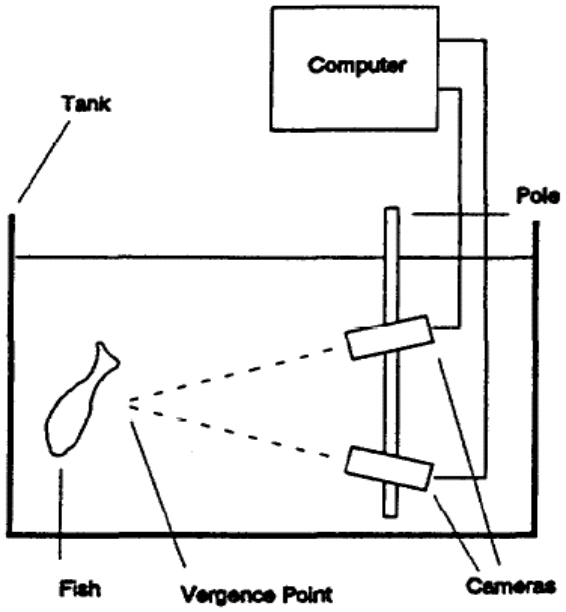


Figura 7 - Montagem do sistema de visão no tanque [20]

No artigo *"Estimating Dimensions of Free-Swimming Fish Using 3D Point Distribution Models"* (Tillett, McFarlane, e Lines, 2000) [21], os autores descrevem o desenvolvimento de um modelo de distribuição de pontos (PDM – *point-distribution model*) tridimensional capaz de segmentar a forma do salmão, em imagens obtidas em vista lateral por um par de câmaras em montagem estéreo. Este PDM definia vinte e seis

pontos de referência correspondentes à forma aproximada do peixe, através de um processo de otimização. Na Figura 9, é demonstrada a montagem deste sistema de câmaras, colocadas verticalmente num tanque com peixes, demonstrado na Figura 10. Antes de iniciar os testes do modelo desenvolvido, foi necessário proceder ao treino do mesmo, recorrendo a imagens obtidas previamente. O processo de treino do PDM foi efetuado obtendo imagens onde figuravam peixes, assinalando manualmente os pontos de referência, tal como demonstrado na Figura 8.

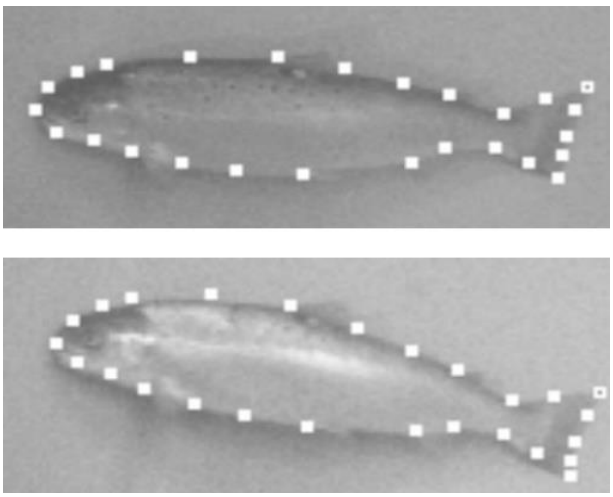


Figura 8 - Par de imagens estéreo com os pontos de referência colocados manualmente [21]

Figura 9 - Montagem do par de câmaras em estéreo [21]

Figura 10 - Vista de topo do tanque com respetiva montagem de câmaras [21]

Após efetuados os testes de medição dos salmões utilizando o PDM, obtiveram-se resultados com uma precisão de aproximadamente 95%, quando comparado com as medições dos peixes efetuadas manualmente.

Em determinadas situações o PDM teve dificuldade em definir corretamente o contorno. Estes casos de falha do PDM são demonstrados na Figura 11. No primeiro par de imagens, mais à esquerda, a falha ocorreu devido à proximidade de outros peixes. No segundo par de imagens, ao centro, a falha ocorreu devido à orientação do peixe, que é muito diferente da do modelo previamente definido. No terceiro par de imagens, à direita, a falha ocorreu pois o modelo é maior que o peixe real.

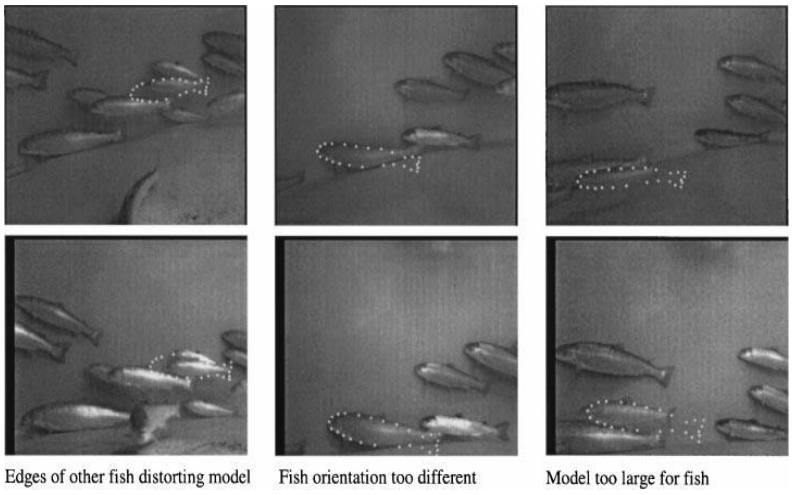


Figura 11 - Casos em que o PDM falhou [21]

Em "*Computer vision and robotics techniques in fish farms"* (J. R. Martinez-de Dios, C. Serna and A. Ollero, 2003) [22], é descrito o desenvolvimento de um sistema de câmaras em estéreo, capaz de estimar a massa de peixes em jaulas, aferindo tal valor a partir do seu comprimento, e ainda de um sistema semelhante colocado sobre os tanques utilizados como berçário de peixes, para efetuar a estimativa de massa corporal de peixes juvenis. Nos testes, foram segmentados com sucesso cento e vinte e dois peixes, observando-se resultados com uma exatidão de 95% e de 96%, respetivamente, para os sistemas subaquáticos e para o sistema colocado sob o tanque peixes juvenis.

A variação da luminosidade é descrita como um dos principais obstáculos para o sucesso deste sistema. Este fenómeno é visível no par de imagens da Figura 12, onde a luminosidade no topo da imagem é muito superior à visualizada na parte inferior da imagem. Antes de efetuar a segmentação da forma dos peixes, é efetuado um pré processamento para compensar as variações de luminosidade nas imagens.

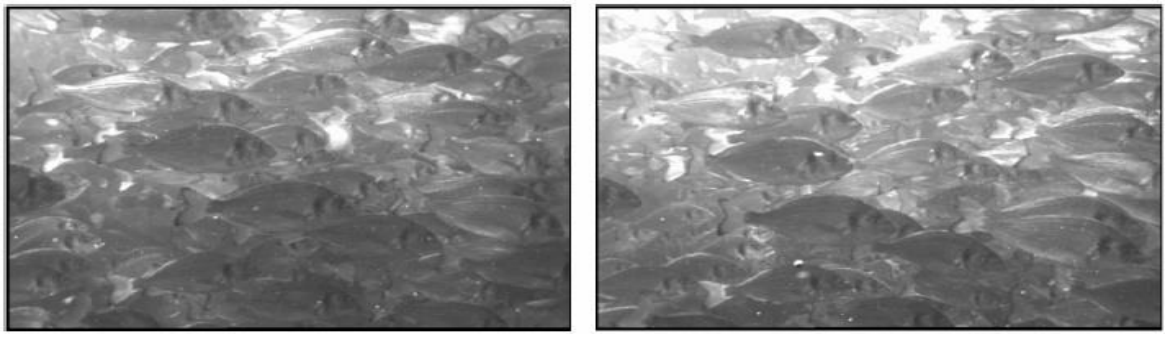


Figura 12 - Par de imagens captadas pelo sistema de câmaras [22]

Os testes deste sistema e respetivos algoritmos foram efetuados em imagens obtidas em jaulas de douradas em mar aberto. No primeiro par de imagens da Figura 12, são demonstradas as imagens obtidas pelas câmaras. Na Figura 13 estão representadas estas mesmas imagens após a compensação da luminosidade. O terceiro conjunto de imagens, na Figura 14, demonstra a identificação das barbatanas de cauda dos peixes, contendo alguns falsos positivos provocados por arestas semelhantes a esta cauda e, ainda, efeitos provocados pela luz. O par de imagens da Figura 15 ilustra o resultado obtido após efetuada a correspondência dos pontos entre as duas imagens do par estéreo, descartando muitos dos falsos positivos. O último par de imagens, na Figura 16, demonstra o resultado obtido após efetuada a correspondência entre pontos referência, barbatana caudal e boca, do mesmo peixe.

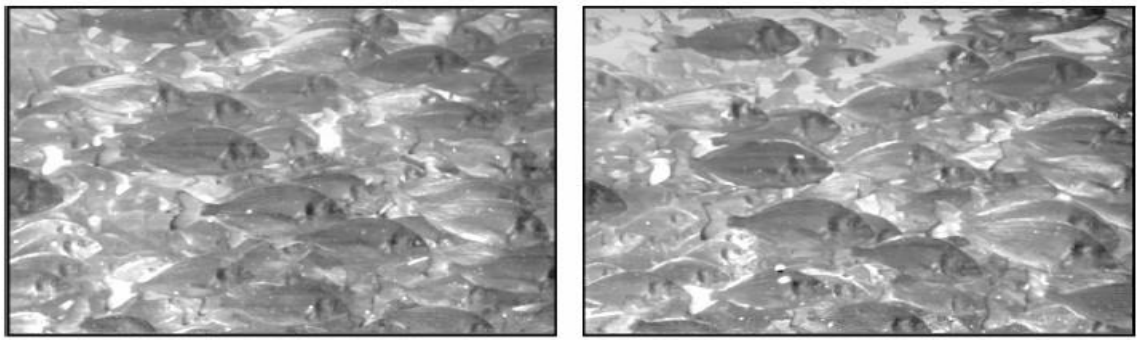


Figura 13 - Par de imagens captadas pelo sistema de câmaras, após compensação da luminosidade [22]

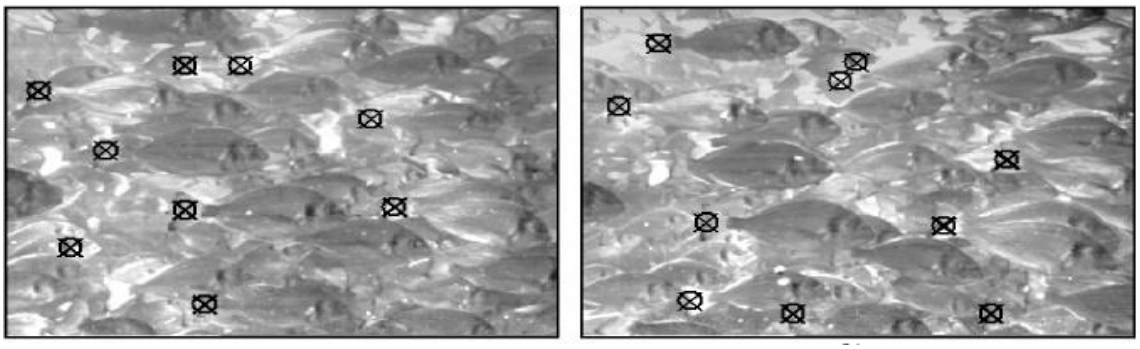


Figura 14 - Imagens onde foram detetadas as caudas dos peixes, incluindo falsos positivos [22]

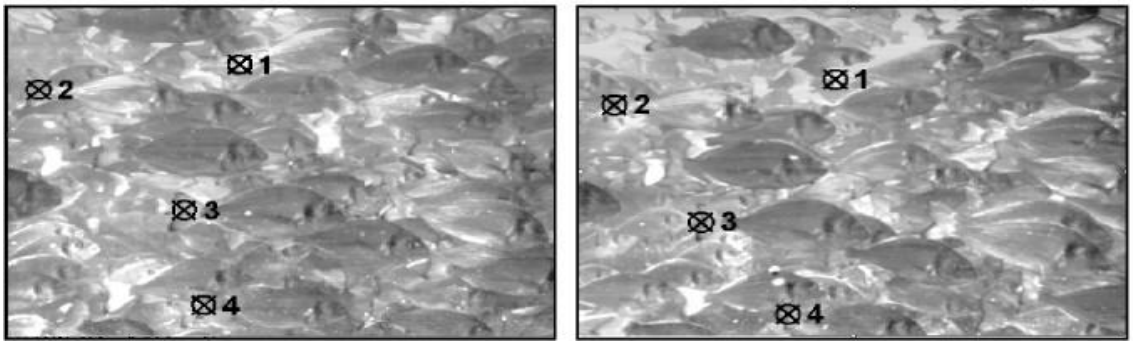


Figura 15 - Imagens com identificação dos peixes detetados e validados [22]

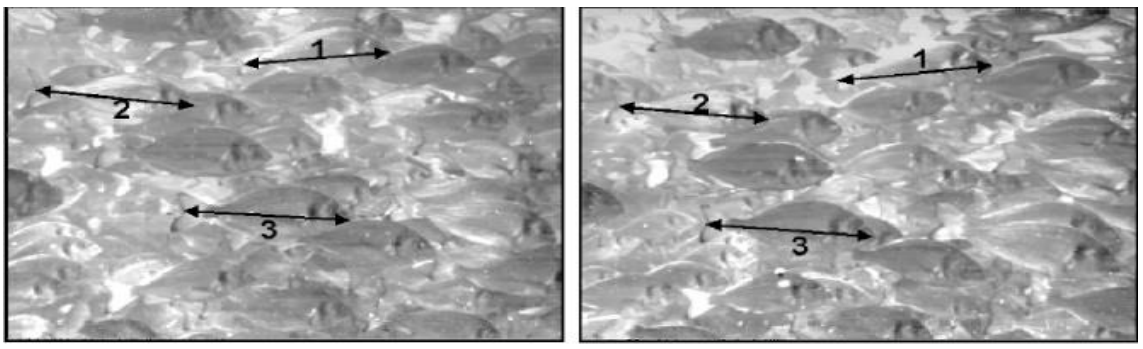


Figura 16 - Imagens resultantes após identificação dos peixes, e associação da respetiva cauda e boca [22]

No artigo "*Extracting fish size using dual underwater cameras*" (C. Costa, A. Loy, S. Cataudella, D. Davis e M. Scardi, 2006) [23], foi proposto um sistema de duas câmaras submersas conectadas a um computador à prova de água equipado com dispositivos de aquisição de imagem (*frame grabbers*). Com esta montagem eram obtidas duas imagens sincronizadamente e, através dos parâmetros de configuração como a distância focal de cada câmara, distância entre câmaras e ângulo relativo entre câmaras, obtinha-se informação relativa ao tamanho e à forma dos peixes. Foi desenvolvido um algoritmo que permitia extrair o contorno dos peixes nas imagens e, a partir desta referência, executar uma reconstrução de um modelo 3D do peixe. Com base neste modelo 3D era efetuada uma estimativa do comprimento do peixe. Utilizando esta montagem e os citados algoritmos, na determinação do comprimento dos peixes, os autores descrevem ter obtido erros de aproximadamente 2%.

Em "*A dual camera system for counting and sizing Northern Bluefin Tuna (Thunnus thynnus; Linnaeus, 1758) stock, during transfer to aquaculture cages, with a semi automatic Artificial Neural Network tool*" (Costa, Scardi, Vitalini e Cataudella, 2009) [24], é descrito o mesmo sistema apresentado no artigo anterior mas aplicado à medição de peixes reais e não de modelos de peixes como no caso anterior. Neste artigo, são descritos os testes efetuados ao sistema e os algoritmos desenvolvidos anteriormente, para efetuar a estimativa de medição de atuns-rabilho. Estas medições foram efetuadas em atuns que estavam a ser transferidos de redes de pesca para jaulas. Os autores descrevem que obtiveram um erro de calibração da estimativa do comprimento inferior a 13%. O erro na estimativa de biomassa dos peixes foi de 50,6%.

Como tem vindo a ser descrito, o uso das tecnologias de visão pode fornecer várias ferramentas que permitem facilitar e melhorar os processos produtivos no âmbito da aquacultura. Em "*Application of machine vision systems in aquaculture with emphasis on fish: state-of-the-art and key issues*" (Saberioon, Gholizadeh, Cisar, Pautsina e Urban, 2016) [13], é reforçada esta ideia. Neste artigo, é efetuada uma revisão do estado da arte relativo a trabalhos que abordem a temática das tecnologias de visão mais recentes, a utilização de vários sensores óticos na gestão da piscicultura e a avaliação, medição e estimativa da qualidade dos peixes e produtos resultantes.

Segundo o que é enunciado neste artigo, podem considerar-se dois tipos de sensores óticos passíveis de serem utilizados na monitorização de organismos aquáticos: sensores

submersíveis ou subaquáticos e sensores não submersíveis. Os sensores óticos não submersíveis são usualmente utilizados para acompanhar a evolução de determinadas características e supervisionar o comportamento dos peixes, como a atividade de alimentação dos cardumes, detetar excesso de alimentação e controlar o processo de fornecimento de alimento. Relativamente a estas aplicações, referem dois exemplos onde foram utilizadas tecnologias de visão à superfície ou nas imediações de tanques ou aquários de peixes. No caso de "*Tracking Multiple Fish in a Single Tank Using an Improved Particle Filter*" (Lee, Osman, Talib, Ogier, e Yahya, 2014) [25], foi utilizada uma câmara colocada sobre um tanque de peixes Koi, para monitorizar os padrões de nado. Nas imagens obtidas, é utilizado um algoritmo de filtro de partículas, que permite detetar alterações de velocidade ou trajetória do peixe no campo de visão do equipamento.

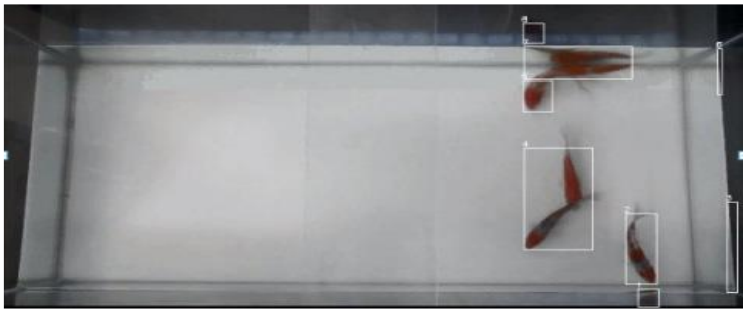


Figura 17 - Identificação de zonas com peixe [25]

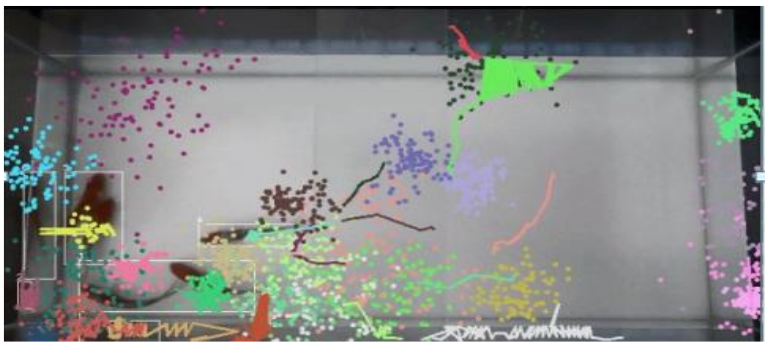


Figura 18 - Padrões de trajetória dos peixes monitorizados [25]

Um outro exemplo deste tipo de cenário é o caso do trabalho descrito no artigo de Atoum *et al*.: "*Automatic Feeding Control For Dense Aquaculture Fish Tanks",* 2015 [26], onde foi colocada uma câmara de vídeo sobre um tanque com um elevado número de peixes, cerca de 10000, que enviava as imagens para um computador onde foi efetuada uma análise em tempo real do comportamento dos peixe no campo de visão da câmara. O

esquema representativo deste sistema é demonstrado na Figura 19. Esta análise comportamental tem como fim controlar o processo de alimentação dos peixes.

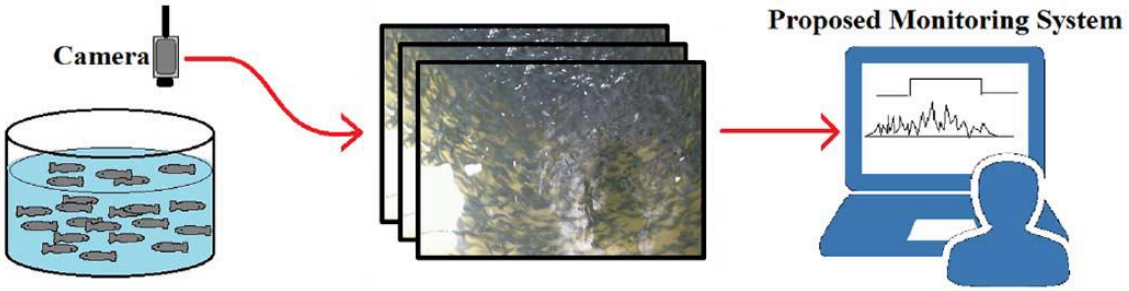


Figura 19 - Esquema representativo do sistema de análise comportamental de peixes desenvolvido [26]

Nos dois casos apresentados, em que os sistemas de visão são instalados fora do tanque, as ferramentas desenvolvidas demonstraram conseguir efetuar as tarefas pretendidas. No entanto, o uso de câmaras fora de água na aquacultura acarreta algumas desvantagens como, por exemplo, a presença de reflexos luminosos na superfície da água em frente da câmara. Este tipo de reflexões facilmente resulta em saturação das imagens obtidas, que dificulta o processamento e pode introduzir erros. Nos casos em que o sistema está instalado em tanques que estejam abrigados da luz solar ou dentro de edifícios, é possível recorrer a iluminação artificial para equilibrar. No caso das aplicações no exterior, torna-se complicado efetuar a compensação da luminosidade [22]. Tendo em vista estas afirmações, em aplicações de visão por computador na aquacultura, geralmente é mais conveniente a utilização de sensores óticos submersos.

Um outro tipo de câmara utilizado para aplicações fora de água é o sistema LIDAR *(light detecting and ranging*). O LIDAR é uma tecnologia ótica de deteção remota que permite obter distâncias e/ou outra informação a respeito de um determinado objeto distante. Este sistema transmite e recebe radiação eletromagnética de alta frequência, na gama de ultravioleta, visível e infravermelho. Este tipo de ferramenta é regularmente utilizado em atividades que podem estar relacionadas com a pesca, para efetuar a determinação da profundidade ou batimetria em águas costeiras, Churnside *et al* (2003) [27] efetuou testes de deteção de cardumes de peixe recorrendo a um LIDAR de forma a comparar com uma sonda eco batimétrica. Para o cenário de testes, o LIDAR, demonstrado abaixo na Figura 20, foi colocado num avião que sobrevoou alguns cardumes, sendo simultaneamente realizado o mesmo percurso por um barco equipado com a sonda.

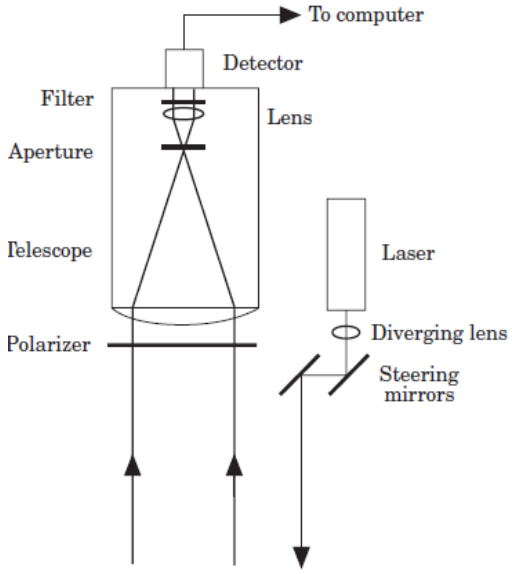


Figura 20 - Diagrama do LIDAR aéreo para deteção de peixes [27]

Os sistemas óticos subaquáticos consistem usualmente em câmaras que operam na gama do visível ou na gama perto do infravermelho e são colocadas dentro de encapsulamentos estanques. Em algumas aplicações recorre-se a câmaras a cores para, por exemplo, efetuar a identificação de espécies de peixes.

Um exemplo de aplicação de câmaras subaquáticas que operam na gama de radiação visível, em atividades marítimas, é o artigo "*Remote High-Definition Rotating Video Enables Fast Spatial Survey of Marine Underwater Macrofauna and Habitats*", (Pelletier, Leleu, Mallet, Mou-Tham, Herve, Boureau e Guilpart, 2012) [28]. Neste documento, é descrito o desenvolvimento de uma ferramenta de gravação de vídeo, denominada pelos autores STAVIRO (do Francês *STAtion VIdeo Rotative*), ou seja, um sistema de vídeo que obtém imagens de todo o espaço que a rodeia, a 360º. Na Figura 21, é apresentado um esquema dos componentes deste sistema (A) e uma imagem do sistema em ambiente real (B). O sistema é colocado no fundo do mar para recolher imagens, com o intuito de quantificar fatores bióticos e abióticos na superfície do fundo marinho e identificar e contar espécies de peixes e outro tipo de seres como tartarugas marinhas. Este sistema foi testado no recife de coral da Nova Caledónia, uma ilha do Oceano Pacifico, permitindo detetar e identificar um grande número de espécies.

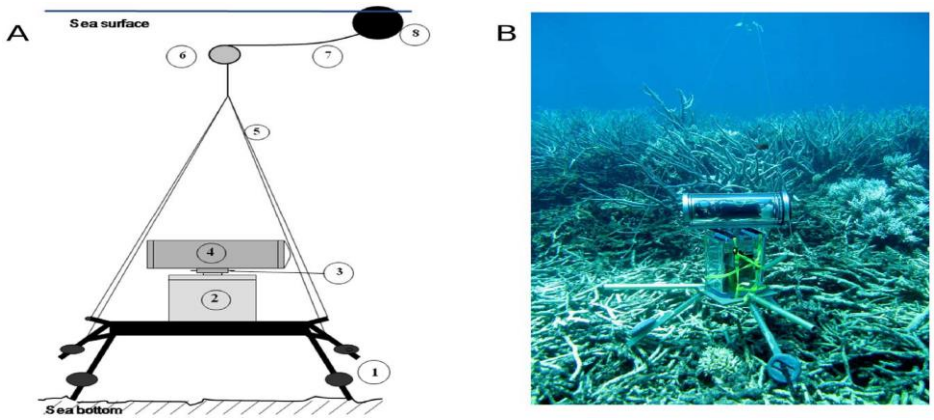


Figura 21 - Esquema dos componentes do sistema (A) e sistema em ambiente real (B) [28]

Nesta revisão de bibliografia de Saberioon [13], são também apresentados alguns exemplos de aplicações de câmaras que registam imagens na gama perto do infravermelho. Um dos exemplos é o artigo "*Automated classification of underwater multispectral imagery for coral reef monitoring*" (Gleason e Voss, 2007). Os autores criaram uma câmara multiespectral subaquática para testar se as faixas espectrais estreitas teriam melhor desempenho em aplicações de classificação automatizada de imagens subaquáticas, que as câmaras típicas, que obtêm imagens RGB. Para obter imagens com seis bandas espectrais de 10 nm diferentes, utilizou-se na lente da câmara uma roda com seis filtros correspondentes a cada banda espectral. Foi desenvolvido um algoritmo baseado na razão da diferença normalizada entre imagens obtidas com os filtros de 568 nm e 546 nm. Juntando este algoritmo com uma segmentação por textura foi possível identificar diferentes espécies de algas e corais, com uma precisão de 80%.

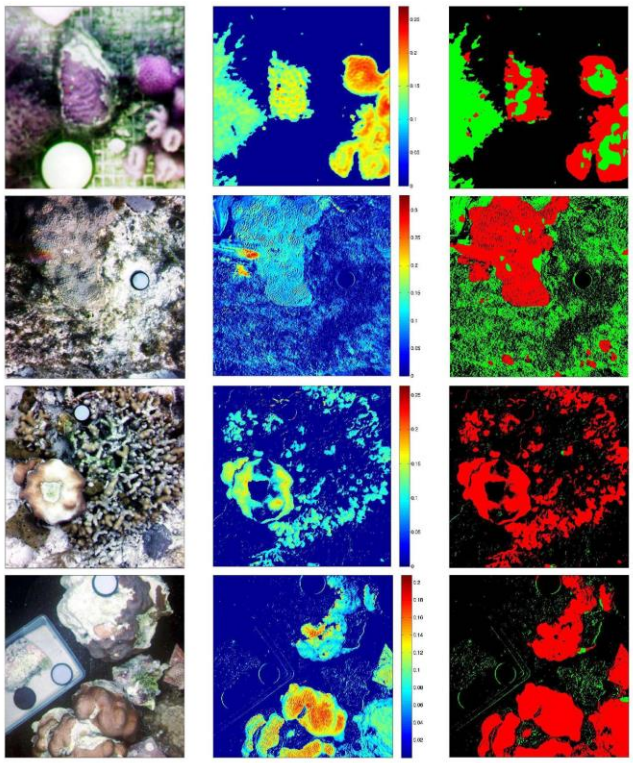


Figura 22 - Resultados obtidos com câmara multiespectral [29]

Na Figura 22, são apresentadas as imagens obtidas com esta câmara multiespectral. Na coluna da esquerda, são apresentadas imagens de cores "falsas", compostas por três bandas de câmara multiespectral (589, 548, 487 nm) exibidas em RGB. Na coluna central, são apresentadas imagens relativas á razão da diferença normalizada de bandas em 568 e 546 nm. Na coluna da direita, são apresentadas as imagens finais suavizadas, onde as algas são representadas a verde, os corais a vermelho e o fundo a preto.

Saberioon [13] refere ainda outro tipo de câmaras subaquáticas, as câmaras acústicas DIDSON (*Dual-Frequency IDentification Sonar*). Estas câmaras são constituídas por um sonar de alta frequência com um sistema de lentes acústicas de focagem do feixe para criação de imagens de alta resolução, sendo utilizadas em ambiente de elevada turbidez e/ou de baixa iluminação. Este sistema foi desenvolvido por Belcher, Hanot e Burch e é descrito no artigo "*Dual-frequency IDentification SONar (DIDSON)*" (2002). Neste artigo, são apresentados alguns exemplos de como este sistema pode ser utilizado. Por exemplo, o sonar poderia ser montado num submarino para obter imagens de objetos bastante detalhadas em água turva. Em muitas situações, recorrer a este sistema poderia evitar o envio de um mergulhador dentro de água, que em condições de fraca

luminosidade e turbidez, teria de tatear as superfícies para identificar ou inspecionar objetos.

No artigo "*Video and acoustic camera techniques for studying fish under ice: a review and comparison*" (Mueller , Brown, Hop e Moulton, 2006) [30], é descrito o uso das câmaras acústicas DIDSON em cursos de água cobertos de gelo para detetar a presença de peixes, determinar o seu tamanho, identificar espécies e determinar a velocidade a que estes nadam. Neste estudo, foram efetuadas algumas comparações com câmaras óticas equipadas com feixes laser. Abaixo, na Tabela 1, são apresentados alguns termos de comparação, identificando com o símbolo "+" (mais) os aspetos em que determinado tipo de câmara é mais vantajoso e com o símbolo "-"(menos) os aspetos menos vantajosos.

Tabela 1 - Comparação de câmaras acústicas com câmaras óticas em cursos de água cobertos de gelo [30]

Application	Acoustic camera	Optical camera with paired lasers
Near field use	–	+
Far field use	+	–
Identifying fish species	–	+
Ease of measuring fish	+	–
Measuring habitat	–	+
Ease of use in cold conditions	–	+
Cost	–	+
Ease of use	–	+
Extended survey period	–	+
Power consumption	–	+
Use in turbid water	+	–
Use in low visibility conditions	+	–
Data storage needs	–	+

Abaixo, é apresentada a Tabela 2, onde são comparados os vários métodos aplicados na medição, quantificação e estudo de comportamento de peixes, entre outros, referidos por Saberioon. A versão apresentada foi traduzida da original presente no artigo.

Tabela 2 - Tabela resumo de tecnologias de câmaras [13]

Técnica	**Princípio de Funcionamento**	**Aplicação**	**Vantagens**	**Desvantagens**
Câmara única	Visível – perto do infravermelho	Estudos de comportamento; análise de características individuais.	Fácil utilização, baixo custo, alta resolução espacial.	Não pode ser utilizada à noite, trabalho intensivo.
Acústica	Sonar multiespectral de alta frequência.	Estudo de comportamento; classificação de espécies.	Útil para ambientes aquáticos com pouca luz.	Baixa resolução espacial.
Visão estéreo	Duas câmaras no espectro visível.	Estudo de comportamento; classificação de espécies; medição de características (tamanho e peso); contagem.	Exatidão; alta resolução espacial.	Requer calibração cuidada e conhecimentos técnicos.
LIDAR	Radiação eletromagnética de alta frequência; opera no espectro UV, visível e infravermelhos.	Estudos de comportamento; identificação de espécies.	Pode ser utilizado à noite; elevada exatidão.	Custo elevado; necessários conhecimentos técnicos.

3 Técnicas de Visão por Computador aplicadas no Projeto Aquatropolis

3.1 Princípios da Visão Estéreo e 3D

No âmbito do projeto Aquatropolis, recorreu-se à utilização da técnica de visão estéreo, particularmente o estéreo binocular, para executar a medição e estimativa da biomassa presente nos tanques de aquacultura. As justificações para a escolha desta técnica serão apresentadas no capítulo relativo aos trabalhos desenvolvidos neste projeto, no capítulo Projeto Aquatropolis - Intelligent Management System for Sustainable Aquacultures. É relevante referir que no desenvolvimento dos algoritmos de visão artificial foi utilizada a biblioteca de *software* de visão MVTec HALCON que, em alguns casos, facilita o desenvolvimento de determinadas tarefas.

Neste capítulo, são descritos os princípios de funcionamento de um sistema de visão estéreo, bem como os princípios teóricos para obtenção de um objeto tridimensional através de imagens adquiridas a partir de um par de câmaras em estéreo.

A visão estéreo é um dos campos mais relevantes no universo da visão por computador. Esta técnica tem vindo a ser utilizada no mais variado tipo de áreas. Com um sistema estéreo, duas ou mais câmaras, é possível obter informação tridimensional do mais variado tipo de objetos, resultando, por exemplo, em imagens de distância, coordenadas 3D ou modelos tridimensionais de superfícies [31], [32] . O estéreo baseia-se no funcionamento da visão humana, no sentido em que, qualquer objeto visualizado é resultado do processamento de dois pontos de vista, ou seja, o mesmo objeto poderá ter várias projeções bidimensionais dependendo do ponto de vista.[33], [34] Assim, obtendo pelo menos duas imagens do mesmo objeto de dois pontos de vista distintos, é possível executar uma reconstrução 3D deste. Para executar esta tarefa, é efetuada uma correspondência entre os pixéis da primeira imagem com os da segunda, através da sua semelhança e, com esta informação, é criado um mapa de disparidades (*disparity map*). Posteriormente, é efetuada a reconstrução 3D de acordo com o mapa de disparidades. [10], [34]

Os procedimentos para obtenção de resultados com um sistema estéreo consistem essencialmente em aquisição de imagem, calibração do sistema, retificação do par de

imagens estéreo, extração de características, correspondência estéreo e processamento da informação de profundidade.[32], [33] Na Figura 23, são apresentados estes passos de um ponto de vista semelhante, mas em que inicialmente são adquiridas imagens de calibração, posteriormente o sistema é calibrado, e de seguida são adquiridas as imagens estéreo.

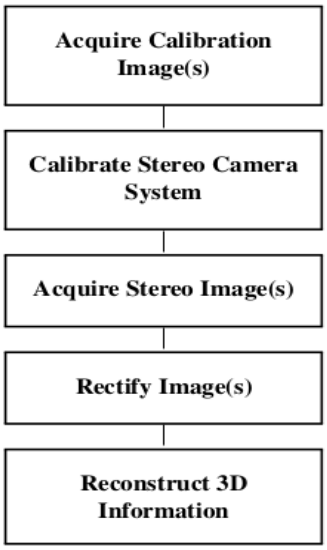


Figura 23 - Processo de reconstrução da informação 3D através de imagens de um sistema estéreo [32]

Considerando um par de câmaras semelhantes entre si, colocadas paralelamente a observar o mesmo objeto de dois pontos de vista distintos, separados por uma distância de base *b***,** obtém-se consequentemente dois pontos de vista diferentes. Considerando a linha de base *b* como uma linha que une os centros óticos de cada câmara, os eixos das coordenadas *x* de cada câmara coincidirão com esta linha. Considerando as coordenadas de dois pontos, um da imagem esquerda e o segundo da imagem da direita, previamente identificadas como correspondentes por um processo de correspondência estéreo, é possível estimar a distância ao objeto ou profundidade (*depth*), através da disparidade entre pontos, ou separação dos dois pontos em coordenadas de pixel da imagem.

Dadas as coordenadas de pixel do ponto na imagem esquerda (X_L,Y_L) e as coordenadas de pixel do ponto na imagem direita (X_R, Y_R), as coordenadas 3D que representam os valores reais (X, Y, Z) são definidas pelas seguintes expressões:

$$X = \frac{X_L b}{d}, Y = \frac{Y_L b}{d}, Z = \frac{f b}{d} \qquad \text{Equação 1}$$

Nestas expressões, *f* representa a distância focal das câmaras e *d* a disparidade entre os ponto da câmara direita e da câmara esquerda [34], [35]. Na Figura 24

Figura 24 - Representação de um sistema estéreo [34]

, é apresentado um esquema de um sistema estéreo que permite perceber melhor a relação entre estas variáveis e as coordenadas *X*, *Y* e *Z*.

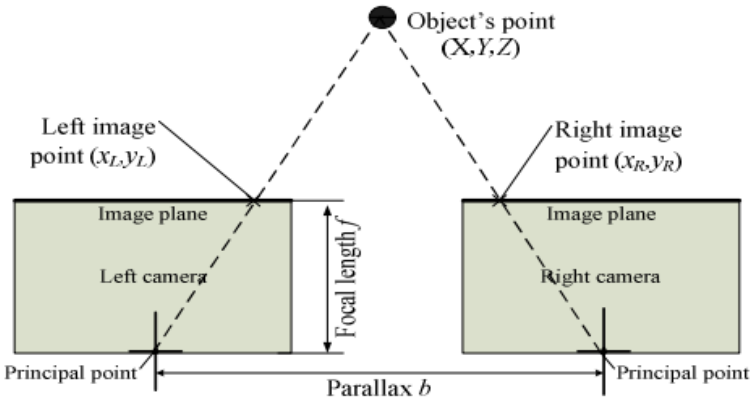


Figura 24 - Representação de um sistema estéreo [34]

Como o mesmo ponto ou objeto é observado de dois ângulos de vista diferentes, a representação deste na imagem obtida na câmara esquerda terá uma localização diferente da representação na imagem obtida na câmara direita. A esta diferença dá-se o nome de disparidade (*d*) ou *disparity* ou ainda paralaxe.

Considerando as coordenadas X_L e X_R, correspondentes ao valor de coordenada *X* da câmara esquerda e de coordenada *X* da câmara direita, respetivamente, o valor de disparidade ou paralaxe (*d*) em pixéis é dado pela seguinte expressão[34]:

$$d = X_L - X_R \qquad \text{Equação 2}$$

É pertinente referir que os sistemas de visão estéreo possuem melhor resolução de profundidade para objetos que se encontrem relativamente próximos às câmaras, ou seja, quanto mais distante o objeto estiver do sistema de câmaras, mais difícil será estimar esta distância. [6], [36], [37] A Figura 25 demonstra esta relação entre a disparidade e a profundidade ou distância ao objeto.

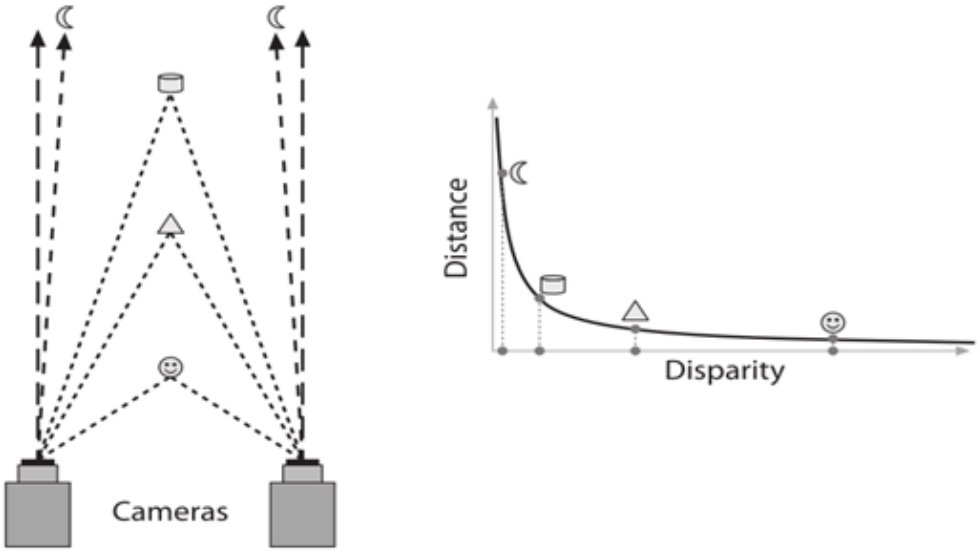


Figura 25 - Relação da disparidade com a distância ao objeto [36]

3.1.1 Erro de Estimativa da Distância ao Objeto

Para definir o erro de estimativa da distância ao objeto, assume-se que o sistema estéreo está calibrado e as imagens analisadas nesta situação estão retificadas. A correspondência entre os pontos do par de imagens do estéreo é simplificada, reduzindo para uma linha a área de procura de correspondência entre pontos. Considerando a expressão para a determinação dos valores de *Z*, na Equação 1, e a expressão para determinação da disparidade, Equação 2, é possível deduzir a expressão para determinar o erro de estimativa da distância ao objeto, a Equação 3. Nesta determinação, não são considerados os erros introduzidos pelas variáveis f e b e assume-se que os valores de X_L e X_R são independentes um do outro. É conhecido também que o erro da estimativa deste valor Z é sensível aos erros provenientes da disparidade d. [34], [37], [38]

$$\Delta Z^2 = \left(\frac{\partial Z}{\partial d}\Delta d\right)^2 \Leftrightarrow \Delta Z = \frac{Z^2 \cdot \Delta d}{f \cdot b} \qquad \text{Equação 3}$$

Atendendo à expressão demonstrada, pode concluir-se que a estimativa da distância ao objeto é menos fiável para objetos que se encontrem mais distantes. Este erro pode ser mitigado configurando o sistema de câmaras de tal maneira que o valor de distância base (b) e de distância focal (d' ou f) sejam elevados, e o sistema seja colocado relativamente perto do objeto em observação, uma vez que a resolução diminui com o quadrado da distância. A distância de resolução é diretamente proporcional à precisão com que a disparidade (Δd) pode ser obtida. [6], [34]. Por exemplo, com uma calibração efetuada

com precisão e com um erro de calibração menor ou igual a 0,1 pixel, obtêm-se valores de disparidade com uma precisão de 1/5 a 1/10 de pixel.[39]

3.1.2 Revisão do Estado da Arte da Calibração Estéreo

A calibração de câmaras no contexto da visão por computador é o processo que visa determinar as características e parâmetros geométricos e óticos da câmara (parâmetros internos) e o posicionamento e orientação espacial da câmara relativamente a um determinado sistema de coordenadas de referência (parâmetros externos). O processo de calibração e seu sucesso têm um grande impacto no resultado das aplicações a que precedem. [40], [41]

No artigo "*Camera Calibration with One-Dimensional Objects*" (Zhang, 2002)[41], é defendido que as várias técnicas de calibração de câmaras podem ser divididas por três secções de acordo com o número de dimensões do objeto utilizado para executar tal tarefa: a calibração baseada num objeto 3D como referência, calibração baseada num objeto 2D e auto calibração ou *self calibration*. Para o presente relatório, o caso mais pertinente é o da calibração com objeto 3D.

Este tipo de calibração é executado colocando a câmara a observar um objeto de calibração tridimensional, com uma geometria previamente conhecida detalhadamente. Nesta abordagem, usualmente, considera-se que a câmara utilizada se baseia no princípio da lente de *pinhole*, representado na Figura 26, que apresenta um funcionamento semelhante à retina do olho humano. Este modelo simplifica o sistema da câmara, definindo-a apenas como uma caixa com um orifício numa das faces e uma placa fotográfica no lado oposto. Este dispositivo efetua uma projeção em perspetiva do espaço 3D num plano, intersetando a linha formada pelos pontos *c* e *p* com o plano de imagem em ***q,*** onde *c* corresponde ao centro ótico do sistema. [42], [43]

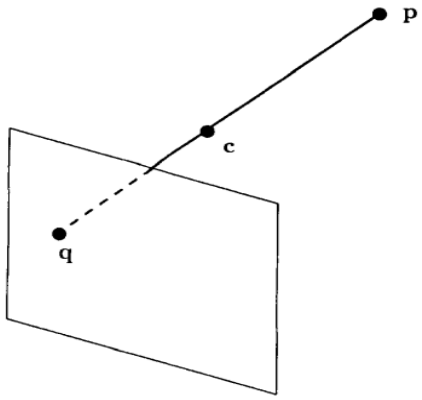


Figura 26 - Modelo de câmara *pinhole* [42]

Este objeto de calibração tridimensional consiste em dois planos ortogonais entre si, com um padrão em grelha, tal como demonstrado na Figura 27. São conhecidas com precisão as coordenadas dos quadrados brancos presentes nestes padrões, que permitem definir um sistema de coordenadas tal como ilustra a mesma imagem. Uma vez conhecidas as características do objeto de calibração, é efetuada a correspondência dessa informação com o que é apresentado nas imagens. A correspondência entre os pontos 3D e as suas projeções nas imagens resulta num mapa linear projetivo de P3 para P2. (P3 é a notação utilizada para referir o espaço projetivo tridimensional e P2 é a notação utilizada para referir o espaço projetivo bidimensional). O ponto de imagem *q* é apurado através da projeção de um ponto *p* no espaço e é obtido recorrendo à expressão $q = M.p$, onde *M* é uma matriz de 3 x 4. A matriz *M* pode ser definida recorrendo a métodos de otimização, dados os vetores de coordenadas p_i dos pontos de interesse no padrão de calibração e os vetores de coordenadas q_i, relativos às projeções desses pontos nas imagens da retina (*cit* por MAYBANK [42], Faugeras 1992 [44]; Horn 1986 [18]). A partir desta matriz, a posição e a orientação da câmara e a imagem da cónica absoluta podem ser definidas e referenciadas relativamente ao sistema de coordenadas apresentado na Figura 27.

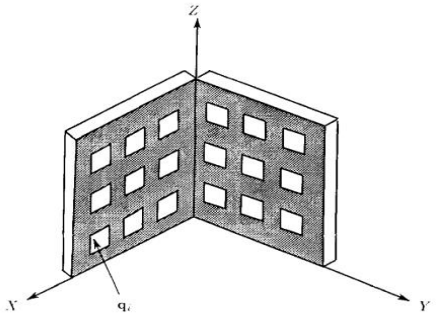


Figura 27 - Padrão de calibração [42]

Citando a dissertação "Método e Ferramenta de Apoio à Realização de Croquis de Acidentes Rodoviários a Partir de Imagens Aéreas" (Castro, 2017) [46], para esclarecer o conceito de "cónica absoluta", "*Uma cónica é a curva no plano gerada quando esse plano atravessa um cone. A lista de curvas que podem ser geradas são a parábola, o círculo, a elipse, e a hipérbole*". Em "*A Theory of Self-Calibration of a Moving Camera*" (Maybank e Faugeras, 1992) [42], também é explicado este conceito, onde se descreve a "cónica absoluta" como uma cónica particular no infinito de um plano, que permanece invariante relativamente aos movimentos (translações, rotações, entre outros) efetuados por este plano no espaço. A imagem da cónica absoluta determina e é determinada pela calibração das câmaras. A posição e a orientação da câmara são os parâmetros externos da calibração. Os coeficientes da equação da imagem da cónica absoluta produzem os parâmetros internos da calibração.

Por vezes, na calibração 3D, é utilizado apenas um plano que é deslocado em frente do sistema de câmara, obtendo imagens com várias orientações em movimento, o que permite igualmente aferir pontos 3D [47]. Segundo Zhang, à data do seu artigo, este tipo de calibração era elaborado do ponto de vista técnico e requeria equipamento caro.

3.1.3 Calibração com a Biblioteca de S*oftware* MVTec HALCON

Tal como referido anteriormente, as soluções de visão no âmbito deste estágio foram desenvolvidas recorrendo à biblioteca de *software* de visão MVTec HALCON, doravante denominado apenas HALCON, tirando partido do IDE e todas as utilidades associadas a este *software*. A utilização desta ferramenta permite agilizar o processo de desenvolvimento, facilitando muitas tarefas, como a calibração, funções de reconstrução 3D e medição de objetos.

O método de calibração do HALCON pode considerar-se semelhante ao método de calibração baseado num objeto 3D descrito por Tsai em "*A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses*" [47], no entanto bastante mais simples uma vez que a ferramenta de *software* executa todo o processamento recorrendo a algumas funções, permitindo que o seu utilizador se abstraia de algumas considerações matemáticas e, por exemplo, expressões de cálculo matricial de dedução dos pontos de interesse dos objetos de calibração. De modo a detalhar o processo de calibração de um sistema estéreo proposto

na documentação do HALCON, são de seguida apresentadas algumas considerações teóricas pertinentes citando o manual do *software* relativo a este processo [39].

3.1.3.1 Considerações Relativamente às Câmaras a Calibrar

O HALCON permite a calibração de dois tipos de câmaras, *area scan* ou *projective area scan* e *line scan*. As câmaras *area scan* obtêm a imagem apenas numa iteração, e as câmaras *line scan* constroem a imagem obtendo a imagem linha a linha. Apenas câmaras do mesmo tipo podem ser utilizadas no mesmo processo de calibração estéreo ou multi-câmara.

No manual "*Solution Guide III C 3D Vision*" [39], são descritos dois tipos de lentes, sendo consideradas as mais relevantes para as aplicações de visão por computador. O primeiro tipo de lente funciona de maneira semelhante ao olho humano, projetando em perspetiva as coordenadas reais dos objetos observados no plano da imagem, ou seja, o tamanho da representação dos objetos na imagem diminui à medida que estes se afastam da lente. Este conceito corresponde basicamente à descrição da câmara *pinehole* enunciada anteriormente. O segundo tipo de lente são as lentes telecêntricas que, ao contrário do caso anterior, efetuam uma projeção paralela dos objetos observados no plano da imagem, dentro de um determinado campo de visão, ou seja, os objetos são representados na imagem com o mesmo tamanho que apresentam no mundo real. Esta combinação de câmara e lente é denominada *telecentric camera model*.

3.1.3.2 Conversão de Pontos 3D do Mundo Real para Coordenadas de Pixel

Essencialmente, uma calibração permite determinar a relação de conversão de pontos 3D do mundo real para coordenadas de pixel no referencial da câmara. A Figura 28 apresenta um esquema que facilita a perceção da transformação das coordenadas de um determinado ponto *P* no mundo real, a sua conversão para o sistema de coordenadas do plano virtual da imagem e a sua relação com o referencial de coordenadas da câmara. Este esquema corresponde ao funcionamento de uma câmara *pinhole*, no entanto, para facilitar a perceção de conceitos, o plano virtual da imagem é representado à frente do centro ótico da câmara à distância *f*, correspondente à distância focal da câmara. Assim, o sistema de coordenadas da imagem estará alinhado com o sistema de coordenadas de pixel, ou seja,

as coordenadas de linha progridem de cima para baixo e as coordenadas de coluna da esquerda para a direita, permitindo simplificar alguns processos de cálculo. Na realidade, o plano da imagem é definido atrás do centro ótico da câmara.

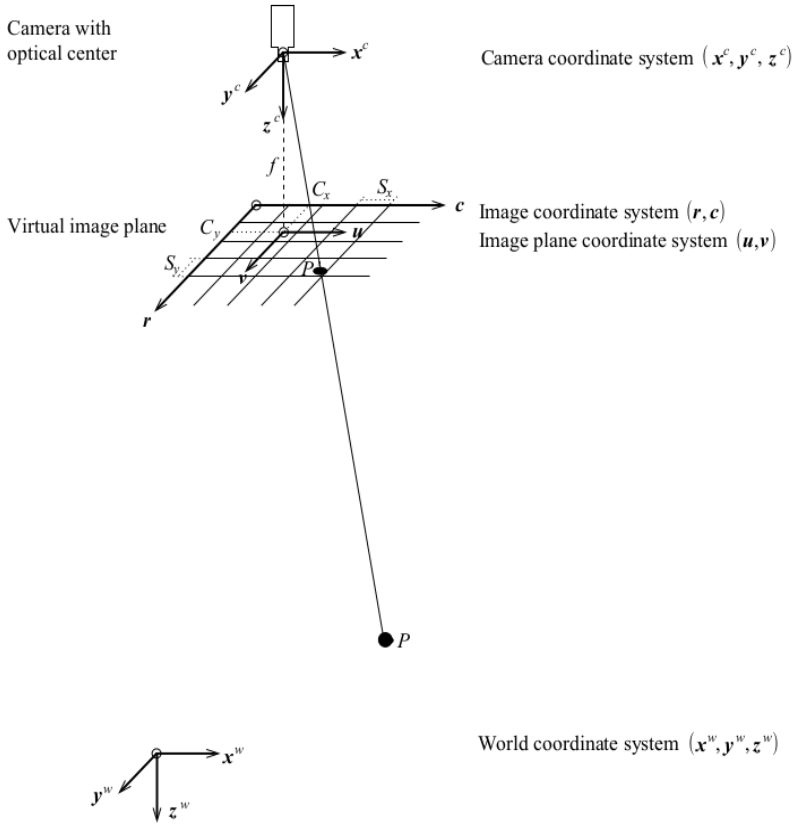


Figura 28 - Representação do plano da imagem e do plano virtual da imagem [39]

Um ponto *P* no campo de visão da câmara é definido em coordenadas relativas às dimensões reais (*world coordinate system* (WCS)). Para converter estes pontos para um referencial bidimensional, é efetuada uma projeção dos pontos e objetos no plano da imagem, convertendo-os para o referencial de coordenadas da câmara (*camera coordinate system* (CCS)). Este CCS é definido de tal forma que os eixos *x* e *y* sejam respetivamente paralelos às colunas e linhas da imagem, e o eixo *z* perpendicular ao plano da imagem. A conversão de WCS para CCS é uma isometria, que pode ser expressa por uma matriz de transformação com coordenadas homogéneas, ${}^cH_w$. As coordenadas da câmara para o ponto *P*, $p^c=(x^c,y^c,z^c)$, podem ser aferidas a partir das coordenadas reais (WCS), $p^w=(x^w,y^w,z^w)^T$ recorrendo à Equação 4:

$$p^c = {}^cH_w \cdot p^w \qquad \text{Equação 4}$$

O resultado desta operação serão seis parâmetros relativos à transformação executada, os parâmetros associados às translações por cada eixo coordenado, t_x, t_y, e t_z, e os parâmetros associados às rotações em cada um dos eixos α, β e γ. Estes parâmetros são denominados parâmetros externos da câmara, pois determinam a posição da câmara relativamente ao referencial do ambiente que a envolve. Num algoritmo desenvolvido recorrendo ao HALCON, esta informação é guardada como uma *pose*, uma variável que armazena, por ordem, os parâmetros de cada translação e rotação.

Uma *pose* descreve uma transformação geométrica tridimensional, correspondente as translações e rotações, definidas por seis parâmetros: *TransX*, *TransY* e *TransZ*, relativos à translação ao longo dos eixos *x*, *y* e *z*, respetivamente, para descrever a rotação são utilizados os parâmetros *RotX*, *RotY* e *RotZ*. Um dos propósitos destas *poses* tridimensionais é descrever a posição e a orientação de um sistema de coordenadas em relação a outro. Por exemplo, a *pose* do sistema de coordenadas de um objeto em relação ao sistema de coordenadas da câmara. As *poses* são empregues igualmente para descrever a transformação de coordenadas entre dois sistemas de referência. Por exemplo, para transformar pontos de coordenadas de um objeto em coordenadas da câmara.

Uma vez determinadas as coordenadas 3D relativas ao CCS do ponto em análise, é efetuada a projeção deste ponto no sistema de coordenadas do plano da imagem (*image plane coordinate system* (IPCS)). No referido manual, a projeção em perspetiva para o modelo de câmara *pinhole*, é dada pela seguinte expressão:

$$q^c = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{f}{z^c} \begin{pmatrix} x^c \\ y^c \end{pmatrix} \qquad \text{Equação 5}$$

Após efetuada esta operação, é necessário considerar o efeito de distorção causado pela lente. Este efeito é demonstrado na Figura 29. A referida distorção provoca uma alteração no posicionamento do ponto de projeção *P'*. Caso não existisse a distorção da lente, a projeção P' ficaria alinhada com o ponto *P* e o centro ótico. Este alinhamento está representado na Figura 29 por uma linha desenhada por pontos.

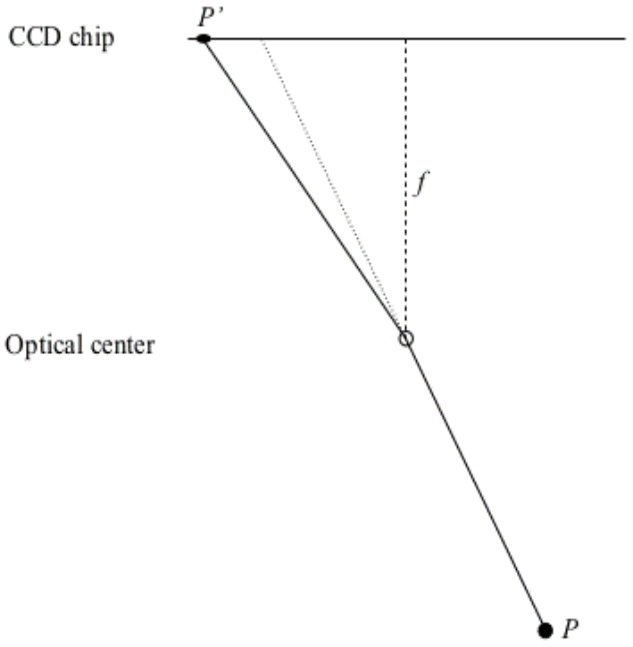


Figura 29 - Esquema ilustrativo do efeito de distorção da lente [39]

O efeito de distorção da lente pode ser modelado no plano da imagem, não necessitando de informações tridimensionais. No HALCON, as distorções podem ser modeladas pelo modelo de divisão, *division model*, ou pelo modelo polinomial, *polynomial model*. O modelo utilizado na maioria dos cenários é o *division model*.

Para definir as distorções radiais, o *division model* utiliza um parâmetro κ que traduz a magnitude das distorções radiais. Se κ for inferior a zero, a distorção será em forma de barril, achatada nos vértices e mais larga no centro (Figura 30, imagem da esquerda). Se κ for positivo, a distorção será em forma de almofada, achatada no meio e mais larga nos vértices (Figura 30, imagem da direita).

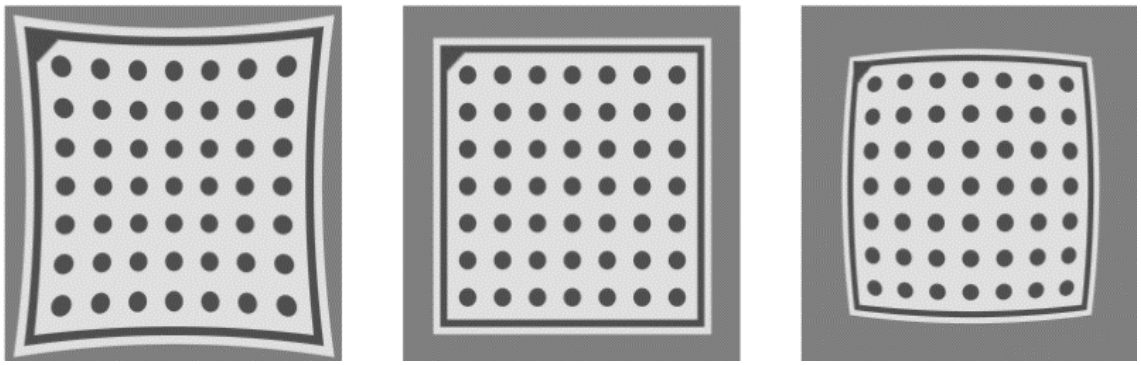


Figura 30 - Efeito da distorção radial modelada pelo *division model* [39]

3.1.3.3 Procedimentos de Calibração de um Sistema e Estéreo ou 3D no HALCON

Uma das grandes vantagens da utilização do HALCON e das ferramentas associadas, em relação a outros *softwares* ou bibliotecas de visão por computador, é o facto de simplificar bastante o processo de calibração. No manual do HALCON relativo às aplicações 3D em visão por computador [39], o processo de calibração é delineado por três passos: preparação, calibração e, por último, o acesso e armazenamento dos resultados de calibração. Este processo é efetuado recorrendo aos objetos ou placas de calibração providenciados pelo fabricante desta ferramenta de *software*.

Utilizando as placas de calibração fornecidas, este *software* permite detetar automaticamente o objeto nas imagens de calibração, efetuar a extração das marcas de calibração e determinar as correspondências entre as marcas de calibração extraídas e as respetivas coordenadas reais 3D. Utilizando estas placas é possível efetuar uma calibração com elevado grau de precisão, até 1 μm ou menos, que é um pré-requisito para aplicações de alta precisão. O HALCON permite a utilização de dois tipos de placa de calibração, placa com marcas dispostas de forma hexagonal (Figura 31 à esquerda) e placa com marcas dispostas de forma retangular (Figura 31 à direita). Existem algumas regras a seguir ao obter imagens de calibração com estes objetos, de modo a garantir a melhor precisão de calibração possível.

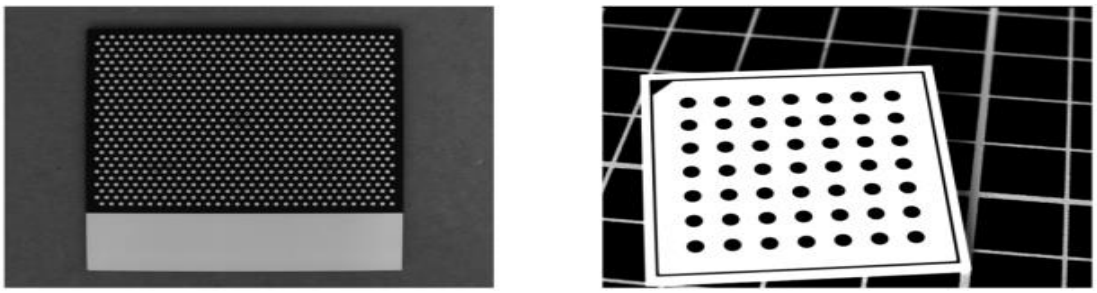


Figura 31 - Placas de calibração HALCON [39]

Para a calibração com a placa com marcas dispostas de forma retangular, é necessário fotografá-la de tal forma que ocupe, pelo menos, um quarto da área da imagem. É necessário obter 10 a 20 imagens diferentes com a placa colocada em várias posições e com vários ângulos de inclinação. A placa contém uma marca triangular e uma moldura retangular que permitem determinar a sua orientação e posicionamento relativos. Como contém apenas uma marca triangular diferenciadora, é necessário que nestas imagens a

placa de calibração esteja completamente visível, de forma a determinar o posicionamento e orientação da placa. O sistema de coordenadas deste tipo de objeto de calibração está localizado no centro de todas as marcas.

Já para as placas com marcas dispostas de forma hexagonal, as marcas estão colocadas numa estrutura hexagonal de tal forma que para cada marca padrão existem outras seis marcas equidistantes. Este tipo de placa figura um número maior de marcas de calibração por placa, comparando com o outro tipo de placa, e assim deve cobrir praticamente toda a área da imagem. Estas placas de calibração contêm um a cinco padrões de localização, que são marcas dispostas de forma hexagonal, tal como as restantes marcas, mas onde quatro ou seis marcas contêm um furo. Para encontrar a placa de calibração na imagem e estimar a sua posição em relação à câmara, é utilizado o operador *find_calib_object*. Este tipo de placa, ao contrário da anterior, não necessita de ser totalmente visível na imagem quando fotografada, sendo que apenas um dos padrões de localização necessita de estar completamente visível na imagem. São necessárias menos imagens, seis ou sete, para obter os mesmos resultados que o tipo de placa de calibração com marcas dispostas retangularmente. O sistema de coordenadas da placa de calibração com marcas dispostas em hexágono está localizado no centro da marca central do primeiro padrão de localização.

Cada placa de calibração é fornecida com um ficheiro de calibração. Os arquivos de descrição para placas de calibração com marcas dispostas em hexágono têm a extensão de arquivo "*cpd*" e os arquivos de descrição para placas de calibração com marcas dispostas retangularmente têm a extensão de arquivo "*.descr*".

Na preparação da informação de calibração é criado o modelo de calibração (*calibration data model*) e é especificado o número de câmaras. A criação do *calibration data model* é efetuada com a função *create_calib_data*, que recebe como argumentos o tipo de calibração, o número de câmaras do sistema e o número de objetos de calibração.

```
create_calib_data ('calibration_object', 1, 1, CalibDataID)
```

De seguida, os parâmetros internos das câmaras são associados ao modelo de calibração recorrendo ao operador *set_calib_data_cam_param*. Para além da variável associada ao modelo de calibração, *CalibDataID*, são introduzidos como argumentos o índice da câmara referente à calibração, *CameraIdx*, começando em 0 até ao número de câmaras

menos um, ou seja, por cada câmara existente é chamado este operador, sendo introduzindo um índice de câmara diferente. É ainda introduzido na função o argumento relativo aos parâmetros internos da câmara, *CameraParam*, sendo que esta variável pode ser inicializada recorrendo a um ficheiro com esta informação, ou uma variável previamente criada com esta informação. O argumento *CameraType* é deixado em branco uma vez que essa informação já é introduzida no *CameraParam*.

```
set_calib_data_cam_param (CalibDataID, CameraIdx,CameraType,CameraParam)
```

Para criar previamente a variável a introduzir no parâmetro *CameraParam*, utiliza-se a função *gen_cam_par_area_scan_division*. Esta função assume a utilização de uma câmara do tipo *area scan* e a distorção da lente modelada pelo *division model*. Nela são introduzidos os argumentos relativos à distância focal da lente usada, *Focus*, ao coeficiente de distorção da lente, *Kappa*, às dimensões do sensor dos elementos da câmara, S_x e S_y, ao ponto central da imagem, C_x e C_y, e às dimensões da imagem, *ImageWidth* e o *ImageHeight*. A função tem ainda um último parâmetro, o parâmetro de saída *CameraParam*, uma variável que irá retornar toda a informação introduzida, onde também constará informação relativa ao tipo da câmara usada.

```
gen_cam_par_area_scan_division (Focus, Kappa, Sx, Sy, Cx, Cy, ImageWidth, ImageHeight, CameraParam))
```

Uma vez relacionada a informação relativa às câmaras a calibrar com o modelo de calibração, é necessário introduzir as características do objeto de calibração, como as dimensões e coordenadas das marcas do padrão de calibração, informação que irá permitir aferir as dimensões reais de objeto presentes nas imagens. Recorrendo ao operador *set_calib_data_calib_object*, é definido o objeto de calibração utilizado, associando-o ao modelo de calibração previamente criado (*CalibDataID*). É introduzido o índice do objeto de calibração, *CalibObjIdx*, parâmetro relevante no caso de se utilizar mais do que um objeto de calibração. Tal como na formulação do modelo de calibração, o índice do objeto de calibração é definido entre 0 e o número de objetos de calibração menos um. A descrição do objeto de calibração, *CalibObjDescr*, introduzida recorrendo aos referidos ficheiros de descrição.

```
set_calib_data_calib_object (CalibDataID, CalibObjIdx ,CalibObjDescr)
```

3.1.3.4 Obtenção de Imagens de Calibração

Uma das fases cruciais do processo de calibração é a obtenção de imagens das placas de calibração. Neste procedimento, são obtidas imagens da placa de calibração, colocada em várias posições e orientações dentro do campo de visão da câmara. Em cada imagem são extraídos os marcadores do objeto de calibração e as suas coordenadas em valor de pixel. A informação é colocada numa tupla que é posteriormente guardada na informação do modelo de calibração, juntamente com os índices de cada câmara, o índice do objeto de calibração e o índice de posicionamento (*pose*) do objeto de calibração.

A localização da placa de calibração na imagem e a respetiva extração das coordenadas de pixel de cada marca da placa de calibração é efetuada recorrendo ao operador *find_calib_object*, que executa a extração de informação automaticamente guardando-a na variável do modelo de calibração *CalibDataID*.

De seguida, é demonstrado um excerto de código retirado do manual do HALCON "*Solution Guide III C 3D Vision*" [39], onde é percorrido um conjunto de imagens extraindo a citada informação, que é guardada no modelo de calibração associada ao índice de câmara, índice de objeto de calibração e ao índice de posicionamento (*pose*) do objeto de calibração.

```
for I := 1 to NumImages by 1
    read_image (Image, ImgPath + 'calib_image_' + I$'02d')
    find_calib_object (Image, CalibDataID, 0, 0, I, [], [])
endfor
```

3.1.3.4.1 Recomendações para Adquirir Imagens de Calibração

No manual [39], são apresentadas algumas recomendações que visam melhorar o processo de calibração de câmaras, de forma a obter resultados com maior precisão. De seguida, são apresentadas estas recomendações:

- Utilizar uma placa de calibração sem sujidade;
- Na imagem obtida, uma marca de calibração deve ocupar pelo menos 20 pixéis;
- A placa de calibração deve ser fotografada com iluminação homogénea;

- As partes escuras e claras das placas de calibração devem apresentar um contraste superior a 140 valores de cinzento;
- A imagem não deve apresentar-se sobre-exposta, o valor de cinzento das partes claras deve estar estritamente abaixo de 255;
- Ao utilizar placas de calibração com marcas dispostas retangularmente, utilizar iluminação que permita que o fundo envolvente seja mais escuro que o objeto de calibração;
- A imagem deve conter o menor ruído possível e ter as arestas dos objetos bem definidas;
- A placa de calibração deve ser colocada em todas as áreas do campo de visão, pelo menos, uma vez;
- Devem variar-se a orientação da placa de calibração, executando rotações em torno dos seus eixos *x* e *y*, permitindo definir as distorções de perspetiva e aferir a distância focal;
- Para placas de calibração com marcas dispostas de forma hexagonal, recomendam-se ângulos de inclinação de aproximadamente 15 graus. Para placas de calibração com marcas dispostas retangularmente, recomendam-se ângulos de inclinação de aproximadamente 45 graus;
- É ainda recomendado obter imagens que cubram de forma homogénea o campo de visão das câmaras. Se for utilizada uma placa de calibração pequena comparada com o campo de visão das câmaras, irá ser necessário obter um maior número de imagens de calibração.

Nos sistemas estéreo, as câmaras adquirem simultaneamente várias imagens do objeto ou objetos de calibração dispostos em vários posicionamentos e orientação, tal como detalhado para a calibração de apenas uma câmara, existindo, no entanto, algumas diferenças neste procedimento para sistemas estéreo e multi-câmara. A grande diferença reside no facto do objeto de calibração ter de estar visível em algumas das partes sobrepostas das imagens que são capturadas por câmaras vizinhas, seguindo as mesmas regras apresentadas anteriormente, relativamente à porção que é necessário ser visível para cada tipo placa. É bastante importante garantir que a configuração das câmaras se mantém inalterada durante a aquisição de imagens de calibração e durante a aquisição das imagens estéreo do objeto a ser investigado, caso contrário a calibração executada deixa de ser válida.

3.1.3.5 Execução da Calibração e Acesso aos Resultados

Até aqui foram demonstrados os vários procedimentos associados à preparação do modelo de calibração. Uma vez reunida toda a informação necessária no referido modelo, a calibração é executada recorrendo à função *calibrate_cameras*, que tem uma variável de entrada, o modelo de calibração *CalibDataID* e uma variável de saída relativa aos erros de calibração, *Errors*. Abaixo figura a utilização desta função:

```
calibrate_cameras (CalibDataID, Errors)
```

A variável de saída relativa aos erros de calibração, *Errors*, corresponde à distância média, em pixéis, entre a retroprojeção dos pontos de calibração e as suas coordenadas extraídas na imagem. Considera-se que uma calibração é bem-sucedida quando o valor de erro obtido é igual ou inferior a 0,1 pixel.

Os resultados da calibração, como os parâmetros internos das câmaras e a *pose* da placa de calibração em cada imagem utilizada para aferir os pontos correspondentes, são armazenados no modelo de dados de calibração e podem ser acedidos utilizando o operador *get_calib_data*. Para guardar toda esta informação numa variável que permita a utilização dos dados posteriormente, é utilizado o operador *write_camera_setup_model*, que guarda as informações da montagem numa variável com extensão "*.csm*". A obtenção dos parâmetros de duas câmaras de uma montagem estéreo e respetiva *pose*, e ainda o armazenamento desses dados numa variável do tipo "*.csm*", deve ser executada conforme demonstrado no excerto de código HALCON demonstrado abaixo.

```
get_calib_data (CalibDataID, 'camera', 0, 'params', CamParamL)
get_calib_data (CalibDataID, 'camera', 1, 'params', CamParamR)
get_calib_data (CalibDataID, 'camera', 1, 'pose', cLPcR)
write_camera_setup_model (CameraSetupModelID, 'stereo_camera_setup.csm')
```

3.2 Estéreo Binocular

No HALCON, é possível recorrer a três métodos de estéreo binocular: estéreo baseado em correlação, estéreo *multigrid* (multi grelha) e ainda estéreo *multi-scanline.* Na aplicação de visão artificial desenvolvida para o projeto Aquatropolis, recorreu-se ao estéreo baseado em correlação, o método tradicional.

O método de estéreo binocular baseado em correlação recorre a técnicas de correlação para efetuar correspondência entre pontos de forma a determinar as disparidades e distâncias dos pontos observados nas imagens. Para determinar as disparidades e as distâncias por este método, no HALCON, recorre-se respetivamente às funções *binocular_disparity* e *binocular_distance.*

As vantagens do estéreo binocular baseado em correlação são:

- Rapidez;
- Invariante perante alterações de valores de cinzento;
- Facilidade de paralisação automática em sistemas com *multicore* e multiprocessador.

A maior desvantagem deste método é o facto de apenas ser funcional para objetos que contenham alguma textura. Caso o objeto em análise tenha poucas áreas com textura, não é possível obter os resultados esperados.

3.2.1 Aquisição de Imagens Estéreo

Ao utilizar um sistema de câmaras para efetuar aquisição de imagens através de métodos de estéreo binocular, é necessário tomar algumas precauções de modo a garantir que os resultados sejam fidedignos e válidos. No manual do HALCON, são referidos alguns dos cuidados a ter ao utilizar um sistema estéreo para aquisição de imagem: [39]

- É muito importante garantir que a montagem e configuração de câmaras se mantém inalterada desde o momento da calibração, caso contrário poderá nem ser possível obter quaisquer resultados com o sistema estéreo;
- É necessário garantir uma boa iluminação do campo de visão das câmaras, evitando reflexões nos objetos em análise.

3.2.1.1 Textura

As coordenadas tridimensionais de cada ponto de um objeto em análise são determinadas através de um processo de correspondência. Esta correspondência é baseada na textura existente nos objetos em análise presentes no par de imagens. Assim, existindo pouca ou nenhuma textura, não é possível determinar os pontos conjugados. Este princípio aplica-se sobretudo no método de estéreo binocular baseado em correlação. Caso o objeto em análise tenha pouca ou nenhuma textura, deve projetar-se um padrão de textura neste, por exemplo, um padrão de linhas projetado por um laser.

Na Figura 32, é demonstrado um par de imagens estéreo retificadas e o resultado da correspondência através do método baseado em correlação estéreo numa área com pouca textura. As regiões onde o processo de correspondência não funciona são exibidas a branco.

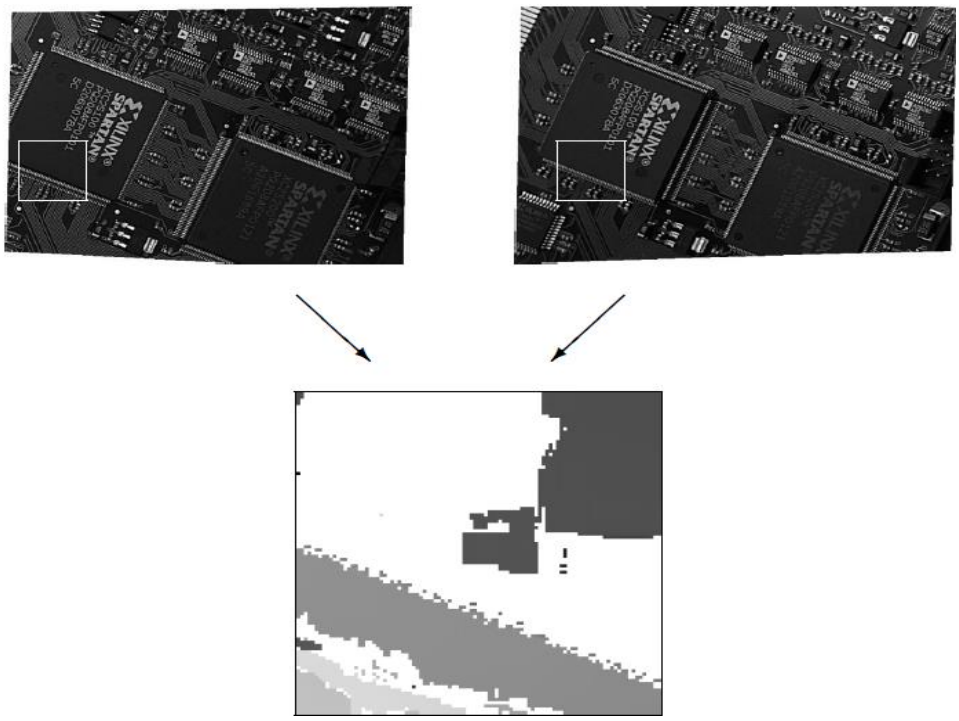


Figura 32 - Imagens estéreo retificadas e resultado da correspondência baseada em correlação estéreo [39]

3.2.1.2 Padrões Repetitivos

O processo de correspondência entre pontos conjugados pode ser afetado negativamente caso existam padrões repetitivos no objeto, devido às semelhanças entre os pontos do objeto. Para agilizar a correspondência entre pontos e torná-la mais eficaz, as imagens estéreo são retificadas de tal forma que os pares de pontos conjugados tenham coordenadas de linha idênticas nas imagens retificadas. Assim, o conjugado de um ponto da primeira imagem é procurado na segunda imagem numa área correspondente a uma linha horizontal. É, assim, aconselhado colocar o objeto (ou o padrão de textura projetado) de modo a que o padrão de textura não esteja alinhado com as linhas das imagens retificadas.

3.2.2 Retificação de Imagens Estéreo

O processo de retificação de imagens obtidas por um par estéreo de câmaras é um ajuste das imagens com base nos parâmetros internos das câmaras e o seu posicionamento relativo, efetuado de tal modo que os pares de pontos conjugados fiquem no mesmo alinhamento horizontal nas duas imagens retificadas. Considera-se que a primeira imagem do par provém da imagem à esquerda e que a segunda imagem provém da imagem à direita.

De seguida, é demonstrado o resultado da retificação de um par de imagens estéreo. Na Figura 33, é apresentado um par de imagens estéreo, num sistema onde as câmaras apresentam um elevado ângulo de rotação entre si. Na Figura 34, é demonstrado o par de imagens retificadas correspondente à retificação do par de imagens da Figura 33.

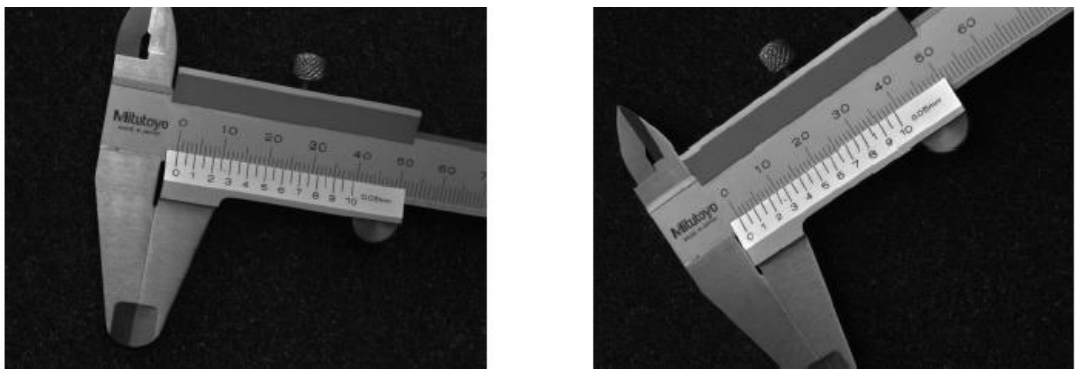


Figura 33 - Imagens estéreo originais [39]

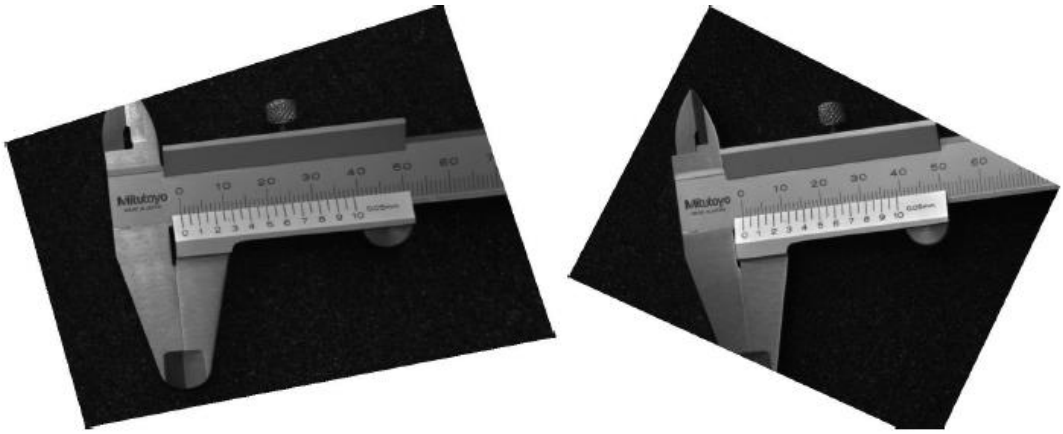


Figura 34 - Imagens estéreo retificadas [39]

No que diz respeito à biblioteca HALCON, para proceder à retificação de imagens estéreo são utilizados os operadores *gen_binocular_retification_map* e *map_image*.

O operador *gen_binocular_retification_map* permitir obter o mapa de retificação das imagens. As variáveis de entrada desta função são os parâmetros internos das duas câmaras, respetivamente *CamParam1* e *CamParam2*, e o posicionamento relativo entre câmaras, *RelPose*, que define a transformação do sistema de coordenadas da primeira câmara para o sistema de coordenadas da segunda câmara. Os restantes parâmetros de entradas são o *SubSampling*, o *Method* e *MapType*.

gen_binocular_rectification_map (Map1, Map2, CamParam1, CamParam2, RelPose, SubSampling, Method, MapType. CamParamRect1, CamParamRect2, CamPoseRect1, CamPoseRect2, RelPoseRect)

O parâmetro *SubSampling* tem como finalidade modificar a resolução e o tamanho das imagens retificadas, quando comparado com as imagens originais. Caso se introduza o fator de *SubSampling* "1", as imagens retificadas irão ter o mesmo tamanho e resolução que as imagens originais, um número superior a 1 resultará numa imagem inferior e um número inferior a 1 resultará numa imagem de tamanho superior. Ao utilizar a redução do tamanho de imagem é possível acelerar o processo de correspondência estéreo, no entanto o resultado obtido terá menos detalhes. No manual do HALCON [39] é aconselhado não usar valores *SubSampling* abaixo de 0,5 nem acima de 2. A utilização de valores não aconselhados poderá promover efeitos indesejados de suavização ou *aliasing*, que dificultarão a correspondência.

Outro dos parâmetros de entrada é o valor *Method*. No referido manual, o processo de retificação é descrito como uma projeção das imagens originais num plano de imagem comum retificado. Para definir esse plano, o HALCON permite a utilização de dois métodos, que são os valores a inserir no lugar do parâmetro *Method*.

No método "*geometric*" a orientação de um plano de imagem comum retificado é definida pelo produto vetorial entre a linha de base das câmaras e a linha de interseção dos dois planos das imagens originais.

Por defeito, é utilizado o método *"viewing_direction"* que assume como eixo *x* do plano de imagem a linha base das câmaras. Neste caso, o plano *xz* retificado é formado pelos eixos *z* das câmaras e pela sua linha base. O eixo *z* retificado resultante é a orientação do plano de imagem comum, assim irá ficar localizado no referido plano e será ortogonal à linha de base, o eixo *x*.

Quando abordadas as câmaras *area scan*, baseadas na teoria da câmara *pinehole*, pode considerar-se que as imagens retificadas são adquiridas por um sistema virtual retificado de câmaras estéreo, simplesmente denominado sistema retificado de câmaras estéreo, demonstrado na Figura 35. O centro ótico de cada uma das câmaras retificadas é o mesmo que é considerado nas câmaras reais. No entanto, neste novo sistema virtual, as câmaras sofrem uma rotação de forma a que os campos de visão de ambas fiquem paralelos e que o eixo *x* de cada uma delas fique alinhado, considerando que as duas câmaras têm a mesma distância focal. Assim, os planos geométricos de cada uma das imagens obtidas pelo par de câmaras irão ser coplanares. A origem do sistema de coordenadas do plano destas imagens retificadas poderá ficar colocado fora da imagem.

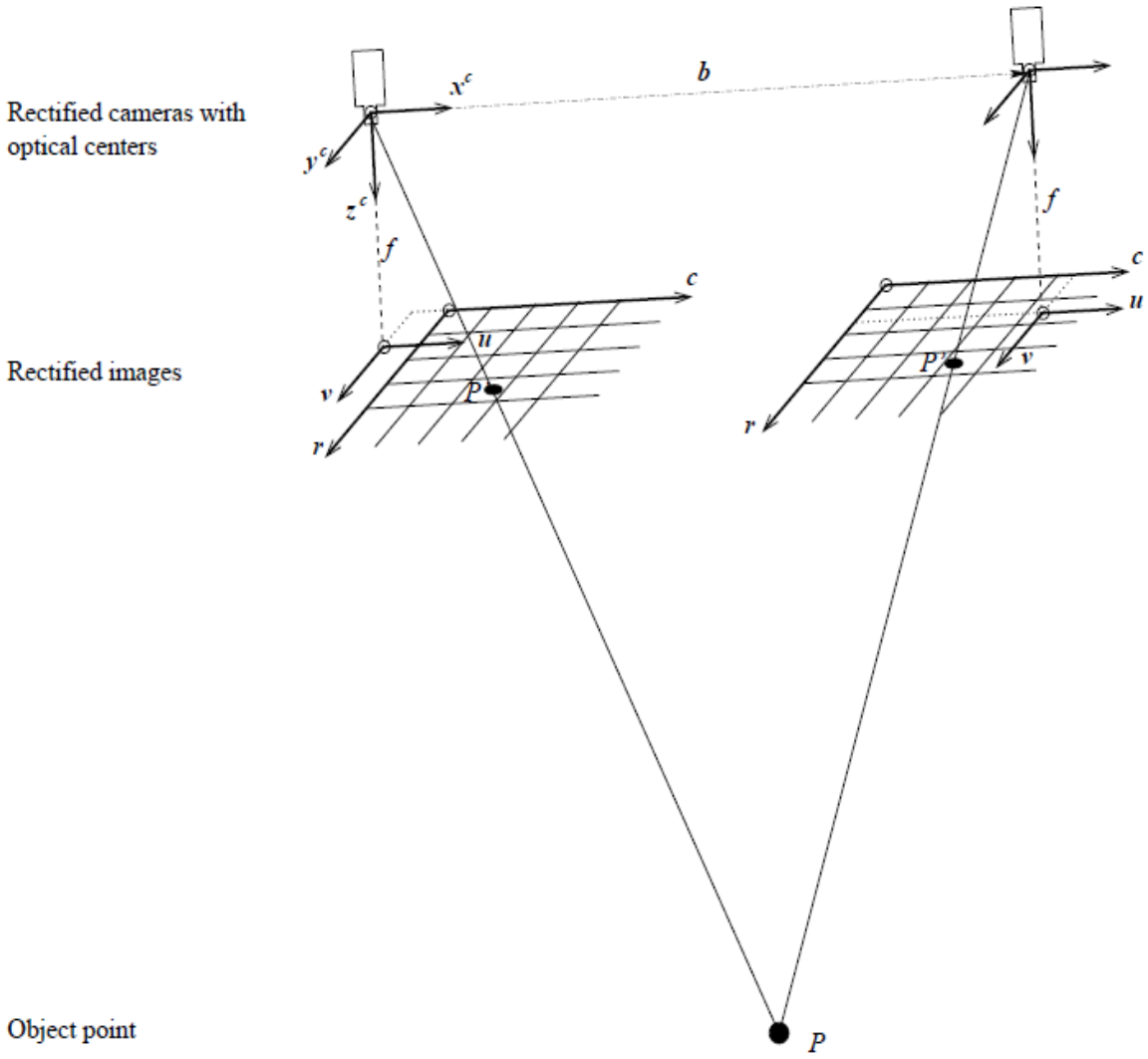


Figura 35 - Sistema retificado de câmaras estéreo [39]

Os mapas de retificação das imagens obtidos pelas duas câmaras são devolvidos nos parâmetros *Map1* e *Map2*, respetivos à primeira e segunda câmara. Para especificar o tipo de mapeamento colocado nestas variáveis, é utilizado o parâmetro de entrada *MapType*. Podem ser escolhidos três métodos de mapeamento: *"nearest_neighbor"*, "*bilinear*" e o "*coord_map_sub_pix*".

O método de interpolação utilizado por defeito é o "*bilinear*", em que cada um dos dois mapas resultantes consistem numa imagem com cinco canais. No primeiro canal de cada pixel da imagem resultante, as coordenadas linearizadas do pixel na imagem original são armazenadas na posição superior esquerda em relação às coordenadas transformadas. Os outros quatro canais contêm os pesos dos quatro pixéis vizinhos das coordenadas transformadas, que são usadas para a interpolação bilinear, e são colocados pela ordem

demonstrada na Figura 36. O segundo canal, por exemplo, contém os pesos dos pixéis que se situam acima das coordenadas após transformação.

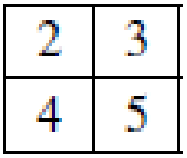


Figura 36 - Ordem dos quatro canais que contêm os pesos dos quatro pixéis vizinhos das coordenadas transformadas [48]

Os parâmetros internos da câmara, após a retificação, são devolvidos nos parâmetros *CamParamRect1*, relativamente à primeira câmara e *CamParamRect2*, relativamente à segunda câmara. A rotação e a translação da câmara retificada em relação à câmara original são especificadas por *CamPoseRect1* e *CamPoseRect2*, alterações relativas à primeira e à segunda câmara, respetivamente. A translação apenas se aplica caso se utilize uma câmara com lente telecêntrica.

O último parâmetro de saída é *RelPoseRect*, que devolve a posição relativa do sistema de coordenadas da segunda câmara após retificação (*ccsR2*) em relação ao sistema de coordenadas da primeira câmara após retificação (*ccsR1*). Considera-se assim que a variável é devolvida no formato ${}^{ccsR1}P_{ccsR2}$.

Uma vez aplicado o operador *gen_binocular_retification_map*, que permite gerar o mapa de retificação de cada imagem do par estéreo, é aplicado o operador *map_image* a ambas as imagens. Tal como demonstra o cabeçalho do operador enunciado abaixo, este operador recebe como parâmetro de entrada a imagem original "*Image*" e o respetivo mapa de retificação "*Map*", e devolve a imagem já retificada "*ImageMapped"*.

```
map_image (Image, Map, ImageMapped)
```

3.2.2.1 Determinação de Disparidades Através de Correspondência Estéreo Baseada em Correlação

Através das disparidades, é possível aferir a distância *z* de pontos de um objeto ao sistema de câmaras estéreo. Dois pontos com igual disparidade estarão equidistantes do sistema estéreo. Caso apenas seja necessário obter as alturas de objetos, distinguindo as alturas localmente, é suficiente aferir as suas disparidades. No HALCON, executa-se a correspondência estéreo baseada em correlação recorrendo ao operador *binocular_disparity*.

binocular_disparity (ImageRect1, ImageRect2, Disparity, Score, Method, MaskWidth, MaskHeight, TextureThresh, MinDisparity, MaxDisparity, NumLevels, ScoreThresh, Filter, SubDisparity)

Os dois primeiros parâmetros de entrada são as duas imagens do sistema estéreo após retificação, *ImageRect1* e *ImageRect2*. As disparidades são aferidas apenas para os pontos conjugados que estão representados no domínio de ambas as imagens. Tomando este princípio, é possível acelerar o processamento das disparidades das imagens, diminuindo para uma região de interesse menor o domínio de uma das imagens retificadas.

O parâmetro *Method* permite definir qual a função de correspondência a utilizar neste processo. Os métodos "*sad*" (*summed absolute diferences* - diferenças absolutas somadas) e "*ssd*" (*summed squared diferences* - diferenças quadráticas somadas) executam a comparação entre os valores de cinzento dos pixéis dentro da área onde é efetuada a correspondência. O método "*ncc*" (*normalized cross correlation* - correlação cruzada normalizada) compensa o valor médio de cinzento e variação deste valor dentro da área onde é efetuada a correspondência, assim é preferível utilizar este método caso as duas imagens tenham alguma diferença de contraste e brilho entre si. No entanto, este método, "*ncc*", é mais lento do que os métodos "*sad*" e "*ssd*", pois estes dois executam um processamento mais simplificado.

Para definir os valores de largura e altura da janela ou área dentro da qual é efetuada a correspondência entre pontos, são utilizados os parâmetros *MaskWidth* e *MaskHeight*. Se for utilizada uma área muito grande, tendencialmente, ocorrem perdas de detalhes das

imagens originais retificadas, enquanto que, ao utilizar áreas pequenas, as imagens de disparidade resultantes irão incluir mais ruído, mas compreenderão mais detalhes.

As áreas com pouca textura são excluídas deste processo, não sendo determinada qualquer disparidade. Para definir a variação mínima de textura dentro da área onde é executada a correspondência, é utilizado o parâmetro *TextureThresh*.

De modo a definir os valores mínimo e máximo da disparidade, utilizam-se respetivamente os parâmetros *MinDisparity* e *MaxDisparity*. A utilização destes parâmetros permite definir uma restrição na área de pesquisa do processo de correspondência. Caso se selecione um intervalo de disparidades que não contenha o intervalo real de disparidades do objeto em análise nas imagens, não será possível aferir os pontos conjugados corretamente. Por outro lado, se o intervalo de disparidade especificado for muito grande, o processo de correspondência será lento e aumenta a probabilidade de falsas correspondências. Assim, é importante definir corretamente os valores dos parâmetros *MinDisparity* e *MaxDisparity*.

No operador *binocular_disparity*, o parâmetro *NumLevels* indica o número de níveis de pirâmides de imagem utilizados no processo de determinação das disparidades. A utilização de níveis de pirâmides de imagem tem como finalidade melhorar o tempo de processamento deste operador. O intervalo de disparidade especificado pelos parâmetros *MinDisparity* e *MaxDisparity* é usado apenas no nível mais alto da pirâmide, indicado pelo parâmetro *NumLevels*. Com base nos resultados da correspondência nesse nível, o intervalo de disparidade é adaptado automaticamente para efetuar a correspondência nos seguintes níveis inferiores da pirâmide. Em termos de aceleração do processo, será ainda mais vantajoso caso os objetos em análise contenham regiões diferentes, entre as quais o intervalo de disparidade apropriado varia fortemente. Ao utilizar um valor de *NumLevels* muito elevado, o processo de correspondência poderá falhar devido à falta de textura no nível superior da pirâmide. Na Figura 37, é apresentado um esquema explicativo do funcionamento dos níveis de pirâmide.

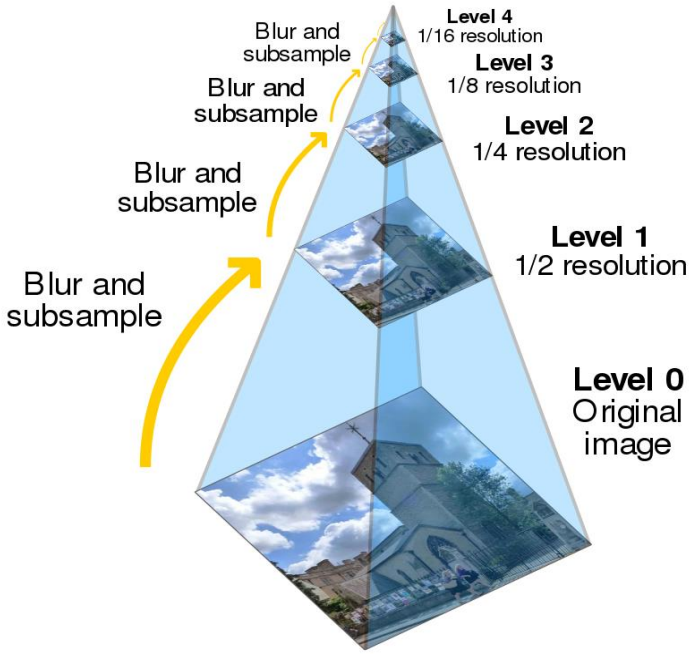


Figura 37 - Esquema representativo dos níveis de pirâmides de imagem [49]

O parâmetro *ScoreTresh* define quais os índices de correspondência adequados. Os pontos que não têm índices de correspondência adequados são excluídos dos resultados, não sendo incluídos no domínio da imagem de disparidade obtida. Este parâmetro, *ScoreTresh*, é definido com base na função de correspondência definida no parâmetro *Method*. Com os dois métodos "*sad* " (0 a 255) e "*ssd* " (0 a 65025) obtêm-se índices de correspondência mais baixos, para correspondências melhores. Por outro lado, o método "*ncc*" (-1 a 1) devolve valores mais altos para correspondências melhores, onde uma pontuação de 0 indica que as áreas selecionadas para correspondência nas duas imagens retificadas são totalmente diferentes e uma pontuação de -1 denota que a segunda área de correspondência é o inverso da primeira área de correspondência.

O parâmetro *Filter* é um parâmetro opcional, que pode ser utilizado para ativar um filtro *downstream*, que permite aumentar a fiabilidade dos resultados da imagem de disparidade. Neste parâmetro, o HALCON permite a utilização do método "*left_right_check*", que permite que seja executada uma verificação dos resultados da correspondência, através de um processo de correspondência semelhante ao descrito anteriormente, mas no sentido inverso, ou seja, os pontos da imagem da esquerda são procurados na imagem da direita. Através deste método, só são aceites os pontos conjugados em que se obtiver a mesma correspondência que no primeiro processo. Esta verificação torna a utilização deste operador bastante rigorosa. Assim, em algumas

situações podem ocorrer algumas perdas de resultados de disparidade, mesmo em áreas com algum nível de textura. Pode optar-se por não utilizar este filtro, colocando o valor "*none*" como argumento do parâmetro *Filter*.

É possível definir uma afinação ao *subpixel* das disparidades. Este processo é efetuado colocando o argumento "*interpolation*" no parâmetro *SubDisparity*. Este parâmetro é opcional e, tal como no parâmetro anterior, caso não se queira recorrer a esta afinação das disparidades, coloca-se o argumento "*none*" no parâmetro.

Os resultados do operador *binocular_disparity* são as imagens *Disparity* e *Score*. A primeira imagem contém a representação das disparidades e a segunda imagem contém as representações dos *Scores*, ou seja, os índices de correspondência para obter as disparidades. Na Figura 38, é apresentado um par de imagens retificadas e de seguida, na Figura 39, são apresentadas a imagem *Disparity* e a imagem *Score*, obtidas a partir do par de imagens retificadas.

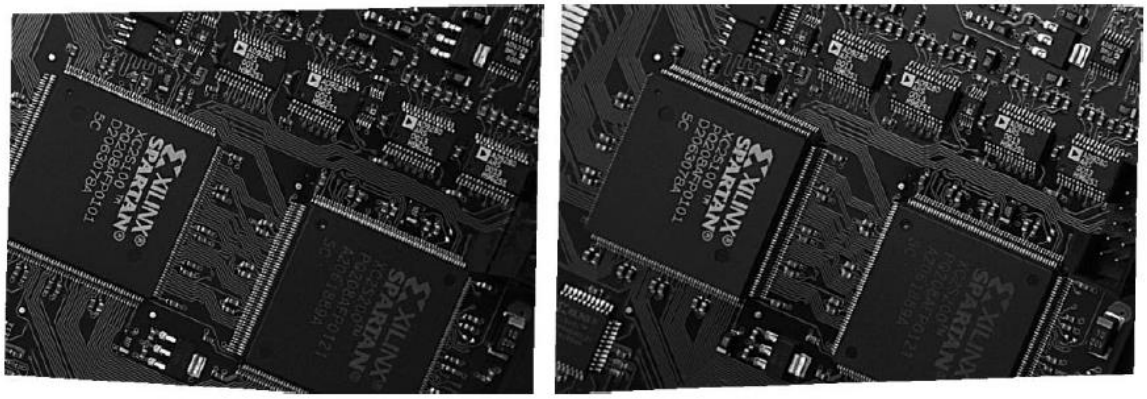


Figura 38 - Imagens estéreo retificadas [39]

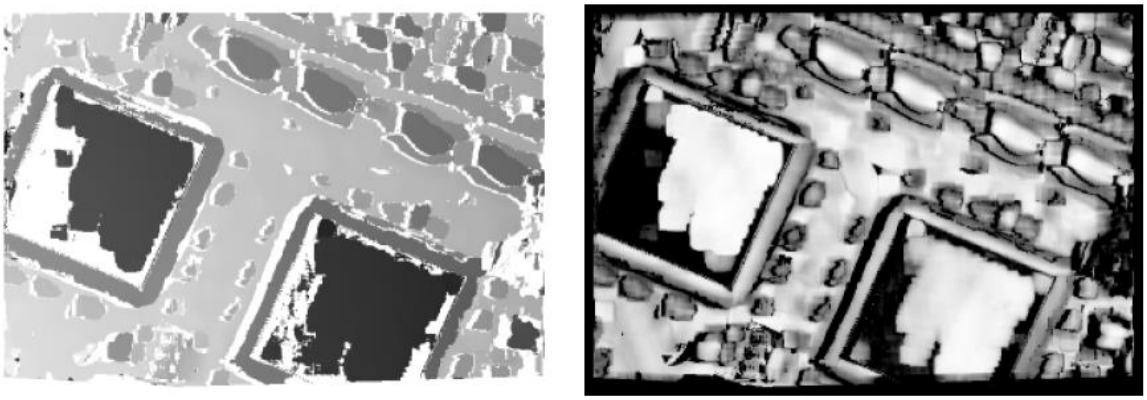


Figura 39 - Imagem de disparidade (à esquerda), imagem dos índices de correspondência (à direita) [39]

As imagens resultantes, *Disparity* e *Score*, têm por base a geometria da primeira imagem retificada. Assim, a disparidade para um ponto (linha, coluna) na primeira imagem retificada é o valor cinza na posição (linha, coluna) da imagem de disparidade. Nas imagens *Disparity* e *Score* apresentadas na Figura 39, as áreas em que a correspondência não foi bem-sucedida, ou seja, as regiões indefinidas das imagens são exibidas em branco na imagem de disparidade e preto na imagem de pontuação.

3.2.2.2 Transformação de Imagens de Disparidade para Coordenadas 3D

Para determinar coordenadas (*x*, *y* e *z*) de pontos específicos ou de um conjunto de pontos, a partir de imagens de disparidade, o HALCON permite a utilização de vários operadores.

Um dos exemplos é o operador *disparity_to_point_3D*. Este operador permite aferir as coordenadas 3D de pontos especificados pelo utilizador a partir da imagem de disparidade. Os dois primeiros parâmetros de entrada do operador são os parâmetros internos retificados das câmaras, *CamParamRect1* e *CamParamRect2*. O terceiro parâmetro de entrada é o posicionamento relativo retificado entre câmaras, *RelPoseRect*. O quarto e quinto parâmetro são as coordenadas, linha e coluna (relativas à imagem da primeira câmara), do ponto ou pontos do(s) qual/quais se pretendem obter as coordenadas 3D correspondentes, *Row1* e *Col1*. O último parâmetro de entrada é a variável com os valores de disparidade dos pontos especificados, *Disparity*. Os parâmetros de saída são três tuplas com as coordenadas 3D dos pontos especificados, *X*, *Y* e *Z*.

```
disparity_to_point_3d (CamParamRect1, CamParamRect2, RelPoseRect, Row1, Col1, Disparity, X, Y, Z)
```

Um dos outros operadores que permite obter as coordenadas 3D a partir das imagens de disparidade é o *disparity_image_to_xyz*. Este operador é semelhante ao anterior, no entanto este é utilizado para determinar as coordenadas 3D de toda a imagem de disparidade.

```
disparity_image_to_xyz (Disparity, X, Y, Z, CamParamRect1, CamParamRect2, RelPoseRect)
```

Os parâmetros de entrada deste operador são a imagem de disparidade, *Disparity*, os parâmetros internos retificados das câmaras, *CamParamRect1* e *CamParamRect2*, e o posicionamento relativo retificado entre câmaras, *RelPoseRect*. Ao processar os valores de entrada, o operador permite obter os parâmetros de saída, que são três imagens, *X*, *Y* e *Z*, com a representação das coordenadas em valores de cinzento, com base no referencial de coordenadas associado à primeira câmara após retificação.

4 Estado da Arte em Técnicas de Triangulação Laser

Tal como descrito no início desta dissertação, um dos projetos que o aluno integrou foi o projeto Vestas. Neste projeto, pretendia desenvolver-se uma ferramenta para a empresa Vestas, que permitisse aferir o desgaste dos dentes das engrenagens (*yaw ring*) de rotação da *nacell* em relação à torre eólica, através de um sistema de visão por computador. Os pormenores relativos ao desenvolvimento deste projeto irão ser detalhados mais adiante neste documento. De forma a introduzir alguns conteúdos teóricos que permitam facilitar a perceção das tarefas de desenvolvimento associadas a este projeto, neste capítulo é introduzido o conceito de Triangulação Laser com a técnica *Sheet of Light*. Serão explicados alguns conceitos teóricos, com base nas recomendações do manual do *software* utilizado para desenvolvimento de tecnologias de visão por computador no LINE.IPT, a biblioteca de *software* HALCON.

4.1 Princípios da Triangulação Laser

As tecnologias de medição através de extração de características 3D, baseadas em visão por computador, têm vindo a evoluir e estabeleceram-se como ferramentas essenciais em várias áreas de trabalho. Este tipo de tecnologia permite executar tarefas de medição de forma não evasiva, o que é bastante vantajoso em áreas em que é necessário manter alguns cuidados com objeto em análise.

Um dos métodos amplamente utilizados para executar medições e obter características de sólidos e superfícies é a triangulação laser com base em iluminação laser em forma de linha (*sheet of light*) em conjunto com uma câmara. Este tipo de sistema utiliza equipamentos e *software* fáceis de configurar, e permite obter medições com elevada precisão. [50] Este tipo de sistema é útil em aplicações na área da medicina, inspeção e verificação de objetos, criação de modelos CAD de peças de arte ou componentes mecânicos, entre outras aplicações. [51]–[54] É vantajoso utilizar iluminação laser neste tipo de triangulação devido a algumas particularidades físicas deste tipo de luz: o laser quase não sofre divergência do foco luminoso, e é monocromático. Isto permite utilizar este tipo de iluminação em áreas pequenas, possibilitando deste modo, obter medidas com elevada resolução. De forma a melhorar o contraste entre áreas iluminadas e não

iluminadas, pode utilizar-se um filtro passa banda com base no comprimento de banda do laser.[51] [55]

4.2 Triangulação Laser "*Sheet of Light*" com o HALCON

A triangulação laser é um método de medição 3D através de câmaras (medição ótica), no qual o objeto em análise é iluminado recorrendo a uma luz laser estruturada. A iluminação laser incide sobre o objeto a partir de uma determinada direção e a câmara obtém a imagem do objeto nestas condições a partir de um ângulo e posição diferentes do laser, tal como demonstrado na Figura 40**Erro! A origem da referência não foi encontrada.**. O ângulo de triangulação, α, corresponde ao ângulo formado entre o eixo ótico da câmara e o plano do laser. A partir desta relação geométrica é possível definir relações matemática que permitem calcular as coordenadas 3D do objeto observado pela câmara. [39], [54] Ao incidir num objeto, a perceção visual da linha laser será afetada pelo perfil de altura do objeto, ou seja, a linha irá acompanhar este perfil. Obtendo imagens do dito perfil, é possível aferir as variações de altura do objeto. Para executar a reconstrução deste sólido, movimenta-se o objeto em relação ao sistema de medição, ou vice-versa, de forma a que a linha laser seja fotografada em toda a sua superfície, obtendo vários perfis da linha laser.

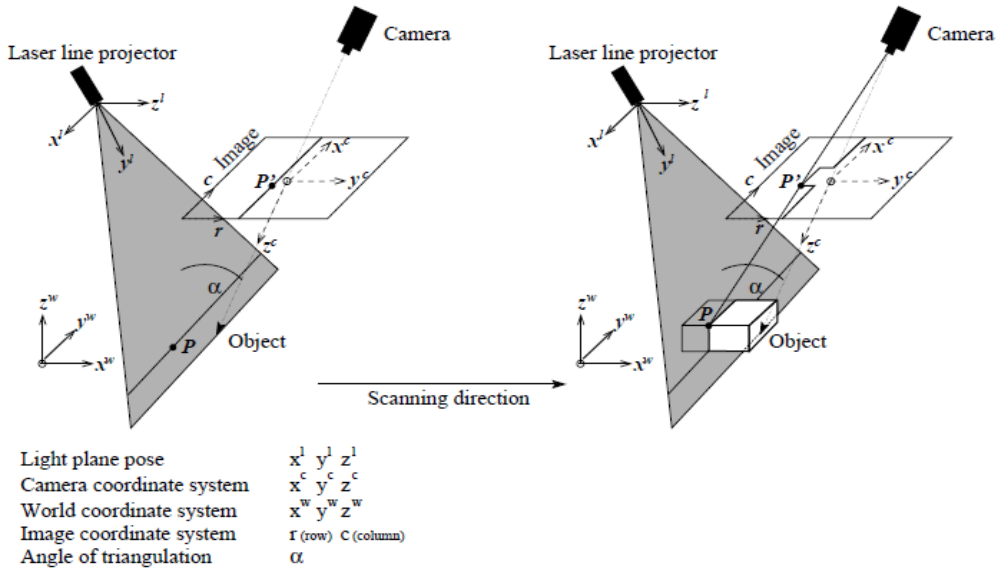


Figura 40 - Esquema exemplificativo do princípio da triangulação laser [39]

Seguindo a abordagem introduzida no sexto capítulo do manual do HALCON "*Solution Guide III C 3D Vision*" [39]: Para executar medições em unidades métricas recorrendo ao sistema de triangulação laser, é necessário proceder a uma calibração do sistema, semelhante em alguns aspetos à calibração de um sistema estéreo, referida anteriormente nesta dissertação. No entanto, é possível utilizar este sistema sem calibração para obter apenas as disparidades e um índice de fiabilidade do resultado obtido.

Ao introduzir informações de calibração, no processo de triangulação, é possível obter as disparidades, as coordenadas *x*, *y* e z deste perfil e um modelo 3D do objeto analisado. O resultado das disparidades é guardado numa imagem em que, cada linha corresponde à disparidade do perfil laser medido. A construção da imagem de disparidades é exemplificada na Figura 41. Para construir este tipo de imagem é necessário garantir que a câmara esteja alinhada de tal forma que os perfis fiquem paralelos às linhas da imagem. No caso dos resultados obtidos para as coordenadas, obter-se-á uma imagem para cada um dos eixos, *x, y* e *z*, com a representação das coordenadas em valores de pixel. O modelo 3D obtido irá conter informação tridimensional do objeto e o respetivo mapeamento bidimensional.

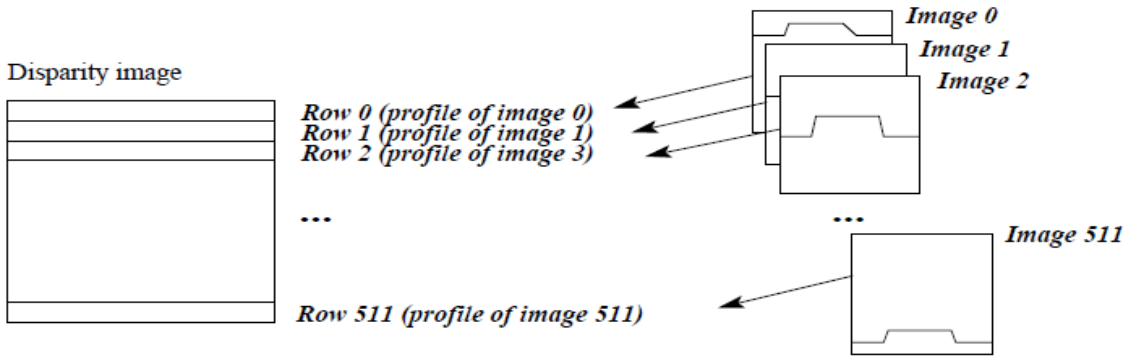


Figura 41 - Formação da imagem de disparidades a partir das imagens do perfil do laser [39]

É importante esclarecer que a noção de imagem de disparidade na triangulação laser através de *sheet of light* é diferente do conceito de imagem de disparidade apresentado anteriormente no capítulo relativo aos sistemas de câmaras estéreo. No estéreo, a imagem de disparidade refere-se à imagem onde são representadas as diferenças das coordenadas de determinado ponto na imagem esquerda e respetiva representação na imagem direita. No caso da triangulação laser através de *sheet of light*, a imagem de disparidade é criada linha a linha com a disparidade obtida em cada perfil do laser fotografado.

4.2.1 Considerações para Configurar um Sistema de Triangulação Laser

Tal como descrito anteriormente, para executar medições e aferir características 3D de um objeto, é necessário que objeto seja movimentado em relação ao sistema. Neste tipo de aplicações, é comum utilizar uma passadeira rolante para movimentar o objeto. No entanto, como este sistema trabalha com base em calibrações, não podem ser alteradas as distâncias e ângulos entre câmara e iluminação, nem entre este sistema e o plano de medida, caso contrário a calibração será inválida.

No manual do HALCON são sugeridas três disposições diferentes de câmara e laser em relação ao objeto a medir. No primeiro caso, utiliza-se a câmara ortogonalmente relativamente ao objeto e o laser inclinado. No segundo exemplo, utiliza-se o sistema ao contrário, ficando o laser ortogonal em relação ao objeto e câmara numa posição inclinada. No terceiro caso, utiliza-se tanto câmara como o laser inclinado em relação ao objeto. Na Figura 42, os esquemas a), b) e c) correspondem respetivamente ao primeiro, segundo e terceiro caso.

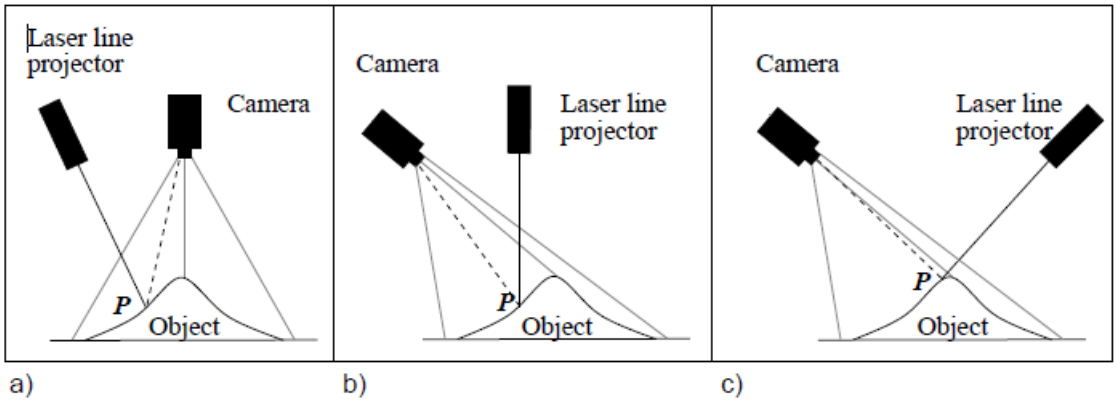


Figura 42 - Diferentes disposições de câmara e laser em relação ao objeto a medir, no manual [39]

A disposição entre elementos do sistema é escolhida com base na medição pretendida e na geometria do objeto. Na Figura 43, são demonstradas as imagens obtidas ao alterar a disposição dos elementos deste sistema. No lado esquerdo da imagem, é demonstrada uma imagem obtida com um sistema disposto conforme o primeiro caso (tal como Figura 43 a)). Neste caso, se o objeto for um paralelepípedo ou um cubo, a sua representação na imagem será um retângulo. Nos outros casos, segundo e terceiro caso, um objeto com

esta fisionomia terá uma representação trapezoidal, tal como demonstrado no esquema mais à direita da Figura 43.

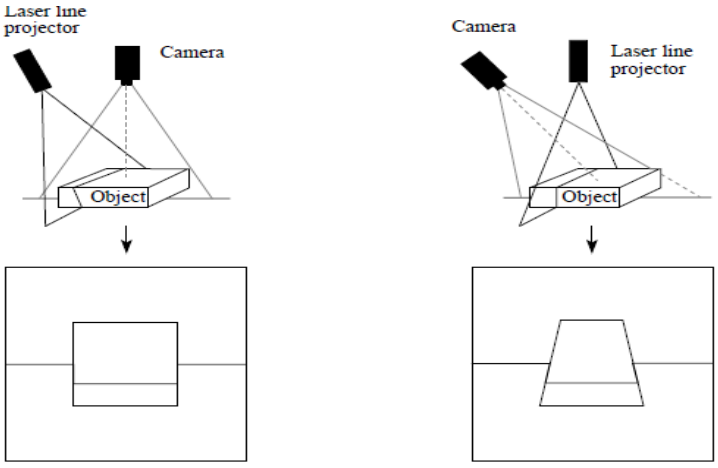


Figura 43 - Representação de um objeto na imagem obtida mediante a alteração da disposição da câmara e do laser [39]

A geometria do objeto a medir tem uma grande importância na definição do posicionamento da câmara e do laser de um sistema de triangulação *sheet of light*. Esta disposição deve ser efetuada de modo a minimizar sombras e oclusões na imagem. Quando determinado ponto é iluminado pelo laser, mas não é visível na imagem, dá-se o nome de oclusão. Este efeito é demonstrado pelo esquema apresentado no topo da Figura 44. Ocorre o efeito de sombra quando uma zona de um objeto é visível na imagem, mas não é iluminada pelo laser, devido à interferência de outras partes do objeto que ficam entre a projeção do laser e o ponto do objeto da imagem.

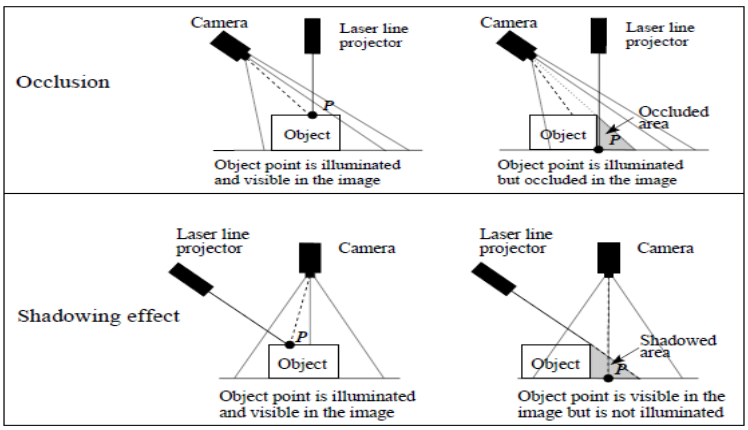


Figura 44 - Problemas na configuração do sistema: oclusão (em cima) e sombra ou efeito de sombreamento (em baixo) [39]

Uma das condições para obter uma boa precisão de medição, com um sistema de triangulação laser, é garantir que o ângulo entre o plano de luz e o eixo ótico da câmara (o ângulo de triangulação α) é superior a 30º e inferior a 60º. Com um ângulo inferior a 30º, a precisão irá diminuir. Já com um ângulo superior a 60º poderá obter-se maior precisão, mas haverá uma maior possibilidade de ocorrerem oclusões e efeitos de sombra. Deve ter-se o cuidado de focar a iluminação laser, de forma a que esteja nitidamente visível nas imagens obtidas, porque a precisão diminui quando o plano de luz está desfocado. É possível que tal suceda quando o ângulo de triangulação for muito pequeno, ou seja, o ângulo entre o plano de luz e o plano de foco ou campo de visão for muito grande e, portanto, apenas uma pequena parte próxima à interseção dos dois planos estará em foco.

4.2.2 Calibrar um Sistema *Sheet of Light*

O HALCON permite executar a calibração do sistema de triangulação laser através do *Sheet of light*, recorrendo a dois métodos, a calibração recorrendo a uma placa de calibração, semelhante à calibração abordada no capítulo relativo aos sistemas estéreo, ou a calibração recorrendo a um objeto específico 3D, um objeto com dimensões previamente conhecidas.

A calibração deste sistema recorrendo a um objeto 3D previamente conhecido é efetuada obtendo a imagem de disparidades do objeto que permite executar a reconstrução tridimensional (imagem obtida sem qualquer calibração prévia). O objeto é criado através de um modelo CAD e o ficheiro desse modelo servirá como ficheiro de descrição da calibração. Este tipo de calibração é mais simples do que a calibração com recurso a placa, no entanto tem um grau de precisão menor. Assim, devido à maior precisão da calibração com a placa de calibração, na aplicação desenvolvida utilizou-se este método.

Para calibrar este sistema com a placa de calibração fornecida pelo fabricante do HALCON, inicia-se o processo executando uma calibração da câmara utilizada, igual à calibração descrita anteriormente para as câmaras em estéreo (neste caso apenas para uma câmara). Após efetuar tal tarefa, é necessário executar a calibração do laser ou do plano de luz, de modo a conseguir definir referências deste sistema em relação às dimensões reais da envolvente do sistema. Por último, é efetuada a calibração relacionada com o movimento do objeto. Para executar estas duas últimas tarefas de calibração é igualmente necessário obter imagens da citada placa de calibração.

O processo de calibração da câmara, tal como descrito anteriormente, permite aferir os parâmetros internos da câmara e definir a relação entre o sistema de coordenadas da câmara e o sistema de coordenadas do mundo real (WCS – *World Coordinate System*).

A definição do plano de luz e do respetivo posicionamento é efetuada pelo menos por três pontos, dois dos pontos contidos no plano Z=0 do WCS, e um terceiro ponto com valor de Z superior. No entanto, quanto mais pontos forem definidos, maior será a precisão da calibração. Na Figura 45, os primeiros dois pontos em Z=0 correspondem a P1 e P2, o terceiro ponto com um valor de Z superior corresponde a P3. Para definir os pontos P1 e P2 é necessário dispor a placa de calibração de modo que fique no plano Z=0 do WCS, se for necessário a placa deve ser disposta em duas posições diferentes deste plano. Para

definir P3, coloca-se a placa numa posição mais alta ou numa posição inclinada, de modo a definir um plano com um valor de Z superior a zero. Para cada uma destas posições da placa de calibração, é necessário obter duas imagens, uma em que seja visível a placa de calibração e outra sem a placa de modo a destacar a linha do laser.

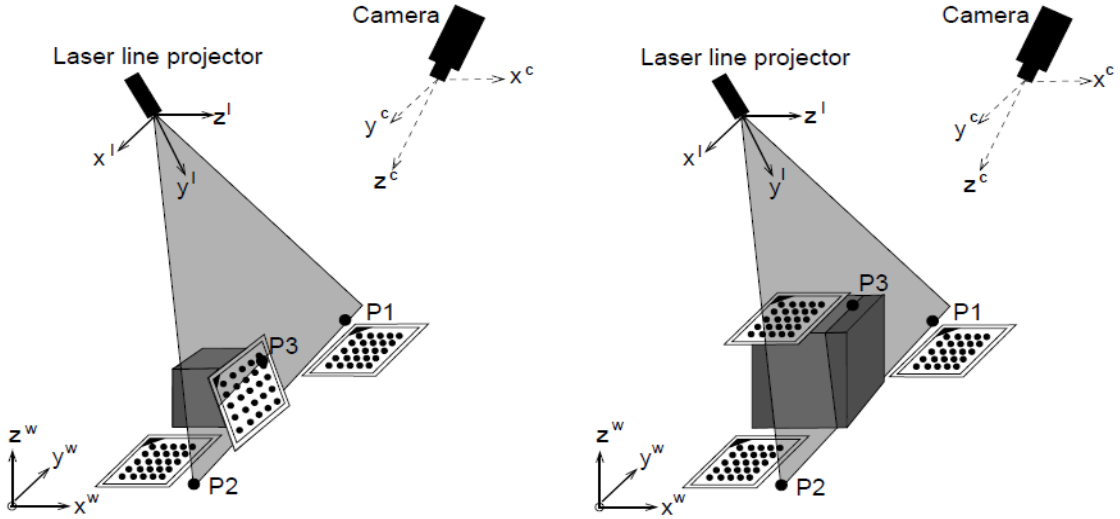


Figura 45 - Esquema exemplificativo de calibração do plano de luz [39]

De forma a obter boas imagens da placa de calibração e obter imagens com uma linha de laser bem definida, deve adaptar-se a iluminação ambiente de forma diferente e adequada a cada um dos cenários a fotografar. No manual do HALCON, é recomendado que o plano definido pelos pontos P1, P2 e P3 tenha a maior área possível, devendo assim fotografar-se a placa de calibração para definir o ponto P3, de tal forma que favoreça tal requisito. A placa deve ser fotografada numa posição tão alta quanto a dimensão em altura do objeto ou objetos que se pretendem medir com o sistema de triangulação laser, precavendo a diferença de alturas máxima que é possível existir em tal situação.

A última tarefa de calibração do sistema é a definição do movimento que o objeto irá realizar entre as imagens de obtenção do perfil de disparidade. Para efetuar a calibração associada ao movimento do objeto, utilizam-se duas imagens contendo a placa de calibração, alterando a posição da placa de uma imagem para a outra, de acordo com o movimento do objeto. A diferença entre imagens corresponde a um número conhecido de iterações do movimento do objeto.

.

4.2.2.1 Operadores HALCON Utilizados para Calibrar um Sistema *Sheet of Light*

A calibração das câmaras é igual à calibração descrita anteriormente no capítulo relativo ao estéreo. Para obter as informações necessárias para o restante processo de calibração do sistema de triangulação laser, nomeadamente, os parâmetros internos da câmara e a *pose* da câmara, recorre-se ao operador *get_calib_data*, também referido anteriormente no citado capítulo.

```
get_calib_data (CalibDataID, ItemType, ItemIdx, DataName, DataValue)
```

Para obter a informação desejada nesta situação, especifica-se no parâmetro *ItemType*, que se pretende obter informação relativa à câmara, para tal colocando o valor "*camera*". Visto ser apenas uma câmara, no parâmetro *ItemIdx* é colocado o valor "0". Para definir a extração dos parâmetros internos da câmara, coloca-se o valor "*param*" no parâmetro *DataName*. Esta informação é guardada na variável de saída colocada no parâmetro *DataValue.*

```
get_calib_data (CalibDataID, 'camera', 0, 'params', CameraParameters)
```

Na segunda fase da calibração, obtêm-se duas imagens com a placa de calibração em duas alturas (Z) diferentes. O posicionamento da placa (*pose*) numa das imagens é utilizado para definir o WCS e a *pose* definida pela placa na outra imagem é assumida como um sistema de coordenadas temporário (TCS – *Tempororary Coordinate System*). Relativamente à *pose* de cada imagem, é necessário adicionar a informação relativa à espessura da placa de calibração. Isto é realizado utilizando o operador *set_origin_pose*, para cada uma das imagens. Nesta função, introduz-se a *pose* que se pretende ver alterada, *PoseIn* e as alterações relativamente a cada um dos eixos coordenados (*DX, DY, DZ)*. O posicionamento resultante (*pose*) é colocado na variável de saída *PoseNewOrigin*.

```
set_origin_pose (PoseIn, DX, DY, DZ, PoseNewOrigin)
```

No excerto de código abaixo, é demonstrado um exemplo de calibração de um sistema de triangulação laser, retirado do manual do HALCON [39], que será útil para futuras demonstrações, onde se consideraram duas imagens com a placa de calibração que teriam os índices de *pose* 19 e 20.

Foi lida a *pose* inicial da placa presente em cada imagem, com o operado *get_calib_data*, e de seguida foi alterado este posicionamento acrescentando a informação da espessura da placa de calibração com o operador *set_origin_pose*, sendo que espessura da placa de calibração é previamente dada pela variável criada para o efeito *CalTabThickness*, que indica que a placa neste exemplo tem 63 mm.

```
CalTabThickness: = .00063
Index: = 19
get_calib_data (CalibDataID, 'calib_obj_pose', [0, Index], 'pose', CalTabPose)
set_origin_pose (CalTabPose, 0.0, 0.0, CalTabThickness, CameraPose)
Index: = 20
get_calib_data (CalibDataID, 'calib_obj_pose', [0, Index], 'pose', CalTabPose)
set_origin_pose (CalTabPose, 0.0, 0.0, CalTabThickness, TmpCameraPose)
```

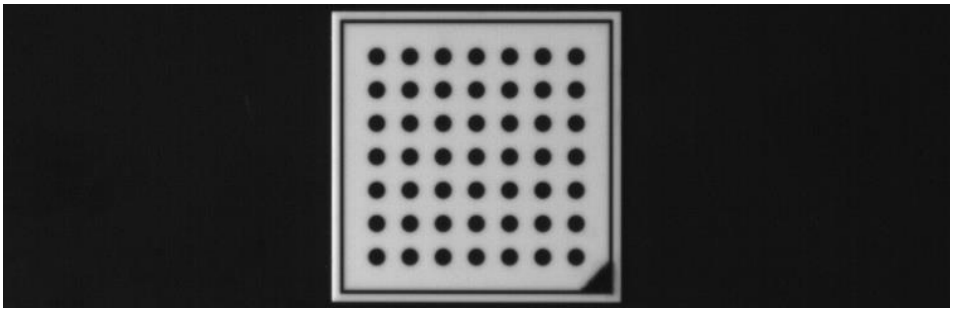


Figura 46 - Posição da placa de calibração no plano *Z*=0 no referencial WCS (índice 19 na sequência de calibração) [39]

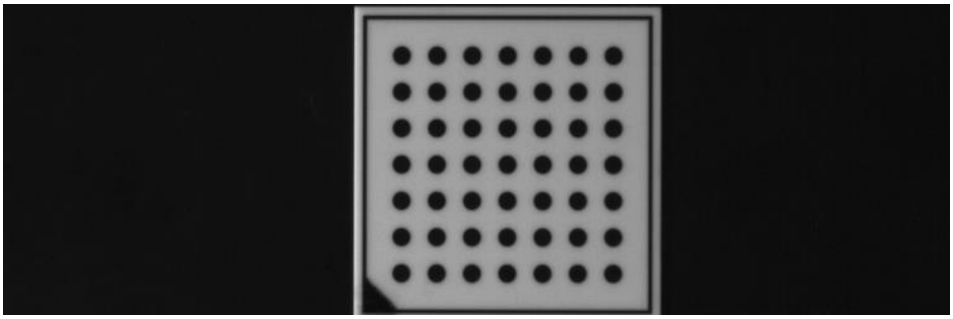


Figura 47 - Posição da placa de calibração no plano *Z*=0 no referencial TCS (índice 20 na sequência de calibração) [39]

Para cada uma destas imagens de calibração, é adquirida uma imagem correspondente, sem placa de calibração, onde a linha laser é projetada no mesmo plano definido pela placa de calibração. Para determinar as coordenadas 3D dos pontos da linha laser, utiliza-se a função *compute_3d_coordinates_of_light_line*. Com este procedimento, é obtido o plano da folha de luz, formado pelos pontos P1 e P2 contidos no plano *Z*=0 do referencial WCS e pelo ponto P3 contido no plano *Z*=0 do referencial TCS.

O excerto de código apresentado de seguida permite perceber quais os procedimentos a efetuar e quais os parâmetros de cada função utilizada. Em primeiro lugar, é lida a imagem com o perfil da linha laser com o operador *readimage* a partir de um ficheiro "*png*", que passa a imagem para uma variável de saída *ProfileImage,* nomeadamente *ProfileImage1* para o primeiro perfil ou *ProfileImage2*, no caso do segundo perfil. De seguida, é utilizada a função *compute_3d_coordinates_of_light_line*, onde o primeiro parâmetro de entrada é a imagem do perfil laser, *ProfileImage,* introduzindo uma das variáveis criadas na leitura da imagem, dependendo dos pontos a processar. O segundo parâmetro de entrada, *MinGray*, define o valor mínimo, em escala de cinzento, dos pixéis que contêm a linha laser, recebendo a variável *MinThreshold*. Os parâmetros seguintes a introduzir são os parâmetros internos da câmara, *CameraParameters*, o parâmetro *LocalCameraPose*, que diz respeito a um posicionamento local da câmara, como o TCS que será definido apenas para o segundo perfil laser fotografado. O último parâmetro de entrada é a *pose* da câmara, *CameraPose*. Os parâmetros de saída são as coordenadas, *X*, *Y* e *Z*, que são colocadas em tuplas, para o primeiro perfil foram denominadas *X19*, *Y19* e *Z19*, e para o segundo *X20*, *Y20* e *Z20*. Na definição dos pontos do primeiro perfil laser, um dos parâmetros de entrada é deixado em branco, o parâmetro *LocalCameraPose*. Este parâmetro só é definido quando é processada a imagem do segundo perfil laser, que diz respeito ao posicionamento do sistema de coordenadas TCS.

```
read_image (ProfileImage1, 'connection_rod_lightline_019.png')
compute_3d_coordinates_of_light_line (ProfileImage1, MinThreshold,
CameraParameters, [], CameraPose, X19, Y19, Z19)
read_image (ProfileImage2, 'connection_rod_lightline_020.png')
compute_3d_coordinates_of_light_line (ProfileImage2, MinThreshold, \
CameraParameters, TmpCameraPose, CameraPose, X20, Y20, Z20)
```

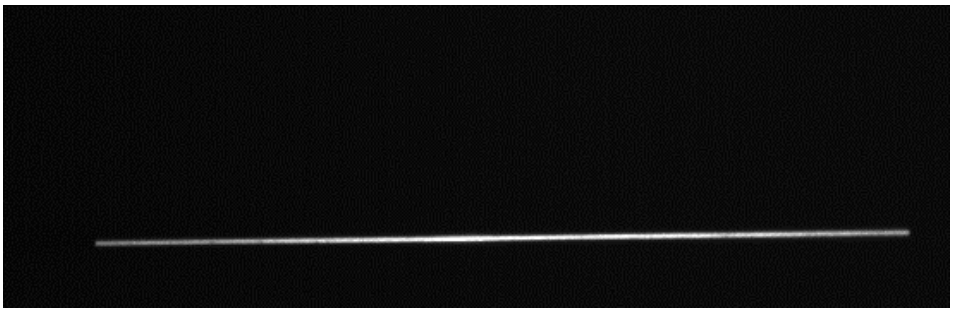


Figura 48 - Imagem do perfil laser no plano *Z*=0 do referencial WCS [39]

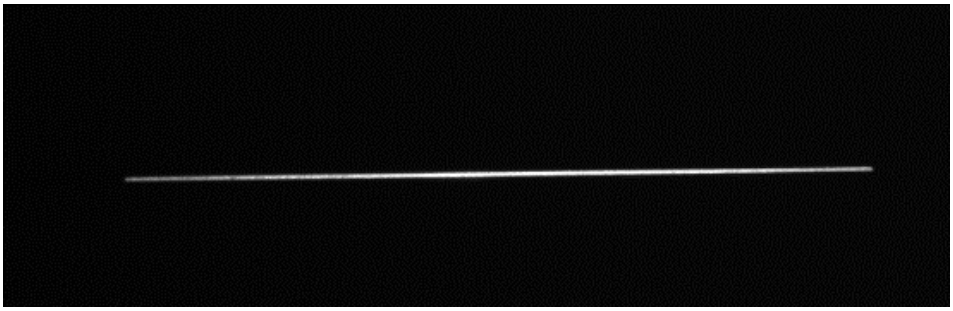


Figura 49 - Imagem do perfil laser no plano *Z*=0 do referencial TCS [39]

Após determinar as coordenadas dos perfis laser, é utilizada a função *fit_3d_plane_xyz*, que permite aferir a nuvem de pontos correspondente ao plano da luz laser. Introduzindo os pontos dos perfis laser obtidos anteriormente, obtêm-se os pontos que definem o centro do plano delimitado pelos pontos dos perfis, *Ox*, *Oy* e *Oz*, e as coordenadas do vetor normal ao plano de luz, *Nx*, *Ny* e *Nz*. Finalmente, o plano é definido com a função *get_light_plane_pose* que, mediante a introdução das coordenadas do centroide do plano e do vetor normal recentemente obtidas, devolve a *pose* do plano de luz, *LightPlanePose*.

fit_3d_plane_xyz ([X19, X20], [Y19, Y20], [Z19, Z20], Ox, Oy, Oz, Nx, Ny, Nz, MeanResidual)
get_light_plane_pose (Ox, Oy, Oz, Nx, Ny, Nz, LightPlanePose)

Tal como descrito anteriormente, a última tarefa deste processo é a calibração associada ao movimento do objeto. Para tal, é executada a leitura de duas fotos com a placa de calibração, que descrevem duas etapas distintas do movimento do objeto. De forma a aumentar a precisão desta fase da calibração, são utilizadas duas imagens não consecutivas com um número conhecido de etapas de movimento entra elas, sendo neste exemplo usadas 19 iterações de movimento. Uma vez lidas tais imagens, são utilizados os operadores *find_cali_obj* e *get_calib_data_observ_points*, que permitem determinar o

posicionamento de cada uma das placas e definir os pontos (linha e coluna) onde cada ponto do padrão de calibração está localizado na imagem.

```
read_image (CaltabImagePos1, 'caltab_at_position_1.png')
read_image (CaltabImagePos20, 'caltab_at_position_2.png')
StepNumber: = 19
find_calib_object (CaltabImagePos1, CalibDataID, 0, 0, NumCalibImages + 1, [], [])
get_calib_data_observ_points (CalibDataID, 0, 0, NumCalibImages + 1, Row1, Column1, Index1, CameraPosePos1)
find_calib_object (CaltabImagePos20, CalibDataID, 0, 0, NumCalibImages + 2, [], [])
get_calib_data_observ_points (CalibDataID, 0, 0, NumCalibImages + 2, Row1, Column1, Index1, CameraPosePos20)
```

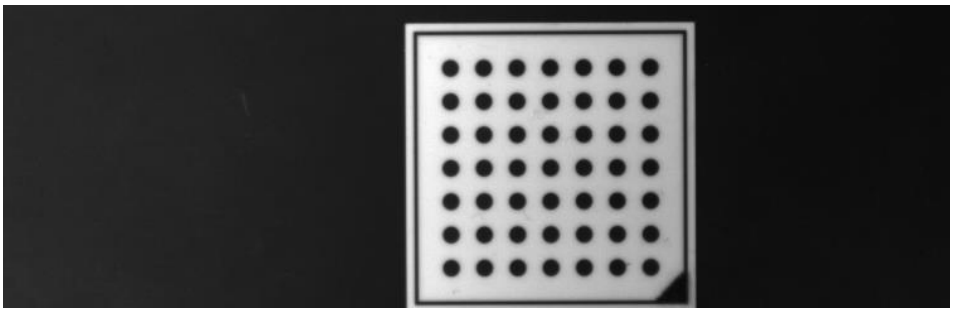


Figura 50 - Primeira posição do movimento do objeto (representado pela placa de calibração) [39]

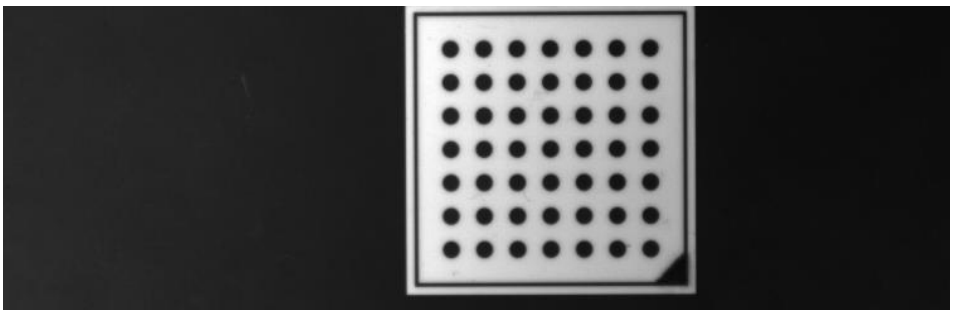


Figura 51 - Vigésima posição do movimento do objeto (representado pela placa de calibração) [39]

No passo seguinte, é determinada a transformação de posicionamento entre as duas imagens, que descreve o movimento efetuado pelo objeto em 19 etapas. No excerto de código abaixo, são calculadas as coordenadas de origem da matriz de calibração em relação ao sistema de coordenadas WCS, para as duas posições, e são determinadas as coordenadas do vetor de conversão correspondente, o vetor que traduz o movimento. O vetor é definido pelos pontos *StartX*, *StartY*, e *StartZ* e *EndX*, *EndY* e *EndZ*. O operador

create_pose traduz este vetor para a transformação geométrica efetuada entre o índice de posição 1 e o índice de posição 20, aqui denominada *MovementPoseNSteps*.

```
pose_to_hom_mat3d (CameraPosePos1, HomMat3DPos1ToCamera)
pose_to_hom_mat3d (CameraPosePos20, HomMat3DPos20ToCamera)
pose_to_hom_mat3d (CameraPose, HomMat3DWorldToCamera)
hom_mat3d_invert (HomMat3DWorldToCamera, HomMat3DCameraToWorld)
hom_mat3d_compose (HomMat3DCameraToWorld, HomMat3DPos1ToCamera, HomMat3DPos1ToWorld)
hom_mat3d_compose (HomMat3DCameraToWorld, HomMat3DPos20ToCamera, HomMat3DPos20ToWorld)
affine_trans_point_3d (HomMat3DPos1ToWorld, 0, 0, 0, StartX, StartY, StartZ)
affine_trans_point_3d (HomMat3DPos20ToWorld, 0, 0, 0, EndX, EndY, EndZ)
create_pose (EndX - StartX, EndY - StartY, EndZ - StartZ, 0, 0, 0, 'Rp+T', 'gba', 'point', MovementPoseNSteps)
```

De forma a definir a transformação geométrica relativa a uma etapa de movimento do objeto, aqui denominada *MovementPose*, divide-se a transformação geométrica obtida anteriormente, *MovementPoseNSteps*, pelo número de etapas a que corresponde. Com o valor *MovementPose*, os parâmetros da câmara (internos e externos) e as informações relativas ao plano de luz, é possível obter medições através do método *sheet of light*.

```
MovementPose := MovementPoseNSteps / StepNumber
```

4.2.3 Executar Medições

Com o HALCON, para executar medições recorrendo a um sistema de triangulação laser pelo método *sheet of light*, é necessário seguir os seguintes procedimentos e utilizar os seguintes operadores:

1. Executar a calibração do sistema de triangulação a laser;
2. Criar um modelo da *sheet of light* (folha de luz) configurada na calibração, recorrendo ao operador *create_sheet_of_light_model*;
3. Aquisição de imagens de cada perfil laser do objeto a medir;
4. Execução da medição do perfil representado em imagem utilizando *measure_profile_sheet_of_light*;
5. Os vários resultados da medição são obtidos com o operador *get_sheet_of_light_result* especificando quais o resultado que se pretende obter nos parâmetros de entrada. No caso específico do modelo tridimensional do objeto, pode obter com recurso ao operador *get_sheet_of_light_result_object_model_3d*.

De forma a demonstrar o processo descrito acima, recorre-se ao exemplo apresentado no manual do HALCON, em que é executada a reconstrução 3D da biela apresentada na Figura 52.

Figura 52 - Biela utilizada no exemplo de reconstrução 3D [39]

Neste exemplo, a introdução da informação obtida na calibração do sistema é introduzida manualmente com parâmetros apropriados ao sistema configurado para a demonstração apresentada. Tipicamente, estes valores são resultado de uma calibração prévia. Como tal, recorre-se ao operador *gen_cam_par_area_scan_polynomial*, que mediante a

introdução dos valores dos vários parâmetros internos da câmara utilizada, devolve a variável, *CamParam*, que contém toda a informação relativa a estes parâmetros internos. No âmbito do exemplo apresentado, utiliza-se o operador *create_pose* para obter os parâmetros externos ou *pose* da câmara, *CamPose*, a *pose* do plano luminoso, *LightplanePose*, e a transformação geométrica executada em cada iteração de movimento, *MovementPose*.

```
create_pose (TransX, TransY, TransZ, RotX, RotY, RotZ, OrderOfTransform, OrderOfRotation, ViewOfTransform, Pose)
```

Tal como explicado anteriormente nesta tese, a *pose* descreve uma transformação geométrica tridimensional, que consiste em translação e rotação definidas por seis parâmetros. Esta pose é variável de saída da função *create_pose*. Relativamente aos parâmetros de entrada, três deles são *TransX*, *TransY* e *TransZ*, que traduzem as translações executadas no âmbito da transformação geométrica em cada eixo coordenado. Os três parâmetros *RotX*, *RotY* e *RotZ* correspondem à rotação efetuada em cada eixo coordenado. Os parâmetros *OrderOfTransform* e *OrderOfRotation* definem, respetivamente, a ordem em que deverá ser efetuada a transformação geométrica e a ordem em que deverá ser efetuada a rotação. O parâmetro *ViewOfTransform* define qual a vista das transformações obtidas. Conforme aconselhado no manual do HALCON, por norma, no parâmetro *OrderOfTransform* deve colocar-se o valor "*Rp+T*" e no parâmetro *ViewOfTransform* utiliza-se o valor "*point*". Estes definem que primeiro é executada a translação e posteriormente a rotação. Caso se utilize o valor alternativo "*R(p-T)*", no parâmetro *OrderOfTransform*, é invertida a sequência, primeiro é executada a rotação e posteriormente a translação, negando a translação. Caso se utilize o valor "*coordinate_system*", no parâmetro *ViewOfTransform*, a sequência de transformações mantém-se inalterada mas os ângulos de rotação são negados.

No excerto de código abaixo, é demonstrado a aplicação dos operadores *gen_cam_par_area_scan_polynomial* e *create_pose*, no exemplo de reconstrução 3D a partir da triangulação laser, que tem vindo a ser demonstrado.

```
gen_cam_par_area_scan_polynomial (0.0126514, 640.275, -2.07143e+007, 3.18867e+011, -0.0895689, 0.0231197, 6.00051e-006, 6e-006, 387.036, 120.112, 752, 240, CamParam)
```

```
create_pose (-0.00164029, 1.91372e-006, 0.300135, 0.575347, 0.587877, 180.026, 'Rp+T', 'gba', 'point', CamPose)
create_pose (0.00270989, -0.00548841, 0.00843714, 66.9928, 359.72, 0.659384, 'Rp+T', 'gba', 'point', LightplanePose)
create_pose (7.86235e-008, 0.000120112, 1.9745e-006, 0, 0, 0, 'Rp+T', 'gba', 'point', MovementPose)
```

No próximo passo, é criado o modelo da *sheet of light*, com o operador *create_sheet_of_light_model*. Este operador recebe como primeiro parâmetro de entrada, uma região de interesse pré-definida, que em largura, deverá ser pouco maior que a largura do objeto em medição, e em altura deverá ser superior à maior disparidade prevista, ou seja, ao deslocamento máximo da linha de laser promovido pela altura do objeto. A região pode ser definida desenhando um retângulo. No HALCON, é possível obter um retângulo com operador *gen_rectangle1*. O operador *create_sheet_of_light_model* tem ainda como parâmetros de entrada *GenParamName* e *GenParamValue*, utilizados, respetivamente, para definir parâmetros genéricos de ajuste do modelo e o valor de ajuste deste parâmetro. No exemplo da reconstrução 3D demonstrada, ao utilizar este operador define-se o valor mínimo em escala de cinzento, que é considerado para efetuar a medição da posição do perfil. Para tal, o valor "*min_gray*" é introduzido no parâmetro *GenParamName*, e no parâmetro *GenParamValue*, o valor de cinzento definido corresponde a 70. Define-se o número de perfis do laser utilizados para efetuar a reconstrução 3D, com o valor "*num_profiles*" no parâmetro *GenParamName* e o valor 290 no parâmetro *GenParamValue*. Por último, é definido o modo de decisão quando determinado perfil laser é de identificação ambígua, com o valor "*ambiguity_solving*" no parâmetro *GenParamName* e o valor "*first*" no parâmetro *GenParamValue*. Este considera o primeiro candidato encontrado, sendo o método mais rápido. O parâmetro de saída do operador é a variável do modelo da *sheet of light*, *SheetOfLightModelID*, que guarda todas as informações introduzidas nos parâmetros de entrada.

```
create_sheet_of_light_model (ProfileRegion, GenParamName, GenParamValue, SheetOfLightModelID)
```

Uma vez criado o modelo, este pode ser alterado mediante utilização do operador *set_sheet_of_light_param*.

```
set_sheet_of_light_param (SheetOfLightModelID, GenParamName, GenParamValue)
```

Este operador apenas tem parâmetros de entrada, o identificador do modelo do perfil laser, *SheetOfLightModelID,* e os parâmetros *GenParamName* e *GenParamValue,* que têm a mesma finalidade que foi descrita para o *create_sheet_of_light_model*. No exemplo, o *set_sheet_of_light_param* é utilizado para definir que a medição efetuada é calibrada, definindo *GenParamName* como "*calibrated*" e *GenParamValue* como "*xyz*", sendo também definida a escala de apresentação dos valores de medição em milímetros, definindo o primeiro parâmetro como "*scale*" e o segundo como "*mm*". De forma a associar a informação de calibração, os valores das *poses* obtidas anteriormente são introduzidos neste modelo. São feitas várias chamadas do *set_sheet_of_light_param*, para associar tal informação. Assim, para introduzir os parâmetros internos da câmara, definem-se os vários valores de *GenParamName* como "*camera_parameter*", para os parâmetros externos da câmara define-se como "*camera_pose*", para o posicionamento do plano de luz "*lightplane_pose*" e para a transformação geométrica associada ao movimento do objeto, "*motion_pose*". Para cada uma destas introduções, o parâmetro *GenParamValue,* recebe respetivamente as variáveis *CamParam, CamPose, LightplanePose* e *MovementPose.*

Abaixo, é demonstrado o excerto do exemplo onde são empregues os operadores *gen_rectangle1*, *create_sheet_of_light_model* e *set_sheet_of_light_param.*

```
gen_rectangle1 (ProfileRegion, 120, 75, 195, 710)
create_sheet_of_light_model (ProfileRegion, ['min_gray','num_profiles',
'ambiguity_solving'], [70,290,'first'], SheetOfLightModelID)
set_sheet_of_light_param (SheetOfLightModelID, 'calibration', 'xyz')
set_sheet_of_light_param (SheetOfLightModelID, 'scale', 'mm')
set_sheet_of_light_param (SheetOfLightModelID, 'camera_parameter', CamParam)
set_sheet_of_light_param (SheetOfLightModelID, 'camera_pose', CamPose)
set_sheet_of_light_param (SheetOfLightModelID, 'lightplane_pose', LightplanePose)
set_sheet_of_light_param (SheetOfLightModelID, 'movement_pose', MovementPose)
```

Na terceira etapa do processo de medição, é executada a aquisição de imagens contendo o perfil laser, uma imagem para cada movimento do objeto. A aquisição de imagens é efetuada através de uma câmara, ou de imagens pré-existentes no ficheiro. Caso se

pretenda utilizar a aquisição de imagens com câmara, recorre-se ao operador *grab_image_async*. Para obter imagens a partir de ficheiros, recorre-se ao operador *read_image*. Na Figura 53 é apresentado o perfil laser tipicamente obtido e um retângulo em volta desta linha luminosa correspondente à região de interesse.

Figura 53 - Perfil laser e respetiva região de interesse [39]

Para executar a medição do perfil laser, que deverá estar contido na região de interesse previamente definida para o modelo, utiliza-se o operador *measure_profile_sheet_of_light*, que determina a disparidade de cada perfil e guarda essa informação no modelo.

```
measure_profile_sheet_of_light (ProfileImage, SheetOfLightModelID, MovementPose)
```

Este operador tem como parâmetros de entrada, a imagem do perfil laser, *ProfileImage*, e o identificador modelo do perfil a que se pretende associar a informação da disparidade da linha de perfil laser representada na imagem analisada, *SheetOfLightModelID*. Existe ainda o parâmetro de entrada, *MovementPose*, onde é introduzida a informação relativa à transformação geométrica associada ao movimento do objeto a medir. No entanto, este parâmetro é, por norma, deixado em branco, uma vez que esta informação deve ser veiculada previamente, na fase de criação e descrição do modelo do perfil de iluminação laser. Abaixo, é demonstrado o excerto de código do exemplo que tem vindo a ser utilizado para demonstração deste processo, onde figura um ciclo de repetição *for*, com 290 repetições correspondentes aos duzentos e noventa perfis do laser utilizados para efetuar a reconstrução 3D. Quanto maior o número de perfis necessários, maior será o tempo de processamento da medição. Nas imagens resultantes da medição (seja a imagem de disparidades ou as imagens relativas às coordenadas *XYZ*), o número de colunas é definido pela largura da região de interesse e o número de linhas é definido pelo número de perfis laser.

```
for Index: = 1 to 290 by 1
    read_image (ProfileImage, 'sheet_of_light/connection_rod_' + Index$'.3')
    measure_profile_sheet_of_light (ProfileImage, SheetOfLightModelID, [])
endfor
```

No último passo deste processo, são obtidos os resultados pretendidos a partir das medições do perfil efetuadas na etapa anterior. Para obter o modelo 3D do objeto analisado pelo sistema, recorre-se ao operador *get_sheet_of_light_result_object_model_3d*, que mediante a introdução do identificador do modelo do perfil laser, *SheetOfLightModelID*, devolve uma variável que corresponde a um modelo 3D do objeto medido através deste sistema de triangulação laser.

O operador *get_sheet_of_light_result* permite obter todos os outros tipos de resultados, imagem de disparidades e imagens de coordenadas *XYZ*.

```
get_sheet_of_light_result (ResultValue, SheetOfLightModelID, ResultName)
```

Para obter a imagem de disparidades utilizando este operador, utiliza-se o valor "*disparity*" no parâmetro *ResultName*, que permite definir o resultado a obter. O parâmetro *ResultValue* corresponde à variável de saída criada por este operador, denominada *Disparity* no excerto de código exemplificado abaixo. Para obter as imagens com a representação das coordenadas *X*, *Y* e *Z*, é necessário empregar o operador *get_sheet_of_light_result* três vezes, uma por cada eixo, colocando respetivamente no parâmetro *ResultName*, os valores "*x*", "*y*" e "*z*". Para a variável de saída, deve nomear-se um nome diferente para a imagem relativa a cada eixo. No excerto de código, as imagens recebem respetivamente o nome de *X*, *Y* e *Z*.

```
get_sheet_of_light_result (Disparity, SheetOfLightModelID, 'disparity')
get_sheet_of_light_result (X, SheetOfLightModelID, 'x')
get_sheet_of_light_result (Y, SheetOfLightModelID, 'y')
get_sheet_of_light_result (Z, SheetOfLightModelID, 'z')
get_sheet_of_light_result_object_model_3d (SheetOfLightModelID,
ObjectModel3DID)
```

O modelo 3D do objeto obtido através da reconstrução calibrada por meio de triangulação laser *sheet of light* está representado na Figura 54.

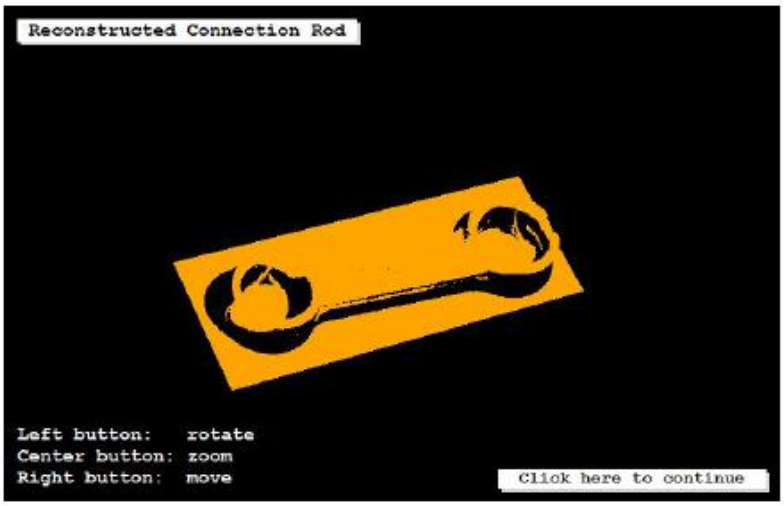


Figura 54 - Modelo 3D resultante da reconstrução calibrada por meio de triangulação laser *sheet of light* [39]

Tanto para a imagem de disparidades como para as imagens relativas às coordenadas *XYZ*, os valores de cinzento são escalados de tal forma que as zonas representadas a preto não sejam consideradas no domínio da imagem resultante, indicando áreas em que não é possível extrair qualquer informação tridimensional. Nestas imagens, as zonas mais claras dentro do domínio em que possível executar a reconstrução do objeto, representam as zonas onde o valor representado é mais baixo, as zonas mais escuras representam partes onde o valor representado é mais elevado. Na Figura 55, estão presentes três imagens com a representação das coordenadas *X*, *Y* e *Z*, e na Figura 56 é demonstrada a imagem de disparidades obtida na reconstrução do objeto, que tem sido utilizada como exemplo. Nestas imagens, retiradas do manual do HALCON, as representações são efetuadas numa escala colorida de forma a facilitar a distinção e diferenciação dos valores obtidos. A relação entre os valores obtidos e a escuridão ou claridade mantém-se igual à relação descrita para uma escala de cinzento.

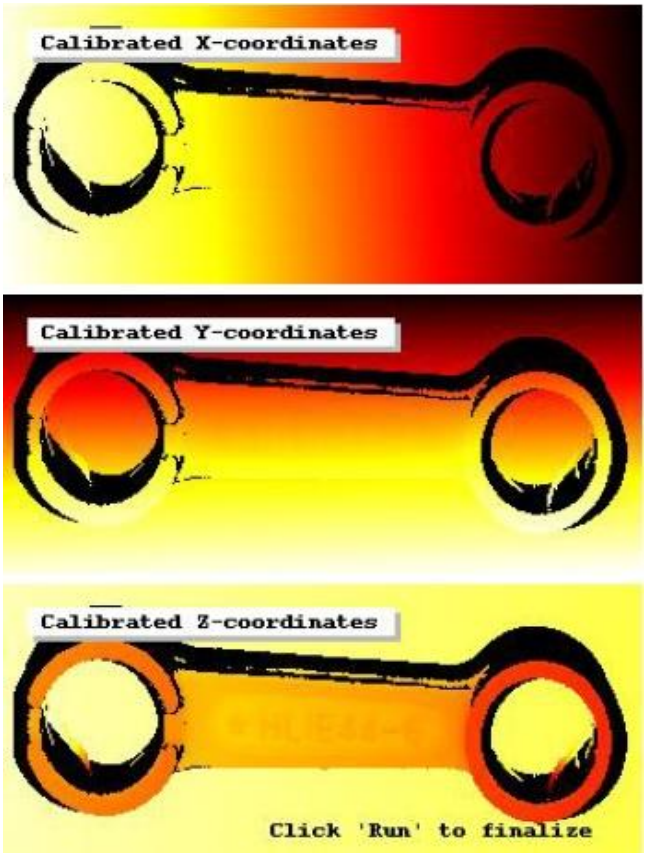


Figura 55 - Imagens das coordenadas *X*, *Y* e *Z* (de cima para baixo) obtidas nesta reconstrução [39]

Figura 56 - Imagem de disparidade obtida na reconstrução 3D efetuada no exemplo presente [39]

5 Projeto Aquatropolis - *Intelligent Management System for Sustainable Aquacultures*

5.1 Descrição e Enquadramento do Projeto

No estágio no LINE.IPT que é descrito neste documento, parte do trabalho desenvolvido pelo aluno esteve inserido no âmbito do projeto Aquatropolis – *Intelligent Management System for Sustainable Aquacultures*. Este projeto consistiu no desenvolvimento de uma solução tecnológica que, de forma inteligente, permitisse facilitar tarefas de gestão, operação e controlo de produções de aquacultura em regime semi-intensivo, particularmente culturas de dourada, robalo e macroalgas.

O Aquatropolis foi desenvolvido por um consórcio de entidades que contribuíram para a execução do projeto. O promotor desta iniciativa foi a Compta Emerging Business, que juntou ao consórcio a ALGAplus, a Domatica, a TAGUSVALLEY, o Instituto Politécnico de Tomar e o Instituto Politécnico de Leiria. A solução tecnológica projetada para o Aquatropolis foi constituída por sete unidades com funções distintas, mas complementares entre si:

- Unidade de Processamento Inteligente (UPI), elemento central e agregador das demais unidades identificadas na arquitetura da solução, compreendendo todas as funcionalidades de análise de dados e suporte à decisão;
- Unidade de Monitorização Ambiental (UMA) consiste num sistema de instrumentação com capacidade de centralizar num só dispositivo toda a capacidade de aquisição de dados relativos à qualidade da água;
- Unidade de Controlo de Biomassa (UCB) é um sistema para dar resposta às tarefas de contagem e monitorização do desenvolvimento da biomassa;
- Unidade de Controlo de Consumos (UCC), rede de sensores de monitorização de consumos energéticos;
- Unidade de Controlo e Automação (UCA) está encarregue de gerar alertas e desencadear procedimentos, mediante os dados recebidos da UMA e da UCB
- Unidade de Observação e Serviços Partilhados (UOSP), unidade inteligente responsável por monitorizar comportamentos e ecossistemas e atuar automaticamente ou remotamente sobre equipamentos;

- Unidade de Relação de Mercados (URM) é um sistema baseado numa plataforma eletrónica que proceder à análise do mercado de forma a controlar a produção, permitindo alguma previsibilidade e redução dos riscos dos aquicultores relativamente à incapacidade de conseguirem fazer chegar os seus produtos ao mercado.

A Compta Emerging Business como promotora deste projeto ficou encarregue do desenvolvimento do sistema central, a UPI, e da UCC e da URM, funcionando ainda como o integrador do conjunto de todas as unidades na solução final.

A Domatica, empresa especializada em soluções tecnológicas, desenvolveu a UOSP e a UCA, unidade que comunica diretamente com os sistemas desenvolvidos pelo LINE e IPT.

O IPL foi o membro do consórcio que contribuiu com conhecimento na área das ciências do mar, colaborando no processo de definição da arquitetura e especificações da solução, e nos testes e experiências dos sistemas desenvolvidos.

A ALGAplus é uma empresa que produz e comercializa produtos baseados em macroalgas e seus derivados, produzidas numa aquacultura multitrófica integrada. Assim, este membro apresenta o perfil de um potencial utilizador da solução desenvolvida.

Coube ao LINE.IPT (TAGUSVALLEY) e ao Instituto Politécnico de Tomar, o desenvolvimento da Unidade de Monitorização Ambiental (UMA) e da Unidade de Controlo da Biomassa (UCB). Foi ainda desenvolvido um veículo anfíbio baseado em robótica móvel autónoma, um ROV, com intuito de transportar a UCB e a UMA dentro e fora dos tanques de aquacultura.

Os objetivos de desenvolvimento da UCB referem que deveria ser criado um sistema móvel robotizado capaz de executar tarefas de contagem e monitorização do desenvolvimento da biomassa. Em termos de desenvolvimento, optou-se por se separar esta componente do desenvolvimento do equipamento de visão por computador, pois a unidade robótica desenvolvida acomodava igualmente a UMA, deixando assim de fazer sentido abordar o desenvolvimento neste veículo apenas na informação da UCB. Assim, o desenvolvimento da unidade robótica será abordado em particular, deixando de ser

considerado como parte da UCB, passando a ser considerado como uma terceira unidade denominada ROV (*remotely operated vehicle*).

A solução desenvolvida para o projeto Aquatropolis segue uma arquitetura modular, tal como demonstrado na Figura 57, de modo a garantir que as três unidades desenvolvidas (UMA, UCB, ROV) possam funcionar em conjunto ou como sistemas individuais e independentes. Tanto a Unidade de Monitorização Ambiental (UMA) como a Unidade de Controlo da Biomassa (UCB) foram instaladas no ROV, permitindo conferir mobilidade a estas unidades, o que possibilita a obtenção de parâmetros e imagens em vários zonas de um tanque e em vários tanques.

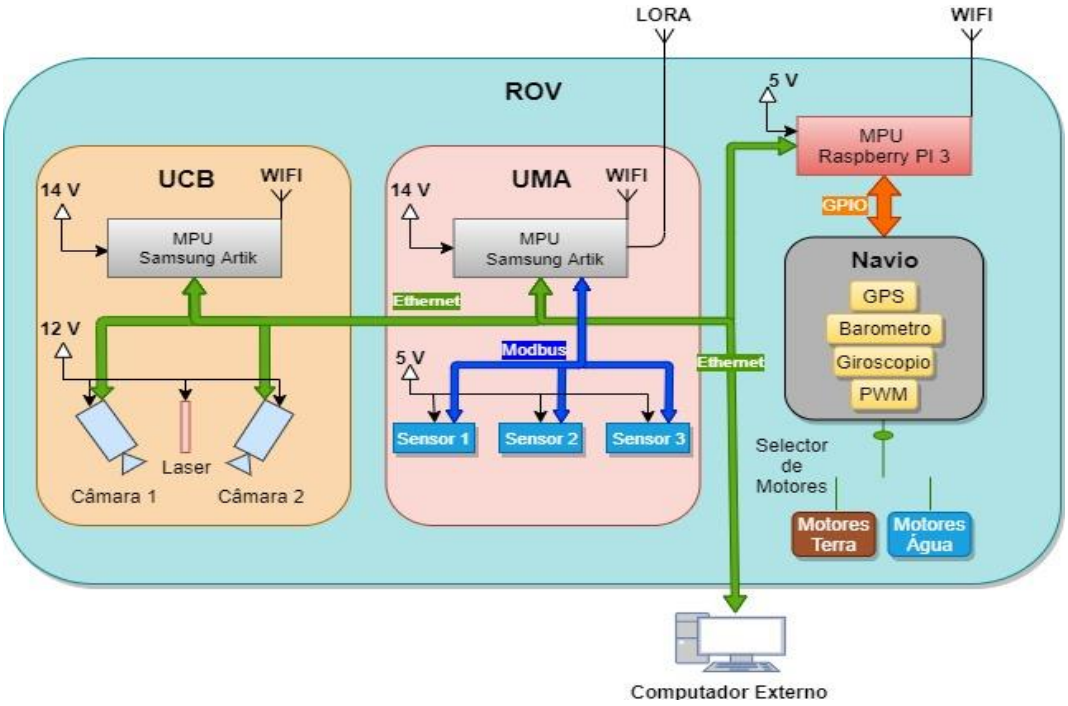


Figura 57 - Arquitetura da solução desenvolvida pelo grupo de trabalho LINE.IPT

No esquema da Figura 57, são apresentados os principais componentes que compõem cada uma das unidades desenvolvidas. Conforme apresentado neste esquema, a comunicação é efetuada através de ligação em rede, via cabo Ethernet. No entanto, as três unidades estão preparadas para comunicação *wireless*. Neste esquema de arquitetura, é considerado ainda um PC externo, um computador independente de todas estas unidades, utilizado sobretudo na fase de prototipagem e testes para executar ajustes nos vários sistemas envolvidos.

A UCB é constituída por um microprocessador Samsung ARTIK 710, duas câmaras ligadas a este via Ethernet e ainda um laser. A UMA tem o mesmo microprocessador ao qual são ligados sensores por meio de um barramento RS-485. Este sistema está preparado para ligação de outros sensores em canais de sinais analógicos (0,10V ou 4-20mA) ou digitais. O ROV possui uma unidade de navegação autónoma, Navio2, uma expansão para Raspberry PI 3, que permite a ligação de controladores de velocidade eletrónicos de motores (ESC), permitindo comandar motores aquáticos ou terrestres. O Navio2 inclui ainda todos os sensores necessários para a navegação (GPS, giroscópio, etc.).

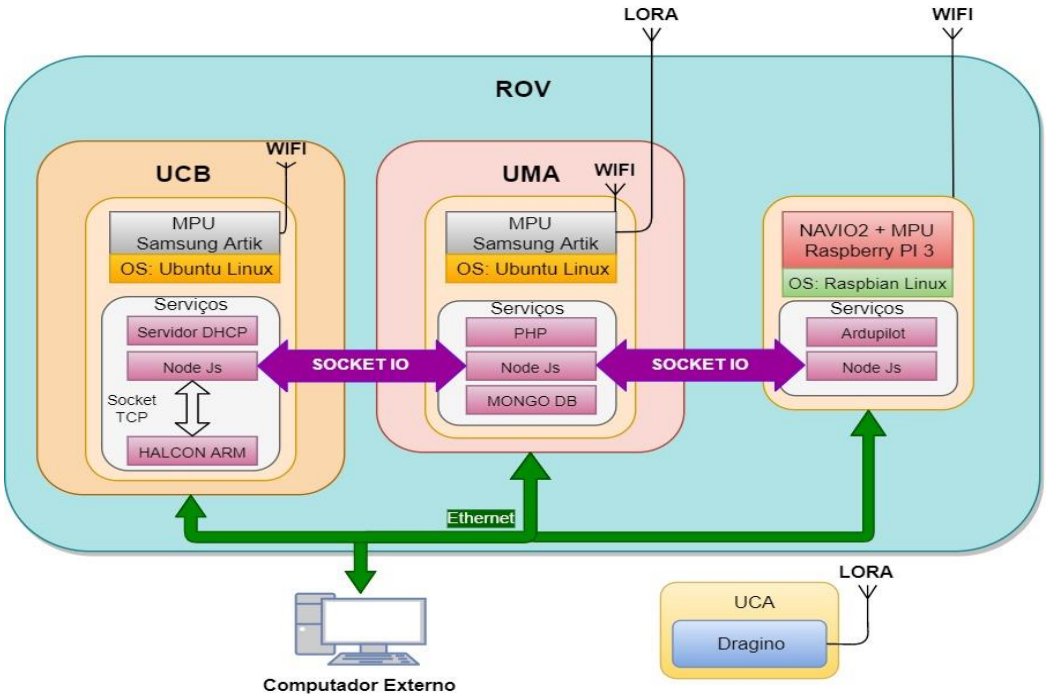


Figura 58 - Arquitetura do *software* e respetivas comunicações

Na Figura 58, é apresentado um esquema relativo aos serviços de *software* utilizados em cada unidade e respetiva interligação. A descrição detalhada do funcionamento de cada unidade será descrita nos capítulos subsequentes, que abordam cada um dos sistemas em particular.

Na arquitetura apresentada na Figura 58, são utilizados alguns protocolos de comunicação diferentes, nomeadamente:

- Modbus: Protocolo de comunicação industrial do tipo mestre-escravo que permite ligar vários dispositivos no mesmo barramento, sendo neste caso específico um barramento RS-485;

- HTTP: O HTTP (*Hypertext Transfer Protocol*) é um protocolo de comunicação entre cliente e servidor. Este protocolo é utilizado na WWW para servir páginas Web. Quando um utilizador acede a determinado site, é efetuado uma requisição ao servidor HTTP que irá responder ao cliente que a requisição teve sucesso e o recurso foi encontrado exibindo a página do site;
- SocketTCP: Um *socket* de rede é um terminal de um dispositivo ou *node* para enviar ou receber dados numa rede. O TCP (*Transmission Control Protocol)* é um protocolo de transmissão um para um. Neste protocolo, os *sockets* estão associados a um endereço IP único e um número de porta para o *node* local, ligados a um outro *node* externo (noutro dispositivo), com o endereço de *socket* (*socket address*) correspondente;
- Socket.IO: Socket.IO é uma biblioteca JavaScript para aplicações Web em tempo real. Esta biblioteca permite executar comunicações bidirecionais em tempo real entre clientes da Web e servidores. O Socket.IO divide-se em duas partes, uma biblioteca do lado do cliente que é executada no navegador e uma biblioteca do lado do servidor para o Node.js;
- LoRa: LoRa é uma tecnologia de radiofrequência que permite comunicação a longas distâncias com consumo mínimo de energia;
- MAVLink: O MAVLink ou *Micro Air Vehicle Link* é um protocolo para comunicação com pequenos veículos não tripulados.

Atendendo à arquitetura apresentada na Figura 58, considera-se uma camada externa e uma camada interna.

Na camada externa, encontra-se o computador e o UCA. O computador está ligado em rede a todas as unidades via Ethernet e WI-FI. A ligação Ethernet é vital para a comunicação entre o computador e o Raspberry Pi 3 equipado com o Navio2, uma vez que o computador é utilizado como estação de controlo e monitorização do ROV. A UCA recebe os dados recolhidos pela UMA e pela UCB e envia instruções provenientes da UPI para todos estes sistemas. A UCA, através do Dragino, um dispositivo de comunicação LoRa baseado em Arduíno, envia comandos e recebe dados do microprocessador da UMA. A UMA agrega a informação proveniente de todos os sistemas e distribui todas as informações provenientes da UCA.

Na camada interna, consideram-se os microprocessadores ARTIK da UMA e da UCB e o Raspberry Pi 3 do ROV. Tal como elucidado anteriormente, estas unidades estão interligadas entre si via Ethernet.

Nas várias unidades desenvolvidas para o Aquatropolis, utiliza-se o Node.js devido às suas capacidades de escalabilidade, rapidez de prototipagem e facilidade de aprendizagem e de programação. Este interpretador é multi-plataforma e possui bastantes pacotes (*node packages*) já implementados pela comunidade aberta deste *software*, que permitem acelerar o processo de desenvolvimento neste ambiente.

Nos microprocessadores das três unidades foram implementados servidores Node.js que executam serviços diversos, dependendo das necessidades de cada sistema. Estes processos de Node.js comunicam entre si por Socket.IO, sendo que a UMA ocupa um papel de "gestão" da informação, nomeadamente ordens de produção, provenientes da UCA para as unidades deste sistema.

5.2 Unidade de Controlo da Biomassa (UCB)

A UCB é um equipamento que permite a execução automática das tarefas de contagem e monitorização do desenvolvimento da biomassa em tanques de aquacultura semi-intensiva.

Atendendo à documentação deste projeto, onde estão presentes os requisitos e objetivos do mesmo, de seguida são apresentados os objetivos específicos do desenvolvimento da UCB:

- Desenvolver um sistema móvel robotizado capaz de executar tarefas de contagem e monitorização do desenvolvimento da biomassa;
- Estimar a biomassa combinando tecnologias de câmaras e sonar, o sonar para detetar a localização do cardume e as câmaras para aferir as suas dimensões e estimar a biomassa total;
- Implementar a estratégia de controlo por duas fases: a localização dos peixes e a estimação da biomassa. Todo o processamento imagem e sinais é efetuado localmente através de algoritmos especificamente desenvolvidos para a aplicação;

- Ter capacidade de aprendizagem e adaptabilidade à espécie/forma de cultura e ser suportado em algoritmos adaptativos para análise de previsões de crescimento e controlo de desvios da biomassa total;
- Apenas deverá existir uma unidade por exploração de aquacultura, mas que permita a monitorização de todos os tanques da produção.

Os objetivos de âmbito geral de desenvolvimento do sistema são:

- Desenvolver métodos para o controlo de biomassa, utilizando multisensores adaptáveis às diversas espécies e com inteligência que, em tempo real, possam fornecer informação relativamente ao desenvolvimento da biomassa em cada "Estação de Trabalho";
- Potenciar um modelo de criação de escalas de automação através da conceção de uma arquitetura de referência para interoperabilidade entre diferentes sensores e atuadores;
- Desenvolver os conetores para a interligação da plataforma com múltiplas fontes de dados (Sistemas de medição de Biomassa; Sensores de Temperatura, Salinidade; Estacões Meteorológicas; etc.) e com os atuadores (Sistemas de Alimentação, oxigenadores, etc.);

Atendendo aos objetivos propostos para a UCB, desenvolveram-se esforços de modo a determinar a solução mais adequada para executar tarefas de quantificação e medição de biomassa em tanques de aquacultura. A equipa do LINE.IPT efetuou uma pesquisa do estado da arte das tecnologias utilizadas no controlo da biomassa em aquacultura, de forma a perceber quais as abordagens mais utilizadas neste âmbito. Foi igualmente efetuada uma pesquisa de produtos existentes no mercado que permitissem desempenhar tarefas de controlo da Biomassa.

Com base na documentação, o grupo de trabalho optou por definir que o tipo de sistema mais adequado para base tecnológica da UCB seria um sistema de visão estéreo com recurso a iluminação laser, em conjunto com tecnologia de sonar. Esta decisão pode ser corroborada consultando a tabela resumo de tecnologias de câmaras, em "*Application of machine vision systems in aquaculture with emphasis on fish: state-of-the-art and key issues*" (Saberioon, Gholizadeh, Cisar, Pautsina e Urban, 2016), no capítulo do estado da arte. Segundo a tabela, de entre as tecnologias avaliadas e mais regularmente utilizadas,

os sistemas de visão estéreo composto por duas câmaras no espectro do visível, são o único simultaneamente aplicável em situações com o estudo do comportamento de peixes, classificação de espécies, medição de características (tamanho e peso) e contagem. São apresentadas como vantagens deste tipo de sistema a exatidão e alta resolução espacial e como desvantagens a necessidade de uma calibração cuidada e de conhecimentos técnicos mais aprofundados que outros tipos de sistema.

A utilização do sonar em conjunto com um sistema de visão estéreo justificar-se-ia, pois, o segundo sistema isolado é pouco eficaz em águas com fraca visibilidade (turbidez e luminosidade desequilibrada) como é o caso dos tanques de aquacultura. Assim, supôs-se que o sonar poderia ser útil para determinar a localização das maiores concentrações de peixe dentro dos tanques. Com esta informação, a plataforma que contivesse a UCB poderia ser deslocada e colocada nas zonas onde fosse possível obter mais resultados de medição e estudo de peixes.

Uma vez decidida a base tecnológica da UCB, iniciaram-se os trabalhos de desenvolvimento desta unidade. O principal foco de desenvolvimento foi o sistema de visão estéreo, uma vez que é um sistema que apresenta alguma complexidade a nível técnico e que necessita de vários cuidados de forma a obter resultados corretos e precisos. O protótipo da UCB desenvolvido acabou por não integrar tecnologia de sonar, devido ao facto dos equipamentos passíveis de serem utilizados neste tipo de solução, uma solução que deverá ser o mais económica possível, não terem resolução suficiente para diferenciar os peixes do fundo e paredes do tanque. Também não foi muito explorado o uso do sonar pelo facto de o sistema estéreo ter consumido a grande maioria do tempo de desenvolvimento dedicado a esta unidade.

A UCB desenvolvida é um sistema de visão estéreo com duas câmaras monocromáticas que permitem obter imagens do ambiente aquático onde os peixes se inserem de modo a analisar e determinar a biomassa presente nos tanques das produções de aquicultura. De modo a facilitar a identificação de contornos e garantir a precisão deste equipamento, é utilizada uma matriz de linhas laser que permite tornar mais nítidos os contornos de objetos presentes nas imagens, especialmente em situações de fraca luminosidade ou com turbidez, condições normais nos tanques de aquacultura.

A grande maioria das componentes desta unidade foi desenvolvida pelo aluno, que procedeu a todas as configurações de *hardware* e sistemas de visão e efetuou toda a programação de *software* de processamento de imagem.

A UCB divide-se em três módulos:

- *Hardware* de captura: Câmaras e iluminação;
- *Hardware* de integração: Unidade de processamento, que executa os algoritmos de processamento de imagem e comunica com as câmaras e com as restantes unidades (UMA e UCB);
- *Software*: *Software* de processamento de imagem para executar tarefas de determinação da biomassa de peixes.

5.2.1 Arquitetura da UCB

Na Figura 59, é demonstrada a arquitetura da Unidade de Controlo da Biomassa, com os detalhes de interligação do *hardware* com os serviços de software no microcontrolador Samsung ARTIK 710.

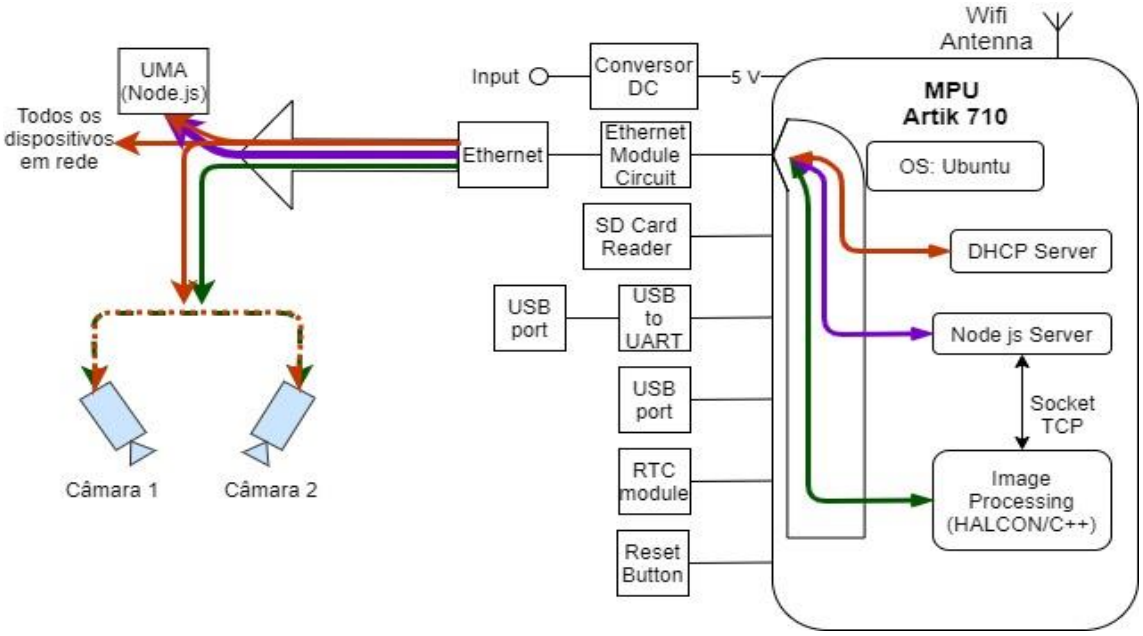


Figura 59 - Arquitetura da UCB (relação entre *Hardware* e *Software*)

5.2.2 *Hardware* da UCB

Na solução desenvolvida pelo LINE.IPT no âmbito do projeto Aquatropolis, muitos dos dispositivos utilizados são baseados em circuitos desenvolvidos personalizadamente para cada uma das unidades. A Unidade de Controlo de Biomassa é baseada num sistema desenvolvido de raiz com recurso a um microprocessador de 64 bits ARTIK 710 da Samsung e sistema operativo GNU/Linux.

Tabela 3 - Especificações do ARKIT 710

Parâmetro	Valor
Tipo de Processador	ARM Cortex A53
Frequência	1.4 GHz
Tamanho da Memória	1 GB
Tensão de Alimentação	3,7V a 5V
Interfaces	Ethernet, GPIO, I2C, JTAG, SDIO, SPI, USB
Memória *Flash*	4 GB
Tipo de Memória	DDR3
Tipo de Produto	*System On Module*: SOM

O ARTIK 710 é um "*System On Module*" (SOM), ou seja, o circuito integrado deste microprocessador inclui um conjunto de funcionalidades embutidas que facilita o desenvolvimento de placas de circuito impresso e diminui o número de componentes externos necessários. Assim, necessita apenas de alguns circuitos auxiliares para estar apto a funcionar.

A utilização de um microprocessador com sistema operativo GNU/Linux de 64 bits para correr os sistemas e *software* necessário garante que, caso no futuro seja necessário um upgrade de *hardware*, ou este componente seja descontinuado, todo o sistema e *software* desenvolvido é 100% compatível com outro sistema operativo ou tipo de microprocessador, desde que tenha a mesma arquitetura (64 bits).

Na Figura 60, está representado o *layout* de um destes circuitos, neste caso, a PCB utilizada para incorporar o microprocessador ARTIK 710 e vários circuitos auxiliares, que são a base do sistema de processamento da UCB.

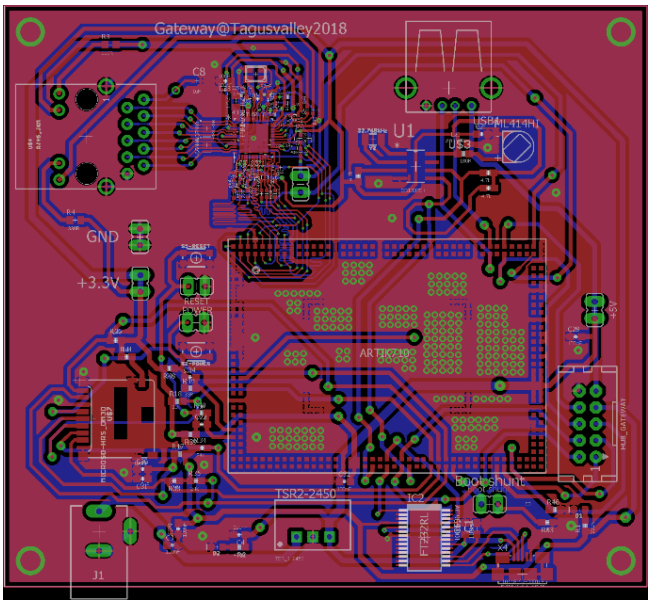


Figura 60 - *Layout* da PCB da UCB

A PCB da UCB, representada na Figura 60, incorpora vários circuitos auxiliares necessários ao funcionamento das funções propostas. Para além do módulo ARTIK710 e dos circuitos auxiliares de funcionamento do módulo, a placa contém o circuito de alimentação, o circuito de Gigabit Ethernet com o chip RTL8211 para a transferência de dados via Ethernet, nomeadamente os dados das câmaras essenciais na leitura da biomassa. Outros dos circuitos fundamentais são o circuito do *Real Time Clock* que é o relógio de tempo real que se mantém em funcionamento mesmo com o sistema desligado, de modo a não perder a hora/data, o circuito de leitura/escrita de cartão micro SD, utilizado, por exemplo, para efetuar a instalação do sistema operativo, e o circuito de ligação USB.

Destacando os elementos mais importantes do circuito da UCB: no Anexo 6 – Circuito do Módulo do ARTIK 710, está representado o módulo do ARTIK 710, respetivo circuito de alimentação e os circuitos auxiliares de funcionamento. O circuito de alimentação desenhado para fornecer energia ao ARTIK, recorre a um conversor DC/DC (Traco Power TSR2-2450) que utiliza como tensão de entrada entre 6,5V a 36V, de forma a obter uma voltagem de saída de 5V. Uma vez alimentado com esta tensão, o módulo ARTIK 710 tem uma fonte de alimentação interna que fornece tensão de 3,3V para o exterior, utilizada para alimentação dos circuitos auxiliares. O módulo é ligado com o botão de pressão, também representado na imagem acima, existindo ainda um botão para reiniciar o sistema caso seja necessário.

O circuito de Ethernet, apresentado no Anexo 7 – Circuito de Ethernet da UCB, é baseado no *transceiver* RTL8211 da Realtek. Este circuito integrado necessita de diversas resistências de *pull-up* e *pull-down* nas suas linhas de comunicação, bem como de condensadores para filtragem e estabilização de sinal, limitação de corrente nas linhas de comunicação, necessitando ainda de um cristal de relógio que define a frequência de funcionamento. A comunicação para o exterior é efetuada com uma ficha RJ45 de 10 pinos (Gigabit), tendo nas suas linhas de comunicação uma proteção ESD com o circuito protetor em *array* de 4 linhas HSP061-4M10 da ST.

A solução desenvolvida recorre a um par de câmaras Teledyne Dalsa Genie Nano M2420, com lentes Kowa LM5JCM e um laser Z-Laser ZM18S3 que projeta uma matriz de vinte e cinco linhas. De seguida, identificam-se as características do material utilizado:

Tabela 4 - Características da câmara Teledyne Dalsa Genie Nano M2420

Característica	**Valor**
Dimensões (C x A x L)	38.9 mm x 29 mm x 44 mm (c/lente e conector Ethernet)
Peso	+/- 46 g (s/lente)
Tensão de Alimentação	10 a 36 VDC
Comunicação	Gigabit Ethernet 1000Mbps (115MB/sec max)
Resolução	2448x2048
Tamanho Pixel	3.45 μm x 3.45 μm
Frame Rate máximo	34.4 fps

Tabela 5 - Características da lente Kowa LM5JCM

Característica	**Valor**
Distância Focal	5 mm
Abertura Mínima	2,8 mm
Distância Mínima	100 mm
Formato	2/3”
Rosca do Filtro	40.0 x 0.5 mm
Comprimento do corpo	45,9 mm

Tabela 6 - Características do laser Z-Laser ZM18S33

Característica	**Valor**
Dimensões (C x Ø):	120 x 20 mm
Categoria de Proteção	IP67
Conector de Alimentação	M12 de 4 pinos
Voltagem de Alimentação	5 a 30 VDC
Consumo	<4VA
Potência de Saída	120 mW
Comprimento de Onda	635 nm
Acessório	Filtro com padrão de vinte e cinco linhas

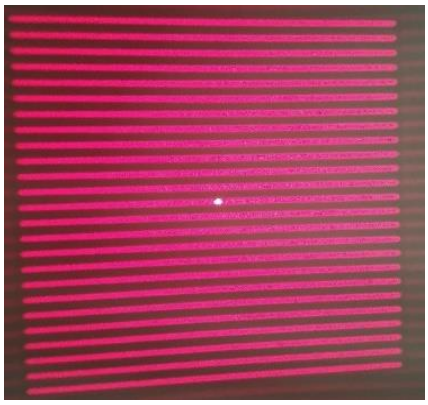


Figura 61 - Projeção do padrão de 25 linhas do laser

As soluções para estimativa da biomassa em tanques de aquacultura atualmente existentes no mercado são baseadas em sistemas de câmaras em estéreo, tal como o representado na Figura 62. Este tipo de sistema, aplicado na aquisição de dados relativos à biomassa em aquacultura, permite obter informação com elevada precisão, sem ser necessário forçar os peixes a tomarem determinada posição/orientação. Este tipo de sistema é ainda vantajoso pois permite que o *setup* esteja montado numa plataforma móvel, uma vez que este é funcional independentemente da distância e ângulo em relação ao objeto em análise.

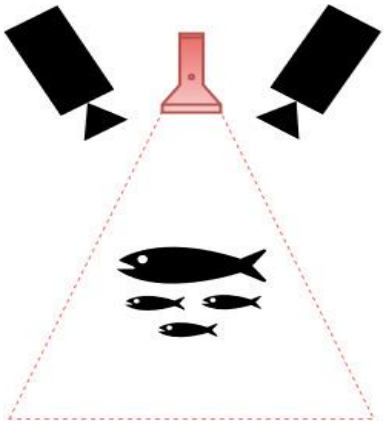


Figura 62 - *Setup* Unidade de Controlo da Biomassa

5.2.3 *Software* da UCB

O sistema operativo utilizado no ARTIK 710 é Ubuntu, neste caso, uma versão deste sistema baseado em GNU/Linux preparada para funcionar em arquitetura ARM.

Na UCB é utilizado um servidor Node.js que efetua a gestão de transmissão e receção desta unidade e ordena o começo ou fim da aplicação de processamento de imagem. O Node.js é um ambiente de execução ou interpretador *open source*, de código JavaScript assíncrono e orientado a eventos, que permite correr numa máquina (servidor) os códigos de JavaScript normalmente executados no lado do cliente.

Na aplicação Node.js criada para a UCB são utilizados dois *packages*: o Net, para utilizar SocketTCP, e o package Socket.IO-client, para implementar a comunicação Socket.IO entre servidor e cliente.

No microprocessador ARTIK inserido na UCB, está um servidor de Node.js encarregue pela gestão da aplicação de processamento de imagem empregue nesta unidade. Este servidor está conectado a esta aplicação via SocketTCP. O processo Node.js "aguarda" pela chegada da ordem de início de aquisição de dados de biomassa, ordem proveniente da UMA que chega via Socket.IO. Chegada essa ordem, é iniciado o processo de recolha de imagens e respetivo processamento. A aplicação de processamento de imagem, desenvolvida recorrendo à biblioteca MVTec HALCON, efetua a aquisição de imagens através das câmaras e efetua a análise destas, de forma a obter dados de biomassa dos peixes. É obtido um número de imagens de peixes previamente determinado e é enviada para o processo de Node.js, via SocketTCP, a massa média, mínima e máxima dos peixes

analisados. Estes dados são colocados no formato JSON e enviados via Socket.IO para o processo Node.js na UMA, que reencaminha estes dados para a UCA por LoRa.

O DHCP (*Dynamic Host Configuration Protocol*) é um protocolo utilizado para gerir de forma rápida, automática e centralizada a atribuição de endereços IP numa rede. O DHCP permite que o endereço IP dos dispositivos ligados a uma rede sejam definidos assim que estes se ligam à rede. A escolha do IP a ser atribuído a cada máquina pode ser manual, automática ou dinâmica.

O microprocessador da UCB é também utilizado como servidor DHCP, designando o endereço IP de todos os dispositivos ligados via Ethernet. Para atribuição dos IPs foi utilizado um servidor DHCP (*isc-dhcp-server*). Este foi configurado com a gama de IPs 192.168.1.x e está configurado para fornecer IPs de 192.168.1.1 até 192.168.1.254, sendo o 192.168.1.1 o *gateway*.

O processamento de imagem associado ao sistema de visão artificial desenvolvido tem a possibilidade de ser executado num microprocessador. Foi escolhido o microprocessador ARTIK 710, pois este cumpre as especificações necessárias para executar os programas desenvolvidos com a plataforma HALCON e suporta a ligação Gigabit Ethernet através da eletrónica que foi desenvolvida neste projeto, que é o meio de comunicação com as câmaras utilizadas. O HALCON necessita de uma licença única, um *dongle* USB, associada a cada dispositivo que tenha de processar programas desenvolvidos com esta ferramenta.

5.2.4 Algoritmo de Processamento de Imagem UCB

Neste subcapítulo, é descrito o processo de desenvolvimento do algoritmo de processamento de imagem da Unidade de Controlo da Biomassa.

Tal como referido anteriormente, no capítulo 3.2. Princípios da Visão Estéreo e 3D, para o desenvolvimento deste algoritmo, recorreu-se à biblioteca de *software* de visão por computador MVTec HALCON. O *software* HALCON é uma ferramenta de referência no âmbito da visão artificial. Este *software* está preparado para trabalhar tanto em Windows como Linux e OSX, é suportado em sistema embebidos com arquitetura armv7a- Linux e oferece suporte para câmaras e *frame grabbers* de vários fabricantes. O HALCON IDE permite o desenvolvimento de algoritmos recorrendo à biblioteca de operadores e funções de processamento de imagem do HALCON, incluindo algumas ferramentas que permitem acelerar o processo de desenvolvimento deste tipo de *software*. Estas ferramentas incluem vários assistentes, como o assistente de aquisição de imagem, o assistente de calibração, o assistente de *matching* (correspondência), entre outros. Através deste IDE, é possível exportar os códigos desenvolvidos para outras linguagens de programação como C, C++, C# e Visual Basic.

Com este algoritmo, pretende-se obter informação tridimensional de objetos, neste caso de peixes, analisando imagens obtidas por um par estéreo de câmaras. Com os dados relativos à dimensão de cada peixe, nomeadamente o comprimento do peixe, é efetuada a estimativa da massa do indivíduo analisado, através de uma das fórmulas referidas na revisão do estado da arte apresentada no subcapítulo Métodos de Medição e Estimativa da Biomassa em Aquacultura.

O início do algoritmo passa pela calibração das câmaras e do sistema estéreo. Cada vez que é executado este programa, é necessário obter os dados de calibração. O método mais célere consiste em adquirir previamente as imagens de calibração, seguindo todos os cuidados e conselhos apresentados no subcapítulo Obtenção de Imagens de Calibração. Uma vez obtidas estas imagens, procede-se à calibração, com o intuito de gerar os ficheiros de calibração, ou seja, o ficheiro dos parâmetros internos de cada uma das câmaras, esquerda e direita, e o ficheiro com a descrição do posicionamento relativo entre câmaras.

Tal como descrito anteriormente, no processo de obtenção de dimensões de objetos com um sistema de câmara em estéreo, a calibração é uma das fases cruciais, uma vez que influencia diretamente a precisão e qualidade dos resultados obtidos. No desenvolvimento desta solução, esta fase do processo foi uma das quais se prestou mais atenção.

Procederam-se a várias tentativas de calibração de forma a minimizar o erro de calibração: foram testados os dois tipos de placa de calibração do HALCON de forma a perceber qual permite obter melhores resultados nesta situação; experienciaram-se alguns ajustes nos ângulos entre câmaras de forma a perceber a melhor posição relativa possível entre estas e procederam-se, ainda, a vários ajustes da iluminação utilizada. Os resultados de calibração obtidos não terão sido os mais promissores, uma vez que o erro de calibração obtido foi de 0,3 pixel, valor superior ao aconselhado. Este valor do erro poderá ser motivado por uma inadequação da iluminação utilizada e, ainda, pela utilização de placas impressas em papel colado em placas de acrílico, sendo o papel um material não ideal, uma vez que, com o tempo, se vai sujando introduzindo ruído nas imagens.

No Anexo 1 – Algoritmo do Projeto Aqautropolis para Calibração do Par de Câmaras Estéreo, é apresentado o algoritmo definido para executar a calibração do sistema utilizado. Recorre-se a um algoritmo de calibração que está separado do algoritmo principal de medição estéreo, de forma a determinar os parâmetros de calibração antes de executar as citadas medições, guardando-os em ficheiros de calibração. Desta forma, o algoritmo principal tem menor complexidade no que à calibração diz respeito, apenas sendo necessário ler tais ficheiros.

Este algoritmo de calibração guarda as imagens mediante a interação do utilizador, através do IDE do HALCON. Assim, é necessário ir alterando a posição da placa, seguindo os conselhos relativos à calibração referidos anteriormente e através deste *software* o utilizador valida as imagens. Antes de ser guardada, a imagem é verificada pelo operador *find_calib_object*, que procura a placa de calibração na imagem e guarda a informação da posição da placa associada a um índice no modelo de calibração.

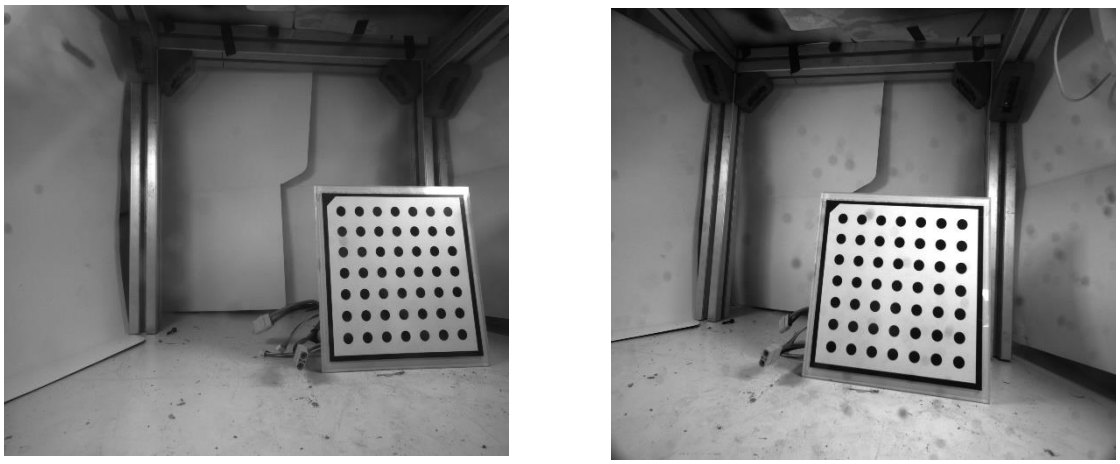


Figura 63 - Exemplo de um par de imagens de calibração (A imagem esquerda corresponde à câmara esquerda e a imagem direita corresponde à câmara direita)

Neste algoritmo, são obtidas vinte imagens de calibração para cada câmara, ou seja, vinte pares de imagens com a placa de calibração vista pelas duas câmaras. A Figura 63 mostra um destes pares de imagens. Após obtidas e analisadas todas as imagens é executada calibração com o operador *calibrate_cameras*, que permite obter o valor dos erros de calibração. Antes de serem guardadas em ficheiros todas as informações de calibração, é efetuada a correção da *pose* do sistema de câmaras, considerando a espessura da placa que, no caso desta aplicação, é de 6,3 mm. Finalmente, são guardadas todas as informações necessárias relativas à calibração em ficheiros: um ficheiro para os parâmetros internos de cada câmara e um ficheiro para a *pose* das câmaras.

Uma vez efetuada a calibração, que permite obter os ficheiros de calibração, é possível obter imagens com o sistema de câmaras em estéreo já calibrado.

No Anexo 2 – Algoritmo Projeto Aquatropolis para Medição de Peixes e Determinação da Respetiva Massa, é demonstrado o código do algoritmo desenvolvido para estimar a massa de um peixe a partir do seu comprimento. Neste programa de medição, o *error handling* ou gestão de erros é efetuada com base nos códigos de erro do HALCON, permitindo verificar que as etapas cruciais, como as conexões com as câmaras ou com o SocketTCP, foram bem sucedidas.

O início do programa de medição é ordenado pelo servidor Node.js a correr no microcontrolador da UCB. Este servidor recebe a informação da massa do peixe medido via socketTCP. O socketTCP é aberto pelo servidor e o algoritmo de processamento de imagem conecta-se a este *socket* utilizando o operador HALCON *open_socket_connect*:

open_socket_connect (HostName, Port, GenParamName, GenParamValue, Socket)

Neste operador, é indicado o *HostName* e *Port* que correspondem respetivamente ao endereço IP e à porta com a qual é efetuada a conexão TCP. Nos parâmetros *GenParamName* e *GenParamValue*, é indicado o *timeout* da conexão e o protocolo a utilizar, neste caso TCP. A variável Socket é o identificador desta conexão para utilização posterior neste programa.

Antes de iniciar a aquisição de imagens com este sistema, é efetuada a leitura dos parâmetros de calibração a partir dos ficheiros criados no programa de calibração. A leitura é efetuada com o operador *read_cam_par*, para os parâmetros internos das câmaras, ou seja, este operador é utilizado duas vezes, uma por cada câmara. Para a leitura da *pose* das câmaras deste sistema é utilizado o operador *read_pose*. Uma vez lida esta informação relativa à calibração, são criados os mapas de retificação para as imagens obtidas por cada uma das câmaras. Estes mapas são dois dos parâmetros de saída do operador *gen_binocular_rectification_map*.

Para iniciar a aquisição de imagens, é necessário efetuar a conexão às câmaras, de modo a que estejam preparadas para receber ordem de aquisição por parte deste programa. Para abrir esta conexão a cada câmara é utilizado o operador open_framegrabber.

open_framegrabber (Name, HorizontalResolution, VerticalResolution, ImageWidth, ImageHeight, StartRow, StartColumn, Field, BitsPerChannel, ColorSpace, Generic, ExternalTrigger, CameraType, Device, Port, LineIn, AcqHandle)

O parâmetro de entrada *Name* é utilizado para definir qual o tipo de dispositivo de aquisição utilizado, estando este parâmetro associado ao DLL do Windows ou biblioteca partilhada do Linux que é utilizado para executar a conexão com a câmara. Para as câmaras utilizadas utiliza-se o DLL ou biblioteca partilhada GigEVision2. Os parâmetros *HorizontalResolution* e *VerticalResolution* permitem definir a resolução da imagem adquirida. Por defeito, utiliza-se o valor 1 nestes dois parâmetros para ter a resolução

máxima permitida pela câmara. O tamanho da imagem adquirida é definido pelos parâmetros *ImageWidth* e *ImageHeight.* Por defeito, utiliza-se o valor 0 para os dois parâmetros, que resulta em imagens com maior tamanho possível. Neste operador, os parâmetros *StartRow* e *StartColumn* definem, respetivamente, as coordenadas de linha e coluna correspondente ao canto superior esquerdo da imagem a obter e, por defeito, utiliza-se para os dois o valor 0. Caso se pretenda obter apenas metade da imagem da câmara, define-se qual a metade da imagem pretendida no parâmetro *Field*. Para obter a imagem inteira coloca-se neste parâmetro o valor "*default*". O parâmetro *BitsPerChannel* permite definir o número de bits por canal da imagem obtida, e neste programa é utilizado o valor "-1", sendo o número de bits definido pela câmara com base nas suas especificações. As câmaras utilizadas permitem obter imagem com 8 e 12 bits por canal, utilizando 8 bits por defeito. As câmaras utilizadas apenas permitem obter imagens a preto e branco, sendo o parâmetro *ColorSpace* definido como "*default*". Caso se utilizasse uma câmara a cores, poderia utilizar-se outro valor como "RGB", dependendo do tipo de câmara. No parâmetro *Device* é introduzido o ID da câmara, diferente para todas as câmaras. O parâmetro *AcqHandle* é uma variável utilizada para referenciar no programa a câmara com que se pretende estabelecer conexão, tendo de ter um nome único e diferente para câmara.

Ao obter um par de imagens com câmaras numa montagem estéreo é necessário garantir que ambas as imagens representam exatamente o mesmo cenário. Na aplicação desenvolvida, em que é pretendido obter imagens de peixes que estão em constante movimento, este detalhe é ainda mais importante, de forma a garantir os resultados esperados. Assim recorre-se à sincronização das câmaras.

A sincronização das câmaras é efetuada configurando uma das câmaras como mestre/*master* e a outra como escravo/*slave*. A câmara mestre recebe a ordem de aquisição de imagem e envia esta mesma ordem para a segunda câmara, que está constantemente a aguardar a ordem de aquisição de imagem por parte da câmara *master*. As ligações do cabo de alimentação e de entradas e saídas são ilustradas na Figura 64.

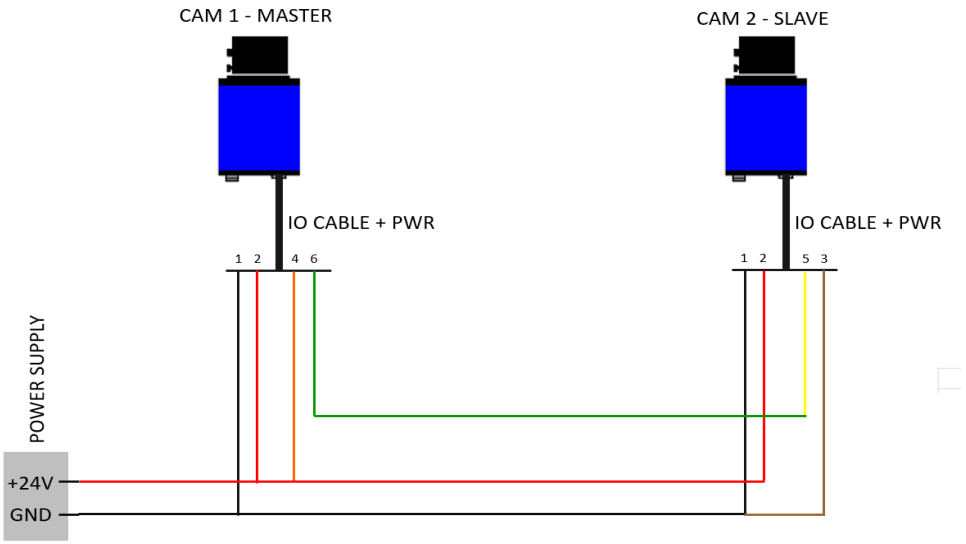


Figura 64 - Configuração da sincronização das câmaras

Depois de ligada a alimentação de ambas as câmaras, o terminal 6 da primeira câmara (mestre), correspondente ao condutor de cor verde, é ligado ao terminal 5 da segunda câmara (escravo), de cor amarela. Na primeira câmara, é ligada a alimentação das portas de saída, o terminal 4, de cor laranja, ao terminal 2, de cor vermelha, que recebe a tensão vinda da fonte de alimentação. O terminal comum das portas de saída, terminal 3, de cor castanha, é conectado à massa da alimentação das câmaras, o terminal 1, de cor preta.

Relativamente às configurações de *software*, na câmara *master* é necessário garantir que os *triggers* externos estão desativados, colocando o *Trigger Mode* desligado (OFF). É selecionada a porta de saída para enviar o sinal para a segunda câmara, colocando o *Line Selector* com o parâmetro *Line* 3. É ainda necessário definir quando é enviado o *trigger* para a segunda câmara, colocando o *Output Line Source* com o parâmetro *Pulse on: Start of Frame*, que coloca a câmara mestre a enviar o sinal para a câmara escravo no início de cada *frame* adquirido. Para garantir que o pulso enviado é detetado, o *Output Line Source* da câmara mestre é colocado a um valor mínimo de 500 microssegundos.

Na segunda câmara, é necessário ativar o *Trigger Mode*, ativado (ON). Seleciona-se o terminal 5 como entrada do sinal de *Trigger*, com o *Trigger Source* e o *Line Selector* com o parâmetro *Line* 1. Finalmente, configura-se o *delay* mínimo reconhecido como sinal da transição da tensão de entrada, *Input Line Debouncing Period*, ao qual foi atribuído o valor de 255 unidades.

O código de configuração típico para esta sincronização é:

```
open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137694', 0, -1, AcqHandle1)
set_framegrabber_param (AcqHandle1, 'ExposureTime', 800.0)
set_framegrabber_param (AcqHandle1, 'TriggerMode', 'On')
set_framegrabber_param (AcqHandle1, 'LineSelector', 'Line1')
set_framegrabber_param (AcqHandle1, 'lineDebouncingPeriod', 255)
open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137535', 0, -1, AcqHandle2)
set_framegrabber_param (AcqHandle2, 'ExposureTime', 800.0)
set_framegrabber_param (AcqHandle2, 'TriggerMode', 'Off')
set_framegrabber_param (AcqHandle2, 'LineSelector', 'Line3')
set_framegrabber_param (AcqHandle2, 'outputLineSource', 'PulseOnStartofFrame')
set_framegrabber_param (AcqHandle2, 'outputLinePulseDuration', 500)
```

Caso a utilização do operador *open_framegrabber* falhe a conexão à câmara, obtém-se um código de erro, neste caso o código 5312 na biblioteca do HALCON. Assim, é verificado se a ligação à câmara foi efetuada com sucesso, caso contrário, é efetuada uma nova tentativa de conexão.

A conexão às câmaras e a definição dos parâmetros de funcionamento é terminada com o operador *grab_image_start* que dá início à aquisição assíncrona de imagens. O operador é utilizado duas vezes, uma para cada câmara. A aquisição de imagens propriamente dita é efetuada dentro de um ciclo infinito. Recorrendo ao operado *grab_image_async* obtém-se uma imagem, para cada câmara referenciada pela variável de identificação da câmara *AcqHandle*. Este ciclo é interrompido quando o servidor Node.js, executado em paralelo, ordena a paragem do programa de processamento de imagem. Após a aquisição, as imagens são retificadas com base no mapa de retificação obtido para cada câmara recorrendo ao operador *map_image*. Nas figuras abaixo, é demonstrada a sequência de retificação das imagens.

Figura 65 - Imagens obtidas em cenário de testes (Imagem esquerda corresponde à câmara esquerda e imagem da direita à câmara direita)

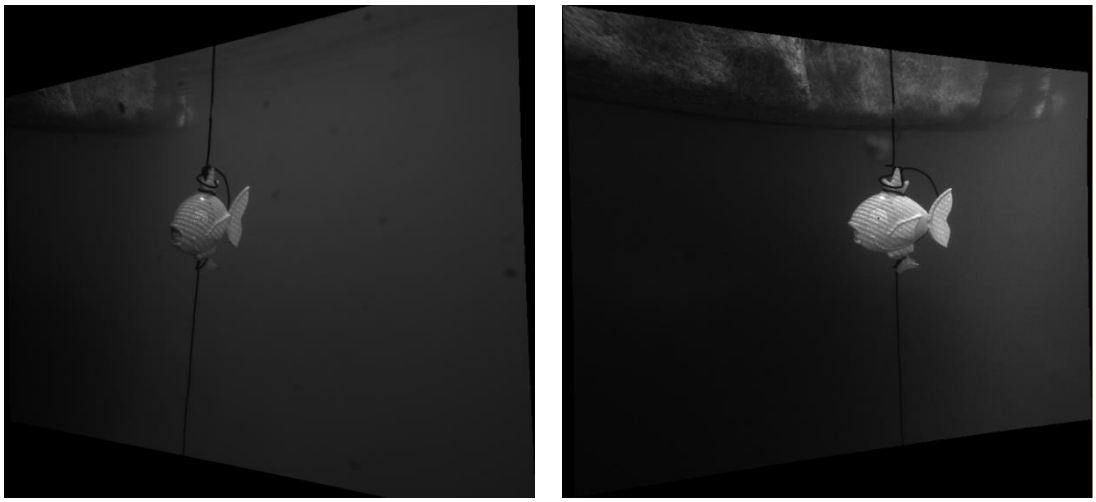


Figura 66 - Imagens da Figura 19 retificadas

Com as imagens retificadas, é utilizada a imagem da câmara esquerda para segmentar a região que deverá corresponder à área ocupada por um peixe ou peixes na imagem. Não se trata de um processo de identificação de peixe, mas apenas de uma segmentação com base na área ocupada por peixes na imagem, assumida com base nos testes efetuados. De seguida, é apresentado o excerto de código correspondente a esta seleção:

```
threshold (ImageRectifiedL, Region, 70, 195)
fill_up (Region, RegionFillUp)
dilation_circle (RegionFillUp, RegionDilation, 60)
connection (RegionDilation, ConnectedRegions)
select_shape (ConnectedRegions, SelectedRegions, 'area', 'and', 70957.6, 350000)
```

union1(SelectedRegions, RegionUnion)
shape_trans (RegionUnion, RegionTrans, 'convex')
area_center (RegionTrans, Area, Row, Column)

Esta segmentação inicia-se com a seleção de um intervalo de valores de cinzento que corresponderem a um corpo presente na imagem, distinto do plano de fundo da imagem. Para tal, utiliza-se o operador *threshold*, que executa a seleção de uma região dentro de um limite de valores de cinzento. A região obtida, que poderá conter várias áreas distintas e separadas, é de seguida preenchida com recurso ao operador *fill_up*, de forma a eliminar pequenas falhas ou aberturas na região. De seguida, é utilizado o operador *dillatation_circle*, que permite dilatar um pouco esta região, de forma a abranger mais alguns pixéis de interesse que tenham sido "ignorados" pela seleção por valores de cinzento. O operador *connection* é de seguida utilizado de forma a unir as áreas com os mesmos valores de cinzento, criando uma ou mais regiões. De forma a selecionar a região com a área dentro do expectável para um peixe nas condições em que este algoritmo foi testado e desenvolvido, utiliza-se o operador *select_shape*. Neste operador, indica-se qual a característica pretendida para fazer a seleção, neste caso a área, e o intervalo de valores da seleção. Neste caso, introduz-se o valor mínimo e máximo da área em pixéis da região. Assim, são selecionadas todas as regiões que têm o valor da área dentro deste intervalo e são unidas num só objeto com o operador *union1*. Finalmente, o resultado desta união, que será uma região com uma forma variável, é transformada numa região em forma convexa, de modo a simplificar as arestas e curvas do contorno existente. A região obtida é utilizada posteriormente para seccionar a imagem de disparidades a obter.

Figura 67 - Exemplo de definição de região de interesse

A obtenção da imagem de disparidades só se inicia caso a área da região anterior seja maior que zero, ou seja, se no processamento efetuado até ao momento tiver sido detetado algum objeto. Caso contrário, são adquiridas novas imagens para análise. Tal como demonstrado no excerto, o valor da área da região selecionada é determinado recorrendo ao operador *area_center* que, para além deste valor, determina ainda as coordenadas em valores de linha e coluna do centro de cada uma das regiões selecionadas. A imagem de disparidades é obtida a partir do par de imagens previamente retificadas. Ao introduzir estas imagens como parâmetros de entrada no operador *binocular_disparity*, é possível obter a imagem de disparidades e a imagem de *score* das disparidades. Na Figura 68, são demonstradas as imagens de disparidades e de *score*, relativas ao par de imagens retificadas apresentadas na Figura 66.

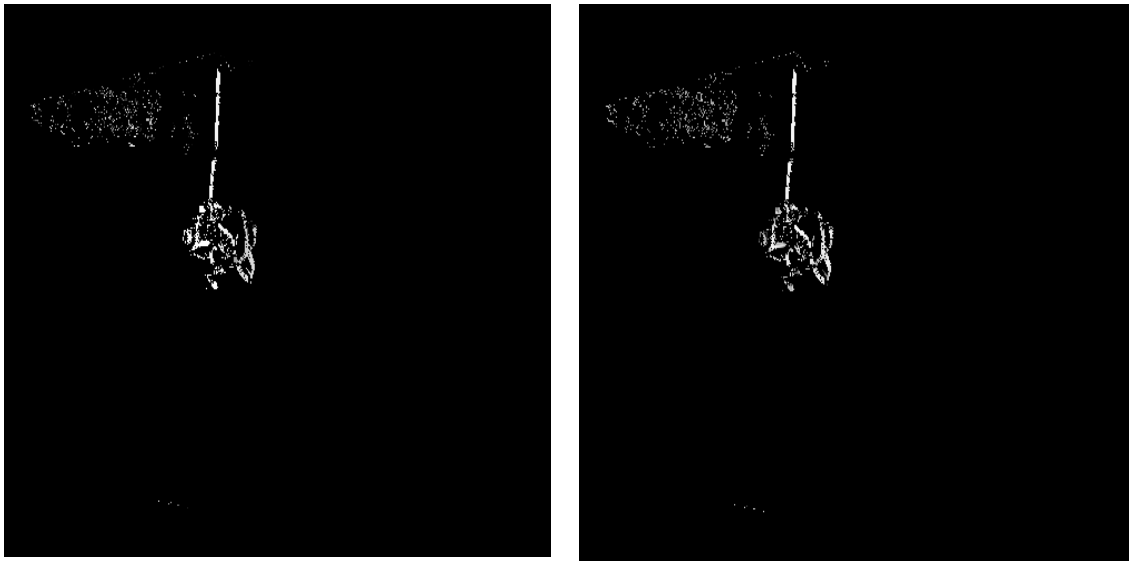


Figura 68 - Imagem de disparidades (esquerda) e imagens de *score* (direita)

A região previamente selecionada é utilizada para reduzir a área de análise da imagem de disparidades obtida, recorrendo ao operador *reduce_domain.* As zonas sem informação de disparidade são apresentadas a preto pois não apresentam textura ou por corresponderem a zonas não cobertas pelo campo de visão de ambas as câmaras. No entanto, para definir corretamente as imagens com os valores de coordenadas a partir da disparidade, é necessário executar uma interpolação de valores de cinzento, que facilitam a definição de tais coordenadas. Isto é efetuado recorrendo ao operador *harmonic_interplation* que atribui valores de cinzento a zonas que não tinham tal informação.

Com esta imagem de disparidade com valores de cinzento interpolados, é possível extrair as coordenadas de *x*, *y* e *z* do peixe ou objeto segmentado. Esta tarefa é executada com a função *disparity_image_to_xyz:*

disparity_image_to_xyz(Disparity, X, Y, Z, CamParamRect1, CamParamRect2, RelPoseRect).

Introduzindo a imagem de *Disparity*, os parâmetros internos retificados de cada câmara (*CamParamRect1*, *CamParamRect2*) e a *pose* do par de câmaras após retificação (*RelPoseRect)*, a função permite obter três imagens que representam, respetivamente, as coordenadas de *x*, *y* e *z* em valores de cinzento.

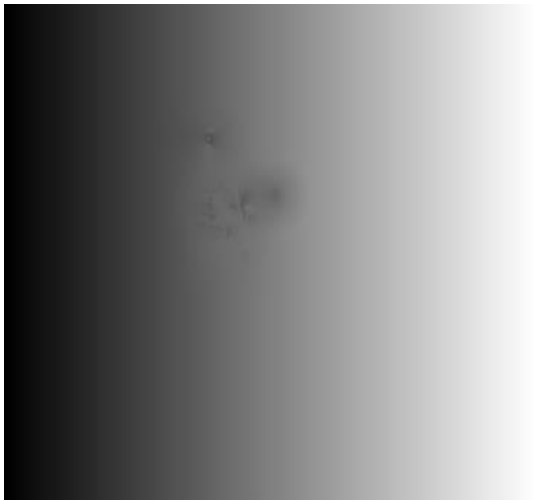


Figura 69 - Imagem de coordenadas *x*

Figura 70 - Imagem de coordenadas *y*

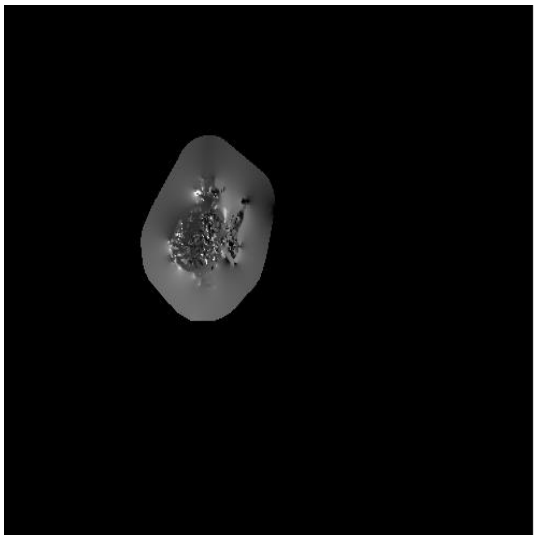


Figura 71 - Imagem de coordenadas *z*

Para obter as dimensões do objeto em análise nas imagens, transformam-se as coordenadas obtidas num modelo 3D, utilizando o operador *xyz_attrib_to_object_model_3d*. A partir do modelo 3D obtido, são aferidas as dimensões aproximadas deste objeto, recorrendo a um operador que permite obter a "caixa envolvente" que mais se adequa ao objeto, de tal forma que as dimensões desta "caixa" são as dimensões aproximadas do objeto do qual se obteve o modelo 3D. Na Figura 72, são apresentadas as imagens que ilustram este conceito.

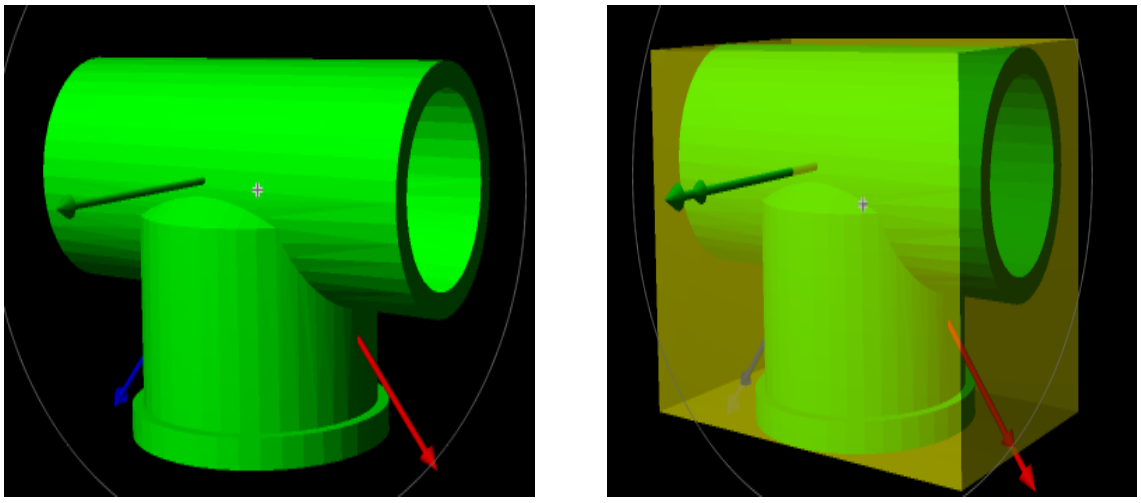


Figura 72 - Exemplo de obtenção da "caixa envolvente" [39]

A Figura 73 é o resultado obtido com as imagens de coordenadas apresentadas anteriormente. O modelo 3D obtido não é muito nítido. Isto poderá estar relacionada com vários fatores, sendo o principal fator o facto de as imagens adquiridas não estarem nas mesmas condições de luminosidade que as imagens obtidas em calibração. A presença de alguma turbidez na água poderá igualmente contribuir negativamente para este resultado.

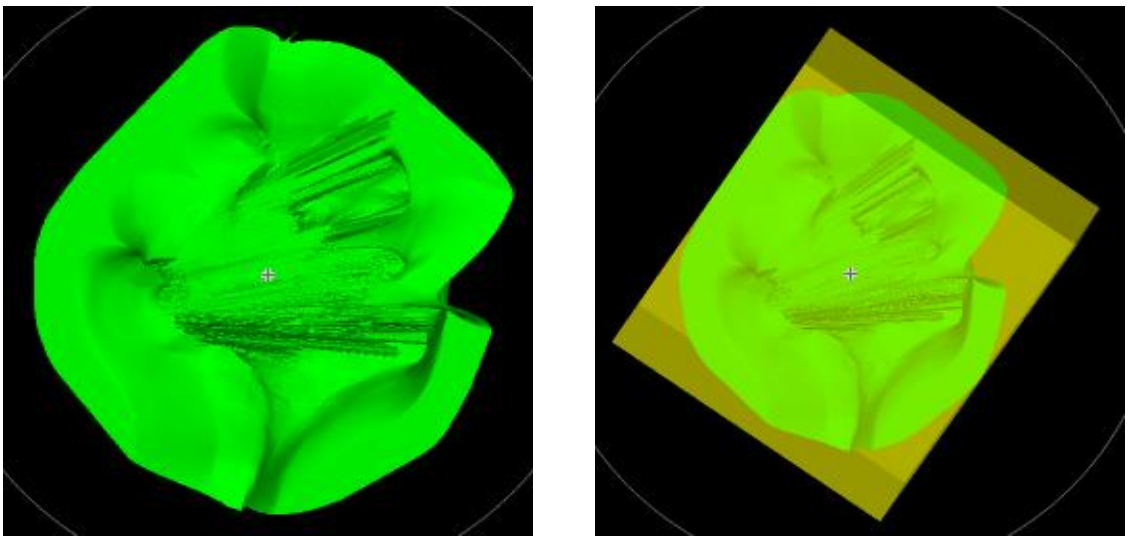


Figura 73 - Modelo 3D obtido e respetiva "caixa" ou objeto envolvente

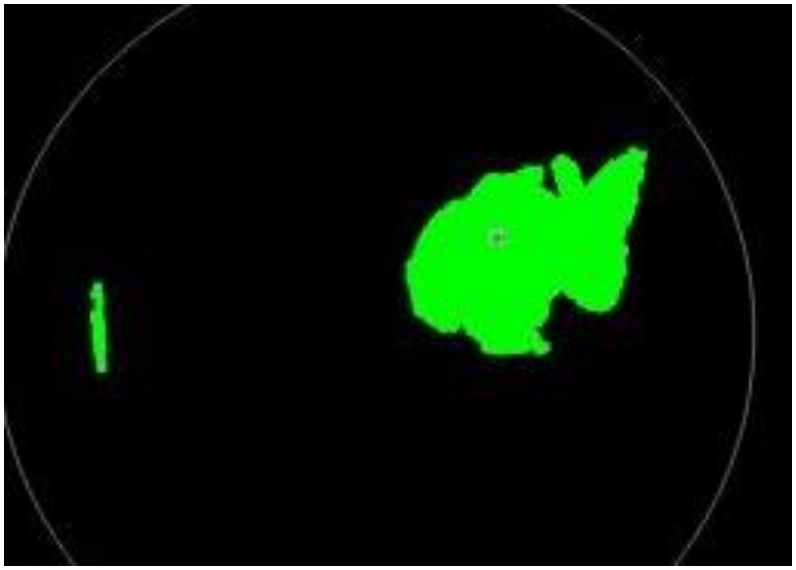


Figura 74 - Modelo 3D do peixe (inclui outro objeto associado a ruído na imagem)

A obtenção da citada "caixa" é efetuada recorrendo ao operador *smallest_bounding_box_object_model_3d*, que tem como variáveis de entrada o objeto 3D (*ObjectModel3D)* e o método utilizado para determinar a "caixa envolvente" (*Type)*:

smallest_bounding_box_object_model_3d (ObjectModel3D, Type, Pose, Length1, Length2, Length3).

A "caixa" pode ser obtida ficando alinhada com eixos coordenados, em que o paralelepípedo resultante é alinhado com os eixos, tendo como argumento do *Type, "axis aligned".* Tendo em conta os comprimentos dos lados, ficará o maior lado alinhado com o eixo *x*, o segundo maior alinhado com o eixo *y* e o lado mais pequeno alinhado com o eixo *z*. Esta caixa pode ficar também orientada aleatoriamente, utilizando neste caso o argumento *"oriented".* A função devolve três medidas, correspondente às dimensões do objeto, *Length1*, *Length2*, *Length3*, e a *pose* da "caixa" obtida, "*pose*". Entre os três valores correspondentes às dimensões do sólido envolvente do modelo 3D do peixe, considera-se o maior valor obtido como sendo o comprimento aproximado do peixe ou objeto em análise.

Dos valores de comprimento da "caixa envolvente" do peixe em análise, seleciona-se o maior valor, que corresponderá ao comprimento aproximando do peixe. Para estimar a massa do peixe em gramas, é utilizada uma das fórmulas referidas na revisão do estado da arte apresentada anteriormente e que foi igualmente sugerida pelos técnicos do Instituto Politécnico de Leiria, parceiros no projeto Aquatropolis. A fórmula relaciona o comprimento com o peso do peixe, onde W representa a massa em gramas, L o comprimento do peixe em centímetros e os valores a e b são coeficientes associados a cada espécie de peixe em estudo, no caso deste projeto, dourada e robalo.

$$W = a \times L^b \qquad \text{Equação 6}$$

Na Tabela 7, encontram-se representados os valores para os parâmetros *a* e *b* para as espécies em causa. Os valores apresentados foram indicados pelos técnicos do Instituto Politécnico de Leiria.

Tabela 7 - Coeficiente *a* e *b* para cada espécie de peixe

	Dourada (6,6 a 37,3 cm)	**Robalo (7,0 a 47,9 cm)**
a	0,01311	0,01090
b	3,040	2,980

Os valores do peso do peixe são enviados para a UMA em formato JSON via socketTCP. A informação a enviar é passada para uma *string, s,* no formato JSON, sendo apresentada da seguinte forma:

s = {"weight":W}

O envio do JSON com a massa é efetuado utilizando o operador *send_data.* Este operador tem como argumentos a variável *Socket*, onde é indicada a referência do socketTCP previamente estabelecido, a variável *Format* onde é indicado o tipo de dados a enviar, que neste caso será uma *string*, e, portanto, introduz-se o argumento *"z"*, permitindo o envio de uma *string* com comprimento variável. Caso se quisesse, por exemplo, enviar um valor do tipo inteiro (*int*) utilizar-se-ia o argumento "*c*" que permite enviar um *signed int* (8 bytes). A informação a enviar é colocada no parâmetro *Data*. O parâmetro *To* é o endereço para onde se pretende enviar a informação, no entanto como tal informação já foi definida no momento da abertura do socket, pode deixar-se este campo vazio.

```
send_data (Socket, Format, Data, To)
```

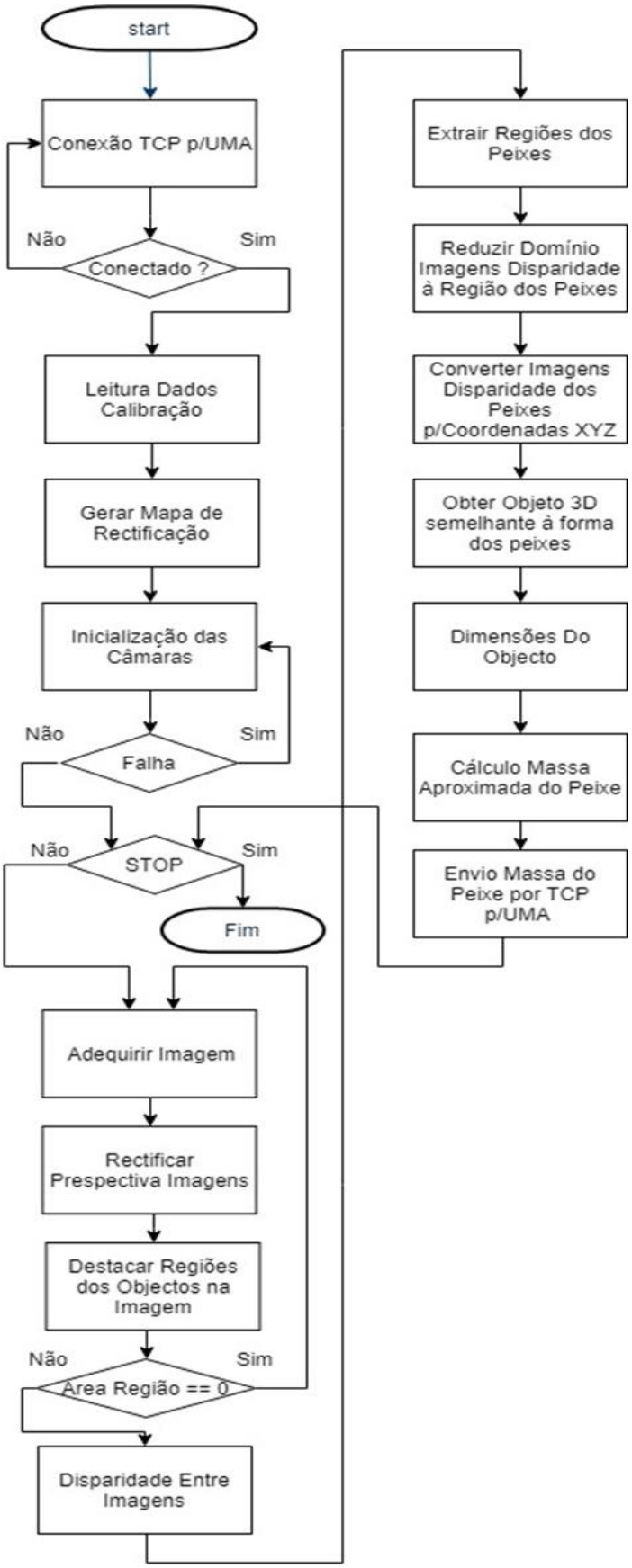


Figura 75 - Fluxograma do algoritmo de processamento de imagem desenvolvido

Durante o processo de desenvolvimento, utilizaram-se várias abordagens para testar o algoritmo, compreender as suas capacidades e como definir determinadas características. Uma das primeiras abordagens de teste do algoritmo foi a utilização de um objeto de teste com formas simples para comprovar o funcionamento do algoritmo para imagens com este tipo de objetos. Um destes objetos era um pequeno disco com 4,3 centímetros de diâmetro e aproximadamente 2 milímetros de espessura, utilizado para determinar se as coordenadas *x* e *y* do par de imagens estavam a ser aferidas corretamente. Neste teste em particular, obtiveram-se resultados muito positivos, uma vez que na grande maioria dos resultados, o desvio foi inferior a meio centímetro. Na Figura 76, são demonstradas imagens relativas a estes testes de medição recorrendo às coordenadas *x* e *y*.

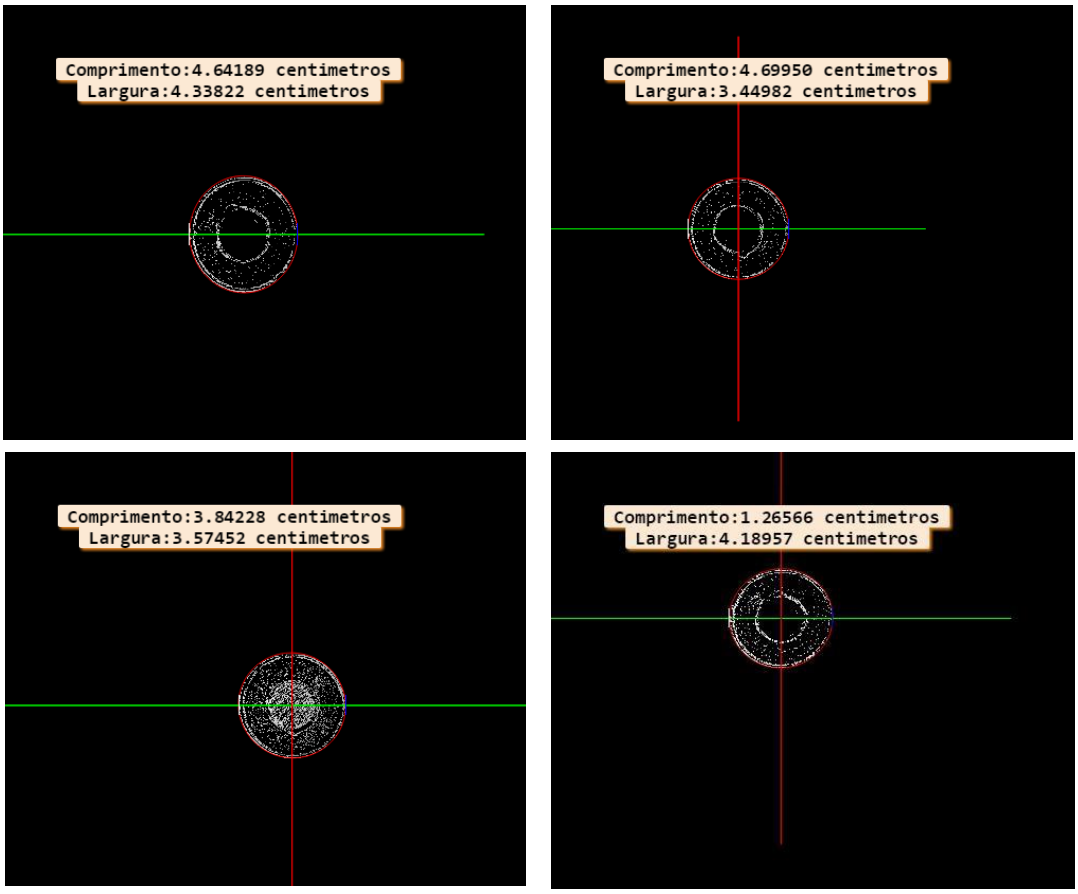


Figura 76 - Testes de medição com um disco de 4,5 cm

Com o mesmo intuito de testar a precisão com que se conseguiria aferir as coordenadas *x* e *y* de um objeto e assim obter as suas dimensões, utilizou-se um peixe feito em cartão do qual se obtiveram imagens para testar tal situação. Ao utilizar um peixe de cartão com comprimento e largura conhecidos pretendia-se também testar métodos que permitissem garantir que o algoritmo consideraria sempre o comprimento como a dimensão a utilizar

posteriormente na fórmula de estimativa da massa. Na Figura 77, são demonstradas imagens deste peixe de cartão, especificamente um par de imagens retificadas e a respetiva imagem de disparidade obtida a partir desse par.

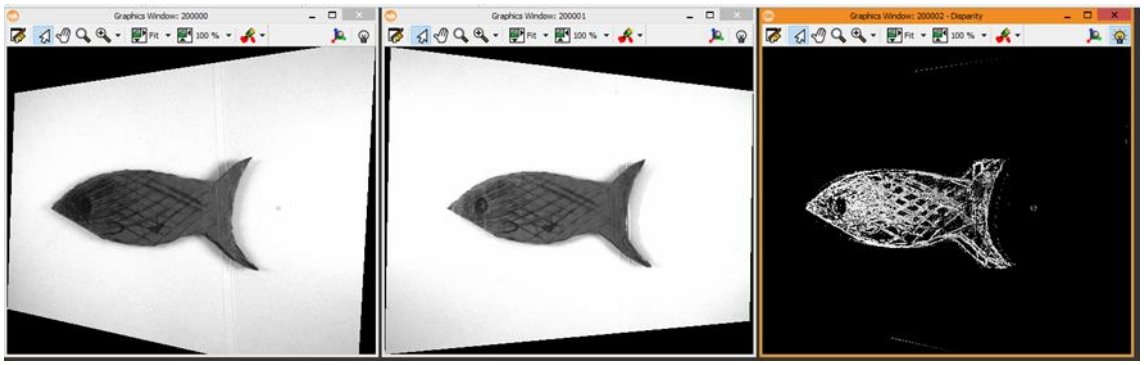


Figura 77 - Imagens retificadas e respetiva imagem de disparidade do peixe de cartão utilizado para os testes

Com este peixe de cartão, iniciaram-se os testes do sistema com o laser. A introdução deste tipo de iluminação, que apresenta um padrão de 25 linhas, permite adicionar mais textura às superfícies, facilitando a obtenção das imagens de disparidade dos objetos sobre os quais o laser incide.

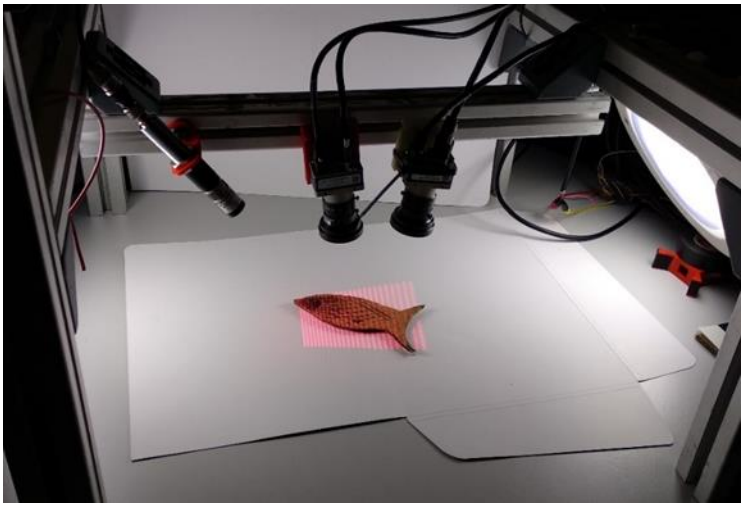


Figura 78 - Primeira montagem de testes do sistema estéreo com laser

Para além destes dois testes em laboratório, utilizou-se também uma caixa, ou seja, um paralelepípedo, de forma a testar e ajustar o processo de obtenção do modelo 3D do objeto em análise pelo sistema e o respetivo sólido envolvente. Na Figura 79, estão algumas imagens associadas a este teste.

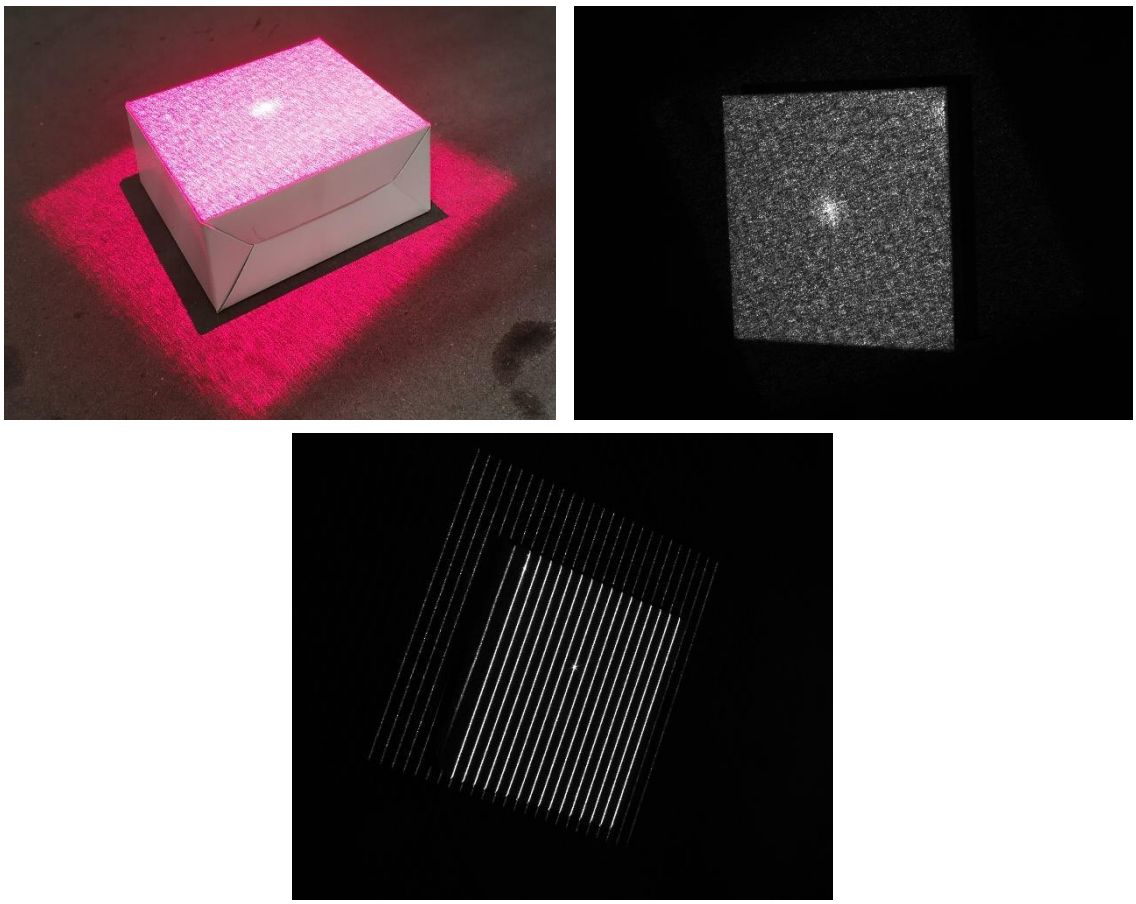


Figura 79 - Testes de obtenção do modelo 3D da caixa com projeção do padrão laser

Com este teste de obtenção do modelo 3D de uma caixa, foi possível perceber que estavam a ser cometidos alguns erros no processo de determinação da imagem de disparidade, que até então, por estar definida de forma errada, resultava num objeto 3D que se distanciava muito da representação da caixa. A correção efetuada, neste caso em particular, foi a introdução do operador HALCON *harmonic_interpolation* no algoritmo, que veio adicionar informação necessária para a definição correta das coordenadas 3D do objeto.

5.2.5 Resultados e Testes

Ao longo do desenvolvimento dos equipamentos e soluções do projeto Aquatropolis, executaram-se vários testes de campo de forma a obter conclusões sobre qual o comportamento destes sistemas em ambiente real de aquacultura semi-intensiva. Usualmente, as sequências de testes envolviam sempre os três sistemas desenvolvidos, o ROV, a UMA e a UCB.

5.2.5.1 Testes de Campo nº1

No caso específico do sistema de visão da UCB, os primeiros testes realizaram-se em várias produções de aquacultura com diferentes características que representam os cenários mais comuns neste tipo de produções. Com estes testes pretendeu-se tomar um primeiro contacto com as condições de luminosidade ambiente e perceber que efeitos esta teria nas imagens obtidas. Igualmente, pretendeu-se perceber se existiriam outros constrangimentos que pudessem afetar o sistema.

Nas primeiras ocasiões de teste, o sistema de visão artificial da UCB foi acoplado a uma plataforma isolada, ou seja, não acoplada ao ROV, possibilitando executar afinações a este sistema. O *software* de visão foi executado num PC externo, ligado por Ethernet às câmaras, permitindo visualizar as imagens em tempo real e proceder a ajustes no algoritmo e na configuração das câmaras (*frame rate*, tempo de exposição, ganho, etc.). As câmaras foram colocadas dentro de água em *housings* metálicos à prova de água, aos quais foram acopladas mangueiras que conduziam os cabos de alimentação e comunicação das câmaras para o exterior. Esta montagem é demonstrada na Figura 80.

Figura 80 - Montagem utilizada nos testes

Figura 81 - Produções de aquacultura em Castro Marim (esquerda) e Ílhavo (direita)

A plataforma com o sistema de visão foi posicionada na água e esteve continuamente a obter imagens. Ao longo dos testes, o posicionamento entre câmaras foi sendo alterado e foram testados vários tipos de configurações de *software*. Os testes foram realizados em períodos diurnos e noturnos para se perceber a influência da iluminação externa no resultado das imagens. Neste aspeto, fizeram-se alguns testes, orientando as câmaras contra e a favor do sol, de forma aferir as limitações existentes ao trabalhar com o sistema durante o dia. Experimentaram-se várias localizações do sistema de câmaras, de modo a tentar aproximar o sistema de zonas com maior concentração de peixes, afastando e aproximando de zonas de entrada e saída de água nos tanques e zonas de alimentação dos peixes.

5.2.5.1.1 Conclusões dos Testes de Campo nº1

Nos testes efetuados, verificou-se que a iluminação solar afeta consideravelmente a obtenção das imagens. A presença da luz solar produz brilhos indesejados e uma luz não homogénea, dificultando consideravelmente a segmentação de quaisquer corpos presentes nas imagens. As imagens subaquáticas obtidas durante o dia apresentaram um grande clarão nas zonas mais próximas do cimo da água.

Em períodos do dia com menor luminosidade ou à noite, em situações em que seja possível evitar a presença de luz indesejada, poderão obter-se melhores resultados. No entanto, será sempre necessário garantir uma iluminação homogénea. Idealmente, a

iluminação deverá ser configurada de forma que as imagens obtidas estejam em condições de luminosidade bastante semelhantes às imagens de calibração que se utilizarem. Só neste cenário se pode garantir sucesso nas medições obtidas com este sistema.

Efetuaram-se alguns testes com a iluminação LED que equipa o ROV, possibilitando algum controlo sobre a luminosidade. No entanto, apesar de se ter revelado útil, o modo como se utilizou esta propriedade não foi o mais correto, sendo que em determinadas situações piorou as condições de imagem. A utilização desta iluminação em águas com alguma turbidez produz um efeito negativo, uma vez que amplifica o efeito de dispersão da luminosidade da imagem causado pelas partículas em suspensão.

Para utilizar este tipo de iluminação, seria necessário recorrer a uma estrutura separada acoplada ao sistema de visão, que suportasse um sistema de iluminação numa posição que não causasse problemas ao sistema. Este sistema de iluminação teria de ter vários focos de luz, de modo a permitir a homogeneização da iluminação na imagem. Contudo, ao utilizar um sistema deste tipo, seria difícil a sua colocação numa plataforma móvel e, muito provavelmente, afugentaria os peixes.

O facto de se trabalhar com um nível de iluminação reduzido dificulta a obtenção das imagens dos peixes. Com base nesta dificuldade procurou desenvolver-se no algoritmo um mecanismo que permita efetuar a compensação da luminosidade. Para além disso, chegou-se à conclusão que poderia ser benéfico recorrer a uma luz laser com padrões em linha, que permite realçar os contornos dos peixes. Um foco de iluminação laser é facilmente acoplado ao sistema de câmaras, não comprometendo a possibilidade de aplicar o sistema de câmaras e luz laser ao veículo móvel robotizado a desenvolver.

Nestes testes, com este sistema de visão, verificou-se também que a turbidez é um fator limitante no que a obtenção de imagens diz respeito. Turbidez elevada provoca a dispersão da luz existente e limita a distância máxima do campo de visão.

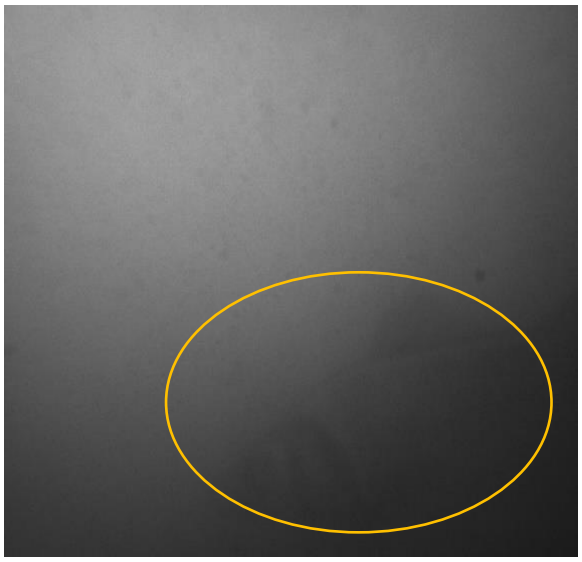


Figura 82 - Testes em tanques de aquacultura (Atlantik Fish, Castro Marim, maio 2018)

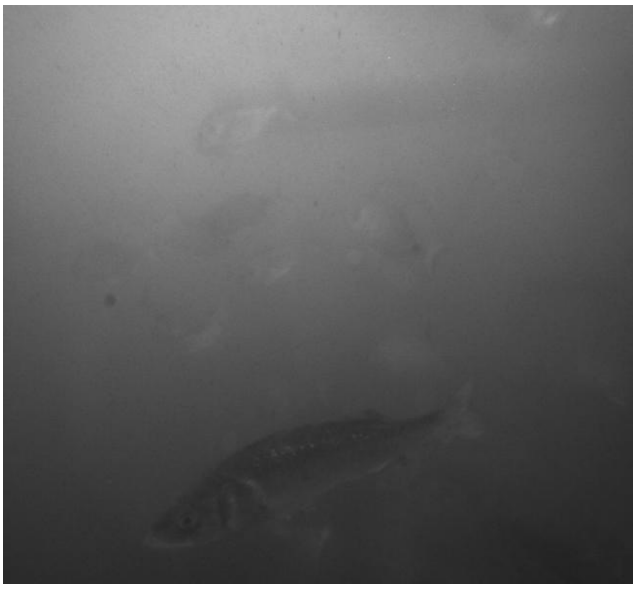


Figura 83 - Imagem obtida na zona de tomada de água (Aqualvor, Alvor, maio 2018)

5.2.5.2 Testes de Campo nº2

Após a realização dos primeiros testes, decidiu-se que a sessão seguinte deveria ser num ambiente mais controlado. Assim, esta segunda sessão de testes foi realizada em Tomar, na albufeira de Castelo de Bode e também numa piscina com 30 metros de comprimento e 10 metros de largura. Executaram-se testes com todos os sistemas tal como no último cenário. O sistema de câmaras manteve a estrutura utilizada nos testes anteriores.

A plataforma com o sistema de câmaras foi colocada dentro da piscina, tendo sido obtidas imagens, permitindo testar a utilização de tal equipamento em condições sem turbidez visível. Neste cenário, efetuaram-se testes mais conclusivos com a iluminação laser com um padrão de 25 linhas para dar realce aos contornos. Uma vez que o objetivo do projeto é analisar peixes, foi utilizado um peixe de plástico para realizar a obtenção das imagens. Na albufeira, efetuou-se o mesmo tipo de testes, porém, neste caso verificou-se alguma turbidez.

Figura 84 - Imagem obtida nos testes em piscina

Figura 85 - Imagem obtida nos testes na albufeira/rio

5.2.5.2.1 Conclusões dos Testes de Campo nº2

Em piscina, visto não haver turbidez, existe uma boa visibilidade. Assim, nas imagens obtidas pelo sistema de visão que foi colocado numa extremidade da piscina, foram facilmente reconhecíveis as paredes e o fundo da piscina, no lado oposto. Tal como se previa neste tipo de condições, como não existe turbidez, existe uma maior facilidade em observar objetos no campo de visão.

Este tipo de cenário seria bastante favorável para o correto funcionamento deste tipo de tecnologia. Nestas condições e com incidência fraca ou nula da luz solar, é possível identificar os contornos do peixe de testes com facilidade. Apenas foi possível tirar proveito da luz laser quando a luminosidade diminuía pois, com a luminosidade usual da luz solar, a luz laser não é visível.

Ao colocar a montagem no rio, obtiveram-se resultados semelhantes, em alguns aspetos, aos obtidos na piscina. Contudo, no rio, as questões de turbidez dificultaram o destaque de contornos em certas condições, uma vez que o fundo de imagem fica bastante escuro. À medida que o peixe se afastava, foi possível vê-lo a distâncias consideráveis, com e sem laser pois o rio na altura dos testes apresentava um nível de turbidez muito inferior à verificada nos tanques de terra no período de realização dos testes. No entanto, no rio a visibilidade foi ligeiramente inferior à que se verificou na piscina. Portanto, concluiu-se que a turbidez é, de facto, um fator bastante limitante, pois pode restringir a distância de obtenção de imagens. Com base nestas conclusões, iniciaram-se esforços de forma a colocar o sistema de câmaras numa plataforma móvel, neste caso acoplado ao ROV. Desta forma, o sistema poderia acompanhar as movimentações dos peixes enquanto obtinha imagens. Após estes testes, achou-se necessário quantificar a limitação de turbidez e da iluminação no sistema de visão artificial.

A sequência de imagens obtida no rio veio a ser bastante útil no processo de desenvolvimento do sistema de câmaras, uma vez que foi o cenário em que se conseguiram obter imagens em melhores condições. Isto porque o algoritmo até determinado ponto foi sempre testado com imagens previamente obtidas. O algoritmo só foi testado com imagens obtidas “em direto” na fase final do desenvolvimento desta solução. O algoritmo foi maioritariamente testado e desenvolvido recorrendo a estas imagens obtidas no rio/albufeira. Com as imagens obtidas nos testes seguintes, tentou-se

testar o algoritmo, no entanto as condições pouco favoráveis de alguns dos cenários de testes e problemas relacionados com o posicionamento e fixação de câmaras, que invalidam qualquer calibração efetuada, impediram obter melhores resultados.

5.2.5.3 Testes de Campo nº3

No terceiro cenário de testes, o sistema de câmaras foi montando na estrutura do ROV desenvolvido. Nestes testes, as câmaras foram colocadas dentro de *housings* criados com impressão 3D, sendo também este um dos pontos a testar nesta sessão.

Os *housings* criaram alguns problemas devido à entrada de água no seu interior, o que dificultou relativamente a sessão de trabalhos. Com esta informação, o Engenheiro Mecânico responsável pelo desenho destes recipientes efetuou alterações de forma a evitar problemas semelhantes no futuro.

Com este sistema, executaram-se testes de forma a determinar quais as condições de turbidez e luminosidade que permitem o funcionamento do sistema.

5.2.5.3.1 Turbidez

Para determinar quais as condições de turbidez (TDS - *sólidos totais dissolvidos*) que permitem obter resultados minimamente aceitáveis, efetuaram-se alguns testes no CETEMARES, instalações do Instituto Politécnico de Leiria, em Peniche. Nestes testes, utilizou-se apenas uma câmara dentro de um dos *housings* metálicos utilizados em testes anteriores.

Verificaram-se algumas complicações em estabelecer as condições de teste, devido à dificuldade em controlar a turbidez existente na água e em manter os sólidos dissolvidos em suspensão na água.

Na primeira experiência, houve uma tentativa em utilizar sacos de plástico com soluções aquosas com várias concentrações de sólidos dissolvidos, que se colocavam em frente das câmaras, mas sem sucesso, uma vez que as mudanças do plástico para água criavam reflexões e refrações da luz que dificultam imenso a obtenção de imagens.

Na segunda abordagem, em que se obtiveram as conclusões necessárias, recorreu-se a um tanque com 500 litros de água, onde foi colocada areia, de modo a obter o valor de sólidos

dissolvidos (TDS) correspondente às condições mínimas de correto funcionamento da tecnologia. Alinhado com o campo de visão da câmara, colocou-se um tubo com marcas de 20 cm em 20 cm.

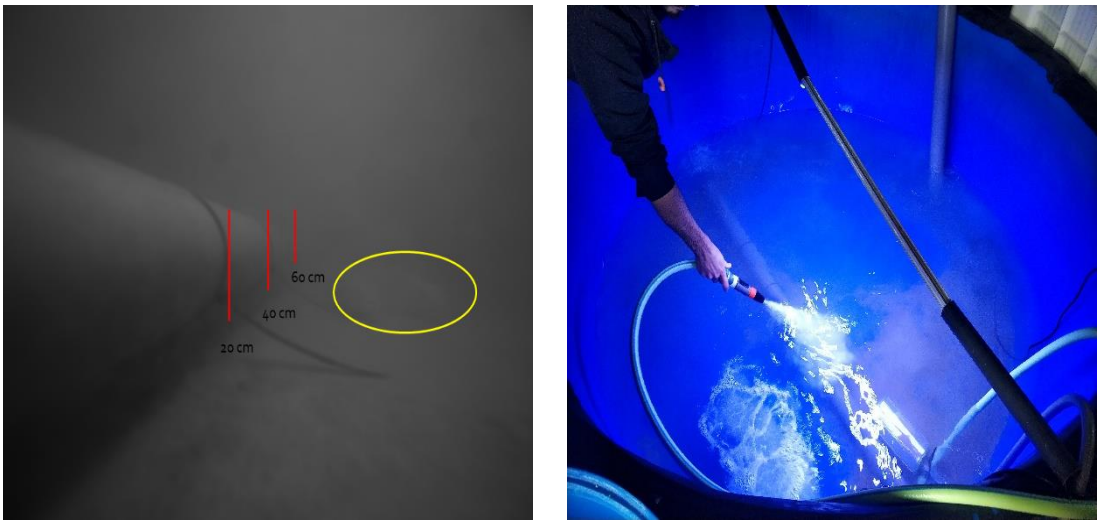


Figura 86 - Condições de teste de turbidez

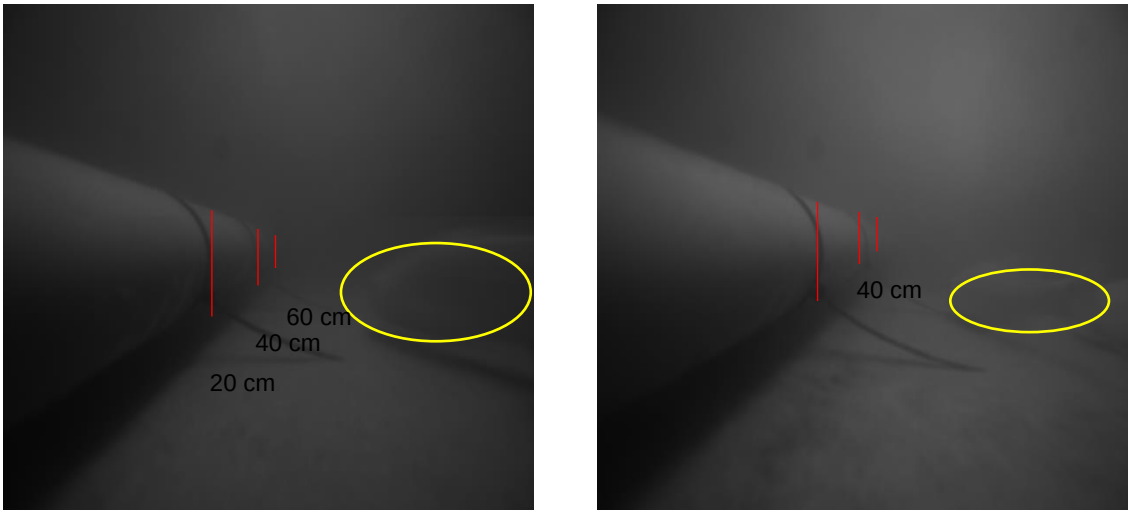


Figura 87 - Demonstração de resultados do teste de turbidez

Assim, determinou-se que para um valor de TDS de 30g/L é possível identificar um objeto que esteja a uma distância de até 40 cm a 60 cm. Com estes testes e com os testes anteriores, foi possível concluir que em condições de turbidez elevada é pouco provável obter resultados fiáveis e significativos com este sistema de visão artificial. Nas condições de turbidez apresentadas nos testes de campo anteriores, é difícil segmentar peixes representados nas imagens, uma vez que os peixes só são visíveis até uma distância de 50 centímetros e, normalmente, estes não se aproximam muito do sistema de visão artificial.

O facto de o sistema estar acoplado ao ROV, um objeto de grandes dimensões, pode trazer algumas complicações, uma vez que, os peixes facilmente se assustam com este objeto. Em condições de menor turbidez, é mais provável obter resultados significativos e que correspondam à realidade.

5.2.5.3.2 Luminosidade

A iluminação é um dos elementos mais importantes das aplicações de visão artificial. Uma das grandes dificuldades na aquisição de imagens dentro de água passou precisamente pela iluminação. Em condições normais nos tanques de aquacultura, a luz solar torna o campo de visão pouco nítido, ficando o topo da imagem muito iluminado e a parte inferior da imagem muito escura.

Para testar o uso da iluminação laser dentro de água, projetou-se um cenário de testes que permitisse eliminar a luminosidade proveniente da luz solar. Dentro de uma piscina interior, demonstrada na Figura 88, fez-se variar a intensidade luminosa de forma a concluir qual o cenário de luminosidade que permite o correto funcionamento do sistema.

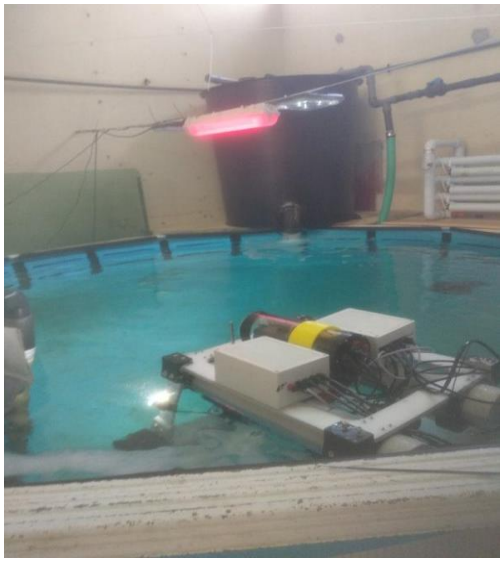


Figura 88 - Testes com luminosidade elevada

Figura 89 - Testes com luminosidade reduzida

Nestas condições em que a turbidez é bastante baixa quando comparada com as condições em tanques de aquacultura, determinou-se que as condições de luminosidade e potência luminosa do laser permitem identificar um objeto que esteja até 90 a 100 centímetros de distância, como demonstrado na Figura 90. No entanto na Figura 91, em que o objeto está colocado a uma distância de 50 a 60 centímetros, os contornos são muito mais nítidos e por isso a esta distância os resultados obtidos seriam muito mais fiáveis.

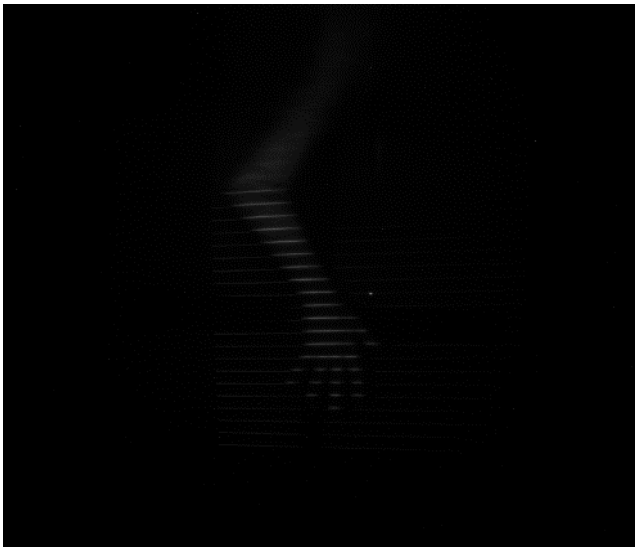


Figura 90 - Testes de luminosidade com objeto a 90-100 cm

Figura 91 - Testes de luminosidade com objeto a 40-60 cm

Com os testes executados, foi possível concluir que, quando existe pouca ou nenhuma luminosidade proveniente do ambiente exterior à água, o laser facilita o destaque de contornos e introduz uma componente de textura adicional ao objeto iluminado. Esta textura beneficia bastante o funcionamento do algoritmo de estéreo binocular, sendo uma das condições essenciais para o seu funcionamento.

5.2.5.4 Testes de Campo nº4

Esta sessão de testes foi realizada nas instalações de um dos parceiros deste projeto, a ALGAplus, em Ílhavo. Esta sessão foi a primeira ocasião em que se testou o conjunto de sistemas desenvolvidos no ambiente para o qual foram concebidos, os tanques de aquacultura. Ambas as unidades (UMA e UCB) já se encontravam instaladas no ROV e prontas a funcionar em simultâneo. Executaram-se testes em tanques de terra de aquacultura e em tanques de cimento utilizados pela ALGAplus para criação de algas. Estes testes foram as sessões que antecederam a apresentação final e entrega do respetivo relatório final do projeto.

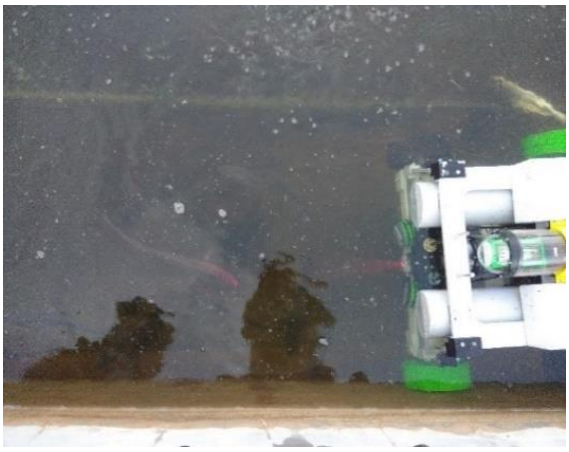


Figura 92 - Testes em tanques de cimento com peixes (ALGAplus, Ílhavo, 2018)

O sistema de visão, equipado com iluminação laser, foi testado nos tanques de terra em período diurno e noturno. Embora durante os testes, o nível de turbidez não ter sido tão elevado como em outros testes (visibilidade até quase um metro de distância), praticamente não se obtiveram imagens onde figurassem peixes. Na altura dos testes, os peixes existentes no tanque eram de tamanho reduzido e raramente se agrupavam em concentrações significativas junto do sistema de visão. Em termos de turbidez, a visibilidade melhorou significativamente no período noturno, permitindo que as imagens obtidas tivessem iluminação mais equilibrada. Independentemente dos níveis de luminosidade, a turbidez permaneceu como um obstáculo que dificulta bastante o sucesso do funcionamento deste sistema.

Testou-se o conjunto de equipamentos em tanques de cimento, com água com turbidez reduzida, onde foram colocados alguns peixes. Visto que nestes tanques não havia

praticamente turbidez nenhuma, as dificuldades oferecidas por este fator eram quase nulas. Na Figura 93, consegue aferir-se a presença de peixes na imagem, no entanto é, mais uma vez, percetível o desequilíbrio da iluminação nesta imagem.

Figura 93 - Imagens de peixes obtidas nos tanques de cimento durante o dia

O equilíbrio da iluminação na imagem continuou a demonstrar-se um fator condicionante. Experimentaram-se várias configurações referentes à aquisição de imagens, relativamente à mudança de tempo de exposição e ganho, no entanto, foi difícil contrariar a ação da luz solar no caso das imagens obtidas em período diurno. Apesar de também apresentar algumas limitações, nas imagens obtidas em período noturno, em que único foco de iluminação era o laser, foi possível identificar peixes nas imagens.

No tanque de cimento, o sistema de visão apresentou resultados bastante positivos pois a turbidez da água era inferior. Com o sistema posicionado numa das extremidades, foi possível visualizar a outra extremidade do tanque bem como os vários peixes no campo de visão (encontravam-se aproximadamente 20 peixes no tanque). Apesar da facilidade em visualizar peixes nestas condições, as imagens obtidas não permitiram testar o algoritmo desenvolvido. Esta impossibilidade esteve sobretudo relacionada com o facto de o posicionamento das câmaras ter sido comprometido, invalidando qualquer calibração anterior. Outro dos problemas foi o facto de ter entrado água no *housing* da câmara direita numa das ocasiões em que se obtiveram melhores imagens, sendo que tal só foi observado após remover o equipamento da água. Ainda assim, efetuaram-se alguns testes com pares de imagens em melhores condições, obtidas em período noturno, no entanto não se obtiveram resultados de funcionamento positivos. Tal estaria relacionado com o grande desequilíbrio de brilhos e luminosidade de imagens, que dificultavam a segmentação de

objetos presentes nas imagens por valores de cinzento, e que resultavam em imagens de disparidade com muito ruido, conduzindo a resultados muito imprecisos. Verificou-se igualmente que o posicionamento das câmaras e o seu direcionamento não seria o ideal. Os peixes concentravam-se no fundo do tanque e as câmaras estavam apontadas para a frente e numa posição um pouco distante do fundo do tanque, levando a que, por vezes, os peixes passassem por baixo do sistema sem serem visíveis nas imagens. Assim concluiu-se que uma das melhorias a efetuar no sistema seria apontar o sistema de visão ligeiramente para baixo.

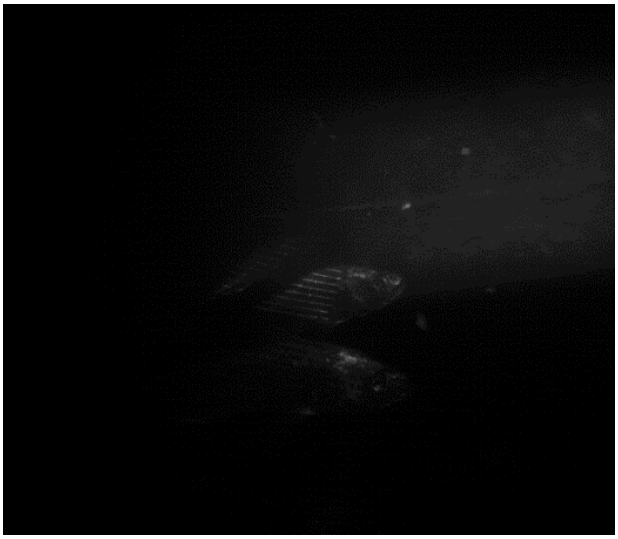


Figura 94 - Imagem noturna nos tanques de cimento

5.2.6 Conclusões Relativas ao Desenvolvimento da UCB

Seguindo as especificações pretendidas para a UCB apresentadas inicialmente, podem tirar-se conclusões relativamente ao trabalho desenvolvido, aos objetivos cumpridos, aos resultados obtidos, melhorias a efetuar e quais os requisitos que não foram cumpridos.

O primeiro objetivo, relativo ao desenvolvimento do sistema móvel robotizado, pode considerar-se um objetivo que foi cumprido. Após algumas decisões no início do desenvolvimento do projeto, este sistema robótico passou a ser a estação móvel que alberga esta unidade e a UMA. Este veículo, bem como o seu processo de desenvolvimento, irão assim ser abordados num outro capítulo à parte.

Nos objetivos apresentados, a estimativa da biomassa deveria ser efetuada utilizando em conjunto um sonar com um sistema de câmaras, no entanto apenas se recorreu ao sistema de câmaras. Durante os testes e mediante pesquisa de soluções de sonar, chegou-se à conclusão de que a utilização deste tipo de tecnologia não teria grande sucesso, uma vez que, para esta aplicação em concreto, seria necessário um equipamento com elevada resolução que permitisse distinguir cardumes de peixes em tanques cujas dimensões são pouco apropriadas ao uso dos sonares mais comuns no mercado. Caso se utilizasse um sonar este teria dificuldade em distinguir cardumes de peixes do fundo dos tanques, a apenas 3 a 4 metros da superfície, e das paredes de tanques com 40 a 50 metros de largura e 150 a 200 metros de largura. Assim, optou-se por utilizar apenas o sistema de câmaras e basear o processo de desenvolvimento nesta tecnologia.

A localização das maiores concentrações de peixes nos tanques foi, de facto, uma particularidade para a qual a solução desenvolvida não conseguiu dar resposta. O sistema e algoritmo desenvolvidos conseguiram determinar as dimensões de objetos nos testes efetuados em laboratório durante o desenvolvimento e, igualmente, obter dimensões aproximadas do peixe plástico utilizado para os testes efetuados no rio. Este peixe tinha aproximadamente 9 centímetros de comprimento, a dimensão mais relevante na fórmula de estimativa de biomassa utilizada, e com o conjunto de imagens obtidas nos testes obtiveram-se resultados aproximados a este valor, com 1 a 2 centímetros de diferença. Relativamente às várias imagens obtidas nos tanques de aquacultura, não foram obtidos resultados muito positivos, tendo existido imensas dificuldades em aplicar o algoritmo desenvolvido nas imagens obtidas em tais conclusões. Assim, concluiu-se que este

algoritmo de processamento de imagem apresenta um funcionamento limitado, apenas apresentando resultados positivos num conjunto específico de imagens e em condições de total controlo da luminosidade. Nos testes efetuados não se obtiveram resultados conclusivos com imagens de peixe obtidas nos citados tanques. Tal como descrito anteriormente, as dificuldades estão associadas à turbidez elevada das águas em aquacultura semi-intensiva e ao facto de existirem grandes dificuldades em controlar a luminosidade em ambiente aquático.

Outro dos objetivos apresentados era o processamento de imagem ser efetuado localmente na unidade a desenvolver. Considera-se que este objetivo foi cumprido, uma vez que o processamento de imagem na UCB é efetuado pelo microprocessador utilizado na solução. Existem alguns constrangimentos relativos à biblioteca de *software* MVTec HALCON utilizada nesta solução, uma vez que para utilizar esta ferramenta é necessário que o equipamento que executa o programa desenvolvido tenha uma licença fornecida pelo fabricante instalada, licença essa que tem um custo associado. Do ponto de vista económico, esta licença eleva bastante o preço da solução final. Assim, uma das sugestões de alteração desta unidade seria a utilização de *software* de processamento de imagem *open source*, que poderia beneficiar igualmente o desenvolvimento, dado que este tipo de *software* tem normalmente comunidades abertas online, que podem ser bastante úteis no fornecimento de conteúdos para o processo de desenvolvimento.

A solução desenvolvida também não cumpre com o objetivo que definia que o sistema deveria ter capacidade de aprendizagem e adaptabilidade à espécie/forma de cultura e ser suportado em algoritmos adaptativos para análise de previsões de crescimento e controlo de desvios da biomassa total. O desenvolvimento não chegou a tal ponto, visto que existiram bastantes dificuldades em estabelecer um sistema capaz de executar as medições e estimativas necessárias.

No início do projeto, pretendia-se que a UCB efetuasse a estimativa da biomassa de peixes e algas. No entanto, o conjunto de entidades do consórcio optou por focar os esforços de desenvolvimento apenas na análise de peixes, em particular a dourada e o robalo.

O processo de desenvolvimento desta unidade foi acompanhado pela empresa representante da MVTec, detentora da biblioteca HALCON, em Portugal. Esta empresa prestou serviços de formação e suporte à equipa do LINE.IPT, em particular ao aluno que

apresenta esta tese de estágio, tendo recorrido a tal entidade para resolução de problemas e esclarecimento de dúvidas relativas a questões de visão por computador e à utilização da biblioteca.

5.3 Unidade de Monitorização Ambiental (UMA)

Ao desenvolver a UMA, pretendia-se criar um dispositivo de monitorização ambiental disruptivo e implementar um método inovador de análise da qualidade da água, através da aquisição de amostras de diferentes "Estações de Trabalho" para um sistema central de sensorização.

De seguida, são apresentados os objetivos específicos que regem o desenvolvimento da UMA:

- Projetar um dispositivo que detenha toda a capacidade de medição de grandezas através de sensores;
- Deve apresentar compatibilidade com diferentes sondas multi-parâmetros que comunicarão entre si e com o sistema de controlo de produção implementado;
- Recolher informação da qualidade da água fiável e efetuada em tempo real;
- Deve fundamentar-se numa arquitetura baseada em padrões abertos ou *open source*;
- Deve permitir a integração flexível, escalável, modular e adaptável à natureza das fases de produção, da espécie a produzir, dos parâmetros a controlar ou até à própria estrutura física do local de produção;
- Permitir que toda a informação recolhida seja também armazenada localmente, de modo a estar preparada para aquaculturas *offshore*, e na *cloud*, permitindo acesso por outros equipamentos em rede;
- Os dados provenientes deste dispositivo poderão despoletar sinais de alarme ou regras de atuação sobre equipamentos, como a ativação/desativação de subsistemas de oxigenação, sistemas de alimentação automáticos, etc.;
- Deverá ser centralizado, numa única unidade de instrumentação para monitorizar diversas "Estações de Trabalho".

Os objetivos de âmbito geral de desenvolvimento do sistema são:

- Potenciar um modelo de criação de escalas de automação através da conceção de uma arquitetura que permita a interoperabilidade entre diferentes sensores e atuadores;

- Desenvolver kits de sensores de multi-parâmetros personalizáveis por fase de produção e espécie;
- Desenvolver os conetores para a interligação da plataforma com múltiplas fontes de dados (Sistemas de medição de Biomassa; Sensores de Temperatura, Salinidade, etc.) e com os atuadores (Sistemas de Alimentação, oxigenadores, etc.).

Os requisitos para os módulos da UMA e respetivas funções no sistema são demonstrados na Tabela 8.

Tabela 8 - Requisitos para os módulos da UMA e suas funções

Módulo	**Função**
Amostragem de dados e transmissão	Recolha e transmissão de dados de qualidade da água com base em RSSF
Monitorização da qualidade da água	Monitorização em tempo real da qualidade da água
Recolha de informação do processo de Aquicultura	Registo diário com recurso a comunicação sem fios, de informações como a alimentação e uso de medicação
Aviso e solução de problemas	Gerar alertas quando for excedido algum indicador
Estatística e suporte à decisão	Gerar relatórios e gráficos para referência de gestão na tomada de decisões
Serviço Web	Fornecer aos consumidores informações sobre os produtos aquáticos na Web

Na documentação deste projeto, foram identificados alguns desafios inerentes ao desenvolvimento da UMA. Na Tabela 9, é demonstrada uma lista com estes desafios.

Tabela 9 - Desafios desenvolvimento da UMA

Desafios - Sistema UMA
• A medição de parâmetros de diferentes tanques aumenta o risco de contaminação, havendo a necessidade de garantir o isolamento das amostras de água recolhidas, bem como o seu descarte
• A calibração dos diversos sensores deve ser assegurada automaticamente através de soluções padrão e apoiada em protocolos de calibração
• A integração dos vários sensores ao nível da eletrónica (compatibilidade eletromagnética, valores de alimentação elétrica, etc.) e da informática (protocolos de comunicação, fiabilidade dos dados face às condições ambientais, etc.) dos diversos equipamentos representa igualmente um desafio, que deve ser ultrapassado com testes de robustez ao sistema

5.3.1 Arquitetura da UMA

Na Figura 95, é demonstrada a arquitetura da Unidade de Monitorização Ambiental, com os detalhes de interligação do *hardware* com os serviços de *software* no microcontrolador Samsung ARTIK 710 e a ligação à UCB, ao módulo de comando e navegação do ROV (RPi+Navio2) e o computador externo onde se utiliza a página Web para configuração do sistema.

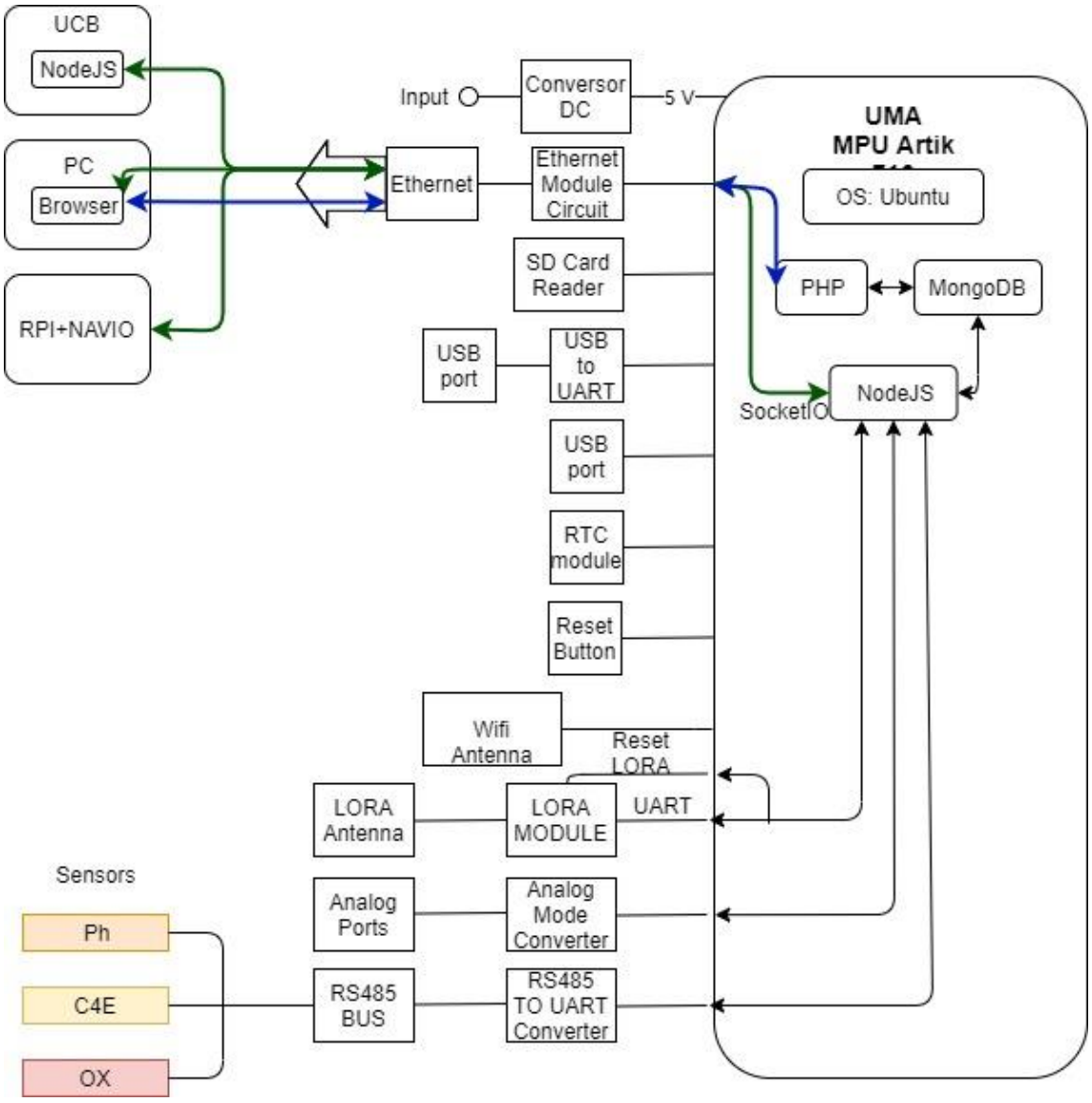


Figura 95 - Arquitetura da UMA (relação entre *software* e *hardware*)

5.3.2 *Software* da UMA

Em termos de arquitetura, a UMA é o ponto de convergência de todas as comunicações no ROV. Esta interage com os diferentes módulos, recebe e envia parâmetros LoRa e gera uma rede Wi-Fi. A unidade de monitorização ambiental é um Gateway desenvolvido de raiz com recurso a um microprocessador de 64 bits ARTIK 710 da Samsung e sistema operativo Ubuntu Linux.

A grande maioria dos desenvolvimentos de *software* desta unidade foram efetuados pelo engenheiro informático da equipa do LINE. Nesta unidade, o aluno, que apresenta esta tese de estágio em conjunto com a restante equipa de engenharia eletrotécnica do LINE, dedicou-se ao desenvolvimento das partes do *software* para configuração dos sensores, aquisição dos dados recolhidos por estes e desenvolvimento dos circuitos e configuração do *hardware* da unidade. O aluno foi responsável pela calibração das sondas e teste de algumas das configurações de *software* iniciais destes sensores.

No microprocessador ARTIK que integra a UMA, é executado um servidor HTTP (Apache) que serve uma página em PHP, que permite efetuar configurações do sistema de sensores e visualizar os dados de qualidade da água obtidos. Neste microprocessador, está também instalada a base de dados MongoDB, que armazena todos os dados recebidos das sondas e ainda um servidor Node.js que processa e reúne toda a informação da estrutura. O servidor Node.js comunica com a referida página de configuração e visualização por Socket.IO, página que é normalmente acedida pelo computador na camada externa.

Inicialmente, o sistema da UMA aguarda um pacote LoRa. Quando este chega, é efetuada a sua validação e é enviado um comando ao ROV para iniciar a operação, de modo a que o veículo siga determinada rota para obter dados de qualidade de água e dados de medição da biomassa. É igualmente enviado um comando à UCB com o pacote de configuração para começar a recolha de imagens. À medida que o ROV se movimenta, são enviados para o processo de Node.js da UMA, os dados de GPS e da *Attitude* (parâmetros relacionados com velocidade, movimentação e rotação do veículo) e os valores da estimativa de biomassa da UCB. Neste processo, ocorre simultaneamente a obtenção de dados relativos à qualidade da água através das sondas.

Os dados das sondas são recolhidos a cada 20 segundos. Ao final de um período de 10 minutos de recolhas, é efetuado o cálculo da média dos valores obtidos neste período. Após este cálculo, são enviados por LoRa, os valores médios dos parâmetros de qualidade da água, juntamente com a informação da localização GPS e a informação da *Atittude*, bem como os valores da estimativa da biomassa. Em tempo real, são também enviadas as leituras das sondas para o *browser* e guardadas em base de dados. Os dados são estruturados em pacotes JSON, e são convertidos para hexadecimal para envio por LoRa.

Este microprocessador, onde está alojado o servidor de base de dados, tem algumas limitações de recursos (processamento e RAM). Assim, uma base de dados relacional iria sobrecarregar demasiado o sistema. Uma vez que o código de servidor (*backend*) foi realizado em Node.js (JavaScript) optou-se por implementar uma base de dados não relacional, MongoDB, que armazena dados já em formato JSON (objeto JavaScript).

A página Web da UMA foi desenvolvida em HTML, PHP e JavaScript, recorrendo a um servidor Apache. Esta foi desenvolvida a partir de um *template* HTML, ao qual foi adicionada toda a parte de processamento com a linguagem de programação PHP e JavaScript. O corpo da funcionalidade da página consiste numa ligação Socket.IO em JavaScript para receber todos os valores em tempo real e para permitir configurar alguns detalhes do sistema da UMA. Toda a parte que exige interação com a base de dados foi executada em PHP, nomeadamente: consultas, visualização de dados e escritas. Na Figura 96, está representada a estrutura da página Web da UMA.

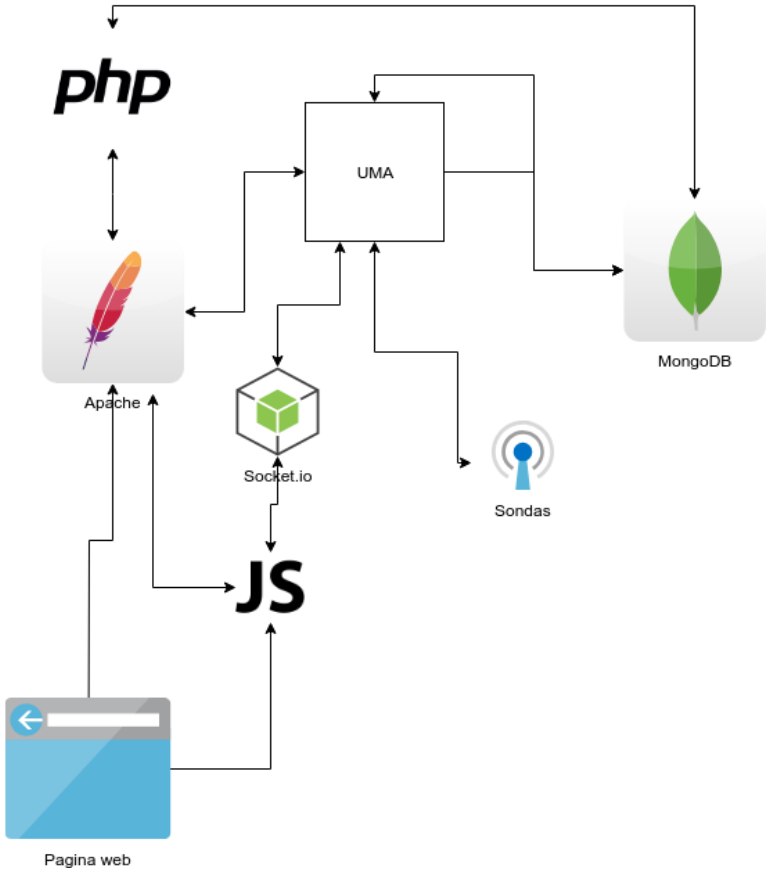


Figura 96 - Arquitetura da página Web

O menu da página Web da UMA está estruturado da seguinte forma

- *Dashboard*
- Opções Sensores
  - Inserir tipo sensor
  - Adicionar sensor
  - Adicionar dispositivo
  - Editar dispositivo
  - Calibrar dispositivo
- Opções Rede
  - Configurar rede
  - Configurar LoRa
- Consultas

No *dashboard*, encontram-se representadas várias informações: os dados dos sensores, em tempo real; quantos utilizadores estão a aceder à página; quantos *nodes* (conjunto de

sensores) estão online e quantos *nodes* estão configurados. No canto superior direito, existem dois botões que permitem mudar o modo de funcionamento entre envio e aquisição do módulo LoRa.

Na aba de opções de sensores, estão disponíveis cinco submenus que permitem inserir um novo tipo de sensores, adicionar sensores, adicionar dispositivo, editar dispositivo e calibrar dispositivo. Em "*Adicionar tipo de sensor"* deve indicar-se a tecnologia e o nome que se quer dar ao tipo em causa. A opção "*Adicionar sensor"* permite adicionar um sensor, definindo o seu nome e a tecnologia (ADC ou Modbus). O menu *"Adicionar dispositivo"* permite a adição de um dispositivo, que corresponde a um conjunto de sensores. No menu "*Editar dispositivo"*, é possível editar dispositivos previamente configurados, de forma a adicionar ou eliminar sensores. Finalmente, o menu *"Calibrar dispositivo"* é utilizado para calibrar os sensores de cada dispositivo.

Nas opções de rede, pode-se configurar a rede *wireless* gerada pelo microprocessador desta unidade, assim como, os parâmetros da comunicação LoRa e ainda visualizar os dados recebidos por LoRa.

A aba "Consultas" permite rever informação guardada na base de dados, relativa à informação recolhida pelos sensores. Estas consultas podem ser realizadas por dispositivo, mostrando, associado a um intervalo de tempo, todos os sensores de um dispositivo, ou ainda, pela referência de determinado sensor de um dispositivo.

5.3.3 Sondas de Aquisição de Parâmetros de Qualidade da Água: *Software* e *Hardware*

A aquisição de parâmetros de qualidade da água é realizada com recurso aos sensores do fabricante Ponsel fazendo uso do protocolo "Modbus RTU". Para os parâmetros necessários, utilizaram-se três sensores (OPTOD, C4E e PHEHT), representados na Figura 97. As características dos sensores serão de seguida apresentadas.

Figura 97 - Sondas de parâmetros (respetivamente OPTOD, C4E e PHEHT)

5.3.3.1 OPTOD

A sonda OPTOD permite medir os valores de oxigénio dissolvido na água através de tecnologia ótica por luminescência. Esta tem uma gama de medidas de 0 a 20 mg/L, 0 a 20 ppm e 0-200%. A sua resolução é de +/- 0,1 mg/L, +/- 0,1 ppm e +/- 1%.

Apesar de o fabricante fornecer o procedimento de calibração no manual deste sensor, esta é desnecessária. Este sensor é baseado em tecnologia ótica e armazena os dados de calibração, não sendo possível haver um desvio de calibração e, assim, a sua calibração é desaconselhada pelo fabricante. No ato de venda, o fabricante emite um certificado autenticado de calibração, que confirma a precisão das medições obtidas.

5.3.3.2 C4E

A sonda C4E permite medir os valores de condutividade e salinidade na água utilizando quatro elétrodos, dois de grafite e dois de platina. Tem gamas de medidas variáveis, que podem ser de 0 a 200 uS/cm, 0 a 2000 uS/cm, 0-20 mS/cm ou 0-200 mS/cm. A resolução depende da gama de valores escolhida, podendo ser de 0,01 a 1 unidade.

Calibração da sonda C4E

A calibração desta sonda é feita em duas etapas: a primeira etapa para definir o valor de *offset* e a segunda etapa para definir o valor de ganho de medida, que dependerá da gama de valores que se pretende medir. Para definir o *offset*, o sensor deve ser colocado exposto ao ar, gravando o valor a 0,0 uS/cm na consola. O ganho é calibrado colocando a sonda mergulhada em soluções previamente estabelecidas pelo fabricante para cada gama de medida.

Tabela 10 - Gama de medição

Gama de Medidas	**Concentração da Solução a Utilizar**
0 a 200 µS/cm	84 µS/cm
0 a 2000 µS/cm	1,413 µS/cm
0 a 20 mS/cm	12,880 µS/cm
0 a 200 mS/cm	111,8 mS/cm

5.3.3.3 PHEHT

A sonda PHEHT permite medir os valores de pH, redox e temperatura. A gama de medida de valores de pH é de 0 a 14, com resolução de 0,01 pH e precisão de ± 0,1 pH. A gama de medida de valores de redox é de -1000 a 1000 mV, com resolução de 0,1 mV e precisão de ± 2 mV. A gama de valores de temperatura é de 0 a 50 °C, com uma resolução de 0,01ºC e precisão de ± 0,5°C.

Calibração da sonda PHEHT

O sensor previamente limpo é imerso numa primeira solução padrão (pH 7,1 a 25 °C, por exemplo), para determinar o ponto 0. Deve manter-se a solução padrão em movimento e aguardar que o valor medido atinja o equilíbrio com a temperatura da solução padrão. O pH da solução padrão varia com a temperatura, consoante a temperatura deve ser registado o valor apresentado na Tabela 11.

Tabela 11 - Gamas de medida de sensor de pH

Solução pH 7,01 a 25ºC	
ºC	**pH**
0	7,13
5	7,10
10	7,07
15	7,04
20	7,03
25	7,01
30	7,00
35	6,99
40	6,98
45	6,98

Para determinar o *slope point* do pH, deve-se primeiro lavar e secar o sensor. De seguida, deve-se mergulhar o sensor numa segunda solução de pH, por exemplo na solução pH 4,01 a 25ºC. O procedimento é igual ao considerado anteriormente para o ponto 0. Na Tabela 12, apresenta-se a variação do valor de pH desta solução em função da temperatura.

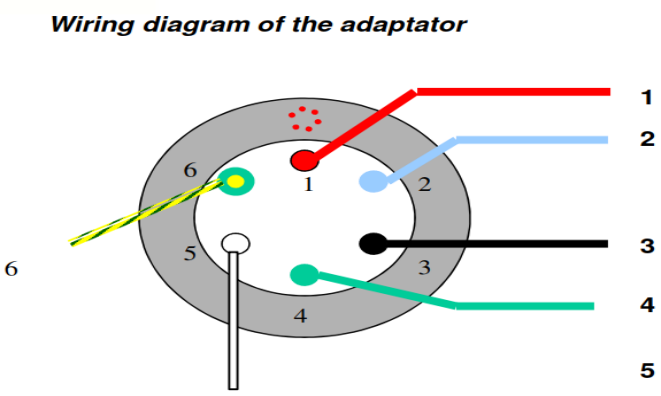
Tabela 12 - Gamas de medida de sensor de pH

Solução pH 4,01 a 25ºC	
ºC	**pH**
0	4,01
5	4,00
10	4,00
15	4,00
20	4,00
25	4,01
30	4,02
35	4,03
40	4,04
45	4,05

5.3.3.4 Configurações e Características do *Software* das Sondas (OPTOD, C4E e PHEHT)

- Modo de transmissão: RS-485;
- 9600 baudrate;
- 8 bits;
- 2 stop bits;
- Sem paridade;
- Sem controlo de fluxo.

Na Figura 98, encontra-se identificada a configuração das ligações elétricas das sondas de monitorização.



1 - red	Power supply V+
2 - blue	SDI-12
3 - black	Power supply V-
4- green	B " RS-485 "
5 - white	A " RS-485 "
6 – yellow/green	Cable shield

Figura 98 - Esquema de ligação das sondas de parâmetros de qualidade da água

Para iniciar a aquisição dos parâmetros das sondas, é escrito para o registo número 1 de cada sensor, o comando para iniciar as leituras. Após algum tempo de espera para terminar a atualização dos valores, é executada a leitura dos bytes dos registos "*holding*" para os parâmetros da qualidade da água que se pretendem obter. Na Tabela 13, são descritos os endereços Modbus que permitem obter as leituras dos parâmetros das sondas. Na Tabela 14, na Tabela 15 e na Tabela 16 são apresentados os parâmetros que cada sensor permite obter. Como os dispositivos de Modbus são conectados a um barramento, é possível conectar mais sondas ao sistema sem ser necessário alterar o circuito do

conversor de Modbus. Para além disso, as sondas apresentam uma boa relação preço/qualidade.

Tabela 13 - Endereços Modbus relevantes para os parâmetros das sondas

	Endereço Modbus	**Memória**	***Read/ Write***	**Descrição**	**Type**	**Bytes**
1	0x0001		w	Ordem para executar medição dos parâmetros	int	2
83	0x0053	RAM	r	Leitura da Temperatura	float	4
85	0x0055	RAM	r	Leitura do Parâmetro 1	float	4
87	0x0057	RAM	r	Leitura do Parâmetro 2	float	4
89	0x0059	RAM	r	Leitura do Parâmetro 3	float	4
91	0x005B	RAM0	r	Leitura do Parâmetro 3	float	4

Tabela 14 - Parâmetros medidos pela sonda OPTOD

Endereço Modbus do dispositivo: 10				
	Medição de Temperatura	**Parâmetro 1**	**Parâmetro 2**	**Parâmetro 3**
Parâmetro medido	Temperatura	Oxigénio	Oxigénio	Oxigénio
Unidade	ºC	%Sat	mg/L	ppm

Tabela 15 - Parâmetros medidos pela sonda PHEHT

Endereço Modbus do dispositivo: 20			
	Medição de Temperatura	**Parâmetro 1**	**Parâmetro 2**
Parâmetro medido	Temperatura	pH	Redox
Unidade	ºC	Unidades de pH	mV

Tabela 16 - Parâmetros medidos pela sonda C4E

Endereço Modbus do dispositivo: 30				
	Medição de Temperatura	**Parâmetro 1**	**Parâmetro 2**	**Parâmetro 3**
Parâmetro medido	Temperatura	C43 condutividade	Salinidade	TDS
Unidade	ºC	µS/cm	ppt	ppm

5.3.4 *Hardware* da UMA

Os módulos que complementam o microprocessador no circuito desenvolvido para a UMA são baseados em USB/UART e um dos módulos é conectado a um ADC do módulo ARTIK. Ao desenvolver um circuito que recorre a tais métodos de conexão, é possível garantir que esta unidade pode ter por base qualquer outro microprocessador, uma vez que, a maioria das ofertas deste tipo de tecnologia presentes no mercado permitem este tipo de conexão. Em caso de necessidade de alteração da unidade central de processamento ou mudança de arquitetura, a eletrónica bem como o *software*/*firmware* associado ao seu funcionamento será assim compatível, não havendo necessidade de alterações em nenhum deles. Na Figura 99, encontra-se representada a PCB do módulo UMA.

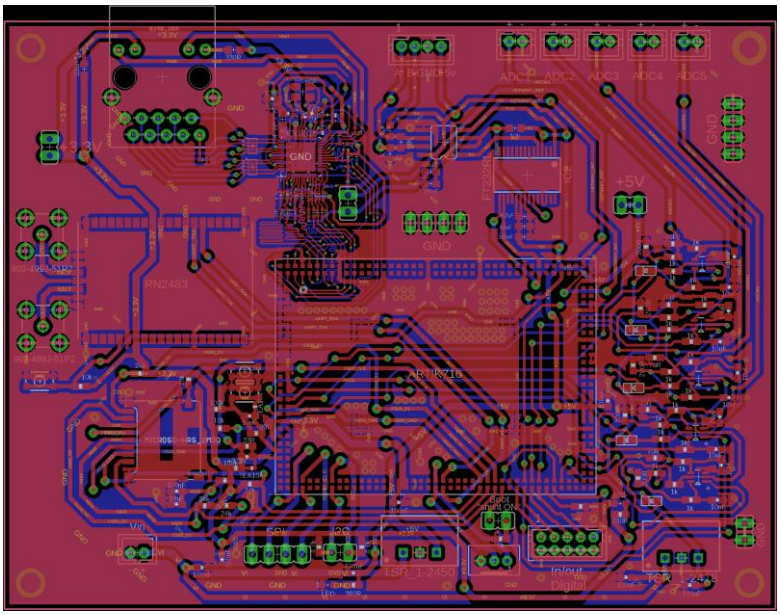


Figura 99 - PCB do módulo UMA

À semelhança do módulo UCB, o módulo UMA incorpora o módulo ARTIK 710, um circuito de Ethernet, assim como o circuito alimentação e o circuito de leitura de cartão micro SD. O módulo UMA possui, além disso, um circuito de Modbus, um circuito de entradas analógicas que permite optar por entradas de sinais de corrente e tensão, um *socket* de entradas/saídas digitais e um circuito de LoRa.

O módulo UMA possui um circuito de Modbus (ver Anexo 8 – Circuito de Modbus da UMA) para leitura dos sensores de análise dos parâmetros da água. Este circuito recebe

o sinal dos sensores em RS485, sendo estes convertidos para TTL e de TTL para USB. Por sua vez, o ARTIK recebe esta conexão USB.

No circuito de comunicação LoRa (ver Anexo 9 – Circuito de LoRa da UMA) é utilizado o módulo de LoRa RN2483. Este módulo comunica com o ARTIK via UART e necessita apenas de alimentação de 3,3V e de um *switch* para efetuar *reset*. No sentido de possibilitar o *reset* de modo remoto, existe também uma ligação a uma porta digital do ARTIK. O módulo LoRa RN2483 possui ligação para duas antenas exteriores. As entradas e saídas do módulo LoRa (GPIO) não foram utilizadas.

5.3.5 UMA: Resultados e Conclusões

Nos trabalhos de desenvolvimento da UMA, os sensores e o respetivo módulo de conversão RS485 para UART foram os primeiros elementos estabelecidos, sendo que a base desta estrutura inicial foi utilizada em todos os protótipos desta unidade. Inicialmente, utilizou-se o módulo de conversão RS485 para USB que, para os testes, foi ligado a vários modelos de microprocessador onde era executado o *software* para recolha de dados dos sensores.

Nos primeiros testes de campo, os sensores foram acoplados com peças criadas em impressão 3D, na estrutura do ROV utilizado para testes, o BlueROV2 do fabricante BlueRobotics. De forma a conectar estes sensores a um equipamento capaz de correr *software* para ler os dados dos sensores, dentro dos *housings* da eletrónica deste veículo, colocou-se um microprocessador BeagleBone Black numa primeira fase e, posteriormente, utilizou-se um Samsung ARTIK 710 (kit de desenvolvimento). Com esta estrutura, efetuaram-se alguns testes, que foram efetuados simultaneamente com os testes dos sistemas da UCB. Esta montagem dos sensores no BlueRov2 manteve-se até ser criado o protótipo final.

No protótipo final da UMA, os módulos que foram utilizados nos testes com os sensores foram integrados numa só PCB, que continha o módulo do microprocessador ARTIK, o modulo RS485, para interligação entre os sensores e esta unidade de processamento, o módulo LoRa e um módulo de ligação de dispositivos analógicos e digitais. Esta PCB, que é o módulo completo da unidade de processamento da UMA, foi colocada numa das caixas estanques do protótipo final do ROV, ao qual foram igualmente acoplados os sensores. A grande maioria dos desenvolvimentos de *software* da UMA, nomeadamente

os serviços da página de configuração da unidade, a base de dados e a comunicação por LoRa, foram efetuados nesta fase. Nos testes com o protótipo final, procurou-se sobretudo configurar corretamente a comunicação LoRa com a UCA, bem como estabelecer com os parceiros do projeto, quais os dados a enviar para esta segunda unidade e qual o formato dos pacotes de dados.

Um dos objetivos era que esta Unidade de Monitorização Ambiental assentasse numa rede de sensores sem fios (RSSF). No entanto, optou-se por criar um único sistema com capacidade de se movimentar pelos tanques de produção aquícola. Não correspondendo exatamente ao objetivo esperado, a solução desenvolvida permite dar resposta à questão de envio de informação sem fios de dados recolhidos em várias localizações. Caso o sistema não consiga enviar os dados por LoRa, os dados são armazenados localmente para posterior comunicação quando a ligação for reestabelecida.

Relativamente aos restantes desafios propostos, o protótipo da UMA conseguiu cumprir com os requisitos apresentados. Esta unidade concentra em si toda a capacidade de processamento e de recolha em tempo real de parâmetros de qualidade da água, permitindo a utilização de vários tipos de sondas, que podem ser conectadas via Modbus ou através das portas analógicas. A unidade desenvolvida é baseada em *software opensource*, permitindo que todos os componentes de *software* criados possam ser utilizados em qualquer microprocessador existente no mercado com características semelhantes ao sistema utilizado.

5.4 *Remotely Operated Vehicle* (ROV): Desenvolvimento do Veículo Aquático

Durante os trabalhos de desenvolvimento, o ROV foi a unidade que sofreu mais alterações, tendo sido implementados três protótipos. As alterações efetuadas de protótipo para protótipo estiveram sobretudo relacionadas com a componente mecânica, para garantir que o veículo desenvolvido cumpria com os requisitos necessários para albergar as outras duas unidades, bem como para deslocar-se ao longo dos tanques de aquacultura. O protótipo final tem por base a arquitetura apresentada na Figura 100.

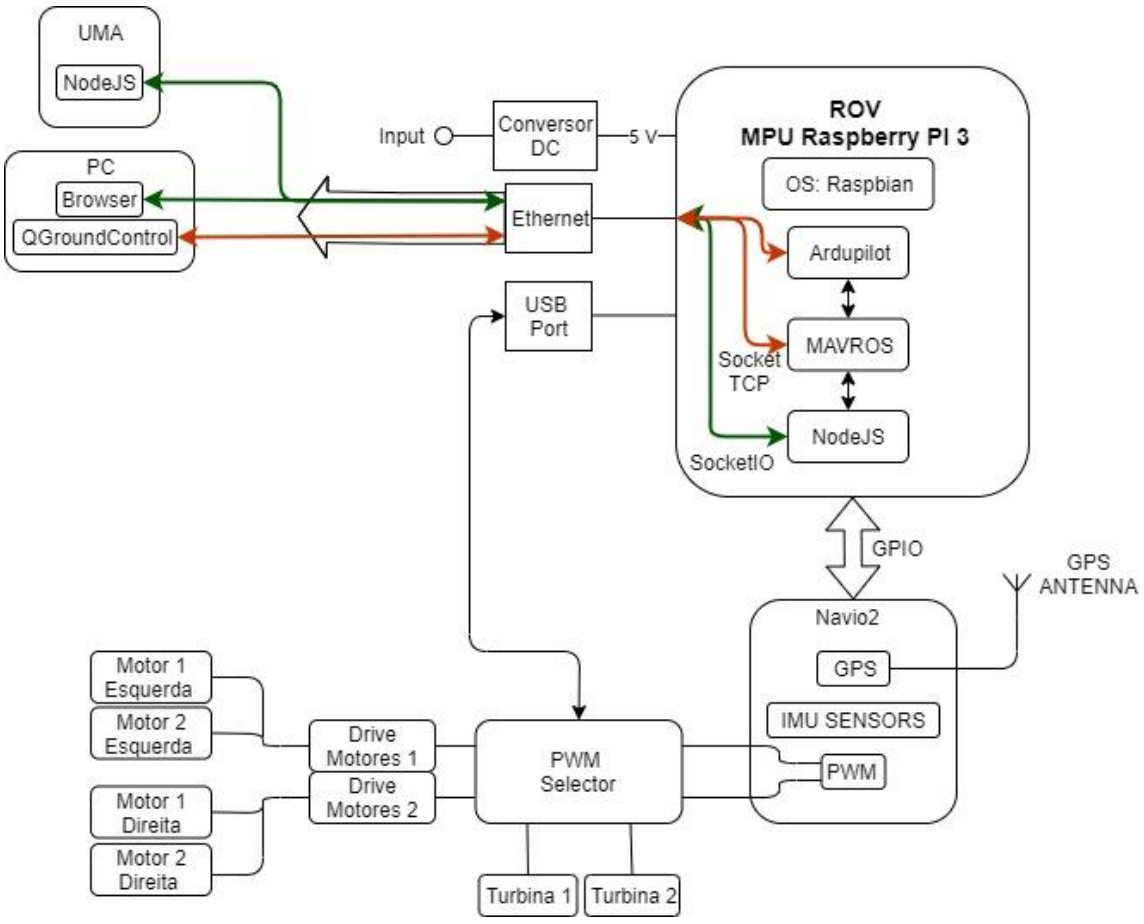


Figura 100 – Arquitetura do ROV

5.4.1 *Software* do ROV

O controlo de locomoção do ROV é efetuado recorrendo ao Raspberry Pi 3 equipado com o Navio2. Neste microprocessador, é executado o Ardupilot, um *software opensource* que reúne um conjunto de ferramentas para locomoção e navegação autónoma de veículos como drones, rovers e submarinos. O RPi3 recebe, via MAVLink da UMA, as instruções para iniciar o movimento autónomo do ROV. No protótipo do veículo desenvolvido, as rotas de movimento autónomo do veículo foram definidas previamente, no *software* de configuração do sistema QGroundControl. Este *software* funciona como estação de controlo do ROV e é executado no computador externo, comunicando com o RPi3, via SocketTCP e sobre esta ligação é utilizado o protocolo de comunicação MAVLink. Este *software* foi utilizado para definir as configurações iniciais do sistema e foi útil no processo de testes e desenvolvimento do ROV pois exibe, em ambiente gráfico, informações como velocidade, estado das baterias, a sua localização no mapa, entre outras informações úteis.

No Raspberry Pi 3, é utilizada uma versão do Raspbian, que é uma versão do Linux baseado em Debian e especialmente desenvolvida para Raspberry Pi. A versão do Raspbian utilizada é fornecida pela Emlid, fabricante do Navio2, sendo pré-configurada com algumas ferramentas para navegação como a *framework* Ardupilot e o ROS (*Robotic Operativ System* – Sistema Operativo para Robô).

Nesta unidade, recorreu-se igualmente ao Node.js para desenvolvimento de *software*. Neste caso, o servidor Node.js recolhe informação relativa à navegação e envia-a para a UMA.

No servidor Node.js do ROV, é usado um cliente de ROS que fornece informação proveniente dos sensores de navegação do Navio2. Neste servidor, também é usado um cliente Socket.IO para a comunicação com a UMA. O sistema recolhe os dados de GPS e de *Attitude* do ROV provenientes de tópicos ROS, enviando esta informação sucessivamente para a UMA via Socket.IO.

No ROS, são subscritos os tópicos "*/mavros/global_position/global*" e "*/mavros/imu/data*" que fornecem a seguinte informação, respetivamente:

1º tópico: "*/mavros/global_position/global*" (informação de GPS)

- *timestamp*;
- latitude;
- longitude;
- altitude.

2º tópico: "*/mavros/imu/data*" (informação da *Attitude*)

- *time_boot_ms*;
- *orientation.x*;
- *orientation.y*;
- *orientation.z*;
- *linear_acceleration.x*;
- *linear_acceleration.y*;
- *linear_acceleration.z*.

A partir de Node.js, são enviados via Socket.IO para a UMA, de 5 em 5 segundos, pacotes JSON, "*attitude_data*" e "*gps_data*", que contêm respetivamente a informação da *Attitude* e do GPS.

A navegação autónoma é efetuada recorrendo à *framework* Ardupilot, um *software open source* presente nesta versão do Raspbian, que interage com os sistemas presentes no Navio2. Esta ferramenta tem um conjunto de "*frames*" pré-definido, para drones, submarinos, barcos e rovers e disponibiliza vários modos de navegação, configuração de parâmetros e mapeamento e navegação por trajetórias. O facto de ser *open source* permite que se possa personalizar conforme o utilizador necessitar, tal como neste caso em que se utilizou para configurar ROV. Para a criação de rotas e parametrizações do sistema, é possível utilizar *softwares* como o QGroundControl, que correm num computador externo ligado por rede ao Raspberry Pi.

No ROV desenvolvido, configurou-se o Ardupilot tendo por base a estrutura pré-existente para barcos, uma das configurações possíveis na versão ArduRover, a versão do Ardupilot

que permite a configuração de barcos e rovers. Através da documentação existente e de alguns testes, efetuaram-se as parametrizações e configurações associadas ao *hardware* utilizado, de forma a garantir o correto funcionamento desta ferramenta.
O Sistema Operativo para Robô (ROS) é uma estrutura flexível de desenvolvimento de *software* para robôs. Este é constituído por uma seleção de ferramentas que visam simplificar a tarefa de gerar *software* para sistemas robóticos e está disponível em licença de código aberto (*open source*).

Uma das características mais importantes do funcionamento deste *software* é a sua arquitetura, baseada em *Point-to-Point* ou *Peer-to-Peer*. Esta arquitetura é centrada no Mestre ROS que irá identificar os nós ou *nodes* e referenciá-los uns em relação aos outros, sendo os nós, os elementos responsáveis pelo processamento.

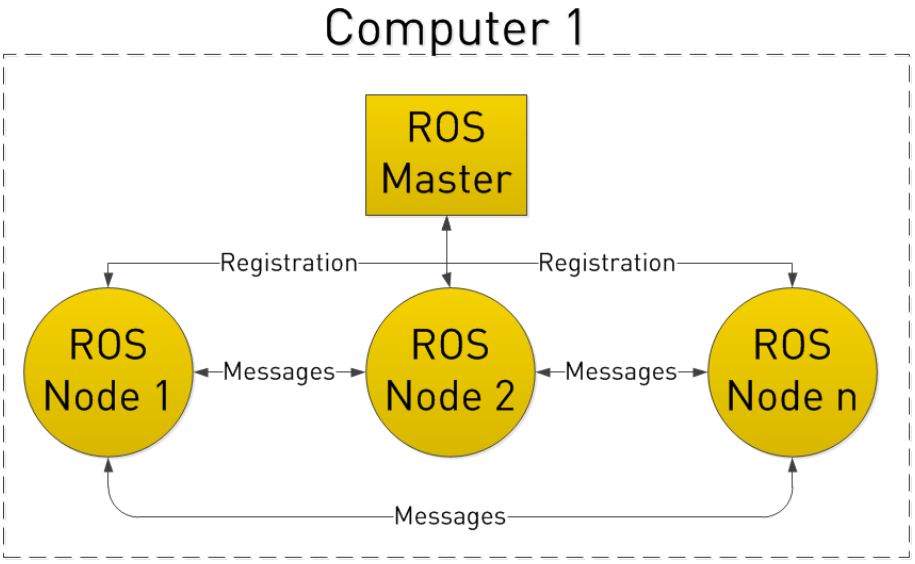


Figura 101 - Arquitetura de funcionamento do ROS

O processo de comunicação entre *nodes* baseia-se numa política de publicação/subscrição, ou seja, para enviar informação, um determinado *node* publica uma mensagem num tópico que é o canal de comunicação entre *nodes*. Assim, um outro *node* que necessite da informação publicada neste tópico, irá subscrevê-lo.

Os *nodes* são, na realidade, processos que executam algum tipo de tarefa de processamento. Por exemplo, um *node* pode controlar um sensor de distância, outro *node* controla outro tipo de sensor ou atuador, havendo um *node* associado a cada dispositivo.

A topologia de troca de informação utilizada vem facilitar a integração entre sistemas pois as comunicações são feitas ao nível dos tópicos, independentemente do tipo de dispositivo ou sistema que publicou a informação, aumentando a flexibilidade da troca de informação. Caso seja necessário enviar uma mensagem que devolva uma resposta, o ROS tem uma topologia de "Serviço" que permite enviar uma mensagem com um determinado formato e receber uma resposta.

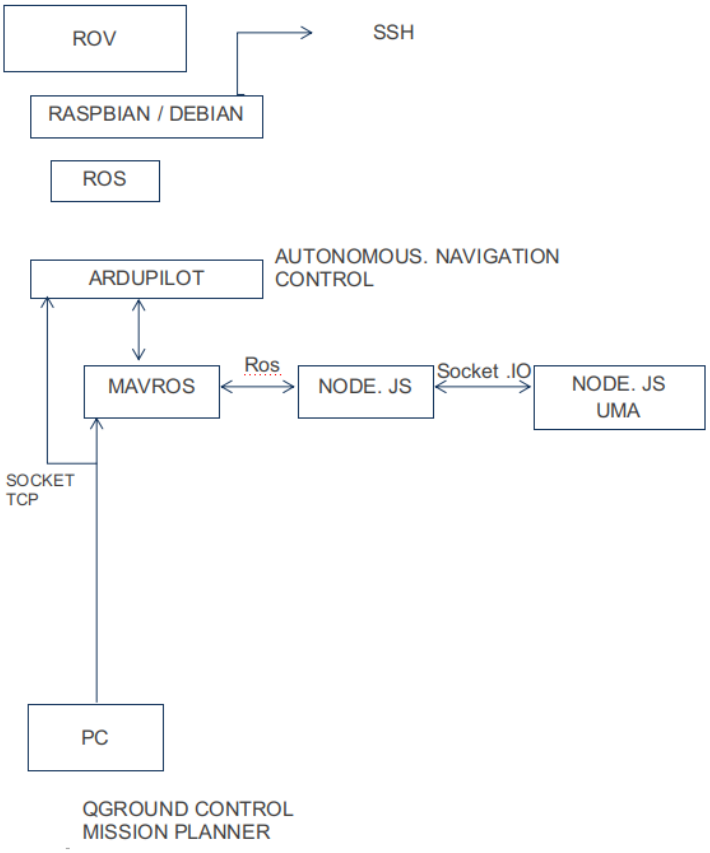


Figura 102 - Arquitetura de comunicações ROV

Tal como demonstrado na Figura 102, a utilização do ROS no sistema de comando de locomoção visa facilitar a obtenção dos parâmetros e dados transmitidos durante a navegação do ROV. Para isso, a aplicação desenvolvida em Node.js subscreve aos tópicos pretendidos e, automaticamente, recebe a informação que é gerada, sem a necessidade de comunicar o protocolo MAVLink pois o módulo "mavros" faz a conversão da informação para mensagens ROS.

Do ponto de vista funcional, o ROV aguarda o envio de uma ordem de produção por LoRa. Após a receção da ordem, a UMA envia a ordem para a UCB e para o ROV. O

ROV, ao receber a ordem, envia para o "Ardupilot" o comando "Arm" para este iniciar o trajeto que foi definido. A partir deste momento, o ROV inicia a missão e, através da UMA, inicia a transmissão, por LoRa, dos parâmetros obtidos até a ordem de produção ser concluída.

5.4.2 *Hardware* do ROV

O primeiro protótipo tinha por base o BlueRov2 da BlueRobtics, que tem por base o Raspberry Pi 3 e o sistema de navegação Pixhawk. A montagem deste primeiro protótipo é apresentada na Figura 103.

Figura 103 - Primeiro protótipo do ROV com base no BlueRov2

Do primeiro protótipo para o protótipo final, optou-se por alterar o *hardware* para o Raspberry Pi 3 B+ e para o sistema de navegação autónoma Navio2. Procedeu-se a esta alteração pois existiram algumas dificuldades em realizar alterações no *software* e no sistema existente do BlueRov2, de forma a ir ao encontro das necessidades pretendidas. Embora o sistema seja *open source* e permita algumas personalizações, a necessidade de adicionar um sistema de locomoção terrestre, bem como a necessidade de integrar no *software* as ações associadas às ordens de produção recebidas, levou a que se alterasse a estrutura do veículo e o seu *hardware* de navegação.

O módulo Navio2 é uma placa de integração para Raspberry Pi que contém os componentes necessários para a navegação de um veículo autónomo, nomeadamente GPS, acelerómetro, giroscópio, magnetómetro, barómetro, entre outros. Para interagir com este módulo, o Raspberry Pi 3 corre uma imagem de sistema operativo

Raspbian/Debian que, por sua vez, contem as funcionalidades necessárias e ferramentas de *software* para navegação e comando de dispositivos associados à locomoção.

A interface com as turbinas e motores é feita com recurso a controladores eletrónicos de velocidade (ESC) que recebem os sinais de PWM do Navio2 e, por sua vez, comandam os motores impondo determinada velocidade e sentido de rotação. Foi desenvolvido um controlador de PWM adicional (ver Anexo 10 – Circuito do Controlador PWM do ROV) para ser executado em paralelo com o Navio2. Este controlador faz uso de um microcontrolador ESP32 da Espressif para gerar vários tipos de sinais de PWM.

Este módulo de PWM permite a comutação entre navegação autónoma, recorrendo ao Navio2, e navegação manual, através de uma aplicação desenvolvida para este efeito. Esta aplicação permite comandar manualmente o ROV e variar velocidades de rotação das turbinas através de uma página Web servida pelo Raspberry Pi 3 desta unidade. A página permite efetuar configurações, enviar comandos de atuação e comandar o veículo, tanto em terra como na água, utilizando o teclado do computador. Os comandos para o controlador são enviados por protocolo Socket.IO para o servidor Node.js no RPi3. Este serviço envia um pacote JSON por USB para o ESP32 que, com base nos comandos contidos neste pacote, atua nos motores de terra ou nas turbinas, dependendo do modo de locomoção.

Para o controlo das turbinas, foram utilizados controladores eletrónicos de velocidade (ESC) "Basic ESC" da Bluerobotics para motores *brushless* de três fases. Este controlador permite alterar a velocidade e sentido de rotação através de sinal PWM. No caso do sistema desenvolvido, o sinal de PWM enviado para este controlador tem origem nos pinos de PWM do Navio2 ou nos terminais de PWM do módulo utilizado para permitir o comando manual do veículo.

O comando dos motores de deslocamento é efetuado utilizando a *drive* de comando "*Sabertooth Dual 5A Regenerative Motor Driver*". Esta *drive* foi escolhida por permitir uma tensão de alimentação de 14V e por estar preparada para responder ao sinal de PWM enviado pelo Navio2, de forma semelhante ao ESC. Esta *drive* tem disponíveis 5A de corrente para os motores conectados nos terminais de saída.

Foi também usada a *drive* Cytron MDD10A para atuar sobre os motores terrestres. Esta *drive* permite a ligação de dois motores em simultâneo e tem características superiores à

drive Sabertooth apresentada anteriormente, nomeadamente 10A de corrente disponível para os motores. Suporta dois tipos de sinal de PWM, "*sign-magnitude mode*" e "*locked anti-phase mode*". Neste enquadramento, foi utilizado o segundo modo pois é mais eficiente (reduz o aquecimento dos motores) e permite um controlo mais seguro tanto de paragem como sentido de rotação.

Para o deslocamento aquático foi escolhida a turbina "T200 Thruster" da Bluerobotics apresentada. Esta turbina é recomendada para veículos com alguma dimensão, sendo composta por um policarbonato resistente, com um isolamento epoxy. Recorre a hélices com plástico de alta performance em vez de metal, conferindo uma maior longevidade em água salgada. Esta turbina é ligada diretamente ao controlador eletrónico "Basic ESC" apresentado anteriormente.

Para o deslocamento terrestre, foram utilizados os motores NeveRest 40. Estes motores têm uma tensão de operação de 12V, compatível com as duas *drives* apresentadas, e uma caixa de velocidades com uma redução de 40:1 e 14W de potência e *encoder* incluído. Tendo em conta que o ROV não necessita de se deslocar a uma velocidade elevada, mas sim ter torque suficiente para se deslocar com o peso e ultrapassar obstáculos, optou-se por esta solução pelo facto de ter uma caixa redutora incluída. Esta redução faz com que se desloque a uma velocidade reduzida e tenha a força necessária para realizar os deslocamentos.

Como fonte de alimentação foram utilizadas baterias de iões de lítio de 4 células com 14,8V. Colocou-se uma bateria para alimentar os diversos componentes de navegação (turbinas, motores, Navio2 e Raspberry Pi) e outra bateria para a UCB, UMA e para o *switch* Ethernet, que interliga todos os sistemas. Estima-se que o conjunto das unidades com o ROV tenha um consumo aproximado de 3,5 a 4,5 Ah, portanto, o conjunto das duas baterias é suficiente para várias horas de utilização.

5.4.3 Estrutura Mecânica do ROV

Tal como descrito anteriormente, o primeiro protótipo deste ROV baseou-se na estrutura do BlueRov2. Devido à necessidade de adicionar características a este veículo, modificou-se a base da estrutura mecânica do ROV, de forma a dar resposta às várias necessidades existentes. A estrutura desta unidade, incluindo os componentes necessários para acoplar as restantes unidades, foi essencialmente projetada e contruída pelo engenheiro mecânico do LINE.IPT, com o contributo da restante equipa nas fases de montagem. Os principais requisitos deste sistema são:

- A capacidade de locomoção dentro e fora de água, permitindo que o veículo se movimente dentro dos tanques e entre eles;
- Sistema eletrónico para controlo da propulsão aquática e terrestre, sendo a locomoção terrestre efetuada recorrendo a quatro motores DC para movimento das rodas e a locomoção aquática com dois motores DC com hélices (turbinas);
- Acoplamento dos sistemas de visão com posicionamento debaixo de água com ajuste de altura;
- Acoplamento dos sensores de medição dos parâmetros da qualidade da água;
- Comando via *wireless* ou comunicação por cabo;
- Sistema de iluminação;
- *Housings* em tubos de acrílico para colocação dos componentes eletrónicos e encapsulamentos para os sistemas da UMA e da UCB;
- *Housings* para isolamento dos motores terrestres;
- Estrutura desenvolvida em 3D e construída com impressão 3D em polímeros (PLA e ABS) e possíveis peças em carbono ou inox.

Durante a fase de projeto da estrutura final, contruiu-se um segundo protótipo. Projetou-se esta estrutura com recurso a tubos e acopladores em PVC, de forma a definir a geometria e verificar tanto a flutuabilidade como a estabilidade em água.

Nas imagens abaixo, Figura 104 e Figura 105, são demonstrados, respetivamente, o modelo 3D desta estrutura e duas imagens deste protótipo, uma delas com o veículo colocado dentro de água.

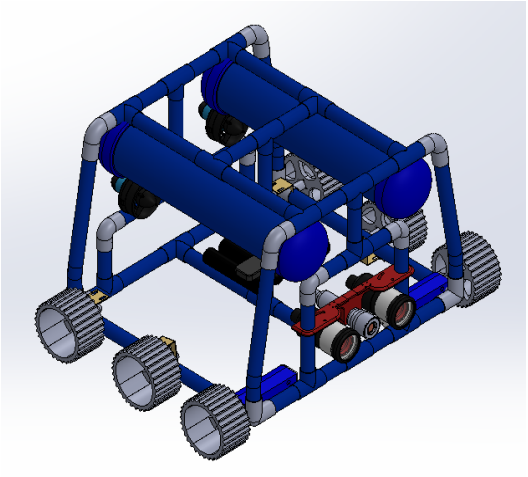


Figura 104 - Modelo 3D do segundo protótipo do ROV

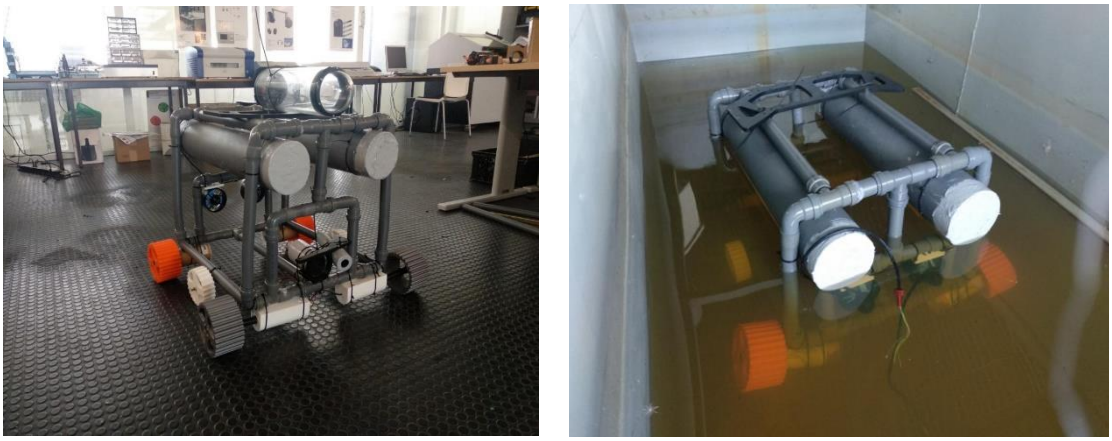


Figura 105 - Segundo protótipo do ROV

Uma vez efetuados todos os testes para definição da geometria, flutuabilidade e estabilidade, realizou-se a modelação 3D de uma estrutura base. Esta estrutura foi desenhada de modo a oferecer rigidez estrutural, de forma a suportar todos os *housings* e componentes necessários, sem que o seu peso impedisse uma boa flutuabilidade. Do protótipo em PVC, foram aproveitados os flutuadores, não só pela sua funcionalidade, mas também pela facilidade de troca por um flutuador de diâmetro maior, se fosse necessária a inserção de mais componentes que, por sua vez, aumentariam o peso da estrutura final.

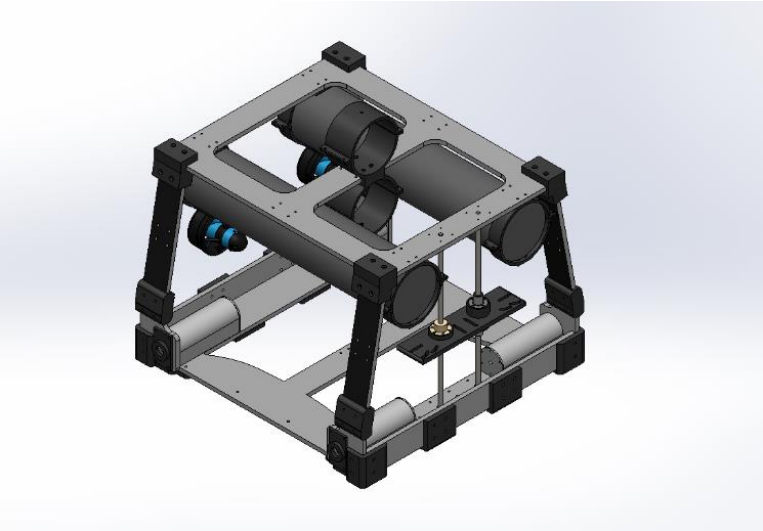


Figura 106 - Modelo 3D da estrutura final

Os elementos estruturais representados na Figura 106 foram feitos em polietileno de alta densidade (HDPE), com os acoplamentos impressos em plástico de poliácido láctico (PLA). O HDPE foi escolhido com base nos materiais usados na estrutura do BlueROV2, sendo um material relativamente leve e que apresenta uma boa rigidez. O PLA foi escolhido por ser uma solução económica, utilizado em fabricação aditiva (impressão 3D), permitindo construção de vários componentes e efetuar alterações às peças construídas com facilidade.

O processo de desenvolvimento de um *housing* apropriado para as câmaras foi um dos mais demorados e com mais obstáculos. Esta dificuldade estava associada à dificuldade de criar um *housing* à prova de água, que aguentasse longos períodos de tempo dentro de água, com peças feitas em impressão em 3D.

Uma peça de impressão 3D normal contém muitas microfissuras, espaços não visíveis ao olho humano, onde a água consegue penetrar. No caso deste *housing*, seria muito prejudicial, pois a água entraria em contacto com a câmara, potencialmente, danificando-a. Assim, todos os *housings* criados para as câmaras, o laser e os motores foram submetidos a tratamento de impermeabilização, impedindo a entrada de água.

Na Figura 107, é apresentado o modelo 3D de todos os componentes que constituem o *housing* das câmaras. Para além do encapsulamento exterior, nesta imagem são apresentados os componentes que permitem fixar o corpo da câmara e a lente dentro deste *housing*.

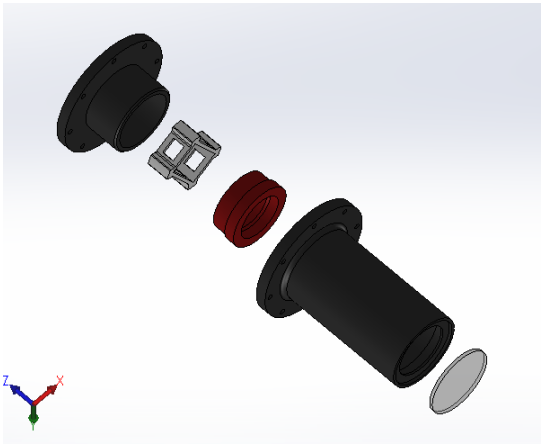


Figura 107 - Visão expandida dos vários elementos do *housing* das câmaras

No caso da câmara, existiram algumas dificuldades em garantir a não entrada de água pelas junções das várias peças do encapsulamento, tendo sido testadas várias versões de peças com diferentes tipo de montagem. No caso do laser, não existiram tantas dificuldades. Neste caso, o corpo do laser foi colocado dentro de um encapsulamento selado com uma tampa, contendo uma lente que permite a passagem da luz do laser. Obtiveram-se bons resultados com esta junção logo na primeira impressão e o método de impermeabilização usado, apesar de afetar o aspeto visual da peça, provou-se igualmente eficaz. Na Figura 108, é apresentado o modelo 3D do *housing* do laser.

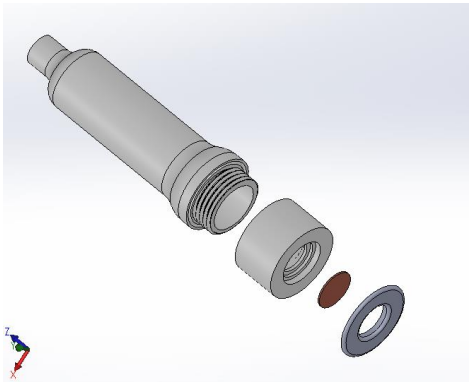


Figura 108 - Visão expandida dos vários elementos do *housing* do laser

Na Figura 109, é apresentado o modelo 3D final do ROV desenhado para o projeto Aquatropolis. Para além da estrutura base, estão incluídos os *housings* das câmaras, do laser e dos motores, bem como os *housings* e caixas colocadas no topo da estrutura, para acomodar os sistemas eletrónicos do controlo de locomoção do ROV, assim como os sistemas eletrónicos da UMA e da UCB. Na Figura 110, é visível este protótipo final, ainda sem o sistema de lagartas para a locomoção terrestre.

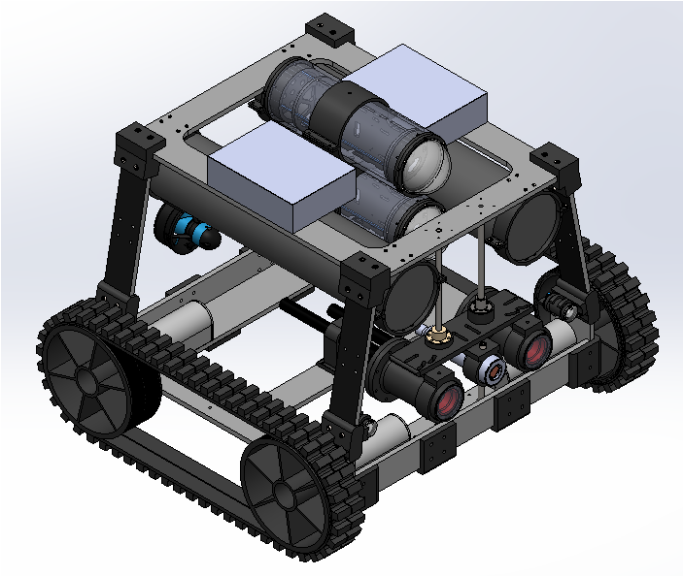


Figura 109 - Modelo 3D final do ROV

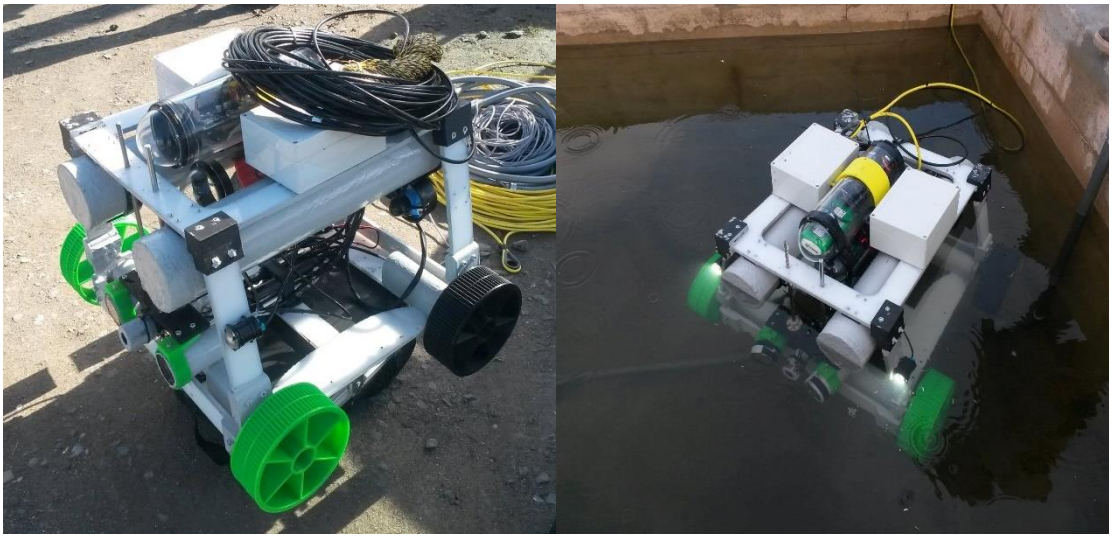


Figura 110 - Protótipo final do ROV

5.4.4 ROV: Resultados e Conclusões

5.4.4.1 Testes com a Estrutura do BlueRov2

Os primeiros testes de campo com ROV, com uma estrutura modificada do BlueRov2, tiveram como objetivo avaliar a mobilidade de um dispositivo móvel ao longo dos tanques de aquacultura e permitir a mobilidade dos sistemas da UMA nestes tanques. Tal como dito anteriormente, estes primeiros testes tiveram lugar em produções de aquacultura nas instalações das empresas ALGAplus, Aqualvor e Atlantikfish. Com esta estrutura, foram também realizados alguns testes em ambiente mais controlado, numa piscina e na albufeira de Castelo de Bode.

Para além de permitir os testes da UMA, estes testes permitiram avaliar qual o comportamento dos peixes quando colocado o veículo dentro dos tanques e perceber quais as implicações energéticas inerentes ao movimento do ROV na água, tanto nos tanques como na piscina e no rio, durante longos períodos de tempo.

Conclusões dos Testes com a Estrutura do BlueRov2

Com base nos testes efetuados, concluiu-se que não é benéfico movimentar o ROV a velocidades elevadas. Para além de, obviamente, levar a um consumo elevado das baterias, movimentar este veículo de forma muito brusca e rápida na água, provoca algum stress nos peixes, afugentando-os e levando a que estes dificilmente se aproximem desta unidade. Tal cenário não seria benéfico uma vez que se pretendia acoplar a UCB ao ROV. De um modo geral, os peixes demonstraram não estar muito à vontade com a presença do ROV no seu habitat, mesmo quando este permanece longos períodos de tempo imóvel.

Nestes testes, existiu alguma preocupação em verificar a estanquicidade dos *housings* utilizados. O facto de ser necessário proceder a algumas alterações da eletrónica dentro dos *housings*, levou a que existissem alguma folgas nas suas tampas e fechos. Posteriormente, tais considerações foram tidas em conta quando se desenhou e construiram os *housings* personalizados para os vários sistemas.

5.4.4.2 Testes do Protótipo Final do ROV com UMA e UCB

Para testar o protótipo final, efetuaram-se várias sessões nas instalações da ALGAplus, em Ílhavo. O objetivo destes testes foi aplicar todo o conhecimento obtido e desenvolvimentos feitos tendo em conta os resultados e experiências dos testes anteriores. O protótipo final do ROV foi aqui testado pela primeira vez num cenário real em tanques de terra e tanques de cimento. Nestes testes, a UMA e a UCB já se encontravam acopladas ao ROV e prontas a funcionar em simultâneo.

Com esta estrutura final para além dos testes de mobilidade aquática, procurou-se testar a mobilidade terrestre deste veículo, com e sem lagartas (as lagartas foram encaixadas em rodas dentadas, que permitem mobilidade sem elas).

Figura 111 - Deslocação do ROV em terra e na água

Relativamente à locomoção do ROV na água, procurou-se testar o sistema de locomoção autónoma e o modo de comando manual do sistema. Durantes os testes, procederam-se a alguns ajustes do posicionamento dos *housings* e caixas, de forma a equilibrar o peso da estrutura e garantir uma boa flutuabilidade.

Testaram-se várias versões de *housings*, de modo a assegurar que o encapsulamento utilizado se mantinha impermeável. A versão final apresentou um comportamento estável, aparentando não permitir a entrada de água. No entanto, seria necessário manter

a estrutura dentro de água durante algum tempo para perceber qual o efeito da água salgada nas junções e encaixas que mantinham o *housing* fechado.

Durante os testes de locomoção autónoma e manual do ROV, procurou-se parametrizar a potência dos motores, de modo a garantir uma locomoção equilibrada. Verificou-se, ao movimentar o veículo de forma muito rápida e brusca, para além das questões anteriores relativas ao consumo energético e ao comportamento dos peixes, que era provocado um desequilíbrio da estrutura durante o movimento. Um movimento mais suave permitia manter o veículo mais equilibrado.

Colocou-se o ROV dentro dos tanques de cimento, para testar a UCB e a UMA. Não foi possível obter resultados conclusivos relativamente à locomoção devido às dimensões reduzidas destes tanques, que dificultaram a movimentação do veículo.

Para testar a locomoção autónoma, utilizou-se o *software* QGroundControl para estabelecer previamente rotas a seguir, como demonstrado na Figura 112.

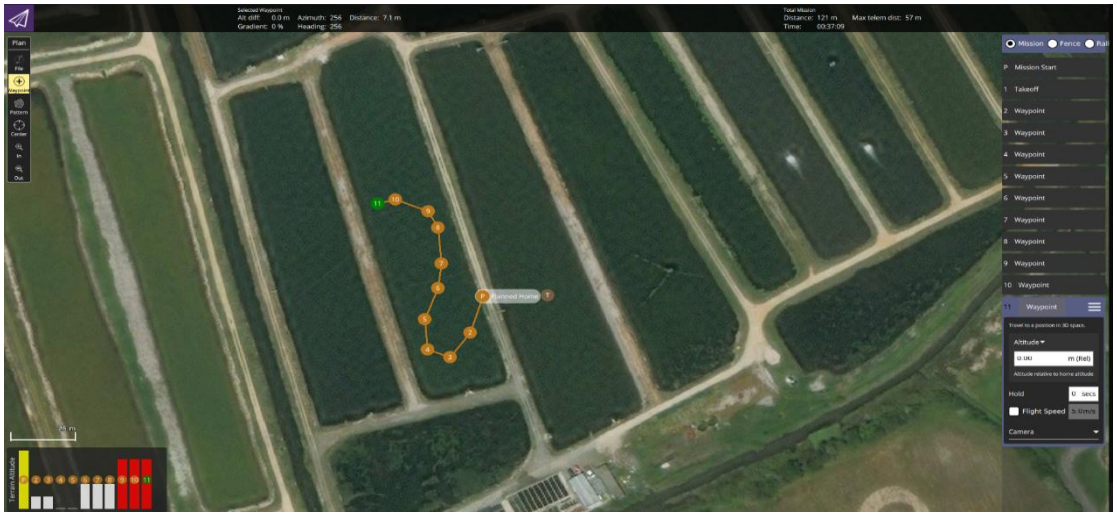


Figura 112 - Planeamento de rotas de movimento

Após a receção correta da rota e obtenção do sinal GPS (necessário para iniciar a navegação) era enviado o comando de "*Start mission*", dando assim início ao movimento autónomo do veículo. Inicialmente, o ROV apresentou um comportamento irregular entre pontos de navegação. Concluiu-se que esta situação se devia a um erro de parametrização do sinal enviado para os controladores das turbinas. Após a correção do erro, o ROV fez a navegação do percurso planeado, mantendo um movimento estável e equilibrado.

Concluiu-se também que a precisão do GPS pode causar limitações numa navegação mais precisa. Idealmente, pretendia-se que o ROV se conseguisse movimentar entre tanques, por terra, saindo e entrado dentro de água por rampas. Nestes testes, verificou-se que o GPS tinha um erro de localização de 3 a 4 metros. O facto de o GPS ser pouco preciso dificultou tal possibilidade, tendo sido descartado este cenário de movimentação. Concluiu-se que o GPS utilizado, bem como o sistema de movimentação autónoma utilizado, seriam um dos pontos a melhorar em trabalhos futuros. Para além da imprecisão do GPS, seria necessário mais alguns desenvolvimentos deste sistema, de forma a permitir que este sistema fosse totalmente autónomo.

Relativamente ao movimento terrestre, o sistema não revelou grandes problemas. O ROV movimentou-se nas imediações destes tanques de aquacultura, em piso irregular. No entanto, é de salientar que acabou por se descartar o uso das correias ou lagartas, visto que se verificou alguma dificuldade em tensioná-las de forma a não saltarem das rodas. Experimentaram-se algumas peças de fixação e tensão destas lagartas mas, em última instância, optou-se por retirar as lagartas das rodas dentadas.

Em suma, o sistema desenvolvido correspondeu a alguns dos objetivos pretendidos. O sistema do veículo desenvolvido é capaz de se mover tanto na água com em terra, transportando a UMA e a UCB. O ROV tem alguma capacidade de locomoção autónoma ou "semiautónoma", uma vez que necessita de uma rota previamente definida e de um sinal de início de movimento. Para desenvolver um sistema autónomo, seria necessário melhorar os algoritmos de locomoção autónoma e equipar o veículo com mais alguns sensores de ajuda à navegação e um GPS com maior precisão.

A nível estrutural, concluiu-se que existem algumas melhorias que poderiam ser efetuadas, nomeadamente na escolha do material para a estrutura. O HDPE apesar de funcional, a longo prazo apresenta deformações devido a fluência, isto é, aplicação de cargas constantes ao longo de um período de tempo.

Concluiu-se também que se poderia melhorar o posicionamento do sistema de câmaras. Neste protótipo, o par de câmaras e o laser eram suportados por uma base que permitia movimentar o sistema verticalmente. No entanto, verificou-se que esta opção era desnecessária. Percebeu-se que o ideal seria fixar as câmaras e o laser na zona inferior do ROV e, em vez de colocar estes equipamentos apontados para a frente, seria desejável

colocá-los ligeiramente apontados para baixo. Ao descartar o sistema de movimentação vertical, seria possível tornar o ROV uma estrutura mais compacta, diminuindo a sua altura total, ao reduzir os espaços vazios entre o elemento estrutural superior e inferior. Esta redução de altura também permitiria melhorar a estabilidade em navegação em terra ao baixar o centro de gravidade do veículo.

Outras das limitações encontradas foi a impressão 3D de algumas das peças. Algumas destas peças acabaram por se partir depois de alguns testes. Esta complicação deveu-se ao peso da estrutura colocado sobre algumas destas peças, bem como a corrosão provocada pela água salgada, que conseguiu penetrar as peças não tratadas, reduzindo a sua vida útil e tornando-as menos resistentes. Assim, seria recomendado recorrer a materiais menos porosos que o PLA. Nesta ótica, concluiu-se que se deveria recorrer a outro processo de produção dos *housings* das câmaras, laser e motores.

O processo utilizado para fabrico e impermeabilização, apesar de eficaz, é complexo e não garante que as peças se mantenham impermeabilizadas a longo prazo. Para além disto, os *housings* desenvolvidos dificultam o acesso aos equipamentos colocados dentro deles, sendo este outro dos pontos a melhorar.

5.5 Conclusões Projeto Aquatropolis

No âmbito deste estágio no LINE.IPT, o aluno integrou os trabalhos de desenvolvimento do projeto Aquatropolis - *Intelligent Management System for Sustainable Aquacultures*, praticamente desde o seu início. Apesar de algumas das limitações relatadas, a solução apresentada distingue-se pela inovação tecnológica e cumpre a grande maioria dos objetivos inicialmente propostos.

Nos trabalhos de desenvolvimento da Unidade de Monitorização Ambiental, UMA, foi desenvolvido *hardware*, *firmware* e *software* com integração das comunicações LoRa, implementação de um sistema de monitorização ambiental, recorrendo a sondas multi-parâmetro. Através de tecnologia modular e facilmente escalável, este sistema possibilita a adição de novas sondas sem necessidade de desenvolvimentos de novo *hardware*. O *software* desenvolvido tem por base linguagens de programação Web *open source*. Devido a esta característica a plataforma Web pode ser utilizada em várias plataformas, com diferentes recursos de *hardware* e diferentes sistemas operativos, sem necessidade de adaptações. O *software* desenvolvido permite facilmente a adição e configuração de novos sensores, reforçando a modularidade e escalabilidade do sistema. Permite ainda visualizar os dados qualidade da água em tempo real, consulta de históricos, escolha do modo de comunicação, entre outros parâmetros. Foi realizada também a integração do sistema desenvolvido pelo IPT/TGV com o parceiro Domatica, recorrendo à tecnologia de comunicação LoRa. A mobilidade de todo o sistema foi assegurada pelo ROV, um veículo anfíbio, que se move com alguma autonomia em ambiente aquático e terrestre.

No que diz respeito à unidade de controlo da biomassa (UCB), recorre-se a um par de câmaras configuradas em estéreo e um foco de iluminação laser com um padrão de vinte e cinco linhas. Recorrendo à biblioteca de processamento de imagem MVTec HALCON, desenvolveram-se algoritmos de processamento de imagem que permitem efetuar a estimativa da massa de peixes presentes nas imagens obtidas com esta montagem estéreo. A configuração utilizada e os algoritmos desenvolvidos permitem, dentro de certos limites de turbidez e luminosidade, identificar os peixes nas imagens e efetuar a estimativa da sua massa com base no comprimento do corpo. Devido às condições de elevada turbidez e iluminação pouco homogénea nos tanques, o algoritmo de estimativa da massa desenvolvido não funcionou corretamente quando utilizado em imagens obtidas em tanques de aquacultura. Apenas se obtiveram resultados positivos com este algoritmo,

quando utilizado em imagens obtidas em ambiente controlado, na bancada de trabalho do laboratório e nos testes efetuados no rio.

O sistema ROV implementado incorpora a unidade UMA e a UCB, permitindo a expansão de cada uma das unidades, caso seja necessário adicionar novo *hardware*. O veículo robotizado desenvolvido tem capacidade para deslocamento aquático e terrestre e pode ser comandado manualmente ou funcionar de forma autónoma, com algumas limitações neste último caso. Foram desenvolvidos componentes e *housings* utilizando métodos de impressão 3D, permitindo uma redução de custos muito considerável comparativamente aos *housings* existentes no mercado. No entanto, há espaço para melhorias na escolha dos materiais utilizados, bem como no processo de fabrico destes sistemas.

De forma a avaliar o comportamento do sistema a longo prazo, o protótipo final foi colocado dentro de um dos tanques nas instalações da ALGAplus, em Ílhavo, local onde se encontra há alguns meses. Deste modo, a UMA tem estado a recolher dados da qualidade da água permitindo perceber a estabilidade do *software* desenvolvido. Este teste é pertinente pois permite igualmente perceber qual a evolução da degradação dos materiais da estrutura do ROV. A UCB encontra-se desativada, pois optou-se por retirar as câmaras desta estrutura, receando que os *housings* não se mantivessem intactos. Tal como se pode visualizar na Figura 113, durante os meses que esteve dentro de água a estrutura veio a acumular bastantes algas e outros organismos marinhos.

Figura 113 - Estrutura do ROV após alguns meses dentro de um tanque de aquacultura

Com base nestes resultados, pode concluir-se que uma estrutura deste género não aguentará muito tempo nestas condições caso os materiais não sejam apropriados à água salgada e não se promovam lavagens regulares da estrutura e sistemas.

No âmbito deste projeto, houve uma sessão de avaliação do trabalho desenvolvido a 22 de outubro de 2019. Assim, preparou-se um protótipo simplificado do sistema desenvolvido para apresentar nesta sessão, de modo a permitir que tanto a equipa do LINE como do IPT possam ter uma base de desenvolvimento para projetos futuros. Esta nova estrutura tem por base o esqueleto do BlueRov2, com dois flutuadores adicionados produzidos a partir de tubos de PVC, tal como a estrutura do protótipo anterior. Nesta estrutura, acoplaram-se igualmente os sistemas da UMA e da UCB. No caso desta segunda unidade, para proteger as câmaras, recorreu-se aos *housings* feitos em metal utilizados anteriormente para testes de campo. Na Figura 114, é possível visualizar a estrutura criada para a apresentação final.

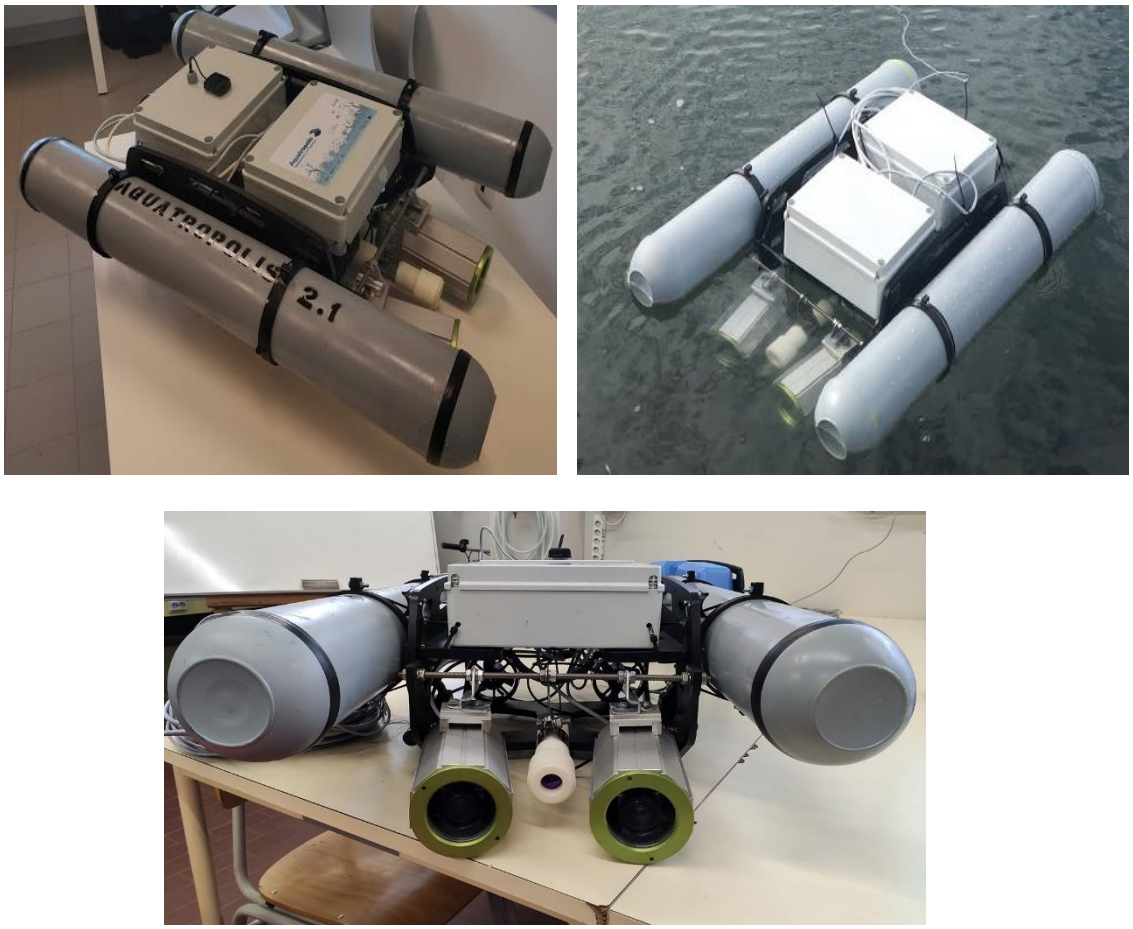


Figura 114 - Protótipo utilizado para apresentação final do projeto Aquatropolis

6 Projeto Vestas

6.1 Descrição e Enquadramento do Projeto

No âmbito do estágio no LINE.IPT que é descrito neste documento, um dos projetos em que o aluno participou foi com a empresa Vestas Portugal. A Vestas é uma empresa do setor das energias renováveis que se dedica ao projeto, fabrico, instalação e manutenção de turbinas eólicas. Esta empresa dinamarquesa é líder do mercado das turbinas eólicas, com um volume de produção mundial de 105 GW em 80 países, correspondente a 17% da produção mundial de energia com turbinas eólicas. Em território nacional a empresa é representada pela Vestas Portugal, que forneceu 384 turbinas distribuídas por vários parques, correspondendo a uma produção total de 698 MW. [56]

No projeto desenvolvido em parceria com a empresa Vestas pretendia-se desenvolver uma solução que, através de câmaras e processamento de imagem, conseguisse quantificar o desgaste dos dentes das rodas dentadas, parte integrante das torres eólicas desta empresa. Ao desenvolver esta solução, pretendia-se criar uma ferramenta que visava acelerar as tarefas de manutenção das citadas rodas dentadas, por parte das equipas de manutenção da Vestas. A roda dentada que se pretende analisar é parte integrante do sistema que permite girar o corpo da *nacelle* da turbina eólica em torno da torre, a chamada *yaw ring* ou *yaw bearing*. Na Figura 115, é demonstrada a posição deste componente em relação aos restantes elementos de uma turbina eólica. Esta engrenagem vai sofrendo algum desgaste ao longo da sua vida útil, associado por exemplo à força aplicada pelos motores a este mecanismo, aquando da rotação do corpo da *nacelle*.

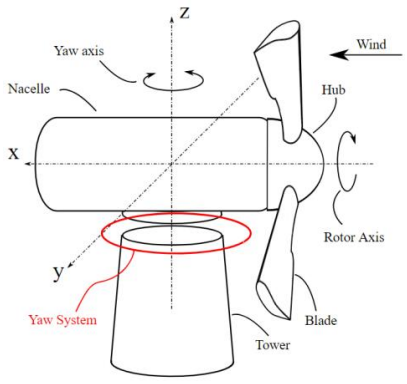


Figura 115 - Esquema dos principais componentes de uma turbina eólica. (Fonte: Wikipédia) [57]

Estas *yaw rings* são engrenagens compostas por 143 dentes. Na Figura 116, é apresentada uma fotografia representativa destas engrenagens.

Figura 116 – Aspeto típico de uma engrenagem de *yaw ring* (Fonte: http://www.china-forging.biz/yaw-gear-1.html)

Na Figura 117, é possível visualizar qual o aspeto dos dentes desta engrenagem quando estão intactos e sem qualquer desgaste.

Figura 117 - Segmento da engrenagem onde se veem três dentes

Ao longo da vida útil deste componente, os dentes vão sofrendo desgastes resultando num estreitamento do contacto das "zonas de ataque" de cada dente. Estas zonas são as que recebem mais força por parte das engrenagens acopladas aos motores que provocam a rotação da *nacelle*. A evolução do desgaste destes dentes leva a que estes sofram deformações (em que deixam de apresentar as sua formas retas habituais, ficando com as arestas arredondadas, em forma de crista), resultantes de um "arrastamento de metal" que é transferido para a frente dos dentes, tal como apresentado na Figura 118.

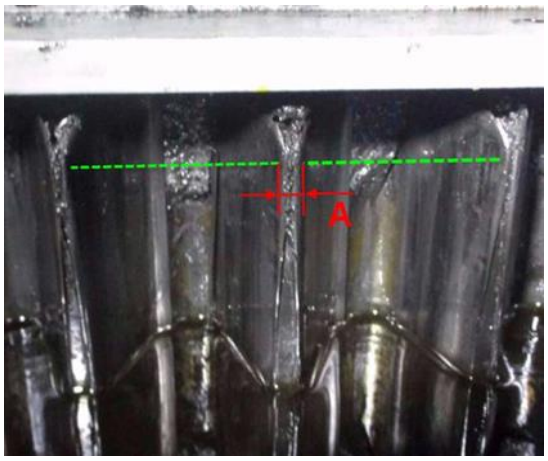


Figura 118 - Dentes da *yaw ring* que sofreram deformação (Fonte: Vestas Portugal)

A avaliação do desgaste do dente é efetuada obtendo várias medidas do seu perfil. Considera-se que um dente deixa de ter condições para permitir o correto funcionamento, quando uma das medidas do perfil está mais de 5 mm abaixo da dimensão original. Nestas condições são aplicadas medidas corretivas, normalmente o dente é removido e é colocado um novo. Caso existam vários dentes consecutivos danificados, pode colocar-se um segmento novo, que corresponde a seis dentes. Caso alguma das medidas do perfil seja 3 mm inferior à dimensão original, o dente é sinalizado de forma a acompanhar atentamente o seu desgaste, visto que um desgaste de 5 mm é considerado severo e poderá levar a falhas ou acidentes no sistema de *yaw*.

Atualmente, as equipas de manutenção da Vestas utilizam um processo manual de medição do desgaste dos dentes, recorrendo a um comparador de precisão. Apesar desta precisão, trata-se de um processo demorado no qual facilmente se podem cometer erros. Atualmente, este processo de medição é precedido por uma observação dos dentes de forma a procurar aqueles que aparentem um desgaste notável ou algum tipo de fissura. Procede-se deste modo porquanto a medição dos dentes toma muito tempo nos trabalhos de manutenção, procurando-se assim evitar medições desnecessárias. Contudo, este tipo de procedimento de observação está igualmente sujeito a falhas humanas, daí a necessidade da utilização de um sistema de medição através de visão por computador, tornando a medição mais célere e acompanhando mais rigorosamente o desgaste das *yaw ring*.

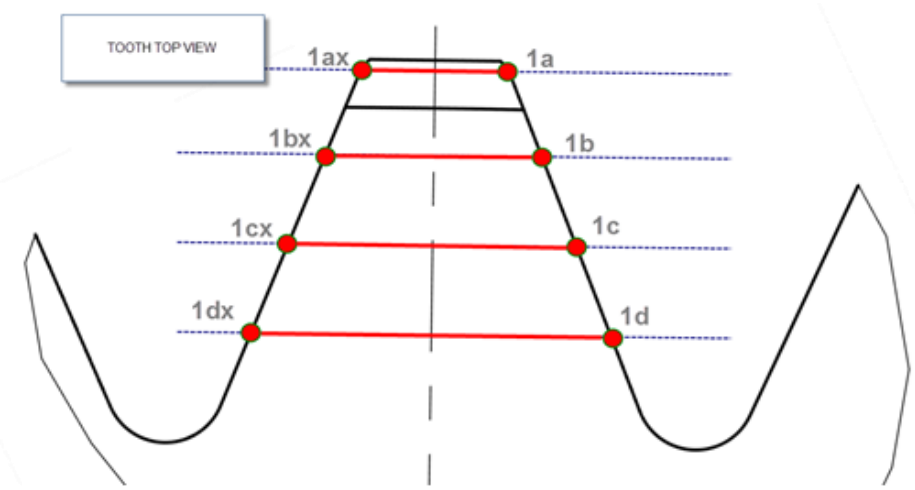


Figura 119 - Vista de topo do esquema de medição do dente (Fonte: Vestas Portugal)

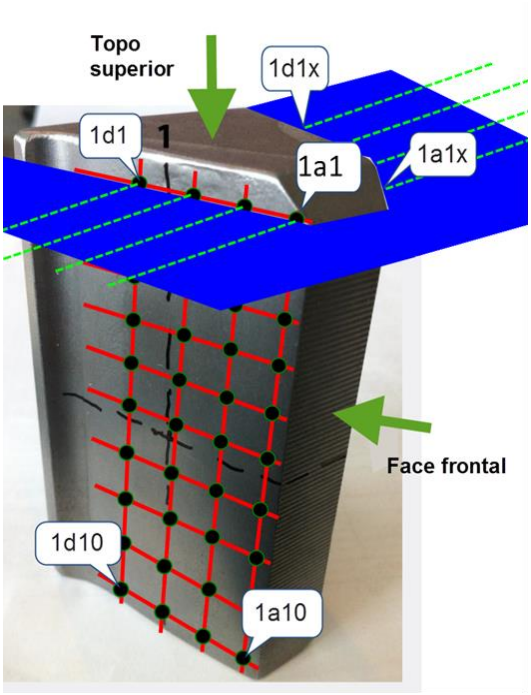


Figura 120 -Esquema para medições do dente "Grelha" (Fonte: Vestas Portugal)

Com o auxílio da Figura 120 e da Figura 119, podem definir-se quais os requisitos de medição que este sistema terá de cumprir. É necessário definir, pelo menos, oito pontos de medição distribuídos ao longo da altura do dente, ou seja, devem existir, no mínimo, oito cotas de medição. Em cada cota de medição é necessário obter quatro valores de distância entre pontos opostos das faces dos dentes, sendo que cada medida deve distar 10 mm da anterior. O primeiro par de pontos de medição tem de distar 3 mm da face frontal do dente e 5 mm do topo superior do dente. Com base nestes requisitos, é possível obter uma "grelha" de medições tal como demonstrado na Figura 120.

Assim, atendendo às referências apresentadas na Figura 120 e na Figura 119, as medidas a obter para cada dente seriam:

Tabela 17 - Distâncias do perfil do dente que devem ser obtidas

	Dente nº1	**…**	**Dente nº143**
Cota 1	Distância 1a1-1a1x	…	Distância 143a1-143a1x
	Distancia 1b1-1b1x	…	Distancia 143b1-143b1x
	Distancia 1c1-1c1x	…	Distancia 143c1-143c1x
	Distancia 1d1-1d1x	…	Distancia 143d1-143d1x
Cota 2	Distância 1a2-1a2x	…	Distância 143a2-143a2x
	Distancia 1b2-1b2x	…	Distancia 143b2-143b2x
	Distancia 1c2-1c2x	…	Distancia 143c2-143c2x
	Distancia 1d2-1d2x	…	Distancia 143d2-143d2x
…	…	…	…
Cota 10	Distância 1a10-1a10x	…	Distância 143a10-143a10x
	Distancia 1b10-1b10x	…	Distancia 143b10-143b10x
	Distancia 1c10-1c10x	…	Distancia 143c10-143c10x
	Distancia 1d10-1d10x	…	Distancia 143d10-143d10x

O sistema a desenvolver teria de efetuar medições mesmo em condições de desgaste severo, como por exemplo na situação apresentada na Figura 121. Neste caso, o sistema deveria ter em conta que o perfil frontal do dente corresponde à reta *t* e a medição a efetuar é referenciada pelo intervalo *c*.

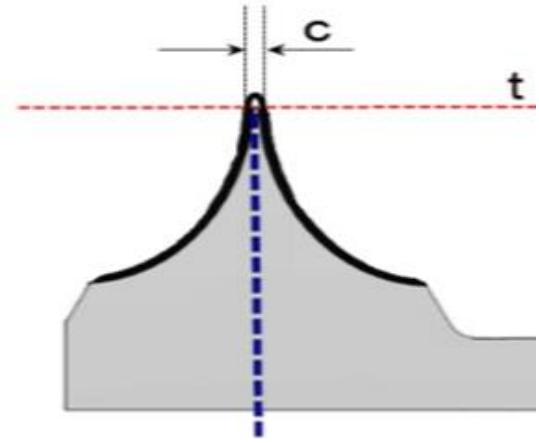


Figura 121 - Esquema representativo de um desgaste severo de um dente (Fonte: Vestas Portugal)

Outras das situações que o sistema teria de conseguir aquilatar seria o caso da Figura 122, em que existe igualmente um desgaste, não tão acentuado como no esquema da Figura 121. Nesta situação, devido ao desgaste, há arrastamento de material para a face frontal do dente, o que dificulta a definição visual desta face. O sistema de medição a desenvolver teria de discernir que a face frontal do dente corresponde à linha *t*, tracejada a vermelho no pormenor *K* da Figura 122. O intervalo de medição correto neste caso é a distância *A*.

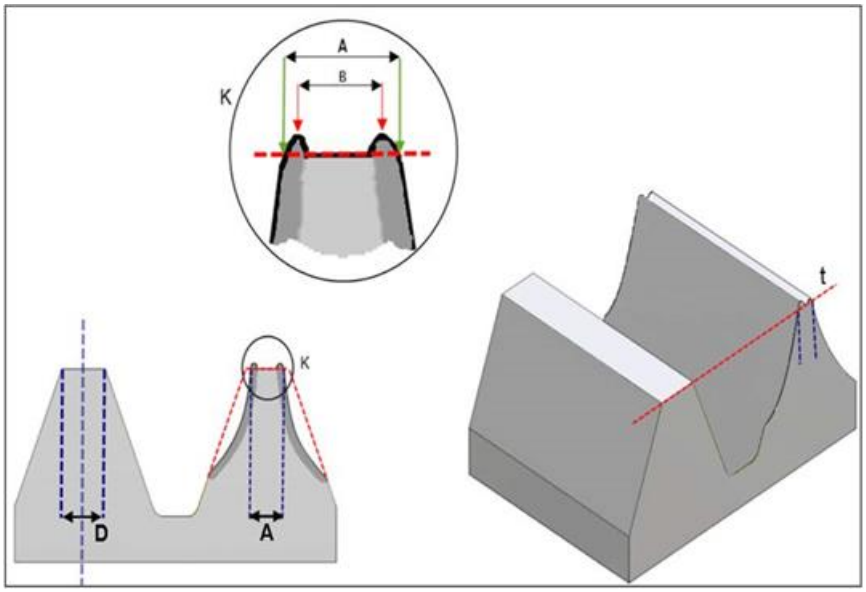


Figura 122 - Esquema representativo do desgaste do dente (Fonte: Vestas Portugal)

O *software* deveria identificar incongruências das medidas, ou seja, alterações anormais entre distâncias de pontos consecutivos que implicariam a existência de um dente partido ou uma falha na medição, e alertar o utilizador do sistema.

O sistema a desenvolver teria de armazenar numa base de dados a informação relativa às medições obtidas, referenciando-as com a data e hora de recolha. Com base nestes registos, o *software* teria de permitir efetuar a consulta de dados recolhidos anteriormente, de forma a permitir comparar um histórico de medições com as medidas atuais. O *software* teria ainda de prever que nem sempre é possível obter acesso à internet quando se opera dentro de uma turbina eólica. Assim, contemplaria um mecanismo que permitisse acesso ao histórico de medições na base de dados, sem ligação à rede.

Finalmente, o *software* deveria gerar um relatório das medições efetuadas, onde para além dos valores medidos, deveriam constar as seguintes informações:

- Nº série da turbina eólica;
- Nome do parque eólico;
- Tipo da turbina;
- Data da última inspeção;
- Data da inspeção atual;
- Hora da recolha das medições;
- Iniciais do técnico que fez a inspeção.

Para além disto, o relatório com os resultados de medição conteria também um comparativo entre a medição anterior e a atual, ou caso não existisse medição anterior, entre as medidas de um dente intacto "padrão" e a medição atual.

Com base nestes pressupostos, iniciou-se o desenvolvimento de um sistema de visão artificial que permitisse obter os valores de medição desejados. O sistema que se veio a desenvolver baseia-se no princípio da triangulação laser descrito anteriormente. No entanto, o sistema desenvolvido não corresponde a uma triangulação laser "tradicional". Na configuração efetuada para este sistema de medição, a matriz laser não permite obter um modelo 3D completo do objeto em análise. Nesta situação a matriz laser é utilizada apenas para aferir um conjunto de pontos num referencial *X*, *Y* e *Z*, coordenadas que permitem quantificar o desgaste dos dentes. Outra das particularidades deste sistema de triangulação laser é o facto de utilizar duas câmaras.

Para o desenvolvimento do programa de processamento de imagem utilizou-se a biblioteca HALCON e o respetivo IDE.

Na solução final pretendia-se que a ferramenta de processamento de imagem interagisse com um *software* de base dados das medições efetuadas, realizasse comparações com valores anteriormente obtidos e gerasse relatórios associados a cada ciclo de medições dos dentes de uma engrenagem. A plataforma de interação com o utilizador é baseada numa página Web em PHP servida por um processo de Node.js. Este servidor Node.js comunica com a aplicação de processamento de imagem via SocketTCP. Os dados são guardados localmente e também num servidor externo, sempre que o computador é ligado à internet, em bases de dados MySQL. Na Figura 123 é apresentado o esquema que

representa a interligação entre o *hardware* de medição, o *software* de processamento de imagem e a plataforma de interação com o utilizador bem como as componentes de *software* inerentes a ela.

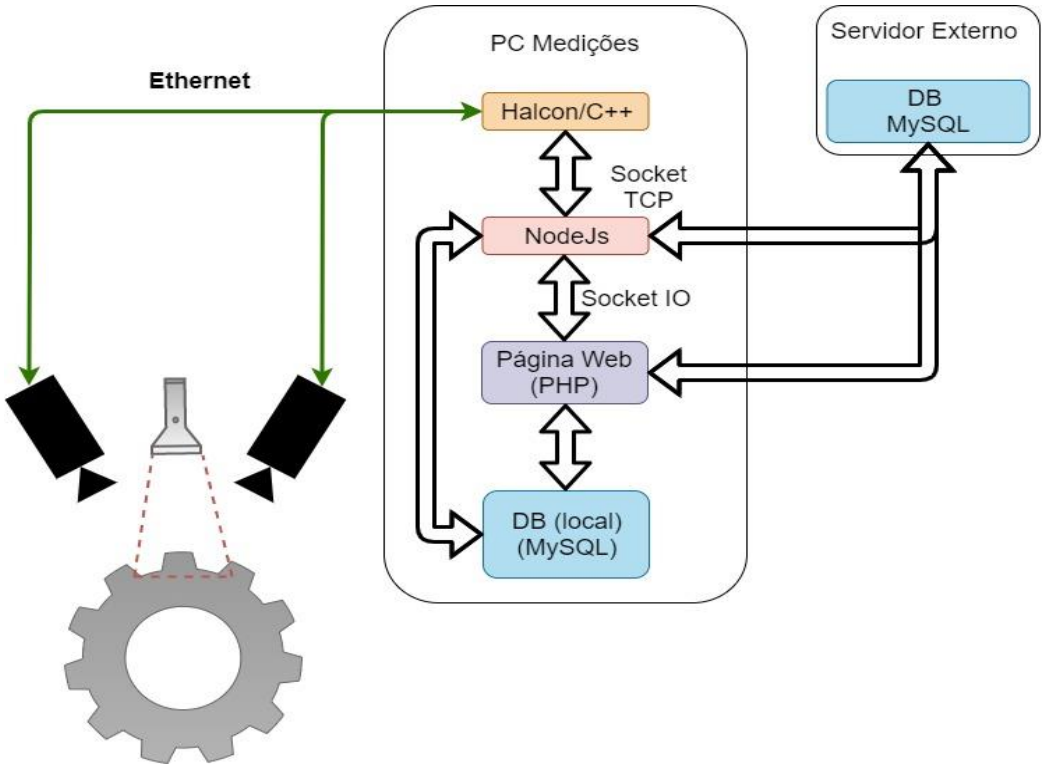


Figura 123 - Esquema da arquitetura do sistema de medição do desgaste dos dentes das engrenagens

6.2 Descrição de *Hardware*

Tal como descrito na introdução relativa a este projeto, o sistema de medição do desgaste de rodas dentadas que se pretendia desenvolver baseia-se num par de câmaras em conjunto com um foco de iluminação laser, ligadas a um computador que permitirá a captura e processamento de imagens.

A solução desenvolvida recorre a um par de câmaras Teledyne Dalsa Genie Nano M2420, com lentes de 12 mm de distância focal e um laser Z-Laser ZM18S3 equipado com filtro para projeção de onze linhas. As especificações das câmaras e do laser utilizado, que têm as mesmas referências que o material utilizado no Projeto Aquatropolis, são apresentadas no subcapítulo *Hardware* da UCB, relativo a esse projeto. Em alternativa ao material utilizado nesse projeto, optou-se pela utilização de lentes com distância focal de 12 mm, após se ter verificado que o campo de visão destas seria suficiente para obter imagens do corpo de um dente destas engrenagens e ainda algum espaço envolvente. Chegou-se a

esta conclusão após consultar a empresa que fornece ao LINE.IPT o material e serviços para os projetos de visão por computador. Foi ainda possível testar várias referências de lentes de forma a escolher a que mais se adequava às dimensões do objeto a analisar.

A utilização de duas câmaras justifica-se porquanto é necessária, em todos os cenários e independentemente do estado do dente analisado, a visualização de todo o corpo do dente bem como a projeção das linhas laser nas suas faces. Com uma câmara apenas, haveria dificuldade em visualizar com clareza as faces laterias do dente e a projeção do laser nelas, e tal dificuldade seria agravada pelo desgaste do dente. Também, com uma única câmara, numa situação de desgaste severo, como a demonstrada anteriormente na Figura 121 e na Figura 122, as faces laterais do dente não seriam simultaneamente visíveis.

Assim, testaram-se várias opções de posicionamento das câmaras. Estas deveriam estar alinhadas, uma com a outra, e acopladas numa peça única que permitisse ajustar o seu ângulo em relação ao plano da face frontal dos dentes.

O laser não foi colocado no mesmo alinhamento para que existisse um ângulo entre as câmaras e o seu feixe, de modo a efetuar uma triangulação. Na Figura 124, é observável uma das montagens testadas e na Figura 125 é demonstrada outra montagem com um posicionamento diferente do laser.

Nestas figuras, são visíveis as peças que compõem os protótipos. Existe uma peça base que fixa o equipamento a uma viga. Acoplada a esta base está uma outra peça que permite ajustar a distância dos equipamentos à engrenagem. O acoplamento desta peça à base de fixação permite algum ajuste, neste caso apenas em altura. No componente que permite o ajuste de distância, é encaixada a peça que sustenta o sistema de fixação das câmaras e igualmente a peça de fixação do laser.

Os encaixes de suporte das câmaras e do laser são desenhados de forma a ser possível o ajuste do ângulo daqueles órgãos. Esta descrição corresponde exatamente ao sistema de montagem da Figura 124. O da Figura 125 é ligeiramente diferente, pois a peça onde deveria encaixar a base das câmaras e a base do laser, apenas tem um encaixe, onde é ligada a peça de fixação do laser. Neste caso, a peça da base das câmaras é ligada à peça do laser.

A última versão deste sistema de fixação de componentes será apresentada no capítulo das conclusões e resultados deste projeto.

Figura 124 - Um dos protótipos de medição montado na zona de acesso à *yaw ring*, permitindo ver o espaço envolvente

Figura 125 - Protótipo do sistema de medição montado na zona de acesso à *yaw ring*

Apesar de ser uma montagem com duas câmaras, esta configuração não constitui um sistema estéreo. As imagens das duas câmaras são utilizadas pelo *software* para obter uma imagem completa do dente, "colando" o par de imagens como se a imagem fosse obtida por uma só câmara.

No modelo mais comum das turbinas eólicas da Vestas Portugal, para as quais se pretendeu desenvolver esta ferramenta, o acesso ao compartimento da *yaw ring*, onde

deve ser colocado este sistema de determinação do desgaste, é um espaço limitado sem capacidade para acolher material de grandes dimensões. Devido a este constrangimento optou-se pela configuração aqui apresentada. Na Figura 125 e na Figura 124, é possível perceber esta limitação espacial.

Os suportes das câmaras e do laser utilizados nos protótipos de desenvolvimento e testes desta ferramenta foram fabricados através de impressão 3D pelo engenheiro mecânico da equipa do LINE.IPT. Estes suportes foram sofrendo várias alterações ao longo do desenvolvimento da solução, devido às dificuldades de posicionamento do sistema de câmaras e laser naquele espaço reduzido, mas também a algumas vicissitudes relacionadas com a calibração do laser e das câmaras.

Nesta fase, optou-se por fabricar peças 3D que pudessem ajudar a simular o ambiente onde este sistema opera. Assim, criou-se uma peça 3D que simula o rebordo do compartimento que envolve a *yaw ring* e que funciona como contentor do óleo de lubrificação desta engrenagem e ainda uma peça que emula o rebordo metálico existente sobre a engrenagem. Para simular a viga onde é fixado o sistema, utilizou-se uma chapa metálica colocada em cima de uma estrutura, permitindo posicionar a ferramenta à mesma altura em que estaria na zona da turbina. Estes elementos são visíveis na Figura 126.

Figura 126 - Montagem do sistema de visão em laboratório que permite simular o cenário real de trabalho

6.3 Descrição de *Software*

O *software* de processamento de imagem que se pretendeu desenvolver para a medição de desgaste dos dentes da engrenagem de *yaw*, baseia-se no método de triangulação laser "*sheet of light*", no entanto com algumas diferenças.

A principal diferença deste sistema, em relação à aplicação de triangulação laser "*sheet of light*", é o facto de não existir uma reconstrução 3D. A matriz de onze linhas laser é projetada sobre o dente para aferir as coordenadas dos pontos de referência, medindo o desgaste. Deste modo, o laser não permite fazer a reconstrução completa do dente, pois a iluminação apenas incide em alguns dos seus pontos, não cobrindo a totalidade do objeto.

Abaixo, na Figura 127, são apresentadas duas imagens que permitem perceber o efeito obtido ao projetar a matriz laser sobre o dente.

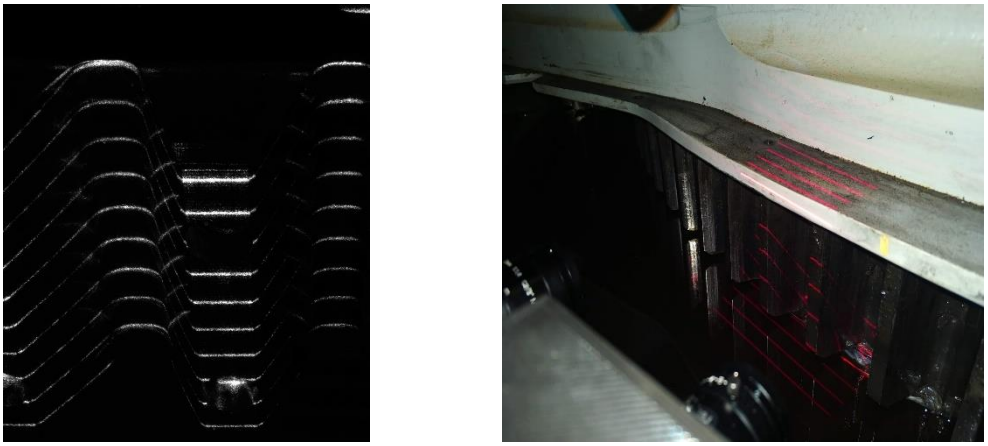


Figura 127 - Projeção da matriz laser num dente da engrenagem

Apesar das diferenças referidas, os procedimentos para configurar e operar um sistema de triangulação laser "*sheet of light*", apresentados anteriormente no subcapítulo 3.5 Triangulação Laser através de Sheet of Light, também se aplicam na montagem utilizada nesta solução. Para além da diferença, mais notável, - o facto da atual solução de medição recorrer a duas câmaras, - o formato do feixe laser também é diferente daquele que é tipicamente utilizado nestas aplicações. Na solução implementada é utilizado um laser que projeta onze linhas ou onze feixes, levando a que os procedimentos de calibração tenham de contemplar cada linha, de modo a permitir que cada uma destas seja utilizada para aferir coordenadas com base em dimensões reais.

Na Figura 128, é apresentado um fluxograma que sintetiza o funcionamento do *software* de processamento de imagem que veio a ser desenvolvido.

Figura 128 - Fluxograma do *software* de processamento de imagem base do sistema de medição

Tal como apresentado no fluxograma, o primeiro procedimento para preparar o sistema de medição em funcionamento é executar a calibração de cada uma das câmaras. Como referido anteriormente, a configuração das câmaras deste sistema não é estéreo, assim a sua calibração é efetuada individualmente. No Anexo 3 – Algoritmo de Medição do Desgaste dos Dentes da Yaw Ring do Projeto com a Vestas é apresentado o algoritmo de medição desenvolvido. O código apresentado resulta da junção de vários exemplos de aplicações do HALCON, mostrando algumas funções que estavam em fase de testes. Assim, a solução aqui apresentada poderá não ser a ideal, no entanto mediante os testes e o desenvolvimento efetuado, ter-se-ão considerado as opções apresentadas.

No processo de desenvolvimento recorreu-se ao assistente de calibração do IDE do HALCON que visa acelerar o processo de calibração e permite verificar ao vivo se as imagens obtidas são adequadas. A base do processo de calibração foi anteriormente descrita no subcapítulo Calibração com a Biblioteca de S*oftware* MVTec HALCON. Com este IDE, é efetuada a calibração de cada câmara e são guardados os respetivos ficheiros, posteriormente utilizados no *software* desenvolvido para obter os parâmetros calibrados, parâmetros internos das câmaras e a sua *pose.*

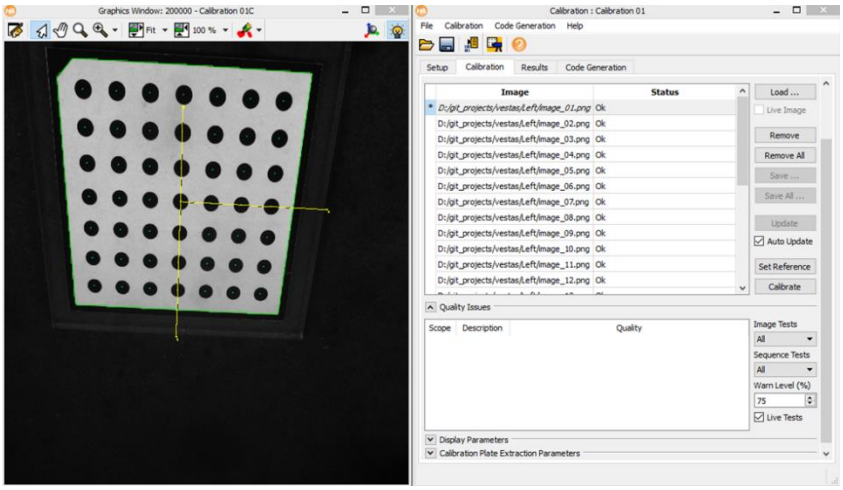


Figura 129 - Layout do assistente de calibração do IDE do HALCON

Uma vez efetuada a calibração e guardados os respetivos ficheiros, são iniciadas as configurações do sistema de câmaras, nomeadamente os procedimentos necessários para calibrar a junção das imagens das duas câmaras. O sistema de câmaras foi configurado de tal forma que a imagem de cada uma sobreponha parte do seu espaço visual. Esta junção da imagem tem por base o exemplo do HALCON *two_camera_calibration.hdev*, apresentado no manual deste *software* "*Solution Guide III C 3D Vision*"[39].

Na Figura 130, é apresentado um esquema que exemplifica este processo de junção das imagens.

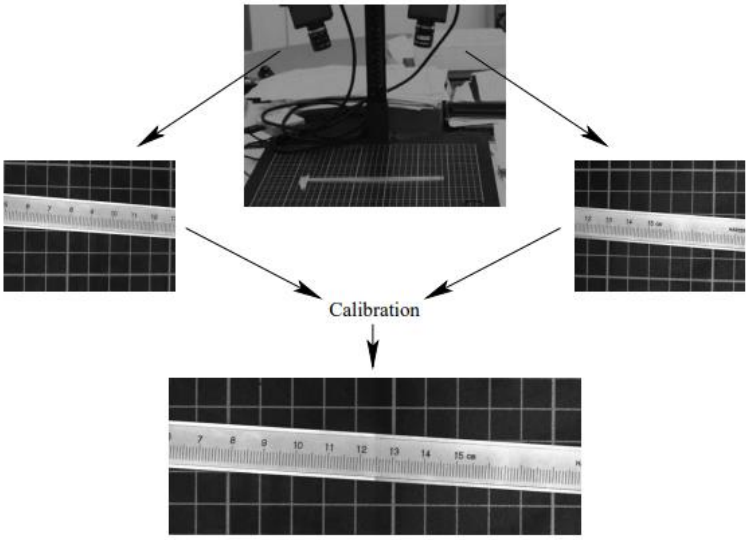


Figura 130 - Esquema exemplificativo da junção de duas imagens [39]

No algoritmo desenvolvido o processo da calibração da junção das imagens é resumido a uma função à qual se deu o nome *sum_two_images_for_laser*. O código desta função é apresentado no Anexo 4. A primeira fase desta função passa por obter os parâmetros de calibração das câmaras da montagem. Neste caso o procedimento é efetuado lendo tal informação a partir dos ficheiros previamente criados. Posteriormente obtém-se duas imagens, uma por câmara, de um objeto de calibração que contém duas placas de calibração com uma distância conhecida entre si, tal como demonstrado no exemplo da Figura 131, por forma a permitir a calibração desta junção

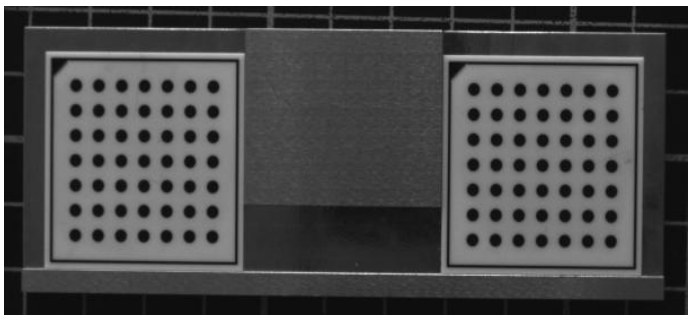


Figura 131 - Objeto de calibração para junção da imagem de duas câmaras [39]

Para executar este procedimento, obtiveram-se previamente imagens deste objeto, guardadas com os ficheiros de calibração e acedidas cada vez que o algoritmo é iniciado de modo a definir a relação de junção das imagens. No algoritmo, estas imagens são lidas e analisadas pelo operador *find_calib_object* que irá aferir a existência do objeto de calibração na imagem de cada câmara. Com o operador *get_calib_data_observ_points* são extraídas as coordenadas dos pontos de cada matriz do objeto de calibração em valores de linha e coluna.

```
*Leitura das imagens do objeto de calibração
read_image (Image1,'double_calplate_left')
read_image (Image2,'double_calplate_right')
*Definição do objeto de calibração
CaltabName := 'caltab_100mm.descr'
create_calib_data ('calibration_object', 2, 1, CalibDataID)
set_calib_data_calib_object (CalibDataID, 0, CaltabName)
*Localização do objeto de calibração nas duas imagens
find_calib_object (Image1, CalibDataID, 0, 0, 0, [], [])
get_calib_data_observ_points (CalibDataID, 0, 0, 0, RowCoord1, ColumnCoord1, Index1, Pose1)
get_calib_data_observ_contours (Caltab1, CalibDataID, 'caltab', 0, 0, 0)
find_calib_object (Image2, CalibDataID, 1, 0, 0, [], [])
get_calib_data_observ_points (CalibDataID, 1, 0, 0, RowCoord2, ColumnCoord2, Index2, Pose2)
get_calib_data_observ_contours (Caltab2, CalibDataID, 'caltab', 1, 0, 0)
```

Antes de efetuar a junção das imagens é necessário efetuar a retificação de cada uma delas, transformação que permitirá encaixar corretamente o par de imagens. Isto é realizado utilizando o operador *gen_image_to_world_plane_map* e, posteriormente, o operador *map_image*, permitindo obter o mapa de retificação das imagens. A junção das imagens propriamente dita é efetuada com o operador *tile_images*, que compõe uma única imagem a partir das duas. Para obter o mapa de retificação de cada imagem, primeiro é necessário definir o canto superior esquerdo (ponto de referência) e o tamanho das imagens retificadas resultantes.

O ponto de referência do canto superior esquerdo da primeira imagem, ou seja, a imagem esquerda é definida automaticamente com base numa percentagem do rebordo, da imagem que é subtraída à imagem original. Através desta percentagem do rebordo são determinadas as coordenadas do ponto, linha e coluna. Posteriormente, estas coordenadas são convertidas para coordenadas associadas ao referencial do "mundo", que fornece as dimensões reais dos objetos presentes na imagem. Uma vez determinadas as coordenadas reais deste ponto, é efetuada uma alteração do ponto de referência do sistema de coordenadas do "mundo", que é definido previamente pela calibração das câmaras. Nesta alteração é introduzida uma correção associada à espessura do objeto de calibração utilizado nesta união das imagens. Esta alteração é efetuada com o operador *set_origin_pose* que assim define a transformação efetuada ao ponto de referência no formato de *pose*.

Abaixo é apresentado um excerto de código que traduz estas operações, e a Figura 132 exemplifica qual o ponto definido.

```
BorderInPercent: =5
get_image_size (Image1, WidthImage1, HeightImage1)
UpperRow: = HeightImage1 * BorderInPercent / 100.0
LeftColumn: = WidthImage1 * BorderInPercent / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, UpperRow, LeftColumn, 'm', LeftX, UpperY)
set_origin_pose (Pose1, LeftX, UpperY, DiffHeight, PoseNewOrigin1)
```

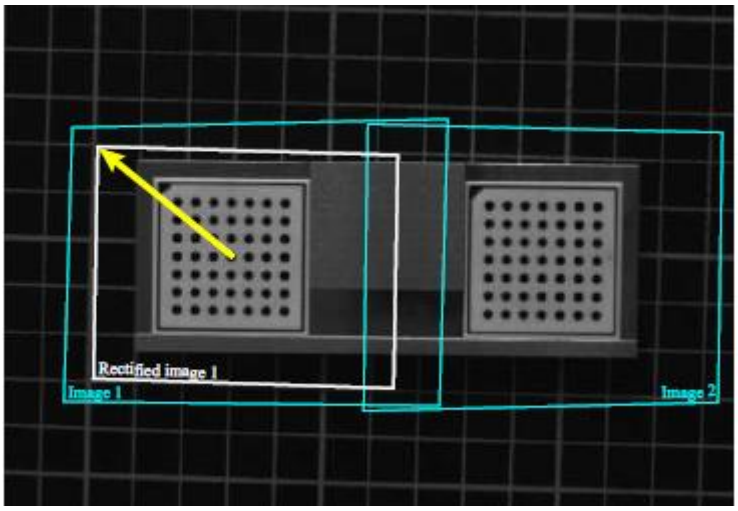


Figura 132 - Definição do ponto de referência no canto superior esquerdo da primeira imagem [39]

Relativamente às dimensões das imagens retificadas, a altura corresponde à distância vertical entre o ponto de origem, no canto superior esquerdo, e um ponto definido na parte inferior da imagem. A largura corresponde à distância horizontal entre o ponto de origem e um ponto definido na área sobreposta das duas imagens, ou seja, um ponto próximo da margem direita da primeira imagem. Estes pontos são definidos automaticamente pelo mesmo princípio utilizado anteriormente, em que é considerada a percentagem do rebordo da imagem que é subtraído à imagem original, tendo em conta a dimensão pretendida dos pixéis para a imagem de retificação. Com a *pose* que traduz a transformação da primeira imagem e a dimensão da imagem retificada, é assim possível obter o mapa de retificação com o operador *gen_image_to_world_plane_map*. Abaixo é apresentado o excerto de código que corresponde a estas operações.

```
PixelSize: = 0.0001
BorderInPercent: =5
get_image_size (Image1, WidthImage1, HeightImage1)
UpperRow: = HeightImage1 * BorderInPercent / 100.0
LeftColumn: = WidthImage1 * BorderInPercent / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, UpperRow, LeftColumn, 'm', LeftX, UpperY)
set_origin_pose (Pose1, LeftX, UpperY, DiffHeight, PoseNewOrigin1)
LowerRow: = HeightImage1 * (100 - BorderInPercent) / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, LowerRow, LeftColumn, 'm', X1, LowerY)
HeightRect: = int ((LowerY - UpperY) / PixelSize)
OverlapInPercent: = 25
RightColumn: = WidthImage1 * (100 - OverlapInPercent / 2.0) / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, UpperRow, RightColumn, 'm', RightX, Y1)
WidthRect: = int ((RightX - LeftX) / PixelSize)
gen_image_to_world_plane_map (MapSingle1, CameraParamLeft, PoseNewOrigin1, Width, Height, WidthRect, HeightRect, PixelSize, 'bilinear')
```

O “ponto de encaixe” da segunda imagem, - da direita -, corresponde ao ponto do canto superior direito da primeira imagem. Este ponto é definido por uma translação do ponto no canto superior esquerdo da primeira imagem, que anteriormente já tinha sido

transformado na origem da placa de calibração. Esta translação, em conjunto com a correção associada à espessura da placa do objeto de calibração, é a transformação representada pela matriz ${}^{cp1}H_{ur1}$, tal como representado na Figura 133.

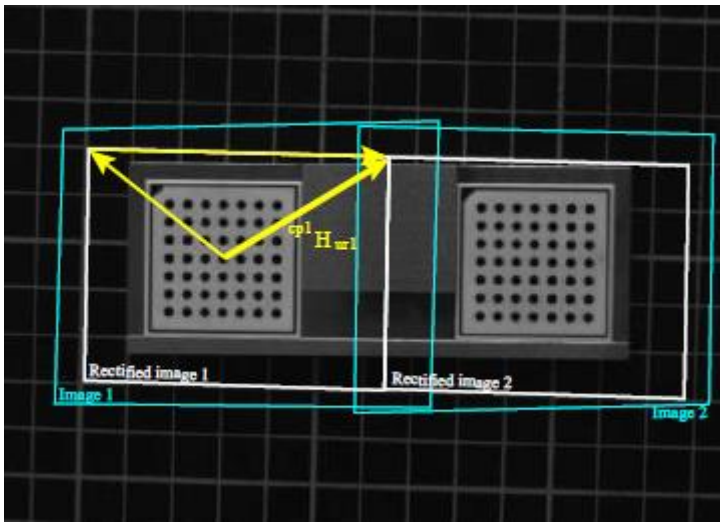


Figura 133 - Definição do canto superior direito da primeira imagem retificada [39]

Seguidamente é necessário definir a transformação entre a placa de calibração da primeira imagem e a da segunda imagem. Esta transformação é efetuada com base na distância fixa entre as duas placas previamente medida, e representada pela matriz de transformação ${}^{cp1}H_{cp2}$. Esta relação de transformação é demonstrada na Figura 134.

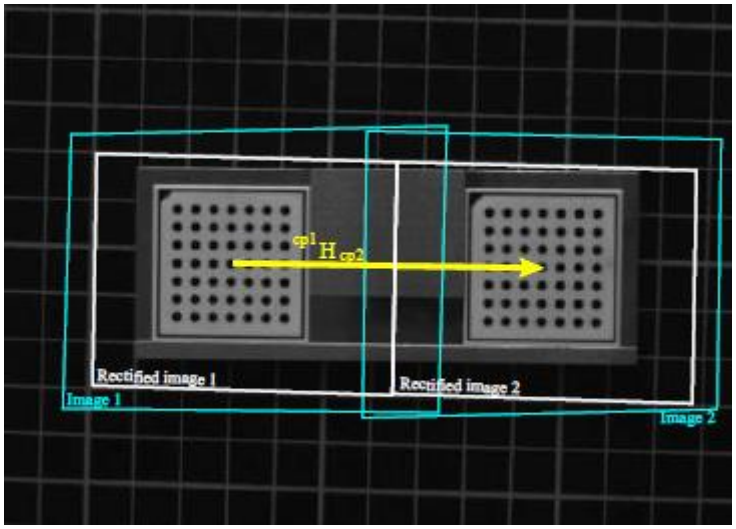


Figura 134 - Relação de transformação do sistema de coordenadas da placa de calibração da esquerda para o sistema de coordenadas da direita [39]

Recorrendo à transformação traduzida pela matriz ${}^{cp1}H_{ur1}$ e à transformação traduzida pela matriz ${}^{cp1}H_{cp2}$, é possível obter a relação, ${}^{cp2}H_{ul2}$, que traduz a transformação do sistema de coordenadas da segunda imagem para o sistema de coordenadas do "mundo", com origem no canto superior esquerdo da segunda imagem retificada. A Figura 135 ilustra esta transformação.

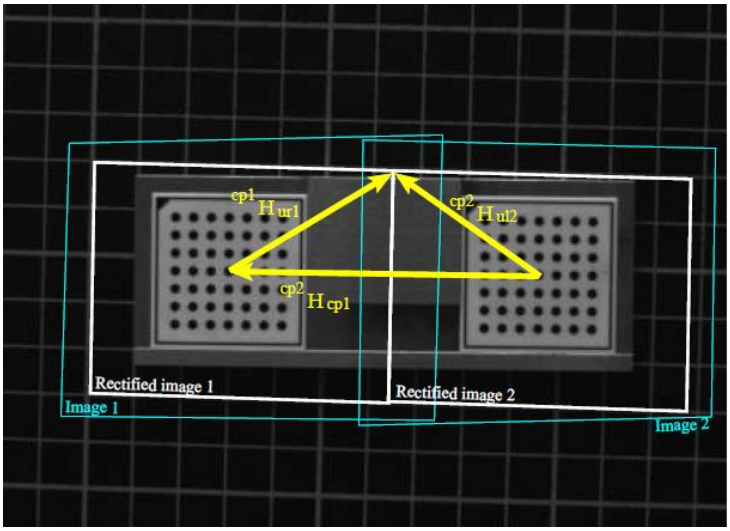


Figura 135 - Definição do ponto de origem do referencial da segunda imagem no seu canto superior esquerdo [39]

No caso desta segunda imagem, é efetuada uma transformação da *pose* da placa de calibração de modo que a origem do seu referencial seja o ponto do canto superior esquerdo da imagem. Efetuadas todas estas operações, é possível obter o mapa de retificação desta segunda imagem. De seguida, é apresentado o excerto de código relativo ao conjunto de transformações associadas à segunda imagem.

```
hom_mat3d_identity (HomMat3DIdentity)
hom_mat3d_translate_local (HomMat3DIdentity, LeftX + PixelSize * WidthRect, UpperY, DiffHeight, cp1Hur1)
hom_mat3d_translate_local (HomMat3DIdentity, DistancePlates, 0, 0, cp1Hcp2)
hom_mat3d_invert (cp1Hcp2, cp2Hcp1)
hom_mat3d_compose (cp2Hcp1, cp1Hur1, cp2Hul2)
pose_to_hom_mat3d (Pose2, cam2Hcp2)
hom_mat3d_compose (cam2Hcp2, cp2Hul2, cam2Hul2)
hom_mat3d_to_pose (cam2Hul2, PoseNewOrigin2)
```

gen_image_to_world_plane_map (MapSingle2, CameraParamRight, PoseNewOrigin2, Width, Height, WidthRect, HeightRect, PixelSize, 'bilinear')

Antes de efetuar a junção das imagens das duas câmaras, é necessário proceder à sua retificação. Só assim será possível uma junção precisa das imagens. Para isso são utilizados os mapas de retificação previamente obtidos e as imagens são retificadas recorrendo ao operador *map_image*. Após a retificação das duas imagens, estas são concatenadas num só objeto, utilizando o operador *concat_obj*. Finalmente, o operador *tile_images* utiliza este objeto para obter a imagem que corresponde à junção das duas. Abaixo é apresentada uma sequência de junção de um par de imagens. Esta calibração de junção das imagens será utilizada novamente mais adiante no algoritmo principal de medição de desgaste para unir um par de imagens.

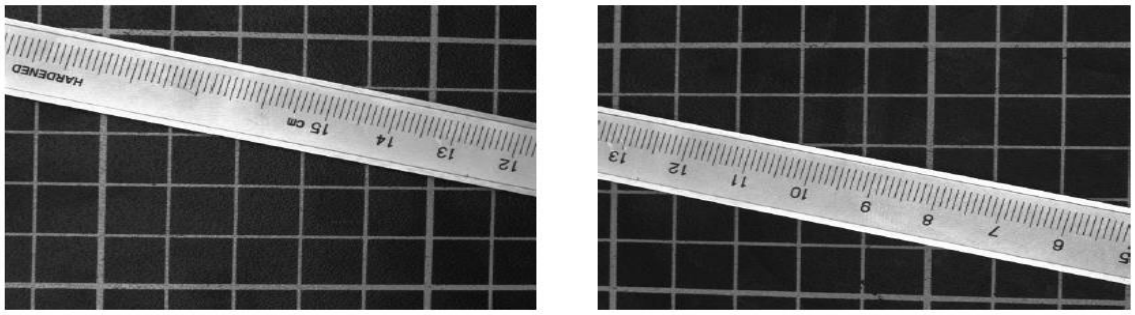


Figura 136 - Exemplo de um par de imagens originais obtidas por um par de câmaras [39]

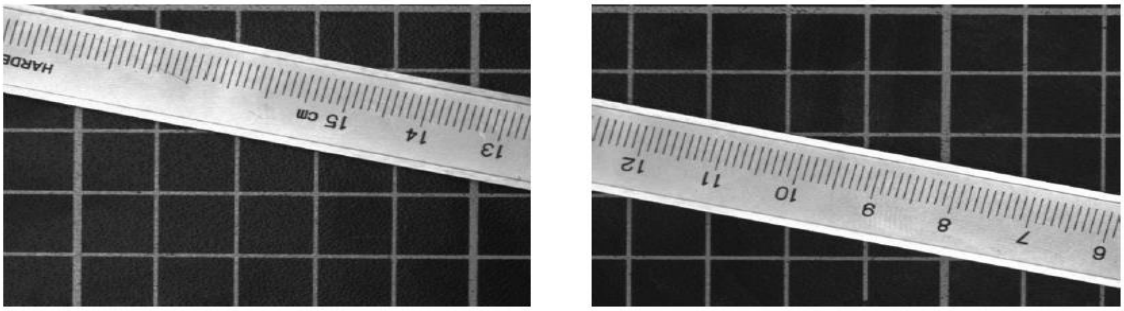


Figura 137 - Par de imagens retificadas [39]

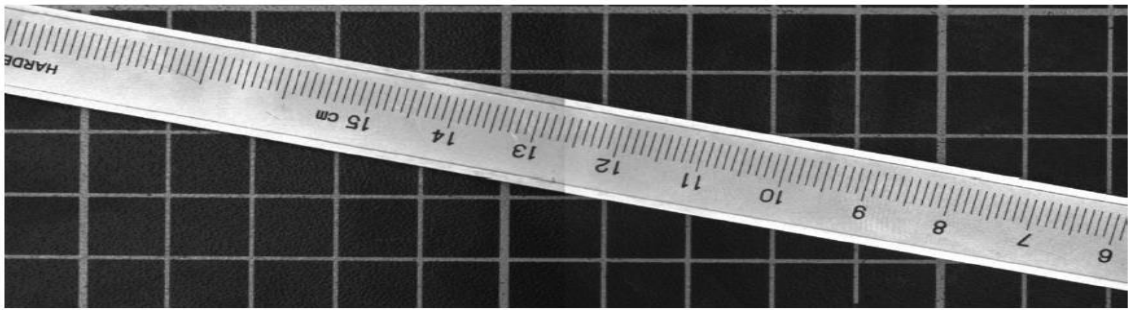


Figura 138 - Resultado da união das imagens [39]

O passo seguinte, no algoritmo de medição, é a calibração das linhas laser. Tal como descrito anteriormente, a grande diferença da calibração utilizada nesta aplicação, em relação à calibração do laser num sistema de triangulação *sheet of light*, é o facto de se utilizarem onze linhas laser. É utilizado o mesmo procedimento de calibração para cada linha, tal como se empregasse apenas uma. Assim, primeiro é efetuada uma calibração das linhas laser em relação ao eixo *Z*, seguindo os procedimentos apresentados no subcapítulo Calibrar um Sistema *Sheet of Light*. Nesta fase é executado exatamente o mesmo procedimento que é descrito nesse subcapítulo, em que são analisadas duas imagens da placa de calibração, ou seja, duas por cada câmara, em que o posicionamento da placa (*pose*) numa das imagens é utilizado para definir o WCS e a *pose* definida pela placa na outra imagem é assumida como um sistema de coordenadas temporário (TCS – *Tempororary Coordinate System*).

No seguimento deste procedimento, que visa determinar o plano de luz (*sheet of light*) de cada uma das linhas laser, são analisadas igualmente duas imagens por cada câmara, em que numa estão as linhas laser projetadas no *Z*=0 do plano WCS (ver Figura 139) e noutra imagem, com as linhas projetadas num plano superior, que corresponde ao *Z*=0 do TCS (ver Figura 140) definido anteriormente. Estas imagens irão permitir definir as coordenadas do perfil laser em cada um dos sistemas de coordenadas, para posteriormente efetuar a referida determinação do plano de luz de cada linha.

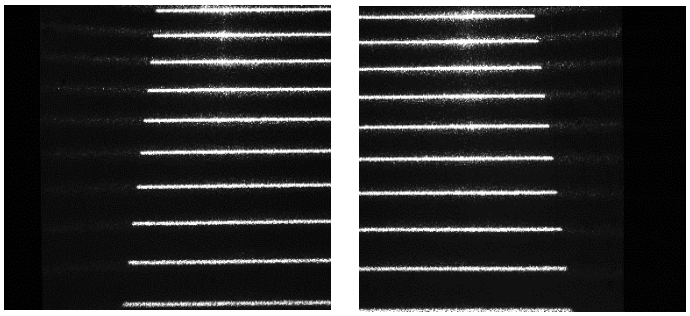


Figura 139 – Par de imagens com a projeção das linhas laser no plano Z=0 do WCS

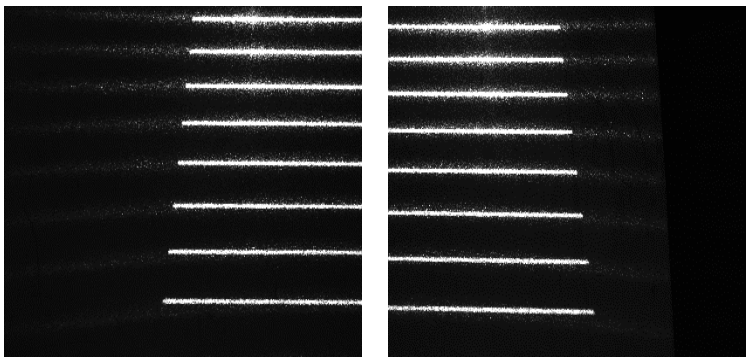


Figura 140 - Par de imagens com a projeção das linhas laser no plano Z=0 do TCS

Os procedimentos para determinação do plano associado a cada linha da matriz laser foram resumidos na função *laser_procedure.* O código desta função é apresentado no Anexo 5 – Algoritmo do Projeto Vestas para Determinação do Plano de Luz de Cada Linha do Laser.

Esta função determina o plano de cada linha do laser, representada em cada uma das imagens das duas câmaras. Assim, a função é utilizada onze vezes, uma por cada linha laser, para as imagens obtidas pela câmara da esquerda e o mesmo número de vezes para as imagens obtidas pela câmara da direita.

A função *laser_procedure* recebe como parâmetros de entrada:

- a variável com os parâmetros de calibração da câmara;
- a *pose* do plano *Z*=0 do WCS;
- a *pose Z*=0 do TCS;
- a imagem com a projeção das linhas no plano *Z*=0 do WCS;
- a imagem com a projeção das linhas no plano *Z*=0 do TCS;

- o valor de cinzento mínimo (*MinThreshold*) para o qual é possível segmentar as linhas do laser.

Para além destes parâmetros, são introduzidas duas variáveis que permitem definir qual das linhas na imagem irá ser analisada:

- uma variável com a linha segmentada na imagem da projeção das linhas no plano *Z*=0 do WCS;
- a variável com a mesma linha segmentada na imagem da projeção das linhas no plano *Z*=0 do TCS.

As linhas laser são segmentadas na imagem, utilizando o operador *lines_gauss* que deteta as linhas na imagem e coloca os resultados numa variável de saída que contém o conjunto de linhas laser no formato de contornos (*XLD-contours*).

De modo a definir os parâmetros *Sigma* e *Low*, utilizados por este operador, para obter o contorno das linhas, é previamente utilizado o operador *calculate_lines_gauss_parameters*. Este operador permite obter os valores a inserir nos parâmetros *Sigma* e *Low*, mediante a indicação dos valores de cinzento mínimo e máximo correspondentes à representação das linhas laser na imagem.

O operador *line_gauss* obtém todas as linhas existentes na imagem. Assim, de modo a filtrar algum artefacto luminoso que possa aparecer na imagem, e de forma a garantir que apenas são selecionadas as linhas do laser, é utilizado o operador *select_shape_xld*. Este operador utiliza-se para selecionar o contorno das linhas através do seu diâmetro máximo.

Determinados os contornos da linha do laser nos dois referenciais WCS e TCS, é selecionado um par de contornos que correspondem à projeção da mesma linha nos dois referenciais. Este par de contornos agora selecionado é utilizado na função descrita anteriormente, *laser_procedure,* que os recebe como parâmetro de entrada, de forma a obter o plano de "folha de luz" ou "*sheet of light*" associado a cada linha do laser.

Abaixo é demonstrado um excerto de código, onde é efetuada a obtenção dos contornos das linhas laser, de forma a poder determinar o plano "*sheet of light*" associado a cada linha.

A fase final do código, em que é selecionado o par de contornos da mesma linha laser e é determinado o plano, é repetida para cada uma das onze linhas e para as imagens das duas câmaras.

```
*Linhas laser projetadas no plano Z=0 do WCS (bottom lines)
read_image (bottom_left, 'images_calib/bottom_left.png')
calculate_lines_gauss_parameters (45, 90, Sigma, Low, High)
lines_gauss (bottom_left, Lines, Sigma, Low, High, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines, SelectedXLD, 'max_diameter', 'and', 800, 2000)
* Linhas laser projetadas no plano Z=0 do TCS (top lines)
read_image (top_left, 'images_calib/top_left.png')
calculate_lines_gauss_parameters (45,90, Sigma1, Low1, High1)
lines_gauss (top_left, Lines2, Sigma1, Low1, High1, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines2, SelectedXLD1, 'max_diameter', 'and', 800, 2000)
*Selecionar 1ª linha laser projetada no Z=0 WCS
select_obj(SelectedXLD, ObjectSelected, 1)
*Selecionar 1ª linha laser projetada no Z=0 TCS
select_obj(SelectedXLD1, ObjectSelected2, 1)
*Definição do plano ou perfil da 1ª linha laser
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold,
CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left,
Left_LightP1)
```

Uma vez efetuada esta primeira calibração, que permite determinar o plano associado a cada linha do laser, é efetuada a segunda tarefa de calibração, relacionada com o sistema de iluminação laser. Este é o processo de calibração associado ao movimento do objeto, onde é executada a leitura de duas fotos com a placa de calibração, que descrevem duas etapas distintas do movimento do objeto.

No caso do algoritmo desenvolvido, o movimento do objeto, a engrenagem de *yaw*, é efetuado em relação ao eixo *X*, definido no referencial associado à calibração das duas câmaras, ou seja, o eixo *X* é o que se alinha com a horizontal das imagens.

No algoritmo, a calibração é efetuada tal como descrito no subcapítulo Calibrar um Sistema *Sheet of Light*. No entanto, devido a dificuldades em quantificar com precisão o

deslocamento associado ao movimento do objeto, as imagens das placas de calibração não representam com exatidão o movimento do objeto.

Existiram efetivamente algumas dificuldades em definir o compromisso de ângulos entre as câmaras e o laser de modo que, quando o sistema é instalado no local para medição, as onze linhas estejam nitidamente projetadas nos dentes das engrenagens, ainda que as câmaras consigam ver claramente o corpo do dente que deverá estar corretamente enquadrado no seu campo de visão. Esta dificuldade deteve o processo de desenvolvimento numa demorada fase de aprimoramento da instalação e posicionamento de câmaras, laser e as suas calibrações. Assim, os procedimentos no algoritmo daqui em diante são afetados negativamente por este "entrave" no desenvolvimento.

Após efetuadas todas as calibrações, relativas às câmaras e ao laser, é efetuada a configuração inicial das câmaras para que estas possam iniciar a aquisição de imagens para proceder às medições necessárias.

A configuração das câmaras efetuada neste algoritmo é semelhante à configuração demonstrada anteriormente no subcapítulo *Hardware* da UCB. Neste caso, o par de câmaras é igualmente configurado de maneira a que a aquisição de imagens seja efetuada sincronizadamente. Como o objeto em análise está em movimento, esta configuração torna-se imprescindível, caso contrário seria impossível conseguir bons resultados. Efetuada a configuração, é iniciada a aquisição de imagens para medição. A aquisição de imagens recorre ao operador *grab_image_async*. Na Figura 141, é apresentado um par de imagens tipicamente obtidas, onde a matriz laser incide sobre os dentes da engrenagem.

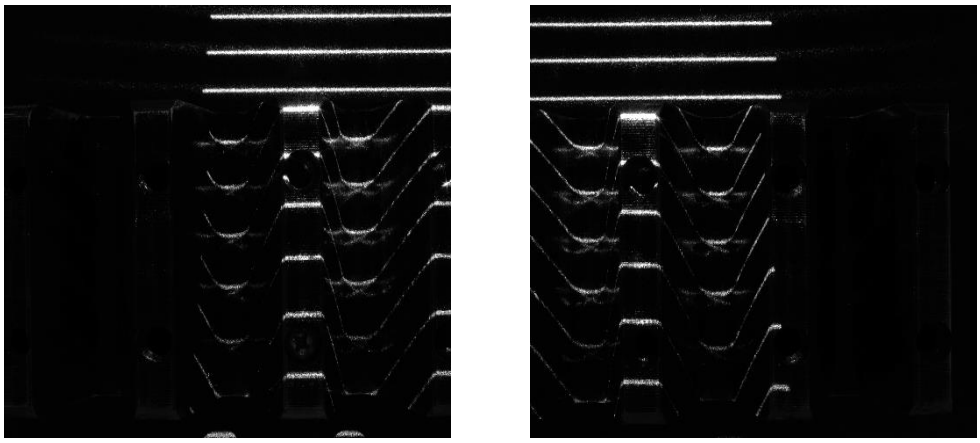


Figura 141 - Imagens obtidas pelo par de câmaras onde a matriz laser é projetada sobre os dentes

Após adquiridas as imagens, é efetuada a retificação das mesmas tendo por base o mapa de retificação definido no início do algoritmo, que irá permitir o seu "encaixe" correto. Tal como descrito anteriormente, a retificação de cada imagem é efetuada recorrendo ao operador *map_image*. O processo a seguir nesta situação será o mesmo que foi descrito anteriormente a propósito da calibração respeitante à junção das imagens. Assim, após a retificação das duas imagens, estas são concatenadas num só objeto utilizando o operador *concat_obj*. Finalmente, o operador *tile_images* utiliza este objeto para obter a imagem que corresponde à junção.

Após a junção das imagens, é efetuada a extração das coordenadas das várias linhas laser projetadas nas faces do dente em análise. Para esta determinação é necessário que a imagem obtida apresente a face frontal do dente paralela ao eixo *X* do referencial das câmaras, ou seja, paralela à horizontal das duas câmaras. Para extrair as coordenadas de uma das linhas laser, é definida uma "região de interesse" que corresponde à zona onde se encontra o perfil da linha laser na imagem. Ao definir esta "região de interesse" parte-se do pressuposto que a configuração do sistema de câmaras e laser bem como a sua posição relativamente à engrenagem se mantiveram inalterados desde a calibração. Caso se alterasse a configuração, ou a posição do sistema, esta região não iria delimitar corretamente a projeção da linha laser pretendida. Para além disso, invalidaria todas as calibrações previamente efetuadas.

Definida a região da projeção da linha laser, as coordenadas são extraídas através da função *get_xyz_from_laser_profile,* uma função que sintetiza o processo de extração das coordenadas apresentado anteriormente no subcapítulo de introdução teórica sobre a triangulação laser, Executar Medições. Abaixo apresenta-se o código que define esta função. Esta permite obter as coordenadas da linha laser dentro desta região, neste caso, em milímetros, sendo apresentadas em três imagens distintas, uma para cada eixo coordenado, onde os valores de coordenadas são anunciados em valores de cinzento.

```
*Função get_xyz_from_laser_profile
reduce_domain (gear, ProfileRegion, ImageReduced)
create_sheet_of_light_model (ProfileRegion, ['min_gray','num_profiles'], [30,1], SheetOfLightModelID)
set_sheet_of_light_param (SheetOfLightModelID, 'calibration', 'xyz')
```

```
set_sheet_of_light_param (SheetOfLightModelID, 'camera_parameter', CameraParameters)
set_sheet_of_light_param (SheetOfLightModelID, 'camera_pose', CameraPose)
set_sheet_of_light_param (SheetOfLightModelID, 'lightplane_pose', LightPlanePose)
set_sheet_of_light_param (SheetOfLightModelID, 'movement_pose', MovementPose)
set_sheet_of_light_param (SheetOfLightModelID, 'scale', 'mm')
* Aplicação das transformações da calibração e extração dos valores das coordenadas
measure_profile_sheet_of_light (ImageReduced, SheetOfLightModelID, MovementPose)
get_sheet_of_light_result (Disparity, SheetOfLightModelID, 'disparity')
get_sheet_of_light_result (X, SheetOfLightModelID, 'x')
get_sheet_of_light_result (Y, SheetOfLightModelID, 'y')
get_sheet_of_light_result (Z, SheetOfLightModelID, 'z')
return ()
```

Apesar de existir um conjunto de pontos pré-definidos (pela Vestas) para executar a medição, desde início se optou por simplificar a localização e quantidade dos pontos de medição de forma a conseguir construir o conceito do algoritmo de medição. Para executar a medição são definidos dois pontos, um de cada lado da face do dente, a uma determinada distância da base do dente, e que coincidam com a zona de projeção de uma das linhas laser, tal como demonstrado na Figura 142.

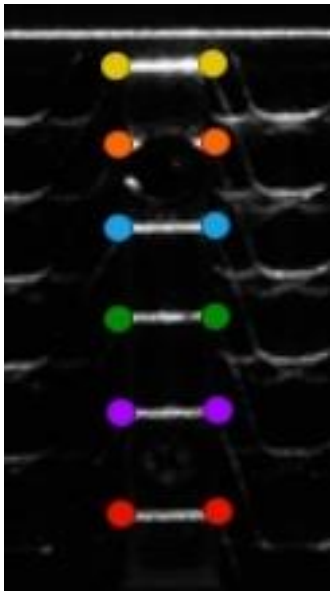


Figura 142 - Esquema ilustrativo da zona onde são definidos os pontos de medição

A distância entre estes dois pontos é o valor utilizado para comparação com as dimensões originais, permitindo assim aferir qual o valor de desgaste do dente. O valor do desgaste é igual à subtração do valor medido ao valor da dimensão original na zona onde foi obtida a medida.

De forma a automatizar a definição do ponto, assumiu-se um pequeno intervalo de valor da coordenada *Z* deste ponto, com o valor em milímetros de 54 mm a 54,99 mm. Este intervalo corresponde a uma distância de 54 mm até 54,99 mm desde a base do dente até este ponto de medição. Para selecionar este ponto, utiliza-se o operador *threshold* que permite selecionar uma região, numa imagem, pelo valor de cinzento. Neste caso é selecionado na imagem, com os valores da coordenada Z, um ponto ou um pequeno intervalo de pontos, cujo valor de cinzento seja 54 a 54,99. Após definidas a linha e coluna que correspondem a este ponto na imagem, é possível aferir o valor das coordenadas *X* e *Y* deste ponto. Isto é efetuado através do operador get_*grayval* que, mediante a introdução de uma imagem, da linha e coluna do ponto ou pontos selecionados, devolve o valor de cinzento correspondente a esta seleção. Deste modo utiliza-se o operador para cada uma das imagens de coordenadas e obtêm-se as coordenadas em valores de cinzento, que neste caso correspondem às coordenadas reais em milímetros.

No excerto de código abaixo é demonstrado o conjunto de procedimentos até aqui descritos para extrair as coordenadas dos pontos de medição.

```
*Laser1
*Imagem esquerda
gen_rectangle1 (ProfileRegion1, 699.275, 2155.73, 777.105, 2471.98)
get_xyz_from_laser_profile (ImageL, ProfileRegion1, Xl, Y1, Z1, ImageReduced1,
CameraParameters_left, CameraPose_left, Left_LightP1, MovementPoseLeft)
threshold (Z1, Regions, 54, 54.99)
connection (Regions, ConnectedRegions)
area_center (ConnectedRegions, Area, Row, Column)
get_grayval (X1,0, Column [0], Grayval_X1)
get_grayval(Y1,0,Column[0],Grayval_Y1)
Column: =0
dev_clear_obj (ConnectedRegions)
*Imagem direita
```

gen_rectangle1 (ProfileRegion21, 733.601, 260.88, 805.925, 583.164)
get_xyz_from_laser_profile (Image2, ProfileRegion21, X21, Y21, Z21, ImageReduced21, CameraParameters_right, CameraPose_right, Left_LightP1, MovementPoseLeft)
threshold (Z21, Regions, 54, 54.99)
connection (Regions, ConnectedRegions)
area_center (ConnectedRegions, Area, Row, Column)
get_grayval (X21,0, Column [0], Grayval_X21)
get_grayval(Y21,0,Column[0],Grayval_Y21)
Column: =0
dev_clear_obj (ConnectedRegions)

A última fase deste algoritmo é a determinação do desgaste existente no dente analisado.

Assumindo que o dente está corretamente alinhado com o referencial, as coordenadas extraídas anteriormente dos dois pontos de cada lado do dente terão o mesmo valor para *Y*, ficando estes pontos no mesmo alinhamento da linha laser. A diferença entre os valores de *X* corresponderá à espessura do perfil frontal do dente. O desgaste corresponderá à subtração dessa espessura ao valor da espessura original da face do dente neste preciso local.

Esta fase não foi alvo de grandes desenvolvimentos, uma vez que existiram outras dificuldades que condicionaram configurações anteriores, nas quais se investiu mais tempo de desenvolvimento. No subcapítulo seguinte, serão detalhados estes problemas bem como a situação atual do desenvolvimento do projeto.

Depois de aferir o desgaste nos vários pontos de medição no dente, o algoritmo deveria enviar estes valores para a plataforma de interação com o utilizador, onde seriam guardados na base de dados. No entanto, na fase de desenvolvimento em que este algoritmo se encontra, este procedimento ainda não é concretizado.

O envio dos valores do desgaste de cada dente, obtidos pelo algoritmo de processamento de imagem descrito, seriam enviados para a citada plataforma via SocketTCP recorrendo ao operador *send_data*. Para estabelecer esta ligação, seria necessário configurar a comunicação dos dois lados: no *software* de processamento de imagem e no servidor Node.js da plataforma. Na biblioteca HALCON a comunicação via SocketTCP é estabelecida recorrendo ao operador *open_socket_connect.*

6.4 Resultados e Conclusões

Desde o início do desenvolvimento desta ferramenta para os trabalhos de manutenção da Vestas Portugal, em meados de 2018, foram efetuadas várias visitas técnicas às turbinas eólicas desta empresa. Estas visitas permitiram tomar conhecimento do processo de medição de desgaste atualmente utilizado pela empresa bem como constatar a exiguidade de espaço na zona onde se pretende aceder à engrenagem. Assim, procurou-se tirar partido destas visitas para efetuar testes do sistema de câmaras com iluminação laser, no sentido de definir, nestas condições, alguns detalhes da solução. Nomeadamente, definir quais as limitações do campo de visão e do campo de projeção da matriz laser. Ao colocar o sistema de câmaras nestas condições efetuaram-se testes de captura de imagem com vista a determinar quais os parâmetros da câmara adequados à fraca luminosidade no local, de modo a visualizar o laser nitidamente.

No que diz respeito à configuração das câmaras, optou-se por manter um valor de tempo de exposição que permitisse visualizar nitidamente as linhas laser, atingido por "tentativas", tendo-se fixado este tempo em 80000, que corresponde a 80 ms. Considera-se, no entanto importante referir que este tempo de exposição poderá não ser adequado ao movimento do objeto em análise. Para tal, seria necessário ter em conta a velocidade de rotação da engrenagem, e assim definir um valor de exposição adequado em que o objeto não fique desfocado e as linhas do laser se mantenham nítidas.

Como referido anteriormente, no desenvolvimento deste projeto surgiram também dificuldades na disposição das câmaras e do laser, devido à limitação de espaço existente na turbina. A estrutura criada em impressão 3D para suportar as câmaras e o laser foi fabricada de forma a permitir o ajuste de ângulos dos equipamentos, ajuste da distância ao objeto e o ajuste da altura da suspensão na zona de acesso à engrenagem na turbina.

Mesmo com esta mobilidade, verificaram-se dificuldades importantes para determinar o compromisso entre ângulos das câmaras e laser, de modo a que se conseguisse projetar o laser na superfície do dente e que este fosse completamente visível na imagem. Pode verificar-se, por exemplo, na Figura 143, que o padrão de linhas não está corretamente posicionado em relação aos dentes da engrenagem, projetando apenas 3 a 4 linhas nas suas faces. Este é um caso em que o laser está extremamente mal posicionado. Contudo, o melhor cenário de enquadramento terá sido alcançado quando apenas ficavam duas a

três linhas fora do objeto. Mesmo nestas situações, nem sempre foi fácil que as câmaras conseguissem enquadrar corretamente o laser na imagem.

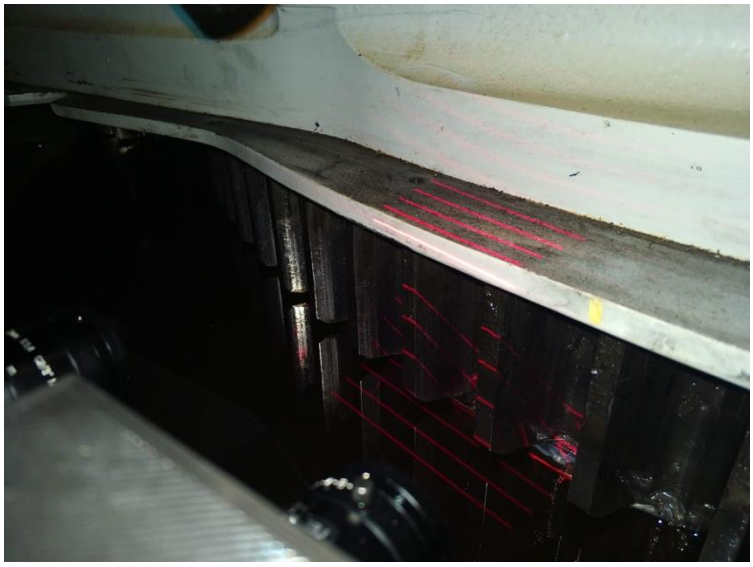


Figura 143 - Laser projetado nos dentes da engrenagem, em que as linhas do padrão não foram bem enquadradas

Após vários testes concluiu-se que o posicionamento ideal para os vários elementos do sistema seria a colocação do laser entre as duas câmaras: Neste caso o laser terá de ficar desfasado do alinhamento das câmaras e o ângulo em relação aos dentes também deverá diferir do das câmaras. Abaixo é apresentada uma imagem exemplificativa do posicionamento ideal do laser. O protótipo atualmente em testes é também apresentado na Figura 144

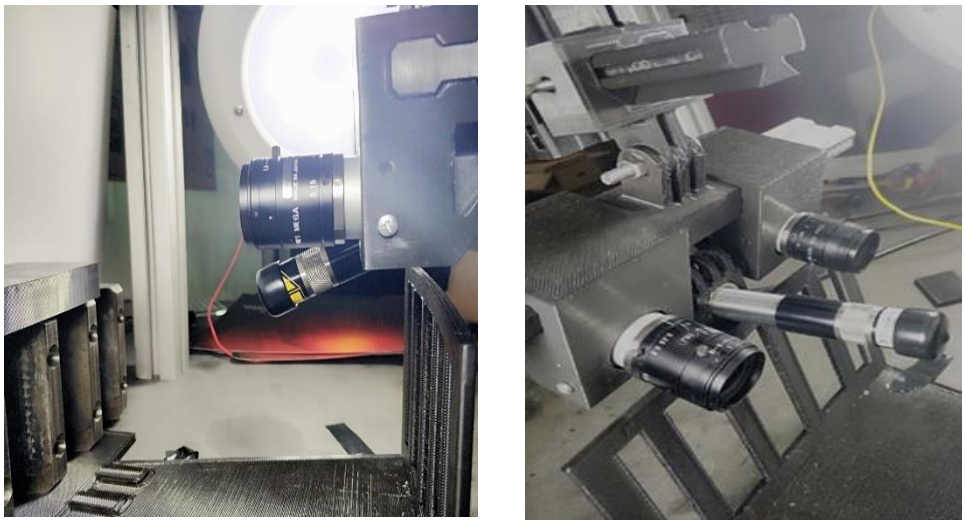


Figura 144 - Montagem ideal das câmaras e do laser

Estas dificuldades levaram a que se criasse uma maquete do espaço disponível para colocar o sistema, tal como descrito anteriormente nas especificações de hardware deste projeto. Os últimos desenvolvimentos, antes do fim do período deste estágio, foram precisamente o fabrico e montagem desta maquete.

A Vestas pretende que esta ferramenta de medição possa ser utilizada em várias turbinas, sendo, portanto, um sistema que é levado para cada local quando necessário, para a determinação do desgaste da *yaw ring* de uma turbina eólica.

Assim, será necessário garantir que, sempre que o sistema é instalado, as câmaras se encontrem corretamente posicionadas e a uma distância do objeto a medir definida pela calibração inicial.

Como se pretende que os ajustes de "distâncias e ângulos entre câmaras" e "ângulos entre conjunto de câmaras e laser" sejam fixos, caso não se alterem tais ajustes, a calibração efetuada previamente, em bancada, manter-se-á válida. A única configuração que poderá ter alguma variação quando o sistema é movido de uma turbina para outra será a distância ao objeto a medir.

De forma a garantir que o utilizador coloca o sistema à distância correta da engrenagem, antes de se iniciar o processo de medição, deverá proceder-se a uma confirmação deste posicionamento, processo que seria efetuado através da plataforma desenvolvida para esta ferramenta.

Durante o desenvolvimento desta ferramenta, testou-se uma abordagem para tentar solucionar este problema, em que o utilizador procedia a uma "pequena calibração" para ajustar a distância das câmaras aos dentes. A Figura 145 ilustra a distância que deverá ser ajustada nesta "pequena calibração".

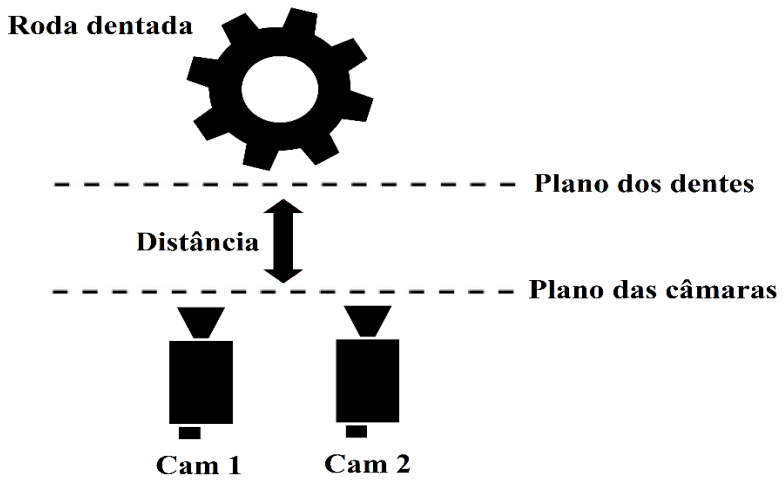


Figura 145 - Esquema de utilização do objeto de calibração

Na plataforma desenvolvida existe um menu específico para efetuar a calibração, em que se pretendia que o utilizador interagisse de forma a receber indicações de COMO proceder e QUAL o ajuste a efetuar ao conjunto de câmaras.

A solução passaria por utilizar um objeto sólido, fabricado em impressão 3D, com as dimensões correspondentes ao espaço entre dentes. Na face frontal deste objeto seria colocada uma placa de calibração de 30 mm. Esta placa deve auxiliar na definição da distância do sistema de câmaras aos dentes. Recorrendo aos operadores utilizados nos processos calibração das câmaras, é possível determinar as coordenadas dos pontos da matriz de calibração, e assim determinar a distância a que o objeto de calibração se encontra.

O objeto foi dimensionado de tal forma que, da base do dente até à face frontal deste objeto distem 55 mm, dimensão intencionalmente definida. Estes 55 mm correspondem à cota *Z* a que deverá ser adquirida a medição do desgaste do dente. Os 55 mm correspondem igualmente à distância da base da aresta que define a face lateral, tal como demonstrado no conjunto de imagens da Figura 146.

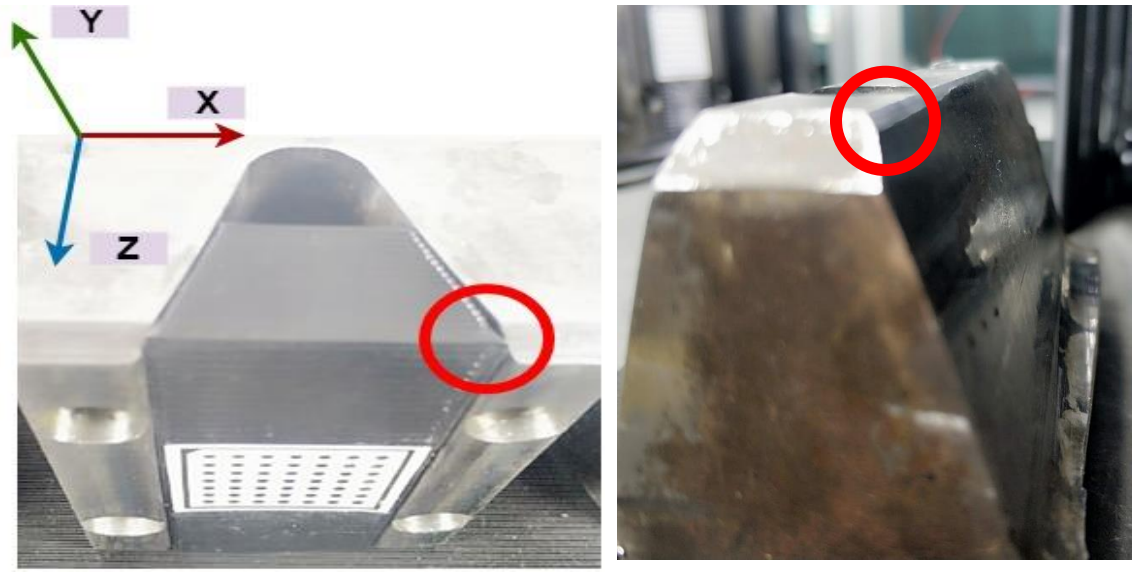


Figura 146 - Pormenor do posicionamento do objeto de calibração

Assim, este objeto de calibração terá dupla utilidade, permitindo o ajuste da distância das câmaras e a definição da cota a utilizar nos pontos para medição.

O procedimento para esta calibração seria baseado no citado menu de calibração da plataforma de interação com o utilizador. Neste menu, o utilizador daria o comando de início de calibração. De seguida, ser-lhe-iam mostradas duas janelas com as imagens que estavam a ser obtidas pelas câmaras. Caso o objeto de calibração estivesse corretamente posicionado entre um conjunto de dentes, apareceria no menu de calibração, ou sobreposto nas imagens, uma mensagem a notificar o utilizador que a matriz de calibração foi reconhecida. Ao reconhecer a matriz, o sistema informaria o utilizador sobre quais as alterações à posição do sistema, em relação à engrenagem, que deveriam ser feitas, ou seja, se as câmaras deveriam ser afastadas ou aproximadas do objeto. Quando o sistema estivesse à distância correta, o utilizador seria notificado e a calibração terminaria e poderia iniciar-se o processo de medição.

Embora a solução para efetuar este ajuste não esteja completamente definida, foram efetuados alguns testes, no entanto esta é uma outra componente da ferramenta de medição que ainda terá de ser desenvolvida.

Tal como tem vindo a ser explicado, a ferramenta para medição do desgaste dos dentes de engrenagens de *yaw* é um projeto ainda em desenvolvimento, tendo principalmente várias componentes do *software* incompletas. Os relatos apresentados a longo deste

capítulo descrevem as várias opções tomadas no decurso do projeto. As conclusões e resultados apresentados são relativos ao estado do projeto aquando do términus do estágio. Esta solução continua a ser desenvolvida por parte da equipa do LINE.IPT, não tendo sido terminada no período em que o aluno efetuou este estágio.

7 Conclusão

No âmbito do estágio no LINE.IPT, o aluno estagiário integrou os trabalhos de desenvolvimento do projeto Aquatropolis - *Intelligent Management System for Sustainable Aquacultures*, bem como do projeto com a Vestas Portugal. Realizou ainda alguns trabalhos no âmbito de outros projetos, sobretudo programação de microcontroladores e fabrico de placas de circuito impresso. No entanto, por se tratar de tarefas pontuais, considerou-se mais relevante dar destaque apenas aos dois projetos principais.

O trabalho desenvolvido neste relatório esteve maioritariamente relacionado com o desenvolvimento de soluções de processamento de imagem e visão por computador. Os algoritmos de processamento de imagem desenvolvidos foram baseados na biblioteca de *software* de processamento de imagem MVTec HALCON. O HALCON é um *software* constituído por muitas ferramentas que facilitam o desenvolvimento de algoritmos de processamento de imagem, no entanto apresenta uma "curva de aprendizagem" longa. O único meio de suporte, para além da documentação do fabricante, é o serviço de suporte do vendedor desta ferramenta. Uma vez que o HALCON não é um *software open source*, não existe uma comunidade aberta que providencie recursos e exemplos de aplicações de outros utilizadores deste *software*, o que reduz as bases de desenvolvimento aos programas modelo fornecidos na instalação do IDE do HALCON.

No projeto Aquatropolis, o aluno foi responsável pelo desenvolvimento da Unidade de Controlo da Biomassa (UCB), uma ferramenta baseada em visão por computador com o intuito de efetuar a contagem e estimativa da massa de peixes em tanques de aquicultura. O desenvolvimento desta solução permitiu expor o aluno ao domínio da visão por computador, área sobre o qual não possuía quaisquer conhecimentos.

No final do projeto, obteve-se um sistema de visão artificial que conseguia extrair a forma tridimensional de um peixe e fazer um cálculo aproximado do seu comprimento. O algoritmo desenvolvido não cumpriu na totalidade alguns dos requisitos inicialmente definidos. As limitações deste sistema estão relacionadas com as condições de visibilidade existentes dentro dos tanques de aquacultura. Para além destas limitações,

existiram algumas dificuldades técnicas no processo de calibração e posicionamento das câmaras e afinação de alguns componentes do algoritmo desenvolvido.

Ainda no âmbito deste projeto, o aluno desenvolveu algumas componentes do *software* e hardware da Unidade de Monitorização Ambiental (UMA) e do veículo aquático, ROV, desenhado para acoplar tanto esta unidade como a UCB. Relativamente a estas componentes do projeto, a UMA e o ROV, é possível concluir-se que foram cumpridos os objetivos propostos inicialmente. No caso particular da UMA, a arquitetura proposta nos objetivos inicias sofreu algumas alterações pois pretendia-se o desenvolvimento de uma unidade baseada em redes de sensores. No entanto, optou-se por conferir mobilidade a esta unidade acoplando-a ao ROV. No caso da UMA, não existiram muitas dificuldades no seu processo de desenvolvimento. Muitos dos módulos de *software* e hardware estabelecidos para esta unidade têm por base soluções criadas para outros projetos desenvolvidos anteriormente pela equipa do LINE.IPT, conhecimento que veio a facilitar o processo de desenvolvimento deste sistema.

Relativamente ao desenvolvimento do ROV, existiram algumas dificuldades relacionadas sobretudo com o processo da impermeabilização dos encapsulamentos das unidades acopladas ao ROV, devido a problemas com entrada de água nos *housings* das câmaras e dos motores de terra. Conclui-se também que o material da sua estrutura base não foi o mais adequado, uma vez que sofreu alguma deformação ao longo dos testes, devido ao peso e à movimentação da estrutura dentro de água. Conclui-se assim que deveriam ser escolhidos outros materiais para os elementos fabricados em impressão 3D, assim como para a estrutura do ROV, materiais esses que deveriam ter melhores características de impermeabilidade e rigidez.

No projeto desenvolvido com a Vestas, o aluno desenvolveu também uma solução de visão por computador e processamento de imagem, baseada no princípio da triangulação laser "*sheet of light*", para quantificar o desgaste dos dentes das *yaw rings*, parte integrante das torres eólicas desta empresa. Neste projeto, a equipa do LINE.IPT colaborou sobretudo no desenvolvimento do *software* Web da plataforma de interação com o utilizador e das estruturas de suporte do sistema de câmaras e laser. Os procedimentos iniciais e todo o processo de desenvolvimento deste projeto foram acompanhados tanto pela empresa a quem se destinava esta ferramenta, a Vestas, como

pela empresa que fornece ao LINE.IPT o material e serviços para os projetos no âmbito da visão por computador.

À semelhança da solução de visão por computador apresentada para o projeto Aquatropolis, também nos trabalhos de desenvolvimento desta solução de medição do desgaste em rodas dentadas, existiram algumas dificuldades relacionadas com a calibração e posicionamento das câmaras. O posicionamento das câmaras revelou-se um dos grandes desafios do projeto. Tal consideração surge devido à complexidade em determinar um compromisso entre os ângulos das câmaras e o laser que permitisse executar a técnica de triangulação laser corretamente e devido à dificuldade em colocar o sistema num espaço tão limitado, como a zona onde deveria ser fixado para executar as medições necessárias. Esta solução continua, atualmente, a ser desenvolvida por parte da equipa do LINE.IPT, não tendo sido terminado ainda todo o projeto no período em que o aluno efetuou o estágio.

Apesar dos resultados obtidos em ambos os projetos apresentarem algumas limitações, desenvolver soluções baseadas em tecnologia de visão e processamento de imagem revelou-se um processo de grande interesse, que contribuíram para o desenvolvimento do conhecimento técnico e científico do aluno. Com estes projetos, o aluno teve oportunidade de interagir com uma equipa multidisciplinar e com várias entidades e instituições, permitindo conhecer um pouco da realidade do mundo empresarial na área da tecnologia e da Engenharia Eletrotécnica, promovendo assim o crescimento profissional do aluno.

8 Referências

[1] TAGUSVALLEY, «TAGUSVALLEY, Tecnopolo do Vale do Tejo». [Em linha]. Disponível em: http://tagusvalley.pt/pt/tagusvalley/apresentacao/. [Acedido: 06-Jul-2019].

[2] TAGUSVALLEY, «INOV.POINT Inovação e Desenvolvimento Empresarial». [Em linha]. Disponível em: http://tagusvalley.pt/pt/servicos/inov-point-inovacao-e-desenvolvimento-empresarial/. [Acedido: 06-Jul-2019].

[3] TAGUSVALLEY, «INOV'LINEA Transferência de Tecnologia Alimentar». [Em linha]. Disponível em: http://tagusvalley.pt/pt/servicos/inovlinea-transferencia-de-tecnologia-alimentar/. [Acedido: 06-Jul-2019].

[4] TAGUSVALLEY, «LINE.IPT Inovação Industrial e Empresarial». [Em linha]. Disponível em: http://tagusvalley.pt/pt/servicos/line-ipt-inovacao-industrial-e-empresarial/. [Acedido: 06-Jul-2019].

[5] mediotejo.net, «Abrantes | Parque Tecnológico – Tagusvalley mostra as suas valências», 2017. [Em linha]. Disponível em: http://www.mediotejo.net/abrantes-parque-tecnologico-tagusvalley-mostra-as-suas-valencias. [Acedido: 23-Set-2019].

[6] B. Jähne, *Digital Image Processing*. 2005.

[7] D. Marr, *Vision*. 1982.

[8] T. S. Huang, «Computer Vision: Evolution and Promise», *Report*, 1997.

[9] V. Wiley e T. Lucas, «Computer Vision and Image Processing: A Paper Review», *Int. J. Artif. Intell. Res.*, vol. 2, n. 1, p. 22, 2018.

[10] R. Szeliski, *Computer Vision: Algorithms and Applications*. 2010.

[11] J. R. Mathiassen, E. Misimi, M. Bondø, E. Veliyulin, e S. O. Østvik, «Trends in application of imaging technologies to inspection of fish and fish products», *Trends Food Sci. Technol.*, vol. 22, n. 6, pp. 257–275, 2011.

[12] B. Zion, «The use of computer vision technologies in aquaculture - A review», *Comput. Electron. Agric.*, vol. 88, pp. 125–132, 2012.

[13] M. Saberioon, A. Gholizadeh, P. Cisar, A. Pautsina, e J. Urban, «Application of machine vision systems in aquaculture with emphasis on fish: state-of-the-art and key issues», *Rev. Aquac.*, vol. 9, n. 4, pp. 369–387, 2016.

[14] T. A. Beddow e L. G. Ross, «Predicting biomass of Atlantic salmon from morphometric lateral measurements», *J. Fish Biol.*, vol. 49, n. 3, pp. 469–482, 1996.

[15] N. J. C. Strachan, «Length measurement of fish by computer vision», *Comput. Electron. Agric.*, vol. 8, n. 2, pp. 93–104, 1993.

[16] M. O. Balaban, M. Chombeau, D. Cirban, e B. Gümüş, «Prediction of the weight of Alaskan Pollock using image analysis», *J. Food Sci.*, vol. 75, n. 8, pp. 552–556, 2010.

[17] M. O. Balaban, G. F. Ünal Şengör, M. G. Soriano, e E. G. Ruiz, «Using image analysis to predict the weight of alaskan salmon of different species», *J. Food Sci.*, vol. 75, n. 3, 2010.

[18] B. Gümüş e M. O. Balaban, «Prediction of the Weight of Aquacultured Rainbow Trout (Oncorhynchus mykiss) by Image Analysis», 2010.

[19] B. P. Hufschmied, T. Fankhauser, e D. Pugovkin, «Automatic stress-free sorting of sturgeons inside culture tanks using image processing», vol. 27, pp. 622–626, 2011.

[20] B. P. Ruff, J. A. Marchant, e A. R. Frost, *Fish sizing and monitoring using a stereo image analysis system applied to fish farming*, vol. 14, n. 2. 1995.

[21] R. Tillett, N. McFarlane, e J. Lines, «Estimating dimensions of free-swimming fish using 3D point distribution models», *Comput. Vis. Image Underst.*, vol. 79, n. 1, pp. 123–141, 2000.

[22] J. R. Martinez-De Dios, C. Serna, e A. Ollero, «Computer vision and robotics techniques in fish farms», *Robotica*, vol. 21, n. 3, pp. 233–243, 2003.

[23] C. Costa, A. Loy, S. Cataudella, D. Davis, e M. Scardi, «Extracting fish size using dual underwater cameras», vol. 35, pp. 218–227, 2006.

[24] C. Costa, M. Scardi, V. Vitalini, e S. Cataudella, «A dual camera system for counting and sizing Northern Blue fi n Tuna (Thunnus thynnus ; Linnaeus , 1758) stock , during transfer to aquaculture cages , with a semi automatic Arti fi cial Neural Network tool», *Aquaculture*, vol. 291, n. 3–4, pp. 161–167, 2009.

[25] W. P. Lee, M. A. Osman, A. Z. Talib, J.-M. Ogier, e K. Yahya, «Tracking Multiple Fish in a Single Tank Using an Improved Particle Filter», *Adv. Intell. Soft Comput.*, vol. 279, pp. 799–804, 2014.

[26] Y. Atoum, S. Srivastava, e X. Liu, «Automatic feeding control for dense aquaculture fish tanks», *IEEE Signal Process. Lett.*, vol. 22, n. 8, pp. 1089–1093, 2015.

[27] J. H. Churnside, D. A. Demer, e B. Mahmoudi, «A comparison of lidar and echosounder measurements of fish schools in the Gulf of Mexico», *ICES J. Mar. Sci.*, vol. 60, n. 1, pp. 147–154, 2003.

[28] D. Pelletier *et al.*, «Remote high-definition rotating video enables fast spatial survey of marine underwater macrofauna and habitats», *PLoS One*, vol. 7, n. 2, pp. 1–13, 2012.

[29] A. C. R. Gleason, R. P. Reid, e K. J. Voss, «Automated classification of underwater multispectral imagery for coral reef monitoring», *Ocean. Conf. Rec.*, pp. 0–7, 2007.

[30] R. P. Mueller, R. S. Brown, H. Hop, e L. Moulton, «Video and acoustic camera techniques for studying fish under ice: A review and comparison», *Rev. Fish Biol. Fish.*, vol. 16, n. 2, pp. 213–226, 2006.

[31] D. A. Forsyth e J. Ponce, *Computer Vision: A Modern Approach*. 2003.

[32] Mvtec Halcon, «Basics». 2016.

[33] K. Lu, X. Wang, Z. Wang, e L. Wang, «Binocular stereo vision based on open CV», *IET Conf. Publ.*, vol. 2011, n. 582 CP, pp. 74–77, 2011.

[34] A. Lipnickas e A. KnyŜ, «A stereovision system for 3-d perception», *Elektron. ir Elektrotechnika*, n. 3, pp. 99–102, 2009.

[35] P. Serafinavicius, S. Sajauskas, e G. Daunys, «Evaluation of hand pointing system based on 3-D computer vision», *Elektron. ir Elektrotechnika*, n. 8, pp. 95–98, 2008.

[36] G. Bradski e A. Kaehler, *Learning OpenCV*. 2008.

[37] G. A. Buonaccorsi, A. J. Lacey, e N. A. Thacker, «Characterisation of a Stereo Matching and Object Location System Report for BAe Systems», *Biomed. Eng. (NY).*, vol. 17, pp. 1–34, 2003.

[38] B. Sridhar e R. Suorsa, «Comparison of Motion and Stereo Methods in Passive Ranging Systems», *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, n. 4, pp. 741–746, 1991.

[39] Mvtec Halcon, «Solution Guide III-C 3D Vision». 2016.

[40] J. Heikkila e O. Silvén, «A Four-step Camera Calibration Procedure with Implicit Image Correction», *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1106–1112, 1997.

[41] Z. Zhang, «Camera calibration with one-dimensional objects», *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2353, n. August 2004, pp. 161–174, 2002.

[42] S. J. Maybank e O. D. Faugeras, «A theory of self-calibration of a moving camera», *Int. J. Comput. Vis.*, vol. 8, n. 2, pp. 123–151, 1992.

[43] A. Heyden, *Three-Dimensional Geometric Computer Vision*. 2005.

[44] O. D. Faugeras, *Three-Dimensional Computer Vision*. 1992.

[45] B. K. . Horn, *Robot Vision*. 1986.

[46] D. H. S. de Castro, «Método e Ferramenta de Apoio à Realização de Croquis de Acidentes Rodoviários a Partir de Imagens Aéreas Duarte Henriques Sodré de Castro Dissertação para obtenção do Grau de Mestre em Engenharia Mecânica Novembro 2017 Agradecimentos», 2017.

[47] R. Y. Tsai, «A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses», *IEEE J. Robot. Autom.*, vol. 3, n. 4, pp. 323–344, 1987.

[48] Mvtec Halcon, «HALCON / HDevelop Operator Reference». 2016.

[49] Wikipedia, «Pyramid (image processing)». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/Pyramid_(image_processing). [Acedido: 21-Set-2019].

[50] C. Mulsow, M. Schulze, e P. Westfeld, «An Optical triangulation method for height measurements on water surfaces», *IAPRS*, vol. 36, pp. 214–217, 2006.

[51] D. Bračun, M. Jezeršek, e J. Diaci, «Triangulation model taking into account light sheet curvature», *Meas. Sci. Technol.*, vol. 17, n. 8, pp. 2191–2196, 2006.

[52] D. Page, A. Koschan, S. Voisin, N. Ali, e M. Abidi, «3D CAD model generation of mechanical parts using coded-pattern projection and laser triangulation systems», *Assem. Autom.*, vol. 25, n. 3, pp. 230–238, 2005.

[53] G. Godin *et al.*, «Active optical 3D imaging for heritage aplications», *IEEE Comput. Graph. Appl.*, vol. 22, n. 5, pp. 24–36, 2002.

[54] S. Kumar, P. K. Tiwari, e S. B. Chaudhury, «An optical triangulation method for non-contact profile measurement», *Proc. IEEE Int. Conf. Ind. Technol.*, pp. 2878–2883, 2006.

[55] L. Inc., *Literature on Laser Diode Structure Light Products*. Quebec, Canada H4R 2K3: LASIRIS Inc., 2002.

[56] Vestas Wind Systems A/S, «Vestas». [Em linha]. Disponível em: https://www.vestas.com/. [Acedido: 28-Out-2019].

[57] Wikipedia, «Yaw System». [Em linha]. Disponível em: https://en.wikipedia.org/wiki/Yaw_drive. [Acedido: 28-Out-2019].

9 Anexos

9.1 Anexo 1 – Algoritmo do Projeto Aqautropolis para Calibração do Par de Câmaras Estéreo

```
*Algoritmo de aquisição de imagens de calibração
dev_close_window ()
open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137694', 0, -1, AcqHandle1)
set_framegrabber_param (AcqHandle1, 'ExposureTime', 10000.0)

open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137535', 0, -1, AcqHandle2)
set_framegrabber_param (AcqHandle2, 'ExposureTime', 10000.0)

*Obter tamanho das imagens fotografadas
grab_image_start (AcqHandle2, -1)
grab_image_start (AcqHandle1, -1)
grab_image_async (ImageL, AcqHandle1, -1)
get_image_size (ImageL, Width, Height)

* Preparação da janela de visualização do HALCON IDE para auxílio na aquisição das *imagens de calibração
Scale: = .2
FontSize: = 14
dev_open_window (0, 0, Width*Scale, Height*Scale, 'black', WindowHandle1)
dev_set_draw ('margin')
set_display_font (WindowHandle1, FontSize, 'mono', 'true', 'false')
dev_open_window (0, Width*Scale + 10, Width*Scale, Height*Scale, 'black', WindowHandle2)
dev_set_window (WindowHandle2)
dev_set_draw ('margin')
set_display_font (WindowHandle2, FontSize, 'mono', 'true', 'false')
Button: = []
NumIgnoredImg:= 0
PoseIndex := 0
dev_update_off ()

*Criação dos Modelos das câmaras
gen_cam_par_area_scan_division (0.005, 135, 3.45e-6, 3.45e-6, Width / 2.0, Height / 2.0, Width, Height, StartCamParL)
StartCamParR := StartCamParL
```

```
*Criação dos modelos de calibração em que irá ser armazenada toda a informação
*relativa à calibração, incluindo a coordenadas na imagem das marcas de calibração e as
*poses de observação da placa de calibração
create_calib_data ('calibration_object', 2, 1, CalibDataID)
set_calib_data_cam_param (CalibDataID, 0, [], StartCamParL)
set_calib_data_cam_param (CalibDataID, 1, [], StartCamParR)
set_calib_data_calib_object (CalibDataID, 0, 'caltab_200mm.descr')

* Aquisição das imagens de calibração
* As imagens são guardadas quando se clica com o botão do rato na primeira janela
*auxiliar, onde deverá se visível a imagem da câmara esquerda
* A segunda janela mostra a imagem de câmara direita
while (PoseIndex < 20)
        grab_image_async (ImageL, AcqHandle1, -1)
        dev_set_window (WindowHandle1)
        dev_display (ImageL)
        grab_image_async (ImageR, AcqHandle2, -1)
        dev_set_window (WindowHandle2)
        dev_display (ImageR)
        disp_message (WindowHandle1, 'Imagem: ' + PoseIndex + ' guardada', 'window',
        12, 12, 'black', 'true')
        try
                get_mposition (WindowHandle1, Row, Column, Button)
        catch (Exception)
        endtry
        if (Button >0)
                disp_message (WindowHandle1, 'Imagem: ' + PoseIndex + ' guardada',
                'window', Row, Column, 'black', 'true')
                find_calib_object (ImageL, CalibDataID, 0, 0, PoseIndex, [], [])
                find_calib_object (ImageR, CalibDataID, 1, 0, PoseIndex, [], [])
                visualize_observation_results (ImageL, CalibDataID, 0, PoseIndex,
                WindowHandle1)
                visualize_observation_results (ImageR, CalibDataID, 1, PoseIndex,
                WindowHandle2)
                write_image (ImageL, 'png', 0, 'stereo_calib/left/' +
                'cam1_img_'+PoseIndex+'.png')
                write_image (ImageR, 'png', 0, 'stereo_calib/right/' +
                'cam2_img_'+PoseIndex+'.png')
                dev_clear_obj (ImageL)
                dev_clear_obj (ImageR)
                stop ()
                Button: = []
                PoseIndex: = PoseIndex + 1
```

```
        endif
endwhile
close_framegrabber (AcqHandle1)
close_framegrabber (AcqHandle2)
stop()

*Calibração propriamente dita
calibrate_cameras (CalibDataID, Errors)
* Obter os parâmetros internos das câmaras determinados pela calibração
get_calib_data (CalibDataID, 'camera', 0, 'params', CamParamL)
get_calib_data (CalibDataID, 'camera', 1, 'params', CamParamR)

*Corrigir o valor de Z da pose relativa entre câmaras considerando a espessura da placa
* de calibração (6,5mm)
thickness: =0.0065
get_calib_data (CalibDataID, 'camera', 1, 'pose', cLPcR)
set_origin_pose (cLPcR, 0, 0, thickness, poseRighttoleft)

*Armazenar todos estes parâmetros em ficheiros individuais
write_cam_par (CamParamL, 'cam_left.dat')
write_cam_par (CamParamR, 'cam_right.dat')
write_pose (poseRighttoleft, 'pos_right2left.dat')
```

9.2 Anexo 2 – Algoritmo Projeto Aquatropolis para Medição de Peixes e Determinação da Respetiva Massa

```
*O início e fim deste programa é executado por uma aplicação externa (servidor nodejs)
*ErroHandling deixa de ser automático, passando a ser efetuado recorrendo ao algoritmo
dev_error_var (Error, 1)
dev_set_check ('~give_error')

* Conexão socket TCP
Protocol: = 'TCP4'
Timeout: = 2.0
open_socket_connect ('192.168.1.1', 8080, ['protocol','timeout'], [Protocol, Timeout], AcceptingSocket)
ReadError:=Error
*Error = 5615 -> Conexão falhada
* Verificação da conexão e tentar nova conexão até ao sucesso
while (ReadError == 5615)
        open_socket_connect ('192.168.1.1', 8080, ['protocol','timeout'], [Protocol, Timeout], AcceptingSocket)
        ReadError:=Error
endwhile

*Leitura de dados de calibração
read_cam_par ('files_calib/left/cam_left.dat', CamParamL)
read_cam_par ('files_calib/right/cam_right.dat', CamParamR)
read_pose ('files_calib/pos_right2left.dat', RelPose)

*Criar mapas de retificação do estéreo para cada câmara
gen_binocular_rectification_map (MapL, MapR, CamParamL, CamParamR, RelPose, 1, 'geometric', 'bilinear', RectCamParL, RectCamParR, CamPoseRectL, CamPoseRectR, RectLPosRectR)

*Conexão às duas câmaras com verificação de sucesso de conexão
ReadError: =5312
while (ReadError == 5312)
        open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137694', 0, -1, AcqHandle1)
        ReadError: =Error
Endwhile

ReadError: =5312
while (ReadError == 5312)
        open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'default', -1, 'default', -1, 'false', 'default', 'S1137535', 0, -1, AcqHandle2)
```

```
        ReadError:=Error
Endwhile
*Definir tempo de exposição das duas câmaras, configurações das câmaras como mestres
*e escravo. A câmara esquerda (AcqHandle1) é o mestre e a câmara direita (AcqHandle2)
* é o escravo
* Início da aquisição assíncrona de imagens
set_framegrabber_param (AcqHandle1, 'ExposureTime', 800.0)
set_framegrabber_param (AcqHandle1, 'TriggerMode', 'On')
set_framegrabber_param (AcqHandle1, 'LineSelector', 'Line1')
set_framegrabber_param (AcqHandle1, 'lineDebouncingPeriod', 255)

set_framegrabber_param (AcqHandle2, 'ExposureTime', 800.0)
set_framegrabber_param (AcqHandle2, 'TriggerMode', 'Off')
set_framegrabber_param (AcqHandle2, 'LineSelector', 'Line3')
set_framegrabber_param (AcqHandle2, 'outputLineSource', 'PulseOnStartofFrame')
set_framegrabber_param (AcqHandle2, 'outputLinePulseDuration', 500)

grab_image_start (AcqHandle2, -1)
grab_image_start (AcqHandle1, -1)

*Ciclo de aquisição de imagens infinito (até que este programa seja interrompido
*externamente)
while(1)
        *Aquisição assíncrona de imagens
        *As câmaras têm de ser configuradas previamente no modo mestre-escravo, sendo
        *considerada como câmara mestre a câmara esquerda e escravo a direita
        *Caso exista erro na aquisição, é reiniciada a conexão à câmara
        grab_image_async (ImageL, AcqHandle1, -1)
        ReadError: =Error
        while (ReadError! =2)
                close_framegrabber (AcqHandleL)
                wait_seconds (2)
                open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'progressive', -1,
                'default', -1, 'false', 'default', 'S1187109', 0, -1, AcqHandleL)
                ReadError: =Error
                grab_image_async (ImageL, AcqHandle1, -1)
        endwhile
        grab_image_async (ImageR, AcqHandle2, -1)
        ReadError: =Error
        while (ReadError! =2)
                close_framegrabber (AcqHandleR)
                wait_seconds (2)
                open_framegrabber ('GigEVision2', 1, 1, 0, 0, 0, 0, 'progressive', -1,
                'default', -1, 'false', 'default', 'S1187124', 0, -1, AcqHandleR)
```

```
        ReadError: =Error
        grab_image_async (ImageR, AcqHandle2, -1)
endwhile

*Retificação do par de imagens com os mapas de retificação previamente criados
map_image (ImageL, MapL, ImageRectifiedL)
map_image (ImageR, MapR, ImageRectifiedR)

*Analise das imagens de forma a verificar a existência de um objeto com área em
*pixéis semelhante ao corpo de um peixe, nas condições verificadas nos testes
threshold (ImageRectifiedL, Region, 70, 195)
fill_up (Region, RegionFillUp)
dilation_circle (RegionFillUp, RegionDilation, 60)
connection (RegionDilation, ConnectedRegions)
select_shape (ConnectedRegions, SelectedRegions, 'area', 'and', 70957.6, 350000)
union1(SelectedRegions, RegionUnion)
shape_trans (RegionUnion, RegionTrans, 'convex')
area_center (RegionTrans, Area, Row, Column)
if(Area>0)
        *Obter imagem de disparidades
        binocular_disparity (ImageRectifiedL, ImageRectifiedR, DisparityImage,
        Score, 'ncc', 17, 17, 5, 400, 800, 2, 0.1, 'left_right_check', 'none')
        *Reduzir a imagem de disparidades à região onde se presume estar o peixe
        reduce_domain (DisparityImage, RegionTrans, ImageReduced1)
        threshold (ImageReduced1, Union2, 400,800)
        dilation_circle (Union2, RegionDilation1,190)
        shape_trans (RegionDilation1, RegionTrans1, 'convex')
        reduce_domain (DisparityImage, RegionTrans1, ImageReduced)

        *Preencher as regiões sem valores de disparidade definidos, por
        *interpolação, de forma a poder obter a imagens de coordenadas
        get_domain (ImageReduced, RegionInpainting)
        complement (RegionInpainting, RegionInpainting)
        full_domain (ImageReduced, ImageReduced)
        harmonic_interpolation (ImageReduced, RegionInpainting,
        ImageReduced, 0.001)
        *Conversão da imagem da secção da imagem de disparidades selecionada
        *para imagens de coordenadas X Y e Z, em valores reais
        disparity_image_to_xyz (ImageReduced, X, Y, Z, RectCamParL,
        RectCamParR, RectLPosRectR)
        *Obter o modelo 3D do peixe segmentado com base nas imagens de
        *coordenadas
        reduce_domain (Z, RegionTrans1, Z)
        median_image (Z, ImageMedian, 'circle', 80, 'mirrored')
```

```
xyz_attrib_to_object_model_3d (X, Y, ImageMedian, ImageRectifiedL, '&amp;amp; gray', ObjectModel3D)
*Preparação do modelo 3D caso se queira visualizar este modelo no IDE
prepare_object_model_3d (ObjectModel3D, 'segmentation', 'true', 'max_area_holes', 100)
*Obter a “caixa" mais pequena que pode envolver o modelo 3D gerado, de
* forma a determinar as dimensões aproximadas do peixe
hom_mat3d_identity (HomMat3DIdentity)
hom_mat3d_rotate (HomMat3DIdentity, 0, 'y', 0, 0, 0, HomMat3DRotate)
affine_trans_object_model_3d (ObjectModel3D, HomMat3DRotate, ObjectModel3DAffineTrans)
smallest_bounding_box_object_model_3d (ObjectModel3D, 'oriented', PoseBoxOri, Length1Ori, Length2Ori, Length3Ori)
*A função abaixo prepara o modelo 3D da “caixa” para mostrar no IDE
gen_box_object_model_3d (PoseBoxOri, Length1Ori, Length2Ori, Length3Ori, BoundingBoxesOriented)
*Conversão das dimensões de metros para centímetros
Length3Ori: =Length3Ori*100
Length2Ori: =Length2Ori*100
Length1Ori: =Length1Ori*100
*O comprimento do peixe será a maior dimensão da “caixa” obtida
comp: = [Length3Ori, Length2Ori, Length1Ori]
tuple_max (comp, Max)
tuple_find (comp, Max, Indices)
tuple_max (Indices, Max1)
length: =comp [Max1]
* Calculo da massa do peixe com base no seu comprimento
*robalo
a:=0.01090
b:=2.980
*dourada
*       a: =0.01311
*       b: =3.040
W: =a*pow (length, b)
s: ='{"weight": '+W+'}'
*Envio do valor de massa obtido via SocketTCP para o servidor nodejs
send_data (AcceptingSocket, 'z', s, [])
*Verificação de erros de envio
*Caso exista erro é reaberto o SocketTCP
ReadError2: =Error
while (ReadError2!= 2)
        open_socket_connect ('192.168.1.1', 8080, ['protocol','timeout'], [Protocol, Timeout], AcceptingSocket)
        ReadError: =Error
```

```
                    while (ReadError == 5615)
                        open_socket_connect ('192.168.1.1', 8080,
                        ['protocol','timeout'], [Protocol, Timeout],
                        AcceptingSocket)
                        ReadError: =Error
                        wait_seconds (1)
                    endwhile
                    send_data (AcceptingSocket, 'z', s, [])
                    ReadError2: =Error
                endwhile
            endif
endwhile
```

9.3 Anexo 3 – Algoritmo de Medição do Desgaste dos Dentes da Yaw Ring do Projeto com a Vestas

```
dev_update_off ()
dev_close_window ()
dev_close_window ()
dev_close_window ()

*Calibração relacionada com a junção do par de imagens
sum_two_images_for_laser (MapSingle1, MapSingle2, Width, Height, WindowHandle1,
WindowHandle2, WindowHandleCombined)

* Calibração das linhas laser, primeiro é calibrado o laser da imagem da câmara esquerda
*posteriormente o laser da imagem da câmara direita

**********Câmara Esquerda**********
*Abrir uma janela auxiliar no IDE
read_image (ProfileImage, 'images_calib/left/image_01.png')
get_image_size (ProfileImage, Width, Height)
Width: =.2*Width
Height: =.2*Height
dev_open_window (0, 0, Width, Height, 'black', WindowHandle)

* Parte 1: Calibração da câmara
read_cam_par ('files_calib/left/cam_param_left.cal', StartParameters_Left)
CalTabDescription: = 'caltab_100mm.descr'
CalTabThickness: = .005
NumCalibImages: = 25
read_calib_data ('files_calib/left/calib_sheet_of_light_left', CalibDataID_left)
calibrate_cameras (CalibDataID_left, Errors)

* Parte 2: Calibração dos planos "sheet of light" de cada linha laser

* Definição do sistema de coordenadas "mundo" world coordinate system (WCS):
MinThreshold: = 50
Index: = 18
get_calib_data (CalibDataID_left, 'calib_obj_pose', [0, Index], 'pose', CalTabPose_left)
set_origin_pose (CalTabPose_left, 0.0, 0.0, CalTabThickness, CameraPose_left)
read_image (CalTabImage1, 'images_calib/left/image_' + Index$'.2')
get_calib_data (CalibDataID_left, 'camera', 0, 'params', CameraParameters_left)

* Definição de um sistema de coordenada temporário (TCS):
Index: = 19
get_calib_data (CalibDataID_left, 'calib_obj_pose', [0, Index], 'pose', CalTabPose_left)
```

```
set_origin_pose (CalTabPose_left, 0.0, 0.0, CalTabThickness, TmpCameraPose_left)
read_image (CalTabImage2, 'images_calib/left/image_' + Index$'.2')

*Linhas laser projetadas no plano Z=0 do WCS  (bottom lines)
read_image (bottom_left, 'images_calib/bottom_left.png')
calculate_lines_gauss_parameters (45, 90, Sigma, Low, High)
lines_gauss (bottom_left, Lines, Sigma, Low, High, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines, SelectedXLD, 'max_diameter', 'and', 800, 2000)

* Linhas laser projetadas no plano Z=0 do TCS (top lines)
read_image (top_left, 'images_calib/top_left.png')
calculate_lines_gauss_parameters (45,90, Sigma1, Low1, High1)
lines_gauss (top_left, Lines2, Sigma1, Low1, High1, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines2, SelectedXLD1, 'max_diameter', 'and', 800, 2000)

*Selecionar 1ª linha laser projetada no Z=0 WCS
select_obj(SelectedXLD, ObjectSelected, 1)
*Selecionar 1ª linha laser projetada no Z=0 TCS
select_obj(SelectedXLD1, ObjectSelected2, 1)
*Definição do plano ou perfil da 1ª linha laser
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold,
CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left,
Left_LightP1)


*Procedimento 2ª linha laser
select_obj (SelectedXLD, ObjectSelected, 2)
select_obj (SelectedXLD1, ObjectSelected2, 2)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold,
CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left,
Left_LightP2)

*Procedimento 3ª linha laser
select_obj (SelectedXLD, ObjectSelected, 3)
select_obj (SelectedXLD1, ObjectSelected2, 3)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold,
CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left,
Left_LightP3)

*Procedimento 4ª linha laser
select_obj (SelectedXLD, ObjectSelected, 4)
select_obj (SelectedXLD1, ObjectSelected2, 4)
```

laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP4)

*Procedimento 5ª linha laser
select_obj (SelectedXLD, ObjectSelected,5)
select_obj (SelectedXLD1, ObjectSelected2, 5)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP5)

*Procedimento 6ª linha laser
select_obj (SelectedXLD, ObjectSelected, 6)
select_obj (SelectedXLD1, ObjectSelected2, 6)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP6)

*Procedimento 7ª linha laser
select_obj (SelectedXLD, ObjectSelected, 7)
select_obj (SelectedXLD1, ObjectSelected2, 7)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP7)

*Procedimento 8ª linha laser
select_obj (SelectedXLD, ObjectSelected,8)
select_obj (SelectedXLD1, ObjectSelected2, 8)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP8)

*Procedimento 9ª linha laser
select_obj (SelectedXLD, ObjectSelected,9)
select_obj (SelectedXLD1, ObjectSelected2, 9)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP9)

*Procedimento 10ª linha laser
select_obj (SelectedXLD, ObjectSelected,10)
select_obj (SelectedXLD1, ObjectSelected2, 10)

```
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP10)

*Procedimento 11ª linha laser
 select_obj (SelectedXLD, ObjectSelected,11)
 select_obj (SelectedXLD1, ObjectSelected2, 11)
laser_procedure (ObjectSelected, bottom_left, ObjectSelected2, top_left, MinThreshold, CameraParameters_left, CameraPose_left, WindowHandle, TmpCameraPose_left, Left_LightP11)

dev_clear_obj (ObjectSelected)
dev_clear_obj (ObjectSelected2)


* Parte 3: Calibração do movimento do objeto efetuado entre a aquisição de duas imagens
*consecutivas

read_image (CaltabImagePos1, 'images_calib/left/image_20.png')
read_image (CaltabImagePos20, 'images_calib/left/image_21.png')
StepNumber: = 20
movement_calibration (CaltabImagePos1, CaltabImagePos20, CalibDataID_left, CameraParameters_left, NumCalibImages, CalTabThickness, CameraPose_left, StepNumber, MovementPoseLeft)

**************************Câmara Direita**************************
* Parte 1: Calibração da câmara
read_cam_par ('files_calib/right/cam_param_right.cal', StartParameters_Right)
NumCalibImages: = 25
read_calib_data ('files_calib/right/calib_sheet_of_light_right', CalibDataID_right)
calibrate_cameras (CalibDataID_right, Errors)

* Parte 2: Calibração dos planos "sheet of light" de cada linha laser
* Definição do sistema de coordenadas "mundo" world coordinate system (WCS):
MinThreshold: = 50
Index: = 18
get_calib_data (CalibDataID_right, 'calib_obj_pose', [0, Index], 'pose', CalTabPose_right)
set_origin_pose (CalTabPose_right, 0.0, 0.0, CalTabThickness, CameraPose_right)
read_image (CalTabImage1, 'images_calib/right/image_' + Index$'.2')
get_calib_data (CalibDataID_right, 'camera', 0, 'params', CameraParameters_right)

* Definição de um sistema de coordenada temporário (TCS):
Index: = 19
```

```
get_calib_data (CalibDataID_right, 'calib_obj_pose', [0, Index], 'pose', CalTabPose_right)
set_origin_pose (CalTabPose_right, 0.0, 0.0, CalTabThickness, TmpCameraPose_right)
read_image (CalTabImage2, 'images_calib/right/image_' + Index$'.2')

*Linhas laser projetadas no plano Z=0 do WCS  (bottom lines)
read_image (bottom_right, 'images_calib/bottom_right.png')
calculate_lines_gauss_parameters (20,80, Sigma, Low, High)
lines_gauss (bottom_right, Lines3, Sigma, Low, High, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines3, SelectedXLD3, 'max_diameter', 'and', 937.61, 2000)
*Linhas laser projetadas no plano Z=0 do TCS  (top lines)
read_image (top_right, 'images_calib/top_right.png')
calculate_lines_gauss_parameters (63 ,100, Sigma1, Low1, High1)
lines_gauss (top_right, Lines4, Sigma1, Low1, High1, 'light', 'true', 'bar-shaped', 'true')
select_shape_xld (Lines4, SelectedXLD4, 'max_diameter', 'and', 875.23, 2000)

*Selecionar 1ª linha laser projetada no Z=0 WCS
select_obj(SelectedXLD3, ObjectSelected, 1)
*Selecionar 1ª linha laser projetada no Z=0 TCS
select_obj(SelectedXLD4, ObjectSelected2, 1)
*Definição do plano ou perfil da 1ª linha laser
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right, MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle, TmpCameraPose_right, Right_LightP1)

*Procedimento 2ª linha laser
select_obj (SelectedXLD3, ObjectSelected, 2)
select_obj (SelectedXLD4, ObjectSelected2, 2)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right, MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle, TmpCameraPose_right, Right_LightP2)

*Procedimento 3ª linha laser
select_obj (SelectedXLD3, ObjectSelected,3)
select_obj (SelectedXLD4, ObjectSelected2, 3)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right, MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle, TmpCameraPose_right, Right_LightP3)
*Procedimento 4ª linha laser
select_obj (SelectedXLD3, ObjectSelected, 4)
select_obj (SelectedXLD4, ObjectSelected2, 4)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right, MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle, TmpCameraPose_right, Right_LightP4)
```

```
*Procedimento 5ª linha laser
select_obj (SelectedXLD3, ObjectSelected,5)
select_obj (SelectedXLD4, ObjectSelected2,5)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP5)

*Procedimento 6ª linha laser
select_obj (SelectedXLD3, ObjectSelected,6)
select_obj (SelectedXLD4, ObjectSelected2,6)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP6)

*Procedimento 7ª linha laser
select_obj (SelectedXLD3, ObjectSelected,7)
select_obj (SelectedXLD4, ObjectSelected2,7)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP7)

*Procedimento 8ª linha laser
select_obj (SelectedXLD3, ObjectSelected,8)
select_obj (SelectedXLD4, ObjectSelected2,8)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP8)

*Procedimento 9ª linha laser
select_obj (SelectedXLD3, ObjectSelected,9)
select_obj (SelectedXLD4, ObjectSelected2,9)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP9)

*Procedimento 10ª linha laser
select_obj (SelectedXLD3, ObjectSelected,10)
select_obj (SelectedXLD4, ObjectSelected2,10)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right,
MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle,
TmpCameraPose_right, Right_LightP10)

*Procedimento 11ª linha laser
```

```
select_obj (SelectedXLD3, ObjectSelected,11)
select_obj (SelectedXLD4, ObjectSelected2,11)
laser_procedure (ObjectSelected, bottom_right, ObjectSelected2, top_right, MinThreshold, CameraParameters_right, CameraPose_right, WindowHandle, TmpCameraPose_right, Right_LightP11)

* Parte 3: Calibração do movimento do objeto efetuado entre a aquisição de duas imagens
*consecutivas

read_image (CaltabImagePos1, 'images_calib/right/image_20.png')
read_image (CaltabImagePos20, 'images_calib/right/image_21.png')
StepNumber: = 20
movement_calibration (CaltabImagePos1, CaltabImagePos20, CalibDataID_right, CameraParameters_right, NumCalibImages, CalTabThickness, CameraPose_right, StepNumber, MovementPoseRight)

**********Medições**********
*Inicialização das câmaras
dev_clear_window ()
*Master Camera- Right
open_framegrabber ('GigEVision2', 0, 0, 0, 0, 0, 0, 'progressive', -1, 'default', -1, 'false', 'default','S1187109', 0, -1, AcqHandle2)
set_framegrabber_param (AcqHandle2, 'ExposureTime', 80000.0)
set_framegrabber_param (AcqHandle2, 'TriggerMode', 'Off')
set_framegrabber_param (AcqHandle2, 'LineSelector', 'Line3')
set_framegrabber_param (AcqHandle2, 'outputLineSource', 'PulseOnStartofFrame')
set_framegrabber_param (AcqHandle2, 'outputLinePulseDuration', 500)
set_framegrabber_param (AcqHandle2, 'ReverseX',1)
set_framegrabber_param (AcqHandle2, 'ReverseY',1)
grab_image_start (AcqHandle2, -1)

*Slave Camera- Left
open_framegrabber ('GigEVision2', 0, 0, 0, 0, 0, 0, 'progressive', -1, 'default', -1, 'false', 'default', 'S1187124', 0, -1, AcqHandle1)
set_framegrabber_param (AcqHandle1, 'ExposureTime', 80000.0)
set_framegrabber_param (AcqHandle1, 'TriggerMode', 'On')
set_framegrabber_param (AcqHandle1, 'LineSelector', 'Line1')
set_framegrabber_param (AcqHandle1, 'lineDebouncingPeriod', 255)
set_framegrabber_param (AcqHandle1, 'ReverseX',1)
set_framegrabber_param (AcqHandle1, 'ReverseY',1)
grab_image_start (AcqHandle1, -1)
```

```
while (1)
    grab_image_async (ImageL, AcqHandle1, -1)
    grab_image_async (ImageR, AcqHandle2, -1)
    map_image (ImageL, MapSingle1, RectifiedImage1)
    map_image (ImageR, MapSingle2, RectifiedImage2)

    *Junção do par de imagens com base na calibração efetuada anteriormente
    concat_obj (RectifiedImage1, RectifiedImage2, Concat)
    dev_set_window (WindowHandleCombined)
    tile_images (Concat, Combined, 2, 'vertical')

    get_image_size (Combined, WidthComb, HeightComb)
    dev_set_part (0, 0, HeightComb - 1, WidthComb - 1)
    dev_clear_window ()
    dev_display (Combined)
    wait_seconds(2)
    *Apesar da junção das imagens, neste algoritmo a medição é feita extraindo os
    *pontos de cada uma das imagens (esquerda e direita) em separado
    * Numa próxima versão deste algoritmo idealmente obter-se-ia a medição a partir
    * da imagem resultante da união

    *Laser1
    *Imagem esquerda
    gen_rectangle1 (ProfileRegion1, 699.275, 2155.73, 777.105, 2471.98)
    get_xyz_from_laser_profile (ImageL, ProfileRegion1, Xl, Y1, Z1,
    ImageReduced1, CameraParameters_left, CameraPose_left, Left_LightP1,
    MovementPoseLeft)
    threshold (Z1, Regions, 54, 54.99)
    connection (Regions, ConnectedRegions)
    area_center (ConnectedRegions, Area, Row, Column)
    get_grayval (X1,0, Column [0], Grayval_X1)
    get_grayval (Y1,0, Column [0], Grayval_Y1)
    Column: =0
    dev_clear_obj (ConnectedRegions)

    *Imagem direita
    gen_rectangle1 (ProfileRegion21, 733.601, 260.88, 805.925, 583.164)
    get_xyz_from_laser_profile (Image2, ProfileRegion21, X21, Y21, Z21,
    ImageReduced21, CameraParameters_right, CameraPose_right, Left_LightP1,
    MovementPoseLeft)
    threshold (Z21, Regions, 54, 54.99)
    connection (Regions, ConnectedRegions)
    area_center (ConnectedRegions, Area, Row, Column)
    get_grayval (X21,0, Column [0], Grayval_X21)
```

```
        get_grayval (Y21,0, Column [0], Grayval_Y21)
        Column: =0
        dev_clear_obj (ConnectedRegions)

        *****
        *O procedimento é semelhante para as restantes linhas
        ******

        *Medição do desgaste
        *Esta fase do algoritmo ainda não foi muito explorada, no entanto apresenta-se
        *um pequeno "esqueleto" daquilo que será a definição do valor de desgaste
        *valor original do perfil do dente ~30mm
        Original:=30
        Desgaste:=Original-(X21-X1)

endwhile
close_framegrabber (AcqHandle1)
close_framegrabber (AcqHandle2)
```

9.4 Anexo 4 – Algoritmo de Junção da Imagem de Duas Câmaras (sum_two_images_for_laser: Função Aplicada no Algoritmo Anterior)

```
*Abrir janelas para visualização no IDE
read_image (Image1,'Right/image_01.png')
get_image_size (Image1, Width, Height)
dev_open_window (0, 0, .2*Width, .2*Height, 'black', WindowHandle1)
dev_open_window (0,.2*Width+10, .2*Width, .2*Height, 'black', WindowHandle2)

*Ficheiros calibrações das camaras
* Camera Esquerda 'S1137535'
read_cam_par('Left/CameraParam_left.cal',CameraParamLeft)
read_pose('Left/Pose_left.dat',PoseLeft)
*Camera direita 'S1137694'
read_cam_par('Right/cam_param_right.cal',CameraParamRight)
read_pose('Right/pose_right.dat',PoseRight)

*Leitura das imagens do objeto de calibração
read_image(Image1,'double_calplate_left')
read_image(Image2,'double_calplate_right')

*Definição do objeto de calibração
CaltabName:= 'caltab_100mm.descr'
create_calib_data ('calibration_object', 2, 1, CalibDataID)
set_calib_data_calib_object (CalibDataID, 0, CaltabName)
set_calib_data_cam_param (CalibDataID, 0, [], CameraParamLeft)
set_calib_data_cam_param (CalibDataID, 1, [], CameraParamRight)

*Localização do objeto de calibração nas duas imagens
dev_set_window (WindowHandle1)
dev_display(Image1)
find_calib_object (Image1, CalibDataID, 0, 0, 0, [], [])
get_calib_data_observ_points (CalibDataID, 0, 0, 0, RowCoord1, ColumnCoord1,
Index1, Pose1)
get_calib_data_observ_contours (Caltab1, CalibDataID, 'caltab', 0, 0, 0)
 dev_display (Caltab1)

dev_set_window (WindowHandle2)
dev_display(Image2)
find_calib_object (Image2, CalibDataID, 1, 0, 0, [], [])
get_calib_data_observ_points (CalibDataID, 1, 0, 0, RowCoord2, ColumnCoord2,
Index2, Pose2)
get_calib_data_observ_contours (Caltab2, CalibDataID, 'caltab', 1, 0, 0)
```

```
dev_display (Caltab2)

disp_message (WindowHandle1, 'Calibration successful', 'window', 12, 12, 'black', 'true')
disp_continue_message (WindowHandle1, 'black', 'true')

Thicknessobj := 5.53/ 100.0
ThicknessPlate := 1.2/ 100.0
DiffHeight := ThicknessPlate - Thicknessobj
DistancePlates:=0.159

PixelSize := 0.0001

BorderInPercent :=5
get_image_size (Image1, WidthImage1, HeightImage1)
UpperRow := HeightImage1 * BorderInPercent / 100.0
LeftColumn := WidthImage1 * BorderInPercent / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, UpperRow, LeftColumn, 'm',
LeftX, UpperY)
set_origin_pose (Pose1, LeftX, UpperY, DiffHeight, PoseNewOrigin1)

LowerRow := HeightImage1 * (100 - BorderInPercent) / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, LowerRow, LeftColumn, 'm',
X1, LowerY)
HeightRect := int((LowerY - UpperY) / PixelSize)
OverlapInPercent := 25
RightColumn := WidthImage1 * (100 - OverlapInPercent / 2.0) / 100.0
image_points_to_world_plane (CameraParamLeft, Pose1, UpperRow, RightColumn, 'm',
RightX, Y1)
WidthRect := int((RightX - LeftX) / PixelSize)
gen_image_to_world_plane_map (MapSingle1, CameraParamLeft, PoseNewOrigin1,
Width, Height, WidthRect, HeightRect, PixelSize, 'bilinear')

hom_mat3d_identity (HomMat3DIdentity)
hom_mat3d_translate_local (HomMat3DIdentity, LeftX + PixelSize * WidthRect,
UpperY, DiffHeight, cp1Hur1)
hom_mat3d_translate_local (HomMat3DIdentity, DistancePlates, 0, 0, cp1Hcp2)
hom_mat3d_invert (cp1Hcp2, cp2Hcp1)
hom_mat3d_compose (cp2Hcp1, cp1Hur1, cp2Hul2)
pose_to_hom_mat3d (Pose2, cam2Hcp2)
hom_mat3d_compose (cam2Hcp2, cp2Hul2, cam2Hul2)
hom_mat3d_to_pose (cam2Hul2, PoseNewOrigin2)
gen_image_to_world_plane_map (MapSingle2, CameraParamRight, PoseNewOrigin2,
Width, Height, WidthRect, HeightRect, PixelSize, 'bilinear')
```

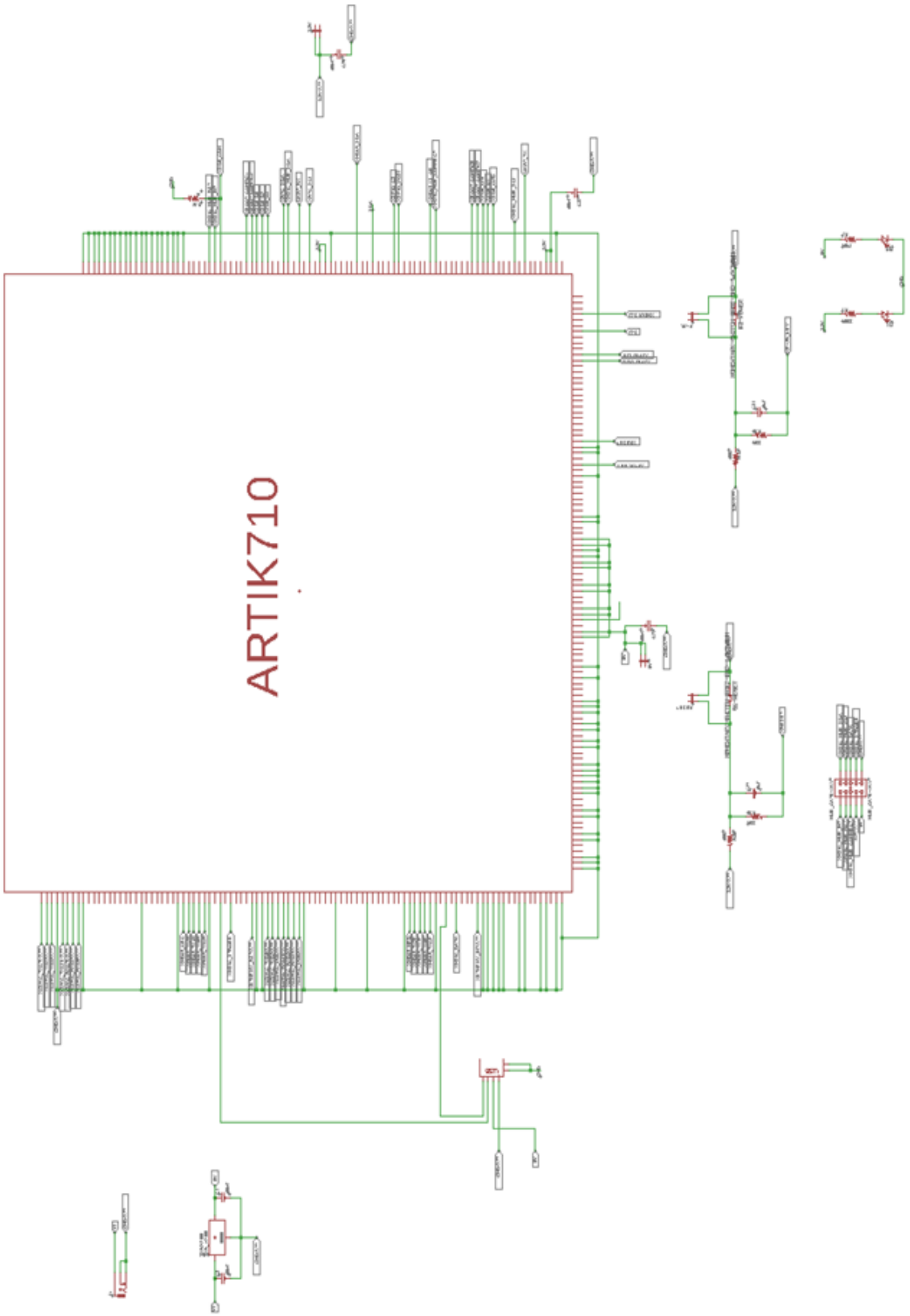
```
dev_open_window (0, 0, Width * 2 * 0.2, Height * 0.2, 'black',
WindowHandleCombined)
set_display_font (WindowHandleCombined, 16, 'mono', 'true', 'false')
dev_set_color ('green')
dev_set_draw ('margin')
ScalePlot := 200
RowPlot := 400
Coord := [0:2000]

dev_set_window (WindowHandle1)
get_image_size (Image1, WidthImage1, HeightImage1)
dev_set_part (0, 0, HeightImage1 - 1, WidthImage1 - 1)
dev_display (Image1)
dev_set_window (WindowHandle2)
get_image_size (Image2, WidthImage2, HeightImage2)
dev_set_part (0, 0, HeightImage2 - 1, WidthImage2 - 1)
dev_display (Image2)

map_image (Image1, MapSingle1, RectifiedImage1)
map_image (Image2, MapSingle2, RectifiedImage2)
concat_obj (RectifiedImage1, RectifiedImage2, Concat)

dev_set_window (WindowHandleCombined)
tile_images (Concat, Combined, 2, 'vertical')

get_image_size (Combined, WidthComb, HeightComb)
dev_set_part (0, 0, HeightComb - 1, WidthComb - 1)
dev_display (Combined)
WindowHandleCombined, RowPlot)
return ()
```

9.5 Anexo 5 – Algoritmo do Projeto Vestas para Determinação do Plano de Luz de Cada Linha do Laser

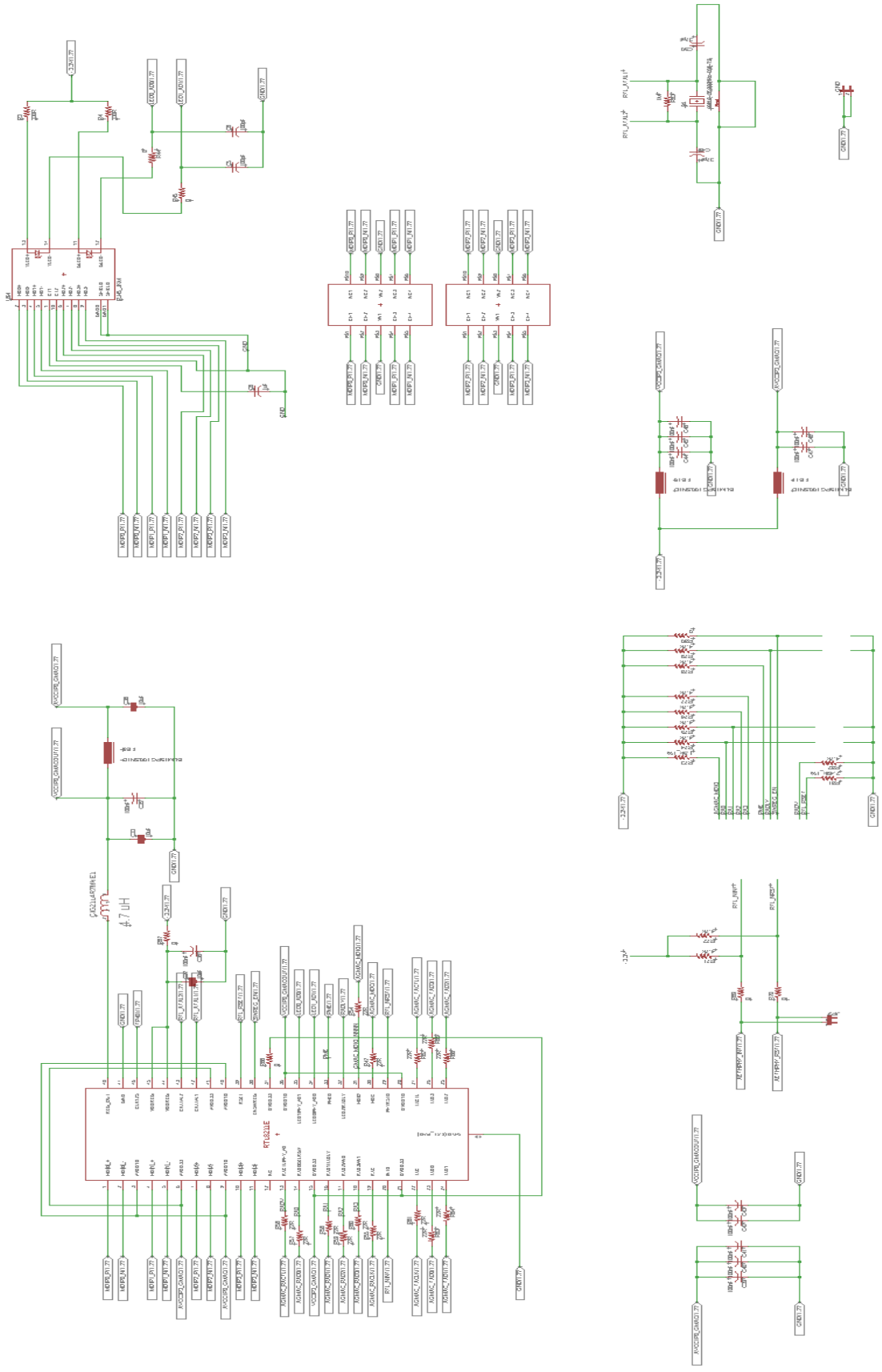
```
gen_region_contour_xld (ObjectSelected, Region, 'filled')
complement (Region, RegionComplement)
paint_region (RegionComplement, bottom, ImageResult, 0, 'fill')
compute_3d_coordinates_of_light_line (ImageResult, MinThreshold,
CameraParameters, [], CameraPose, X19, Y19, Z19)

if (|X19| == 0 or |Y19| == 0 or |Z19| == 0)
        disp_message (WindowHandle, 'The profile MUST be oriented horizontally\nfor
        successfull processing!\nThe program will exit.', 'window', 12, 12, 'black', 'true')
        stop()
        return ()
endif

gen_region_contour_xld (ObjectSelected10, Region10, 'filled')
dilation_circle (Region10, Region10, 4)
complement (Region10, RegionComplement10)
paint_region (RegionComplement10, top, ImageResult10, 0, 'fill')
compute_3d_coordinates_of_light_line (ImageResult10, MinThreshold,
CameraParameters, TmpCameraPose, CameraPose, X20, Y20, Z20)
if (|X20| == 0 or |Y20| == 0 or |Z20| == 0)
        disp_message (WindowHandle, 'The profile MUST be oriented horizontally\nfor
        successfull processing!\nThe program will exit.', 'window', 12, 12, 'black', 'true')
        stop()
        return ()
endif

fit_3d_plane_xyz ([X19,X20], [Y19,Y20], [Z19,Z20], Ox, Oy, Oz, Nx, Ny, Nz,
MeanResidual)
if (|Nx| == 0 or |Ny| == 0 or |Nz| == 0)
        disp_message (WindowHandle, 'Too few 3d points have been provided to fit the
        light plane,\nor the points are (nearly) collinear!\nThe program will exit.',
        'window', 12, 12, 'black', 'true')
        stop()
        return ()
endif
if (MeanResidual &gt; 5e-5)
        disp_message (WindowHandle, 'The light plane could not be fitted
        accurately!\nThe mean residual distance between the 3d-points and the\nfitted
        plane is too high (' + (MeanResidual * 1000)$'.3' + 'mm). Please check
        the\nquality and the correctness of those points.\nThe program will exit!',
        'window', 12, 21, 'black', 'true')
```

```
        stop()
        return ()
endif

get_light_plane_pose (Ox, Oy, Oz, Nx, Ny, Nz, LightPlanePose)
if (|LightPlanePose| != 7)
        disp_message (WindowHandle, 'The pose of the light plane could not
        be\ndetermined. Please verify that the vector\npassed at input of the
        procedure\nget_light_plane_pose() is not null.\nThe program will exit!', 'window',
        12, 12, 'black', 'true')
        stop()
        return ()
endif
String := ['LightPlanePose: ',' Tx    = ' + LightPlanePose[0]$'.3' + ' m',' Ty    = ' +
LightPlanePose[1]$'.3' + ' m',' Tz    = ' + LightPlanePose[2]$'.3' + ' m',' alpha = ' +
LightPlanePose[3]$'.4' + '°',' beta  = ' + LightPlanePose[4]$'.4' + '°',' gamma = ' +
LightPlanePose[5]$'.4' + '°',' type  = ' + LightPlanePose[6]]
 disp_message (WindowHandle, String, 'window', 12, 12, 'black', 'true')
 disp_continue_message (WindowHandle, 'black', 'true')
return ()
```

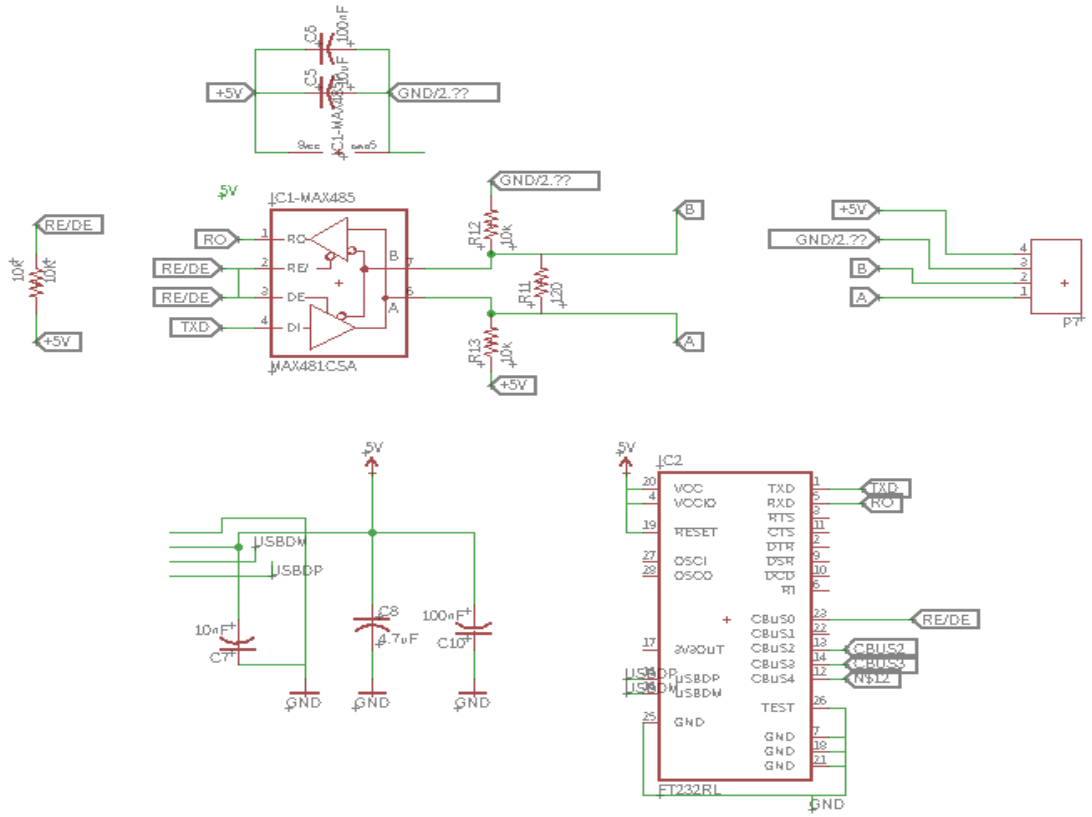
9.6 Anexo 6 – Circuito do Módulo do ARTIK 710 da UCB



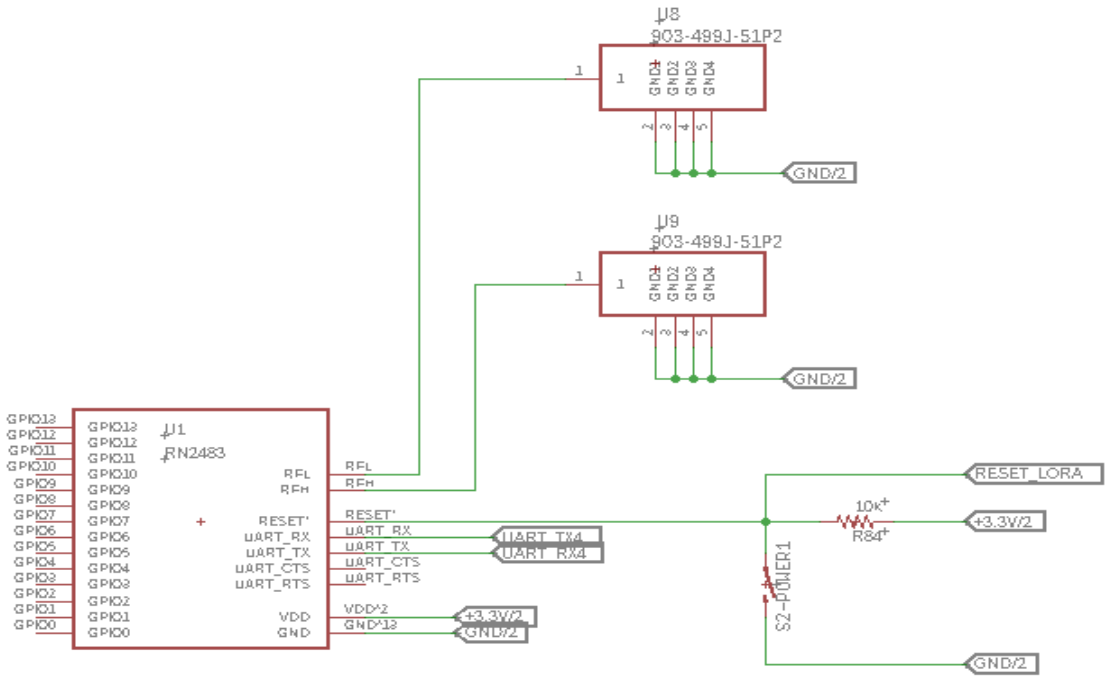
9.7 Anexo 7 – Circuito de Ethernet da UCB



9.8 Anexo 8 – Circuito de Modbus da UMA



9.9 Anexo 9 – Circuito de LoRa da UMA



9.10 Anexo 10 – Circuito do Controlador PWM do ROV