UNIVERSIDADE FEDERAL DE RORAIMA

PRÓ-REITORIA DE ENSINO E EXTENSÃO

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

PEDRO DANIEL DA SILVA GOHL

KRAKEN: DETECÇÃO DE OBJETOS PARCIALMENTE OBSERVÁVEIS EM AMBIENTE AQUÁTICO COM ALTA TURBIDEZ

Boa Vista - RR

2019

Dados Internacionais de Catalogação na publicação (CIP)
Biblioteca Central da Universidade Federal de Roraima

G614k Gohl, Pedro Daniel da Silva.

Kraken : detecção de objetos parcialmente observáveis em ambiente aquático com alta turbidez / Pedro Daniel da Silva Gohl. – Boa Vista, 2019.

53 f. : il.

Orientador: Prof. Dr. Herbert Oliveira Rocha.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Roraima, Curso de Ciência da Computação.

1 - Sistemas computacionais. 2 - Sensores. 3 - Classificação de imagens. I - Título. II - Rocha, Herbert Oliveira (orientador).

CDU - 681.31

Ficha Catalográfica elaborada pela Bibliotecária/Documentalista:
Maria de Fátima Andrade Costa - CRB-11/453-AM

PEDRO DANIEL DA SILVA GOHL

KRAKEN: DETECÇÃO DE OBJETOS PARCIALMENTE OBSERVÁVEIS EM AMBIENTE AQUÁTICO COM ALTA TURBIDEZ

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Roraima como requisito para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Dr. Herbert Oliveira Rocha

Boa Vista - RR

2019

PEDRO DANIEL DA SILVA GOHL

KRAKEN: DETECÇÃO DE OBJETOS PARCIALMENTE OBSERVÁVEIS EM AMBIENTE AQUÁTICO COM ALTA TURBIDEZ

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Roraima como requisito para a obtenção do grau de Bacharel em Ciência da Computação.
Defendido em 10 de julho de 2019 e aprovado pela seguinte banca examinadora:

Prof. Dr. Herbert Oliveira Rocha
Orientador / Curso de Ciência da Computação - UFRR

Prof. Dr. Luciano Ferreira Silva
Curso de Ciência da Computação - UFRR

Prof. Filipe Dwan Pereira
Curso de Ciência da Computação - UFRR

Dedico esse trabalho a minha mãe, Telma Maria de Jesus Silva.

AGRADECIMENTOS

Agradeço primeiramente a minha mãe, Telma Maria de Jesus Silva, que sempre fez tudo por mim e pelos meus irmãos, e também a minha família, nisso incluo meu pai, Jefferson Göhl, tio, Adalberto da Silva, meus irmãos Jefferson Göhl Jr. e João Vitor Silva Göhl.

À família da minha amiga, Eduarda de Castro Guterres, que fez parte da minha vida, sua mãe, Patrícia Macedo de Castro, que é uma pessoa incrível, seu pai, Luís Fernando dos Reis Guterres, que foi meu parceiro de conversas.

Aos meus amigos, que sempre me motivaram, mesmo quando eu não era a pessoa mais fácil de se lidar, sempre me suportaram e foram fiéis até o fim.

Aos professores que sempre me guiaram e orientaram a dar o melhor de mim nessa longa jornada da faculdade, Em especial Professor Herbert Oliveira.

Por último e não menos importante, à minha amiga e namorada Andressa Rodrigues Diógenes, que tem sido uma ótima companheira nesse momento tumultuado e complicado da minha vida e por aturar uma pessoa hiperativa sob pressão.

RESUMO

No Brasil há uma grande diversidade de redes fluviais, muitas dessas redes fluviais tendem a ter águas brancas, que possuem alta turbidez dificultando a visibilidade, locomoção e exploração dentro da água, comprometendo o desempenho de profissionais e pesquisadores. Este trabalho propôs a construção e avaliação de um sistema computacional móvel, capaz de detectar peixes em ambientes aquáticos com baixa visibilidade, calculando a estimativa da distância e as dimensões de peixes utilizando rede neural convolutiva pré-treinada. Foi observado através de experimentos, que o sistema proposto pode ser aplicado para contabiliza a quantidade de peixes em um açude, diminuindo a mão de obra e facilitando a contagem e triagem dos peixes, provendo um modo de superar as limitações da visão ou atuação humana.

Palavras-chave: Sistemas Computacionais; Sensores; Classificação de Imagens.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

No Brasil há uma grande diversidade de redes fluviais que são divididas em regiões hidrográficas. Uma dessas regiões é a Bacia Amazônica, considerada a mais extensa do mundo (PORTAL BRASIL, 2009). As águas da Bacia Amazônica são classificadas em branca, preta e clara (SIOLI, 2012). Segundo Barthem e Fabré (2004) os rios de água branca tem turbidez elevada dificultando a visibilidade dentro da água. De acordo com Santos e Santos (2005) essa coloração branca é causada pela riqueza de minerais na água, e frequentemente encontrada nos rios da região norte do Brasil.

A demanda pela identificação de objetos submersos tem crescido, dado ao grande desenvolvimento nas observações oceânicas, que resultam em grandes quantidades de dados visuais de ambientes submersos a serem processados. Reconhecimento de espécies de de peixes e resolver os problemas na detecção de objetos são as tarefas mais importantes na observação oceânica, beneficiando cientistas e biólogos marinhos, bem como aplicações comercias na área da piscicultura (QIN et al., 2016).

Em aplicações para identificação de peixes, câmeras colocadas em redes de observação oceânicas enfrentam dificuldades extremas causadas pelos ambientes naturais, tal como atenuação da luz e recifes de corais. Segundo Qin et al. (2016), com a utilização de matriz de decomposição para processar os vídeos, é possível extrair as linhas dos peixes em primeiro plano, assim eliminando o fundo da imagem, facilitando o processo de reconhecimento dos peixes.

Segundo Damme (2015) é possível fazer a captação de dados em ambientes submersos, utilizando técnicas elementares como desenhos em escala, trilateração de fita métrica, medidas de deslocamento e fotografia simples. Esses métodos são ideais para fazer reconhecimento de sítios arqueológicos submersos. Porém esses métodos não são precisos, com exceção da fotografia, e levam muito tempo para serem construídos, e são propensos a erros humanos. Essas técnicas produzem apenas representações bidimensionais e tridimensionais com baixo nível de detalhes do local estudado.

Segundo Lu et al. (2017), o som pode ser usado para mapear ambientes, emitindo um pulso que reflete no fundo do oceano criando um sonograma, que é a representação gráfica através de frequências de som. As imagens obtidas por este sonar se assemelham a imagens óticas, com níveis de detalhes bem superiores. O reflexo criado por esse sonar tem formato de leque, com a medida que o pulso se movimenta, os reflexos irão criar séries de linhas de imagem, perpendiculares ao feixe. Dependendo do ambiente estudado, o sonograma pode ser confuso, sendo necessário ter muita experiência para identificar as imagens.

Na análise feita por Damme (2015) é afirmado que há técnicas avançadas de coleta

tridimensional submersa, que são eficientes e altamente precisas. De acordo Watson et al. (2005), uma dessas técnicas é o remote *stereo-video technique*, que consiste em duas câmeras controladas remotamente no fundo do oceano. Damme (2015) também descreve outra técnica como *Computer Vision Photogrammetry* (fotogrametria de visão computacional), que permite que uma série de imagens sejam carregadas em um software dedicado para gerar uma modelo tridimensional da cena ou objeto.

Visando contribuir com o desenvolvimento de sistemas computacionais para processamento de imagens, o contexto deste trabalho está situado em projetar e desenvolver um sistema computacional móvel que seja capaz de detectar peixes submersos em águas turvas, utilizando: uma câmera ótica submersa, visão computacional e algoritmos de classificação de padrões. Assim, os dados colhidos através do sistema serão enviados para um computador ou aplicativo móvel através de uma conexão sem fio, e então serão processados e classificados. O sistema será controlado a partir de um computador e irá fazer uma estimativa da distância e dimensões dos peixes de acordo com a classificação. O sistema irá facilitar o trabalho de identificação e detecção de peixes em açudes.

1.1 Definição do problema

Com a dificuldade de realizar operações em ambiente com águas turvas, onde há pratica de piscicultura, faz-se necessário o uso de métodos para auxiliar a visibilidade dos peixes submersos, e reduzir a mão de obra necessária. Como exemplo, a remoção de peixes aptos ao consumo em um determinado açude, por meio de decisões baseadas em relatórios técnicos.

Neste sentido, o problema considerado neste trabalho é expresso na seguinte questão: **Como projetar um sistema computacional para identificar peixes, oriundos da região norte do Brasil, submersos em ambiente aquáticos com alta turbidez e parcialmente observáveis, de forma que a distância e as dimensões dos peixes sejam estimados?**

1.2 Objetivo Geral

O objetivo principal deste trabalho foi projetar e avaliar um sistema computacional móvel que seja capaz de detectar peixes (da região norte do Brasil) submersos em ambientes aquáticos com baixa visibilidade, e que possa calcular a estimativa da distância e as dimensões dos peixes baseado em processamento de imagens, de tal forma que o sistema proposto possa ser aplicado para auxiliar em atividades de piscicultura, provendo um modo de superar as limitações da visão ou atuação humana.

1.3 Objetivos Específicos

Os objetivos específicos são os seguintes:

1. Identificar métodos para a modelagem do sistema proposto em termos de software e do hardware;
2. Propor um método para identificar peixes em ambientes aquáticos com baixa visibilidade, utilizando processamento de imagem;
3. Propor um método ou técnica para classificar os peixes identificados pelo sistema, visando calcular uma estimativa de sua distância e dimensões;
4. Projetar um sistema computacional móvel capaz de capturar dados de ambientes aquáticos submersos;
5. Validar o sistema proposto, pela análise de testes práticos e simulados, a fim de examinar a sua eficácia e aplicabilidade.

1.4 Organização do trabalho

A introdução deste trabalho apresentou: o contexto, definição do problema, e objetivos deste trabalho. Os próximos capítulos estão organizados da seguinte forma:

No Capítulo 2, **Conceitos e Definições**, são apresentados os conceitos abordados neste trabalho, especificamente: Sistemas Embarcados, Modelos formais para verificação de sistemas, Visão Computacional, *Frameworks* para identificação de imagem.

No Capítulo 3, **Trabalhos Correlatos**, serão apresentados o método de pesquisa bibliográfica utilizado, sendo ele a revisão sistemática, seguido do resultado encontrado com esta pesquisa e, por fim, a contribuição dos artigos utilizados no desenvolvimento do projeto.

No Capítulo 4, **Método Proposto**, são descritas as etapas de execução do método proposto que consiste na construção de um sistema computacional de suporte a coleta de imagens em ambientes fluviais. Que irá ser apto a detectar peixes e classificá-los.

No Capítulo 5, **Avaliação Experimental**, são descritas as etapas de execução do método proposto que consiste validação do Kraken, com cenários para responder as questões de pesquisas criadas na seção.

No Capítulo 6, **Considerações Finais**, serão apresentados as considerações finais e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo apresentar os conceitos e definições abordados neste trabalho, tais como: Sistemas Embarcados, Modelos Formais para Verificação de Sistemas, Visão Computacional e Reconhecimento de Padrões.

2.1 Sistemas Embarcados

Sistemas computacionais estão por toda parte e muitos deles estão em computadores pessoais, já sistemas embarcados são um tipo específico de sistema computacional que pode ser encontrado em uma grande gama de dispositivos, desde *notebooks, tablets* a jogos eletrônicos portáteis, máquinas de fax, entre outras coisas como eletrodomésticos (VAHID; GIVARGIS, 2002).

Segundo Vahid e Givargis (2002), um sistema embarcado é qualquer sistema computacional que não seja um computador pessoal ou *workstation*. Heath (2002) define sistemas embarcados como sistemas baseados em microprocessadores ou microcontroladores (ver Seção 2.1.1), construídos para executar uma **função específica** ou grupos de funções programadas. Os sistemas embarcados não são projetados para serem reprogramados da mesma forma que computadores pessoais. Um dos exemplos citados por (HEATH, 2002) como sistema embarcado é uma máquina de lavar, que possui vários ciclos de lavagem, painéis para acompanhar os ciclos e controle dos motores, e bombas de água.

Já Schlett (1998) diz que um sistema embarcado precisa executar uma tarefa específica com o menor custo energético e monetário possível. Neste sentido, segundo Vahid e Givargis (2002), algumas características de sistemas embarcados são:

- Função única: Sistemas que são programados para um tarefa específica, que pode se repetir várias vezes.
- Fortemente limitado: Sistema com limitações definidas nas suas métricas de projeto, como de desempenho, consumo de energia, preço e dimensões.

Mesmo com a necessidade de desenvolvimento de *chips* mais rápidos e mais baratos, a necessidade de adicionar ou remover funcionalidades de um projeto de sistema embarcados também é muito importante. Encapsulando partes do sistema ao *software* é possível fazer atualizações no sistema sem a necessidade de mudar o *hardware*, isso diminui os custos de fabricação, fazendo com que vários sistemas embarcados utilizem o mesmo *hardware* (HEATH,

2002). Esses artifícios são vantagens que fazem com que muitos dos produtos utilizados pela população (como micro-ondas e alguns eletrodomésticos) se tornem mais baratos.

2.1.1 Microcontroladores x Microprocessadores

Segundo Heath (2002), os computadores modernos (como *desktops* e *laptops*) são baseados em microprocessadores, permitindo-os realizar várias funções, enquanto outros sistemas baseados em microcontroladores, são limitados a apenas um ciclo repetido da mesma tarefa. Os microprocessadores foram originalmente desenvolvidos para substituir as primeiras calculadoras, com chips que de alguma forma pudessem ser reprogramáveis, que ao invés de desenvolver outro chip só mudasse apenas o código.

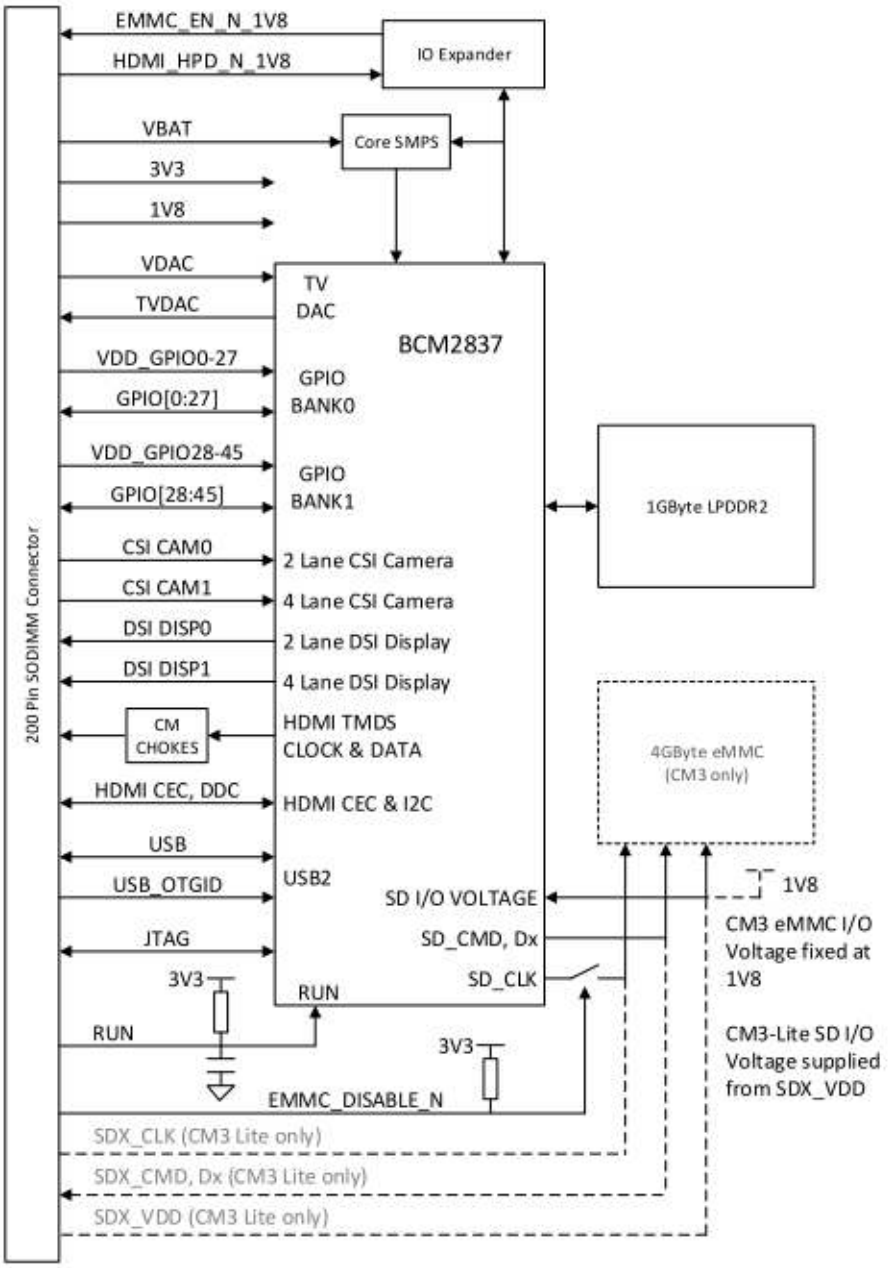
Ainda por Heath (2002), o termo design de sistemas embarcados abrange uma enorme gama de projetos de microprocessadores, não sendo exclusivamente para um microcontrolador simples, podendo ser um computador pessoal que executa certos *softwares*. Microcontroladores são compostos por vários componentes integrados a ele, como RAM, interface de entrada e saída e uma memória re-programável (WHITE, 2011).

Sistemas embarcados derivaram de processadores desenvolvidos para o mercado de computadores domésticos, com algumas diferenças no consumo energético, preço e componentes atrelados à CPU. Outras diferenças são o tempo de resposta de interrupções, a quantidade de memória e portas paralelas (SCHLETT, 1998).

Um exemplo de um sistema baseado em microcontrolador é o *Arduino Uno* (ARDUINO UNO REV3, 2018) que é uma microcontrolador que possui 14 pinos digitais, 6 portas analógicas, cristal de quartzo com frequência de 16Mhz, conexão USB e entrada de alimentação eletrica. O *Arduino* pode ser reescrito várias vezes utilizando um computador e no pior dos casos você pode substituir o microcontrolador com um preço acessível (ARDUINO UNO REV3, 2018).

Existem também os *Single Board Computers*, um bom exemplo é o *Raspberry PI*, que de acordo com Raspberry Pi (2016), possuem processador e memória integrados, os caracterizando como microcontroladores ou modulo computacional. As vantagens da utilização são por seu baixo custo, baixo consumo energético e alta rentabilidade e grande quantidade de portas para uso geral como pode ser visto no *datasheet* da Figura 1.

Figura 1 – CM3/CM3L Diagrama



Fonte:Raspberry Pi (2016)

Um bom exemplo de um sistema construído utilizando um *Single Board Computer*, é o sistema proposto por Jain et al. (2014), que utiliza Raspberry Pi para automação residencial através da internet utilizando e-mail.

2.1.2 Internet das Coisas(IoT - Internet of Things)

A Internet das Coisas (IoT - *Internet of Things*) é paradigma que está ganhando espaço no cenário moderno de telecomunicações sem fio. Sendo seu conceito básico conectar uma vasta

quantidade de coisas ou objetos (casas, carros, sinais de trânsito e outros), fazendo-os interagir entre si para alcançar um objetivo em comum (ATZORI et al., 2010).

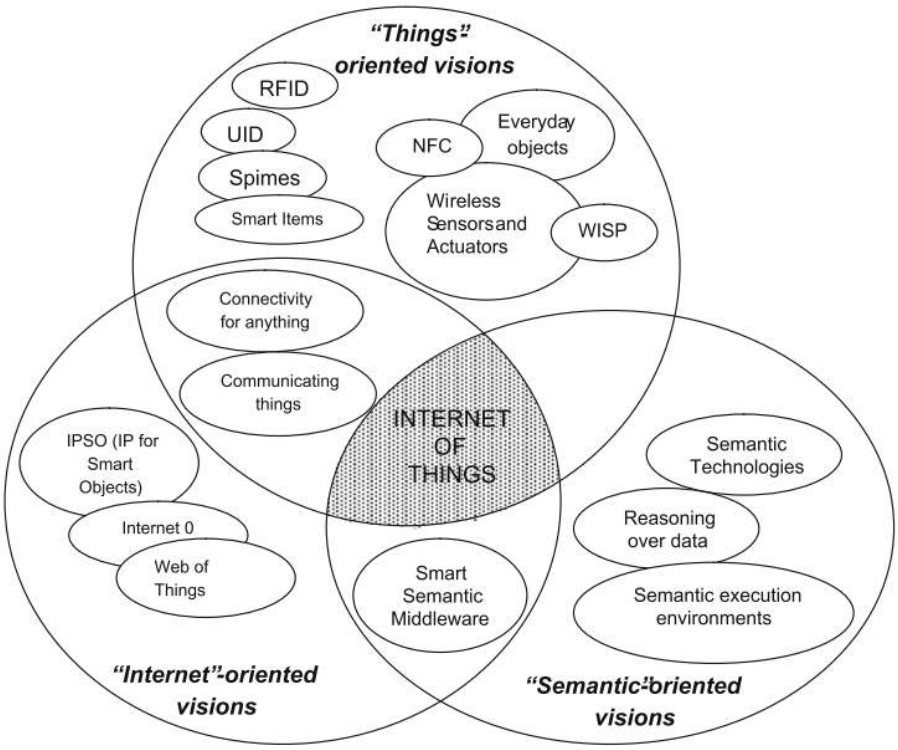
Segundo (XIA et al., 2012) a Internet das Coisas é um conceito que interligará todos os dispositivos através de sistemas embarcados, formando uma rede de dados inteligente, ubíqua e distribuída. Os dispositivos podem trazer avanços que irão melhorar a qualidade de vida, facilitando a comunicação entre pessoas e dispositivos.

Todos os dispositivos ao nosso redor estarão conectados a internet, isso resultará numa enorme quantidade de dados que terão de ser processados e apresentados sem falhas de forma eficiente. A computação em nuvem pode oferecer uma infraestrutura de ponta a ponta, entre o dispositivo e o servidor satisfazendo essa demanda de qualquer lugar (GUBBI et al., 2013).

Internet das Coisas é sobre ampliar o escopo de conexões para objetos que não são convencionais de estarem conectados à rede. Por esse motivo uma grande quantidade de soluções de comunicação foram introduzidas, visando o melhor custo energético (SIEKKINEN et al., 2012).

Na tentativa da caracterização da Internet das Coisas, A Figura 2 coloca os principais conceitos, tecnologias e padrões na visão desse paradigma, destacando e classificando as diferentes visões que resultam na Internet das Coisas, mostrando o melhor resultado de uma convergência.

Figura 2 – Paradigma "Internet das Coisas"(IoT) como resultado da convergência de diferentes visões

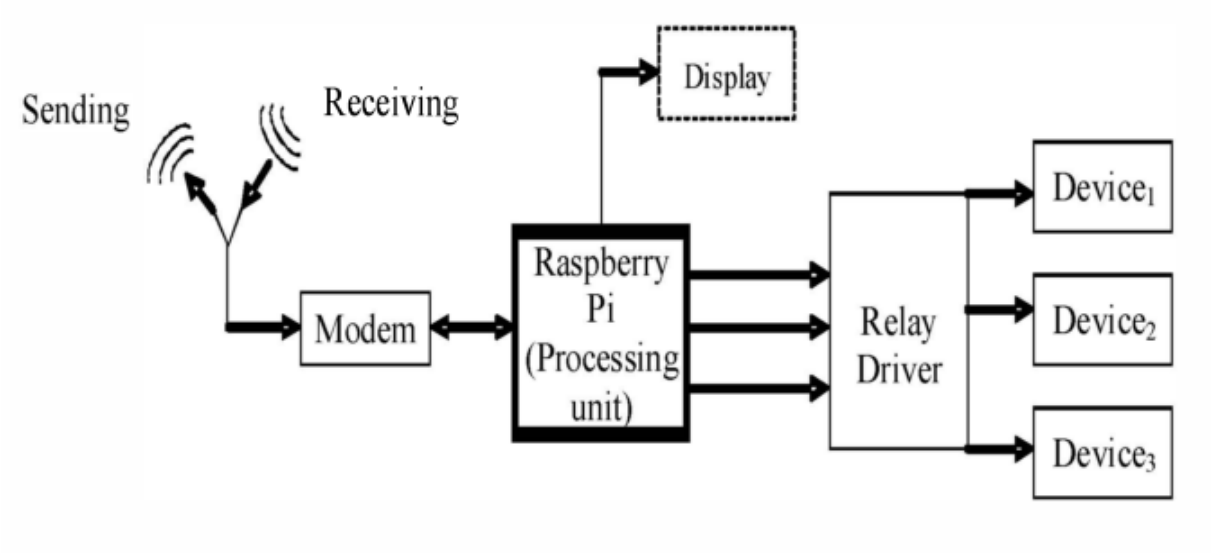


Fonte: (ATZORI et al., 2010)

O sistema proposto por Jain et al. (2014), mostra uma placa de sistema integrado, que possibilita automação residencial através da utilização de e-mails. O sistema funciona inicializando o ambiente e logando o e-mail do usuário, em seguida ele lê os assuntos dos e-mails, comparados com os comandos armazenados anteriormente em um banco de dados. Esses comandos são usados para controlar os dispositivos da casa. Segundo Jain et al. (2014), o *Raspberry Pi* foi escolhido como unidade de processamento por seus benefícios e custos econômicos.

A Figura 3 ilustra como a configuração do sistema proposto, sua conexão e controle dos relês que controlam os dispositivos interligados.

Figura 3 – Visão do sistema proposto por Jain et al. (2014)



Fonte: (JAIN et al., 2014)

2.1.3 Baixo consumo de energia

Os dispositivos que compõem a Internet das Coisas são caracterizadas por utilizar poucos recursos, em termos de computação e capacidade energética. Isso requer certa atenção, já que novos projetos terão que levar em consideração à eficiência dos recursos e problemas de escalabilidade (ATZORI et al., 2010).

Com o avanço da Internet das Coisas, uma grande demanda de dados surgiu e por sua vez a alocação e gerenciamento de dados se tornaram um problema crítico. Atualmente a internet utiliza 5% da energia gerada com a solução desses tipos de problemas, e tende a crescer cada vez mais (GUBBI et al., 2013).

Um sistema embarcado é projetado para executar tarefas específicas, limitando alguns recursos como memória RAM, armazenamento, periféricos e ciclos de processamento e velocidade de processamento. Um exemplo prático é adicionar mais linhas de códigos, usando

mais armazenamento, melhorando o tempo de execução do código, outro exemplo é reduzir a velocidade de processamento para reduzir o consumo energético (WHITE, 2011).

De acordo Marwedel (2010), que descreve uma série de requisitos e abordagens para especificar um dado sistemas embarcados, essas especificações geram modelos do sistema em desenvolvimento. Esses modelos são descritos em linguagens, essas linguagens devem ser capaz de representar alguns características, bem como:

- **Sincronização e Comunicação**: onde os componentes devem se comunicar e estarem sincronizados. sem comunicação não há cooperação entre eles.
- **Comportamento Temporal**: levando em consideração há existência de muitos sistemas embarcados em tempo real. Sendo uma das características mais importantes de alguns sistemas embarcados. Na ciência da computação o tempo é dado de maneira abstrata, a notação O é um desses exemplos, que reflete o grau de crescimento de uma função, usado frequentemente para medir o tempo de execução de algoritmos, mas falha ao descrever a execução real de tempo. Para sistemas em tempo real, é necessário quantificar o tempo. Para isso algumas técnicas podem ser utilizadas, como: **medir o tempo utilizado**, meios de **atrasar um processo** por determinado período de tempo, **especificar *timeouts***.
- **Propriedades não-funcionais**: Sistemas embarcados em desenvolvimento devem possuir um número de propriedades não-funcionais, como tolerância a falha, tamanho, extensibilidade, perspectiva de vida, consumo energético, peso e outros.

Marwedel (2010) ainda afirma que melhorar as tecnologias de baterias, nos ajudará a conseguir uma melhor duração para baterias, mas a limitação térmica nos impede de utilizar essa energia por muito tempo, devido a problemas de superaquecimento do dispositivo. Devido a esses problemas de temperatura, dispositivos estão sendo projetado com sensores para detectar quando há superaquecimento, regulando o dispositivo para o evitar que isso aconteça.

2.2 Modelos formais para verificação de sistemas

De acordo com Baier e Katoen (2008) sistemas de Tecnologia da Informação e Comunicação (TIC) estão crescendo rapidamente, e seu funcionamento de forma eficaz é fundamental. Esses sistemas estão se tornando cada vez mais complexos e estão ganhando espaço no cotidiano através da internet e de todos os tipos de sistemas embarcados. Sistemas TIC estão por toda parte, eles controlam a maioria dos dispositivos de uso cotidiano (como por exemplo um sistema para controle da casa, como o sistema proposto por Jain et al. (2014)).

A dependência do uso de sistemas embarcados tornam sua eficácia fundamental para sociedade. Esses sistemas oferecem um bom desempenho em tempo de resposta e capacidade de processamento. O mal funcionamento desses sistemas podem ameaçar nossas vidas, e podem ter

consequências financeiras substanciais para o fabricante. Sistemas TIC eficazes são fundamentais para a sobrevivência de uma empresa (BAIER; KATOEN, 2008).

Um exemplo clássico de uma falha catastrófica, é a falha do lançamento do Ariane 5. Em 4 de Junho de 1996, o lançamento do foguete Ariane 5 terminou com uma falha catastrófica, com apenas 40 segundos após o início da sequência de lançamento. As condições de lançamentos eram ideais, nenhum problema climático ou pertubações no campo magnético. O problema no lançamento foi o **sistema de controle de voo**, mais precisamente no **sistemas de referência inercial**, causando uma série de reações em cadeia até a auto-destruição do foguete (LIONS et al., 1996).

Após testes posteriores foi comprovado que o causa técnica foi um erro de operando no momento de conversão do variável do viés horizontal, fazendo o sistema de referência inercial sobrecarregar e parar. foi comprovado que nem todas as operações estavam protegidas após uma análise em todos os códigos que demonstravam risco (LIONS et al., 1996).

O crescimento na complexidade dos sistemas TIC mostram que eles não estão mais isolados, agora podem fazem parte de um grande sistema, conectando e interagindo com vários outros outros componentes, tornando esses sistemas vulneráveis a erros. A concorrência e o não determinismo que são fundamentais para a modelagem dos sistemas integrados, tornando a utilização de técnicas padrões muito difícil (BAIER; KATOEN, 2008).

Técnicas de verificação são aplicadas a sistemas TIC de forma mais confiável. A verificação de sistemas trabalha pra estabelecer que o produto possua certas propriedades, como certificar que o sistema não entre em uma situação de *deadlock*. A falha é encontrada quando o sistema não cumpre todas as propriedades especificadas (BAIER; KATOEN, 2008).

2.2.1 UML

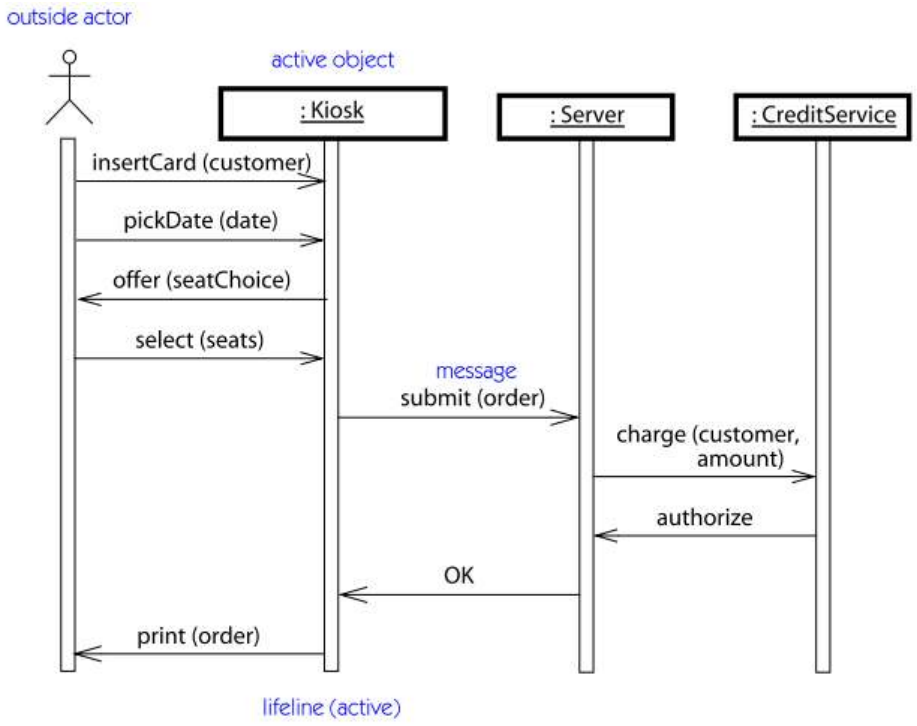
Segundo Rumbaugh et al. (2017) UML (*Unified Modeling Language)* é uma linguagem de modelagem visual que tem como intuito de utilizar todos os métodos de desenvolvimento de software, extraindo melhor para especificar, visualizar, desenvolver e documentar os artefatos de um sistema em nível de software, capturando estáticas informais e dinâmicas do comportamento de um sistema. Alguns diagramas da UML de acordo com Rumbaugh et al. (2017) são:

Diagrama de Classe: Os principais componentes do diagrama de classe são classes e suas relações, associação e generalização, mostrando algumas dependências e utilização de cada elemento.

Diagrama de Caso de Uso: Os principais componentes do diagrama de Caso de Uso são os atores e as funcionalidades. O diagrama modela o sistema, expressando a relação dos atores com o sistema em si.

Diagrama de Sequência: O diagrama de sequência, que será utilizado nesse trabalho para validar algumas ações. Como ilustra a Figura 4, o diagrama de sequência possui um grupo de mensagens que estão organizados em uma sequência de tempo, toda classe é definida como uma linha do tempo, no exemplo da Figura 4, os objetos ativos são linha temporais exibidas pelas linhas verticais, mostrando os processos que são relacionados com elas durante seu tempo de vida. As mensagem são representadas por setas entre as linhas de tempo.

Figura 4 – Diagrama de sequência



Fonte: (RUMBAUGH et al., 2017)

Diagrama de Colaboração: O diagramas de colaboração modelam os objetos e links que são significativos dentro de uma interação. O papel das classes descreve o que acontece com um objeto e o papel do associador descreve a relação com a colaboração dos objetos.

Diagrama de Estados: A máquina de estados modela as possibilidades de um objeto de uma classe. os principais componentes são os estados e as transições. Cada estado modela uma linha temporal dado determinada condição, condição essa que pode levar o objeto a um novo estado.

Diagrama de componentes: Os diagramas de componentes modelam os componentes a serem utilizados na implementação de um sistema, definindo dependências entre os componentes, a nível *software*, podendo avaliar o impacto de uma possível mudança nos componentes.

A utilização de UML nesse trabalho se dará pela utilização de diagramas de sequência que mostrarão como o fluxo do software irã ocorrer. De acordo com Rumbaugh et al. (2017), diagramas de sequência mostram a iteração em uma forma bidimensional, onde o eixo vertical é o tempo de vida do processo e o eixo horizontal mostra as regras de transição como pode ser visto na Figura 4.

Já Cunha et al. (2011) diz que nos diagramas de sequência as ações são organizadas por tempo, mas não exploram as relação dos objetos. Os diagramas de sequência mostram todo o processo de trocas de mensagens dos objetos, mapeando todo o ambiente mostrando como os objetos colaboram entre si para obter sucesso. Um exemplo é a Figura 4, que mostra um diagrama de sequencia, com ator, processos e ações que são executas em certa ordem.

2.3 Visão Computacional

A visão humana não tem dificuldades em identificar pequenas variações na translucidez e sombreamento em imagens, distinguindo os objetos da imagem que é fundo. Temos certa facilidade em identificar estruturas tridimensionais, podemos diferenciar formatos, texturas, cores e pessoas em imagens, e até mesmo dizer que sentimentos eles podem estar sentindo, apenas olhando fotos (SZELISKI, 2010). A visão computacional é uma forma de emular a visão humana, possuindo imagens como entrada, e sua saída é a interpretação dessa imagem (MARENGONI; STRINGHINI, 2009).

Bradski e Kaehler (2008) definem uma parte da vasta área da visão computacional como uma transformação de dados de uma imagem em uma nova representação para atingir um objetivo específico, como deixar a imagem em escala de cinza. Essas transformações precisam de contexto, como especificar distância, referências de onde a imagem foi tirada, ou o que pode conter na imagem.

De acordo com Marengoni e Stringhini (2009), a necessidade de uma etapa de pré-processamento é importante quando queremos extrair informações. Essa etapa de pré-processamento envolve o processamento de imagens, que muitas vezes precisam ser convertidas para certo formato ou tamanho com a necessidade de serem filtradas para remoção de ruídos provenientes dos processos de aquisição de imagem, como por exemplo o tipo de sensor utilizando na hora da captura ou até mesmo a iluminação do ambiente assim como condições climáticas.

Os filtros são ferramentas básicas que podem remover ruídos de imagens, filtros podem ser espaciais, atuando diretamente na imagem, ou filtros de frequência, necessitando de uma transformação no seu domínio, geralmente essas transformações de domínio são feitas utilizando das transformadas de Fourier, para assim então serem aplicados os filtros e em seguida transformadas novamente para o domínio espacial (MARENGONI; STRINGHINI, 2009). Algum desses filtros são demonstrados na Figura 5a e 5b.

De acordo com Gonzalez et al. (1992), existem três níveis de processamento de imagens, baixo nível, nível médio, e alto nível. O processamento em baixo nível envolve operações primitivas como processamento para redução de ruídos, aprimoramentos de contraste e aprimoramentos de nitidez. Esse nível é caracterizado por receber como entrada uma imagem e retornar uma imagem como saída. Processamento em nível médio em imagem envolvem tarefas como segmentação que particionam a imagem em regiões, classificando objetos. Esse nível é caracterizado por receber imagens como entrada e retornar atributos extraídos das imagens de entrada, como contornos. O processamento de imagem em alto nível envolve a compreensão do computador na hora do reconhecimento de objetos, realizando funções cognitivas normalmente associadas a visão.

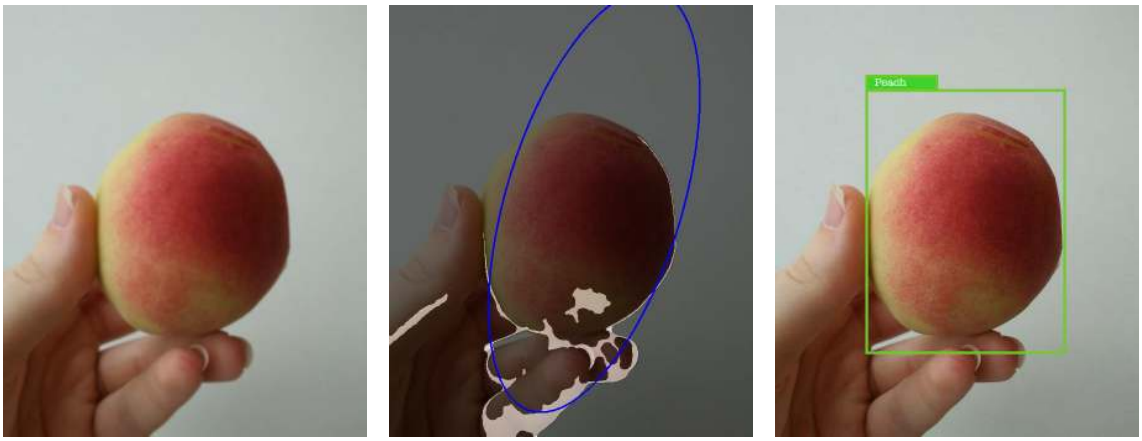
Alguns exemplos de níveis:

Figura 5 – Exemplificação dos níveis descritos por Gonzalez et al. (1992).

(a) Aplicação de filtro Gaussiano para remoção de ruídos

(b) Segmentação com base nas cores da imagem com sobreposição para identificação de pêssegos.

(c) Identificação de objetos utilizando aprendizado de máquina.



Fonte Própria

- Baixo-nível: Pode ser associado a operações como redução de ruídos ou níveis de contraste da imagem (MARENGONI; STRINGHINI, 2009). Um exemplo pode ser visto na Figura 5a, onde foi aplicado um filtro gaussiano para remoção de ruídos encontrados na imagem.
- Nível-médio: São operações associadas a operações de segmentação de imagem ou reconhecimento de objetos na imagem (MARENGONI; STRINGHINI, 2009). Um exemplo é ilustrado na Figura 5b, onde foi feita a aplicação de um algoritmo de segmentação com base nas cores da imagem.
- Alto-nível: São relacionados a tarefas de cognição associadas com a visão humana (MARENGONI; STRINGHINI, 2009). Um exemplo pode ser visto na Figura 5c, onde um

framework identificou a fruta com base em um modelo de dados, classificando e exibindo o resultado.

2.3.1 Representação de Imagens Digitais

Uma imagem pode ser definida como uma função bidimensional, $f(x, y)$, onde x e y são coordenadas espaciais, e a amplitude de f em qualquer par de coordenadas (x, y) é chamada de intensidade da imagem, naquele determinado ponto. Quando x, y e os valores de amplitude de f são finitos, chamamos a imagem de imagem digital. Uma imagem digital é composta por um número finitos de elementos, cada um tendo seu próprio ponto e posição específicos. Esses elementos se referem à elementos de imagem, *pels* e *pixels* (GONZALEZ et al., 1992).

O termo processamento de imagens digitais normalmente se referem ao processamento de imagens bidimensionais por computadores. Uma imagem digital é um vetor de números reais ou números complexos representados por uma quantidade finita de *bits*. Na representação de imagem devemos nos preocupar com a caracterização e quantidade de elementos que representam a imagem (*pixels* ou *pels*). Em geral qualquer função bidimensional que contém alguma informação pode ser considerada uma imagem. O principal requisito para o processamento de imagens digitais é que ela seja amostrada e logo em seguida quantificada. A taxa de amostragem, que são os *pixels* por unidade de área, tem de ser grande o suficiente para preservar uma quantidade de informações úteis na imagem. A quantização da imagem é a conversão analógica para digital de uma imagem amostrada para um número finito de níveis de cinza (JAIN, 1989).

Na Figura 6 é possível ver a convenção de coordenadas utilizada para representar imagens digitais segundo Gonzalez et al. (1992). O resultado da amostragem e quantização é uma matriz de números reais. Assumindo que a imagem $f(x, y)$ é amostrada, resultando uma imagem digital que possui M linhas e N colunas.

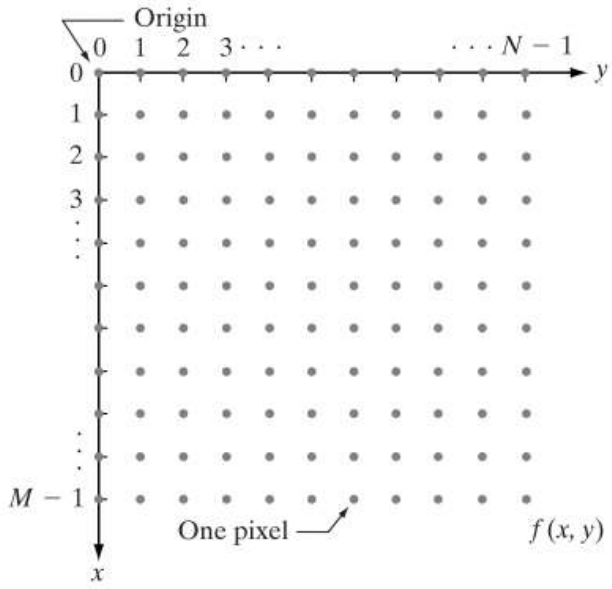
2.4 *Frameworks* para identificação de imagem

Nesta sessão iremos abordar *frameworks* para identificação e classificação de imagens, onde são usando, as vantagens e desvantagens em relação a métodos convencionais e tecnologias que utilizam *frameworks* para suas atividades.

2.4.1 *Tensorflow*: Um *Framework* de Código Aberto para Aprendizagem de Máquina

Tensorflow é um *framework* para operações computacionais, tendo uma grande biblioteca permitindo o desenvolvimento para diferentes plataformas, de *clusters* de servidores até dispositivos móveis. Tensorflow tem grande facilidade de lidar com problemas de aprendizado de máquina e *deep learning*, podendo ser usado em diversos domínios científicos (Martin˜Abadi et

Figura 6 – Amostragem de imagem e quantização



Fonte: (GONZALEZ et al., 1992)

al., 2015). A utilização do *framework* será de extrema importância nas classificações de imagens que serão processadas.

O campo de aprendizagem de máquina teve grandes progressos nos últimos anos, conseguindo alcançar níveis de abstração de imagem que alcançam a percepção humana. Um tipo de modelo que tem conseguido bons resultados, alcançando e até mesmo ultrapassando a capacidade humana em tarefas de reconhecimento de imagem são as **redes neurais convolutivas** (*Convonlutional neural network*) (IMAGE RECOGNITION, 2018). Redes convolutivas estão no centro da maioria das soluções para problemas de reconhecimento de imagem (SZEGEDY et al., 2015). A Figura 7 ilustra um exemplo gráfico de uma rede neural profunda criada utilizando *Tensorflow* [1], demonstrando três predições.

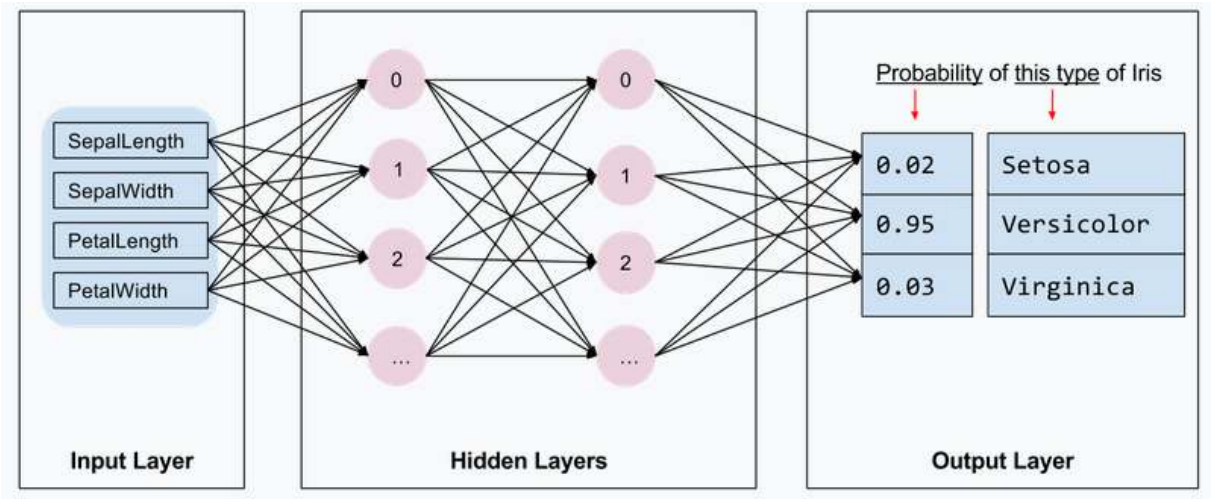
O modelo que será utilizado no desenvolvimento desse trabalho será utilizado o *Inception*, que segundo Szegedy et al. (2015) tem custo computacional bem menor que outros modelos, tornando a sua utilização viável para cenários com recursos limitados como em processamento móvel.

2.4.2 OpenCV: Uma biblioteca de visão computacional

OpenCV é uma biblioteca que permite a utilização de funções para manipulação de dados. A biblioteca contém mais de 2500 algoritmos otimizados para diversas funções, como

[1] <http://tensorflow.org>

Figura 7 – Rede Neural com atributos, camadas ocultas e predições.



Fonte: TensorFlow (2018)

rastreamento de objetos e rastreamento de movimentos de sensores ópticos (OPENCV, 2019).

No desenvolvimento desse trabalho será utilizado algumas funções de transformações, que servirão para manipulação dos dados obtidos pelos sensores de captura óptica.

3 TRABALHOS CORRELATOS

Neste capítulo serão apresentados e discutidos os principais trabalhos existentes na literatura que se relacionam com os tópicos que formam a base para o desenvolvimento deste trabalho, com ênfase em: Pré-processamento de Imagem, Segmentação e Detecção de Objetos, Pós-processamento e Reconhecimento de Padrões.

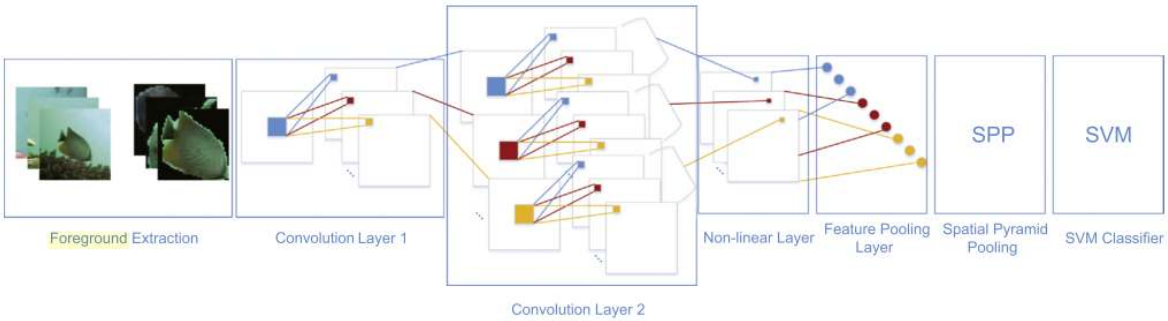
3.1 DeepFish: Accurate underwater live fish recognition with deep architecture

Um *framework* baseado em uma rede neural convolutiva foi proposto por Qin et al. (2016) para reconhecimento de peixes em vídeos gravados submersos em águas oceânicas. No trabalho as imagens de fundo são removidas utilizando métodos baseados em decomposição de matrizes esparsas e de baixo nível, removendo assim apenas o fundo da imagem, deixando somente o peixe.

Os dados extraídos são passados para camadas de convolução depois para uma camada não linear, em seguida é feita a classificação utilizando SVM (Support Vector Machine)(CORTES; VAPNIK, 1995). A taxa de aprendizado diminui a medida que a quantidade de parâmetros são adicionados, necessitando assim de uma otimização no filtro de aprendizado.

O método do trabalho de Qin et al. (2016), pode ser visto na Figura 8, onde os passos do *framework* são mostrados. Onde os filtros são escolhidos, as imagens processadas pela rede neural profunda e processos para extração de atributos, após essas etapas a saída final alimenta uma máquina de vetor de suporte linear.

Figura 8 – *Pipeline* do *framework* proposto.



Fonte: (QIN et al., 2016)

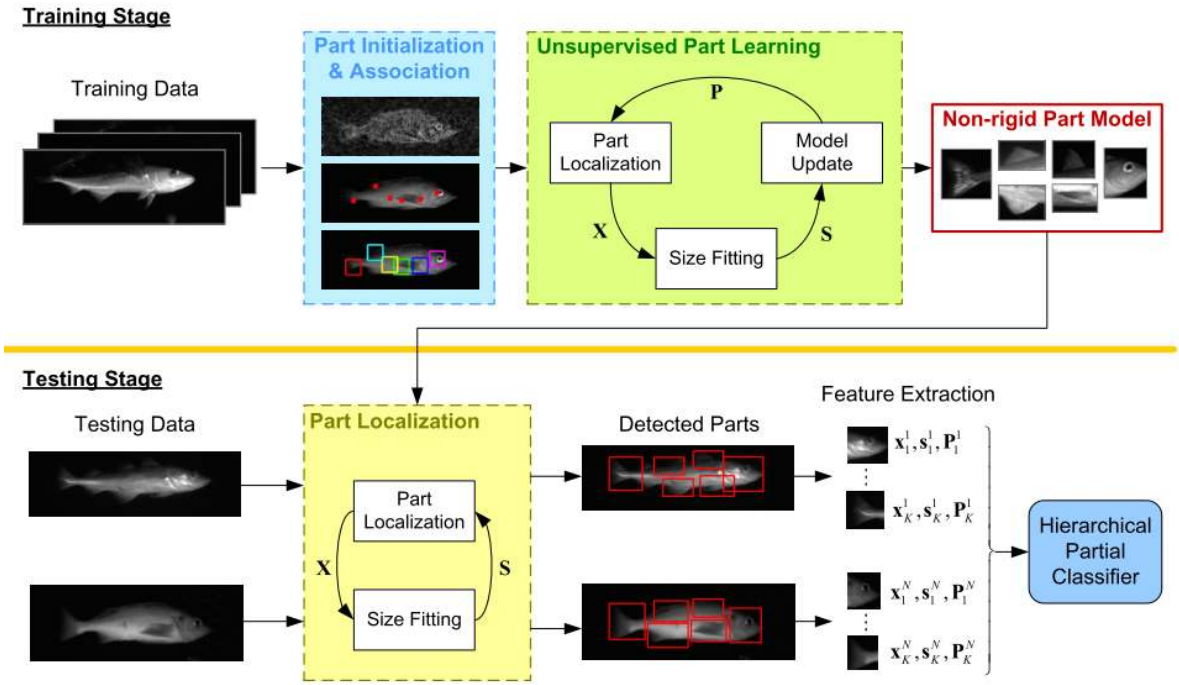
Similar ao trabalho de Qin et al. (2016), o método apresentado nesse trabalho de conclusão de curso propõe um método para detecção de imagens, onde a classificação será feita utilizando métodos baseados em *Convolutional Neural Network* assim como no trabalho de Qin et al. (2016).

3.2 A Feature Learning and Object Recognition Framework for Underwater Fish Images

Chuang et al. (2016) propõe um *framework* para reconhecimento de peixes em baixo d'água, que consiste em uma técnica de aprendizagem totalmente não supervisionada e um classificador resistente a erros, conforme Figura 9. Os dados são iniciados com base em algumas características (como formato do contorno do peixe), depois passados por alguns critérios de classificação com base nessas características.

A Figura 9 ilustra o classificador, que tem uma abordagem não supervisionada que gera uma classe de hierarquia binária, onde cada nó é um classificador. Os experimentos mostram que o *framework* proposto é preciso tanto em situações onde a base de dados é pública quanto em ambientes controlados com alta incerteza.

Figura 9 – Proposta de treinamento não supervisionada.



Fonte: (CHUANG et al., 2016)

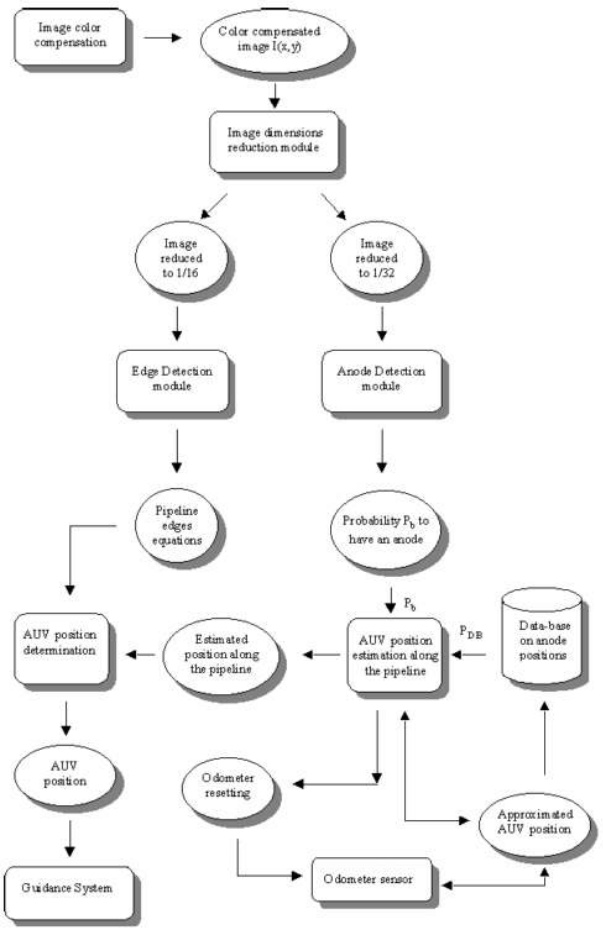
A utilização do trabalho de Chuang et al. (2016) como referência irá auxilar em uma

abordagem de aprendizado resistente a erros, que irá auxiliar na confiabilidade de informação, uma vez que o sistema proposto neste trabalho irá auxilar mergulhadores em lugares de riscos, em águas com alta turbidez.

3.3 A Vision Based System for Object Detection in Underwater Images.

O trabalho de FORESTI e GENTILI (2000) propõe um sistema (veículo autônomo) para detecção e rastreio de objetos submersos em água, conforme fluxograma na Figura 10, o sistema faz detecção automática de canos (que podem se estender por quilômetros) e posicionamento autônomo baseado nas detecções.

Figura 10 – Proposta do sistema.



Fonte: (FORESTI; GENTILI, 2000)

No método proposto por FORESTI e GENTILI (2000) é aplicado uma técnica para compensação de cor (alterando valores nos canais de cores) devido as pertubações luminosas, e

dimensionamento na imagem. Depois do dimensionamento essas imagens ganham tamanhos de $1/16$ para detecção de bordas (para encontrar canos) e $1/32$ para encontrar anodos em canos.

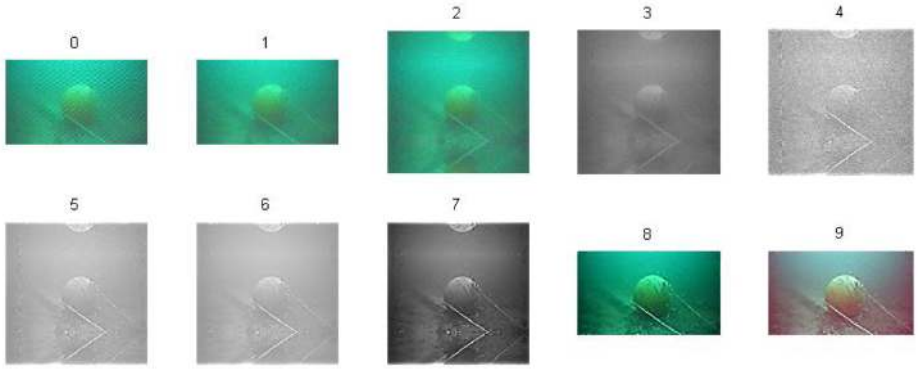
A identificação dos objetos é feita utilizando redes neurais, que classificando a imagem em tempo real, verificando se há obstáculos. Essas informações auxiliam na navegação automática do veículo, traçando rotas e caminhos através das imagens processadas e ressonância geométrica.

O sistema proposto por Gentili e S. (2000) consegue detectar canos e outras estruturas, mesmo com os problemas causados pela falta de luminosidade, dispersão e atenuação da luz e problemas como areia e detritos sobre canos.

3.4 Automatic underwater image pre-processing

Bazeille et al. (2006) propõe um algoritmo para processamento e restauração de imagens de ambiente submersos. Imagens de ambiente submersos sofrem pela perda de distância na captura, distorção de luz, baixo contraste, baixa saturação de cor e outros problemas. Os métodos de processamento de imagens atuais focam em distorção e atenuação causados pela luz e precisam de conhecimento do ambiente que está sendo estudado. O algoritmo proposto pelo trabalho de Bazeille et al. (2006) é um algoritmo que automatiza o pré-processamento de imagens submersas. O algoritmo proposto no trabalho de Bazeille et al. (2006) reduz as pertubações nas imagens submersas, e melhorando a qualidade da imagem.

Figura 11 – Aplicação do algoritmo proposto por (BAZEILLE et al., 2006) passo-a-passo.



Fonte: (BAZEILLE et al., 2006)

A Figura 11, mostra o resultado da aplicação de diversas técnicas (como filtro gaussiano e anisitrópico) para processamento de imagem e remoção de pertubações, como atenuações de luz. O algoritmo proposto por (BAZEILLE et al., 2006) utiliza algumas técnicas que serão

importantes para o pré-processamento das imagens do sistema computacional proposto nesse trabalho de conclusão de curso, algumas dessas técnicas são:

Remoção do padrão *Moiré* : O padrão Moiré é removido através de uma análise espacial detectando picos nas nas transformadas de *Fourier* e deletando-as assumindo que eles representam padrões Moiré.

Redimensionamento de imagem : Essa transformação padroniza a imagem facilitando a aplicação de algoritmos como as tranformadas de *Fourier*.

Filtro homomórfico : Utilizado para correção de iluminação e aprimorar o contraste da imagem.

4 MÉTODO PROPOSTO

Este capítulo apresenta as etapas do método proposto para o funcionamento do sistema computacional do Kraken e será explicado como os objetivos deste trabalho serão satisfeitos.

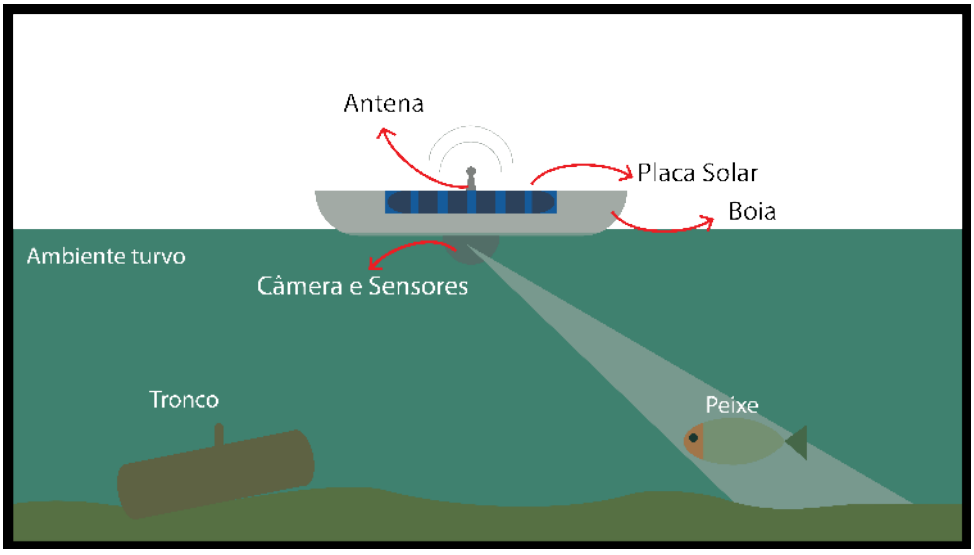
4.1 Visão geral do método proposto

Esse trabalho apresenta um sistema computacional para classificação, contagem e estimativa de tamanho de peixes dentro de ambientes aquáticos com alta turbidez, visando gerar relatórios para facilitar a leitura dos resultados para o usuário final.

O sistema computacional proposto consiste em vários módulos interligados que se comunicam-se por rede sem fio, e foi estruturado em forma de boia aquática, composta de sensores para captura ótica, podendo ser um sistema embarcado como pode ser visto na Figura 12. Um exemplo de sistema embarcado que poderia ser utilizado para adquirir imagens subaquáticas seria um *smartphone*, acoplado a boia, enviando os dados para a central processá-las gerando um relatório, contendo contagem e tamanho dos objetos classificados para o usuário final.

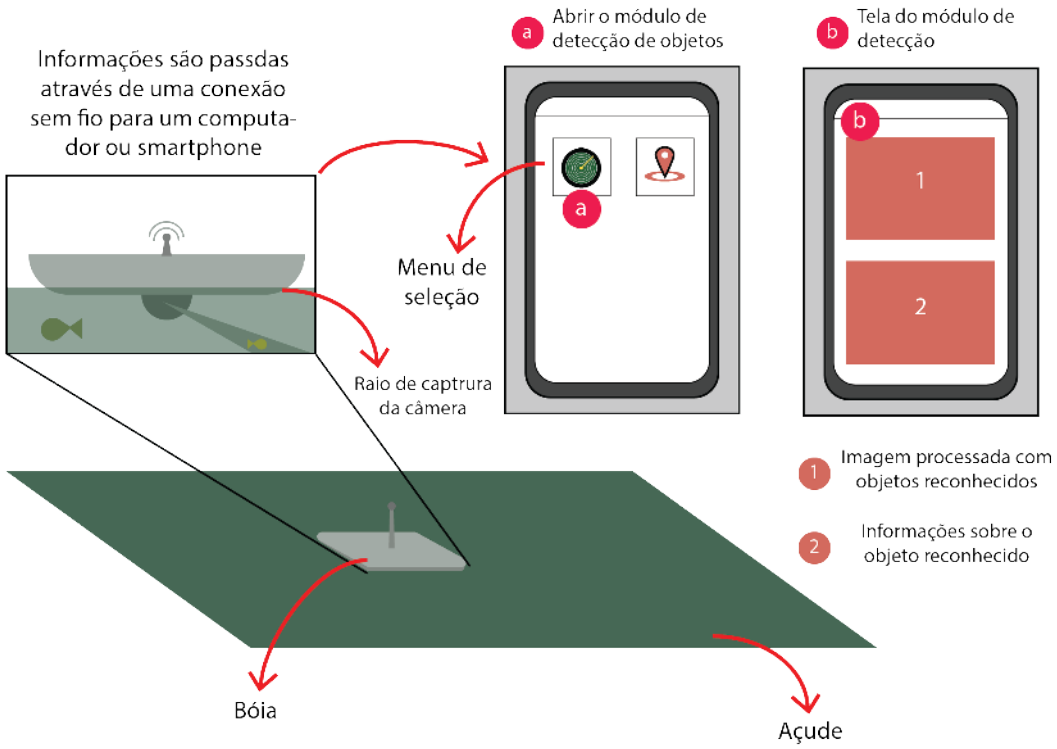
Na Figura 12, pode-se ter a ideia do funcionamento do Kraken em situação real, coletando imagens sub-aquática de peixes enquanto conectado a central. A placa solar de auxílio é opcional e serve para carregar ou alimentar o sistema embarcado acoplado a boia. A interação com o Kraken é feita através de uma conexão sem fio com a central. A central pode ser acessada através de um *Web app*, podendo utilizar um computador ou celular, como é presentado na Figura 13.

Figura 12 – Visão Geral



Fonte Própria

Figura 13 – *Storyboard*



Fonte Própria

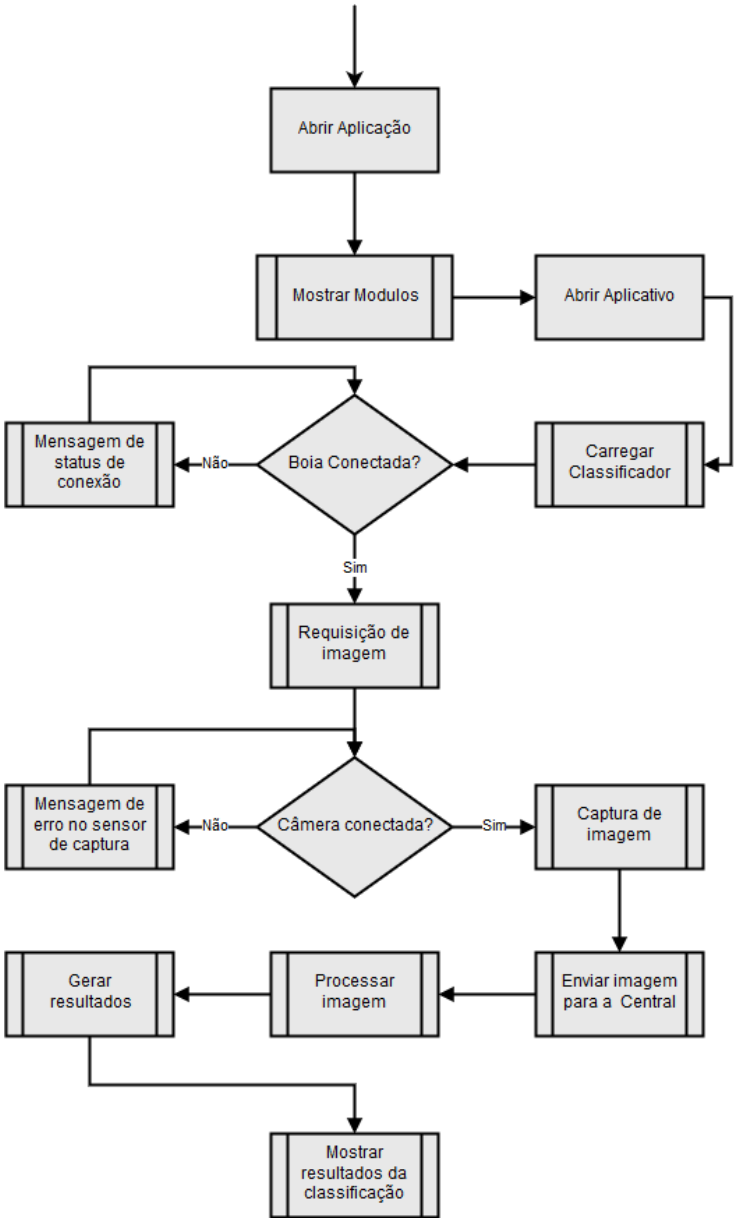
4.2 Sistema computacional de coleta e processamento de imagens fluvais em ambiente sub-aquático com baixa visibilidade: Kraken

A Boia proposta contará com microcontroladores que serão responsáveis por gerenciar os sistemas integrados à mesma, como motores e/ou estabilizadores e sensores para detecção de peixes e captura de imagens dentro do ambiente fluvial com alta turbidez, classificando peixes e especulando seu tamanho. A Boia servirá para facilitar a atividade da piscicultura, normalmente é necessário uma mão de obra excessiva para realizar a triagem dos peixes.

Os objetos classificados terão seus tamanhos especulados. A imagem será pré-processada antes de ser enviada ao classificador, isso irá garantir que a imagem esteja em perfeitas condições de classificação. Além do aplicativo de auxilio, o sistema conta com um módulo de classificação de imagens, em especifico neste trabalho iremos adotar o *framework* Tensorflow, por ter apresentado melhor resultados em testes prelimanares em relação a outros *frameworks* com o

Yolo[1].

Figura 14 – Fluxo de funcionamento do sistema proposto.



Fonte Própria

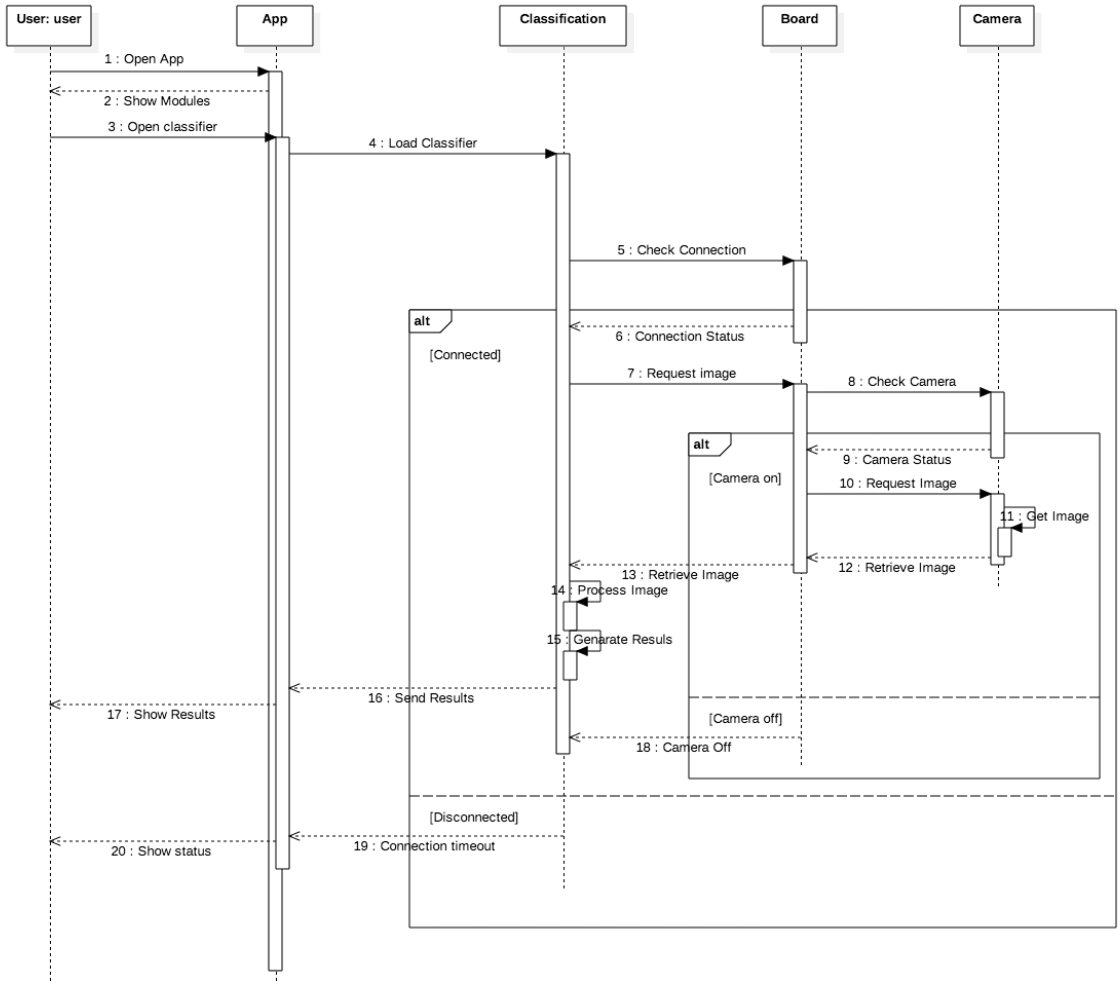
Como demonstra a Figura 14, o fluxo do sistema com a inicialização da aplicação, onde os módulos serão exibidos, dando opção para usuário abrir a aplicação para classificar e gerar o relatório da imagem. Em seguida a verificação se a Boia está conectada a Central e caso não esteja uma mensagem é retornada informando para o usuário que a conexão falhou, caso contrário, uma requisição de imagem é solicitada à Boia, onde mais uma vez será verificada a conexão com o sensor óptico, enviando mensagem de falha caso esteja desconectado. Uma vez

[1] https://pjreddie.com/darknet/yolo/

que o a imagem é enviada para a Central, o módulo de classificação irá pré-processar a imagem caso seja necessário, e quando a imagem estiver adequada será processada e um relatório será gerado e exibido para o usuário.

O diagrama de sequência representeado pela Figura 15 mostra a sequência de interações do sistema, tais como: a interação do usuário com a Central; e da Central com a Boia. As condições caso haja falha nas conexões com os sensores e como proceder caso essas falhas sejam encontradas.

Figura 15 – Detalhando o diagrama de sequência



Fonte Própria

- Usuário: o usuário receberá os *feedbacks* através do aplicativo;
- Aplicativo: o aplicativo será responsável de entregar as mensagens para o usuário, e manusear a Central (conexão e sensores);

- Módulo de Classificação: Consiste em um módulo localizado na aplicação que será responsável por analisar, processar e classificar a imagem;
- Boia (Board / Câmera): A Boia que estará conectada à aplicação, onde estarão contidos os sensores e será responsável por adquirir imagens e as enviá-las.

4.3 Modelo de classificação

A parte central do Kraken é o modelo de classificação de objetos, gerado a partir do treinamento de uma rede neural Inception V2[2]. Ao decorrer da seção serão explicados com o *dataset* foi criado, como o *software* foi aplicado para geração do modelo e integração com o sistema proposto.

Para o treinamento do modelo de classificação foi necessário muitos recursos de *hardware*, foram preparadas máquinas equipadas com processadores gráficos que aceleraram o a velocidade do treino, uma vez que processadores gráficos conseguem facilmente lidar com processos matemáticos. Levando isso em consideração as configurações das máquinas pode ser vista na Tabela 1 e 2:

Tabela 1 – Especificações máquina 1.

Processador	i3 - 2100 3.1Ghz
Memória	8 Gigabytes DDR3 1333Mhz
Processador gráfico	GTX 760 2 Gigabytes GDDR5 1152 núcleos CUDA 980Mhz
Sistema operacional	Ubuntu 18.04
Versão do framework	Tensorflow-Gpu 1.6, Cuda toolkit v7

Fonte Própria

Tabela 2 – Especificações máquina 2.

Processador	i7 - 2600K 3.4Ghz
Memória	8 Gigabytes DDR3 1333Mhz
Processador gráfico	RTX 2060 6 Gigabytes GDDR6 1920 núcleos CUDA 1365Mhz
Sistema operacional	Manjaro 18.0.4
Versão do framework	Tensorflow-Gpu 1.9, Cuda toolkit v7

Fonte Própria

4.3.1 Treinamento do modelo

Como parte essencial do Kraken, o modelo foi utilizado como a principal parte da classificação, nessa seção será abordado como foram executado as etapas de desenvolvimento do modelo, descrevendo os passos e recursos utilizados para geração do mesmo.

[2] <https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_v2.py>

Para geração do modelo foi utilizado um *dataset* criado com imagens de peixes de água doce da fauna amazônica, totalizando 319 imagens, obtidas em *websites* variados, a partir da busca de peixes da região amazônica. O treino foi feito usando como base uma rede R-CNN, utilizando Inception v2 dividido em duas etapas, separadas em 3 treinos.

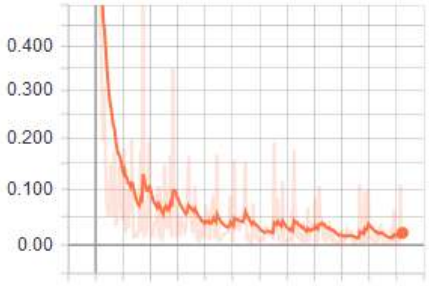
Antes da execução do treino as imagens foram rotuladas com as classes que iriam alimentar a rede neural, no caso do modelo do Kraken, só havia a classe peixe, contendo peixes diversos da fauna amazônica. Para rotulação dessas imagens foi utilizado o *software* LabelImg [3], esses dados servem para o *software* de treinamento saber onde os objetos estão situados na imagem, podendo assim treinar uma nova classe na última camada de uma rede neural.

Foram executadas 80.000 iterações totais no Tensorflow, dessas, 35.000 na máquina especificada na Tabela 1 e o restante na máquina especificada na Tabela 2. O tamanho das imagens foi limitado para 300 *pixels*, por limitações do recurso disponivel nos cmpoutadores utilizados, mantendo a proporção para não perder as características dos peixes.

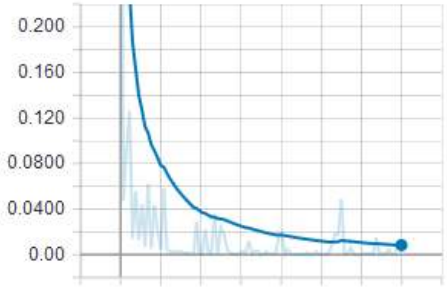
Os gráficos representados pelas Figuras 16 (quanto menor melhor) mostra se a “caixa"de classificação está no objeto treinado ou classificando elementos em plano de fundo. Após o treino o classificador obteve uma acurácia de 96%.

Figura 16 – Gráfico de perda

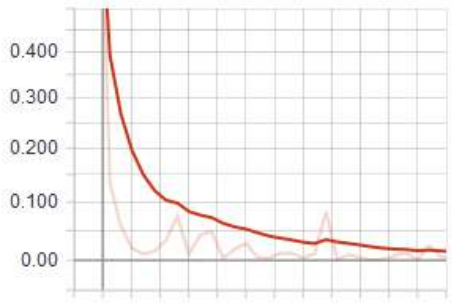
(a) Perda do classificador no primeiro treino.



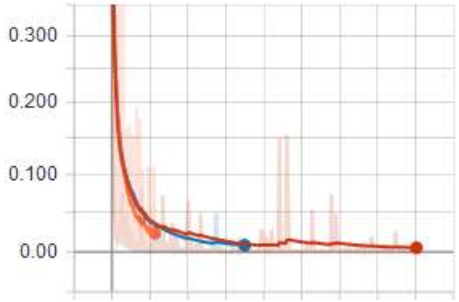
(b) Perda do classificador no segundo treino.



(c) Perda do classificador no terceiro treino.



(d) Perda do classificador sobrepostos.



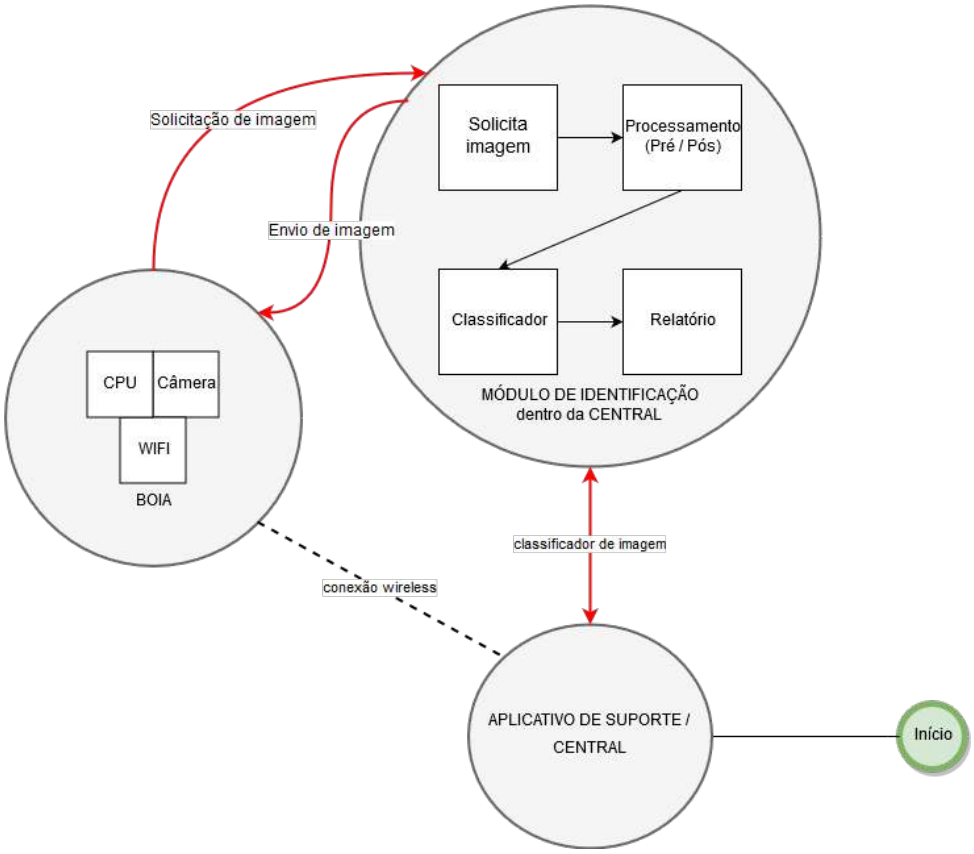
Fonte Própria

[3] <https://github.com/tzutalin/labelImg>

4.4 Análise dos dados na central de processamento

Como ilustra a Figura 17, após a coleta das imagens captados pelos sensores de captura (como câmeras) as imagens serão enviadas através de uma rede *Wireless* ou por *upload* de arquivo para a Central, onde serão processados pelo módulo de Classificação, que utilizará *OpenCV* (BRADSKI, 2000) para os filtros de pré-processamento, afim de melhorar a qualidade da imagem coletada. A classificação é então executada nas imagens já processadas, realizando a contagem e estimativa de tamanho, usando o *Tensorflow* que foi previamente modelado utilizando *Inception*, para retreinar as últimas camadas da sua rede neural. Após o processamento da imagem, será gerado um relatório sobre os dados processados dessa imagem e o resultado da classificação, exibindo um infográfico mostrando os resultados.

Figura 17 – Fluxo de Processos do Sistema Computacional Proposto



Fonte Própria

4.4.1 Módulo de pré-processamento

A etapa de pré-processamento consiste em verificar a qualidade da imagem para decidir se ela, vai passar por certas etapas de processamento para melhorias na imagem, caso a imagem esteja apta a classificação a mesma será enviada ou não para o módulo de pré-processamento. Dada a situação onde o ambiente observado não tem suas águas turvas o envio para o módulo de classificação

O módulo de pré-processamento foi desenvolvido utilizando a linguagem de programação Python[4](3.6) que utiliza as bibliotecas OpenCV[5](versão 3.2) e PILLOW[6](versão 6.0), integrando o software de classificação de imagens proposto neste trabalho.

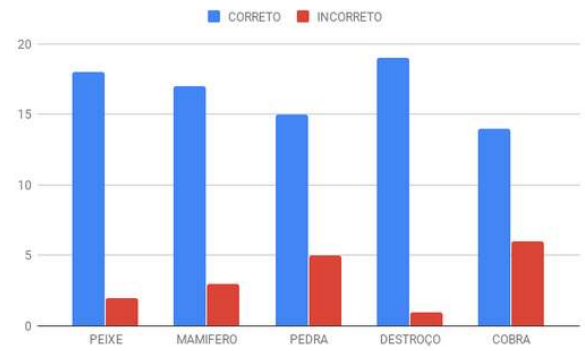
Para avaliar o impacto do filtro de pré-processamento foram executados alguns testes. Os testes foram executados com uma base de imagens, contendo cinco classes diferentes (exemplo, Peixes e Destroços), cada classe possuindo 20 (vinte) amostras diferentes. O experimento então foi dividido em duas partes, com as imagens pré-processadas e não pré-processadas.

Analisando os resultados dos experimentos, pode-se verificar que na Figura 18 (imagens pré-processadas), a utilização do *framework* Tensorflow com a biblioteca Inception v3, usando o modelo pré-treinado com imagens da base de dados ImageNet, que possui imagens de contextos da América do Norte, reduziu a taxa de acertos do classificador.

Figura 18 – Histograma de taxa de acerto do grupo de imagens pré-processadas

Fonte Própria

Figura 19 – Histograma de taxa de acerto do grupo de imagens não pré-processadas



Fonte Própria

Nas Figura 18 e Figura 19, pode-se observar a quantidade de classificações corretas e incorretas resultantes do experimento utilizando as amostras controladas. É possível observar

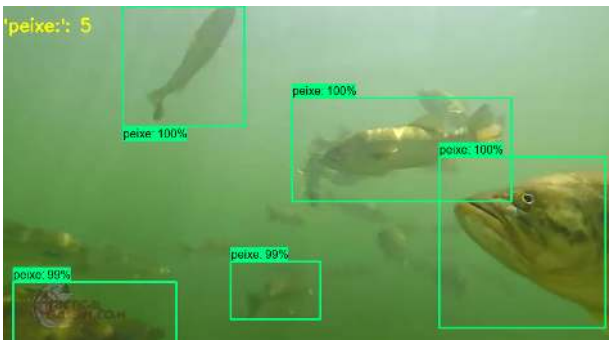
[4] https://www.python.org/
[5] <https://opencv.org/>
[6] <https://pillow.readthedocs.io/en/latest/>

uma sutil diferença entre o experimento utilizando as amostras pré-processadas contra as imagens não processadas. Contudo, pode-se notar que o número de acertos em ambas as abordagens com as imagens foi maior que os resultados incorretos.

4.5 Módulo de contagem

A contagem é feita atráves de uma API implementada utilizando o *framework* Tensorflow escrito em Python[7] (Versão 3) por Özlü (2018). O software recebe o arquivo de vídeo enviado pelo usuário, esse vídeo é processado usando o modelo (específicado na Seção 4.3.1). A contagem é feita durante a execução do vídeo, que é divídido em *frames* para facilitar o processamento, uma vez que só será utilizado a imagem do vídeo.

Figura 20 – Contagem de objetos



Fonte Própria

4.6 Estimativa de tamanho

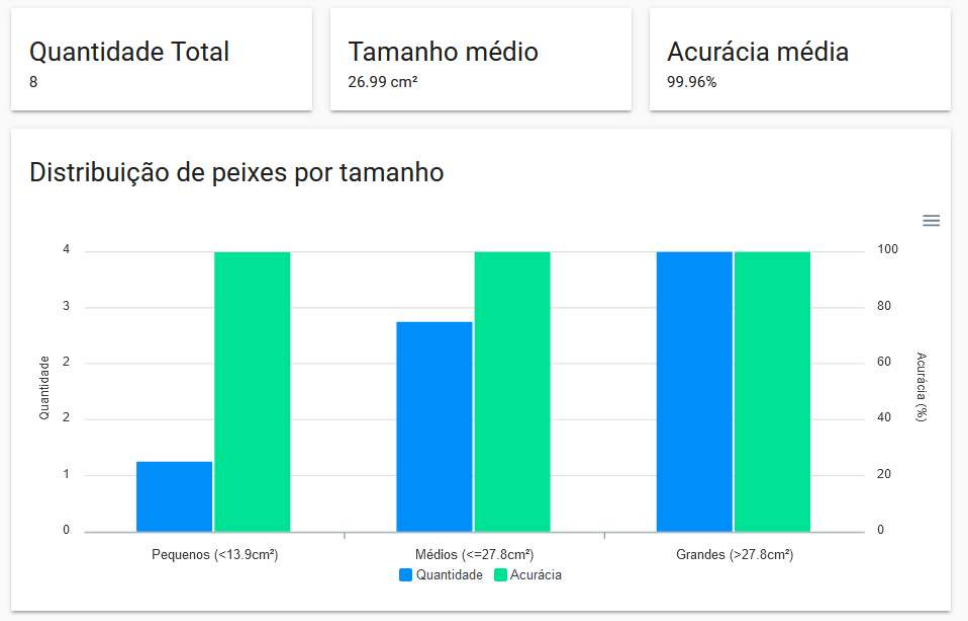
Estimativa de tamanho é realizada a partir da coleta da dados dos demarcadores de identificação de cada objeto detectado na imagem. Os dados são coletados pelo mesmo *script* de classificação, como pode ser visto na Figura 20. Os dados de tamanho então são transformados em estimativas de tamanho baseado numa constante de normalização, calibrada para calcular o tamanho com até 30cm de distância.

4.7 Módulo de exibição de relatório

Para tornar simples ao usuário a visualização dos resultados foi criada uma *interface web* que exibe os resultados de forma dinâmica, dando a possibilidade de fazer novas classificações a partir de arquivos de vídeo ou *streaming* de vídeo. A Figura 21 mostra como é a *interface* do relatório.

[7] www.python.org

Figura 21 – Relatório gerado a partir de dados coletados do Kraken

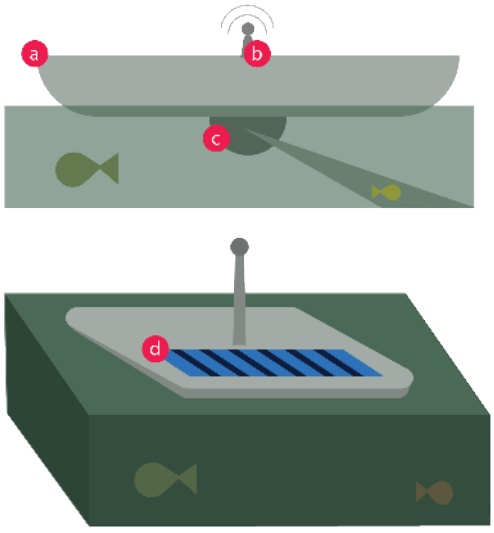


Fonte Própria

4.8 Construção da Boia

A estrutura da Boia, que é ilustrada na Figura 23, que visa baixo custo, utilizará materiais que podem ser encontrados com facilidade, como canos de PVC para a sustentação dos sensores e placas, tendo grande potencial para extensões ou modificações como alterar a estrutura para adicionar outro tipos de motores.

Figura 22 – Descrição da construção da Boia.



Fonte Própria

Como ilustra a Figura 22 e 23, os principais pontos a serem observados na construção da Boia são os seguintes:

a. Corpo da boia: feito com materiais de baixo custo e de fácil acesso, como canos de PVC. Está será uma parte importante, uma vez que todos os componentes elétricos da boia estarão armazenados nesse contêiner. Sensores como:

- Placa de circuito integrado (*Single Board Computer*) como um *Raspberry Pi zero*[8], ou um *smartphone*, que por ter conexão *wireless* embutida por padrão, diminuindo os custos com extensão para tal finalidade;
- Sensor de captura óptica (câmera);

b. Sensor de captura óptica (câmera): será utilizada a câmera *Raspberry Pi Camera*[9] *Module v2* ou um *smartphone*;

c. Antena (caso use um computador de proposito geral): a antena que será utilizada no projeto está embutida no computador de placa única *Raspberry Pi zero W*;

d. Placa Solar (opcional): um painel solar (genérico) que irá alimentar parte acoplada a boia.

Figura 23 – Protótipo da boia Kraken.

Fonte Própria

A utilização de um computador de proposito geral de pequeno porte, ou *smartphone*, podendo ou não conter várias entradas (digitais e analógicas), caso tenha, irá garantir o suporte para novas extensões como um oxímetro, que irá medir o nível de oxigênio na água. O uso do microprocessador será o meio para interligar a maioria dos sensores e fará a conexão com a central, garantindo que os dados sejam tratados e enviados ao usuário final.

[8] <https://www.raspberrypi.org/products/raspberry-pi-zero/>
[9] <https://www.raspberrypi.org/products/camera-module-v2/>

5 AVALIAÇÃO EXPERIMENTAL

Este capítulo tem como objetivo apresentar o planejamento, a execução e a análise dos resultados obtidos a partir da implementação do método proposto do Kraken. O foco desta avaliação se concentra no funcionamento do protótipo do Kraken por meio de experimentos em um cenário controlado com a utilização de amostras externas, levando em consideração a eficácia e a eficiência do sistema em situações de utilização nominal.

5.1 Planejamento e projeto dos experimentos

Esta avaliação experimental investiga a habilidade, do sistema computacional proposto, em classificar e estimar a medida dos peixes em um ambiente submerso com alta turbidez. Neste sentido, o sistema proposto foi implementado em um software denominado Kraken disponível em <https://github.com/danielgohl13/Kraken>. Considerando isso, pôde-se definir as seguintes questões de pesquisa:

QP1: A acurácia e a precisão do sistema Kraken é suficiente para provar sua confiabilidade?

QP2: O sistema Kraken é capaz de estimar o tamanho unitário do peixe via captura de video, quando em um ambiente submerso há mais de uma unidade?

QP3: O sistema Kraken é capaz de contabilizar o número de peixes em um ambiente submerso via captura de video?

Afim de responder as questões de pesquisa, foram elaborados experimentos para validação dos módulos. Para esses experimentos foram separados 7 (sete) vídeos com duração variada (entre 3 e 6 segundos) e resolução 640x360 pixels, adotada como padrão. Os vídeos usados para validar o sistema **Kraken** foram capturados em ambiente controlado e em trechos de gravações de pesca sub-aquática, obtidos no YouTube[1].

Para o ambiente de gravação do primeiro cenário utilizou-se uma câmera de 12 *megapixel* com a resolução fixada em 640x360 *pixels* para manter a mesma proporção em todos os vídeos, facilitando a estimativa de tamanho. A gravação foi feita com uma distância de 30 centímetros dos peixes, que se moviam, simulando o movimento dos peixes.

Por sua vez o segundo cenário é composto por testes em grupos de vídeos. O primeiro grupo da amostra contém os trechos que foram extraídos de vídeos recolhidos de gravações de pesca sub-aquática, com duração variando entre 3 e 5 segundos. A duração foi limitada pela

[1] <https://www.youtube.com/watch?v=HxjLc9OJNqU&t=94s>

limitação de *hardware* encontrada nas experimentações. No segundo grupo da amostra estão presentes os vídeos com reconhecimento humano. Os peixes foram analisados e contabilizados, com o objetivo de comparar ao primeiro grupo da amostra. Os vídeos são processados e as informações são exibidas em um relatório, como é representado pela Seção 4.7, com as quantidades e a acurácia de cada grupo de peixes.

5.2 Execução dos experimentos e análise dos resultados

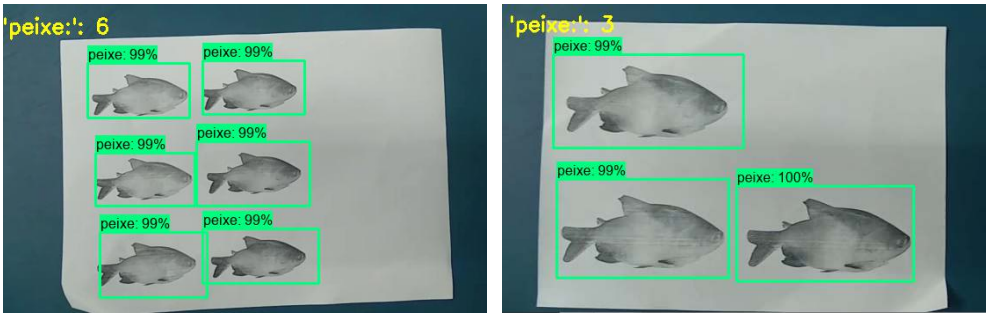
Nessa seção analisa-se os resultados obtidos pelos experimentos, procurando responder as perguntas de pesquisa criadas para validação dos módulos do sistema Kraken.

5.2.1 Prova de conceito

Visando responder a segunda questão (**QP2**) proposta nesse capítulo, foram executados testes com dois vídeos gravados em ambiente controlado, sendo cada cena do vídeo previamente definida para testar aspectos de funcionalidade do sistema proposto, exemplo, com imagens fora do ambiente submerso. Alguns trecho dos vídeos são expostos na Figura 24.

Figura 24 – Primeiro experimento

(a) Experimento controlado - Vídeo 1 (b) Experimento controlado - Vídeo 2

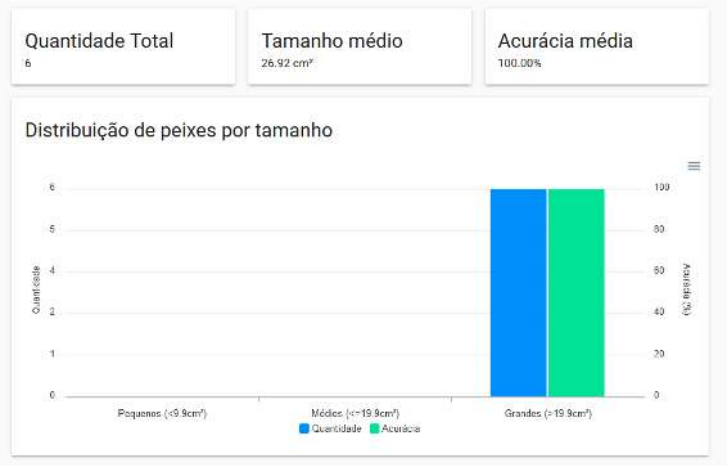


Fonte Própria

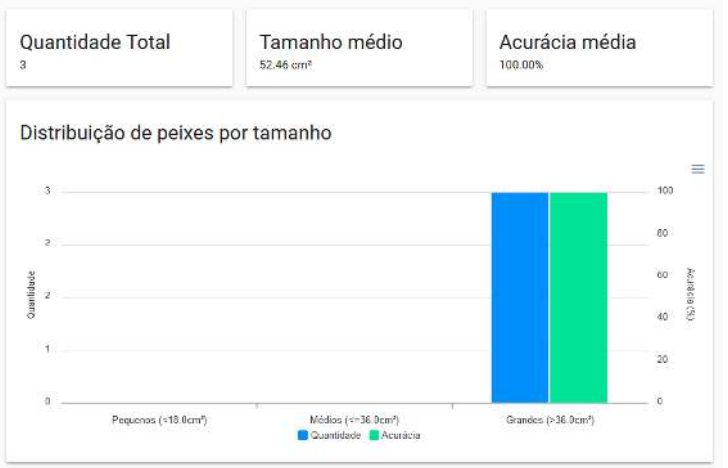
Analisando os resultados obtidos pelo sistema Kraken (na Figura 25), nota-se que o sistema foi capaz de identificar os peixes. Para calibrar o módulo de estimativa de tamanho no Kraken, foi definida a constante no valor de 2.75, que ao ser multiplicada pelo valor em *pixels* do tamanho da imagem coletada no vídeo de classificação, resulta no tamanho estimado do objeto em centímetros, o que não afeta a acurácia da classificação, alterando somente a estimativa de tamanho. O módulo de estimativa tem eficiência a uma distância de 30cm do objeto (distância definida na gravação do vídeo), e a acurácia de detecção média foi 99%, como pode ser visto na Figura 24, em ambiente controlado. Assim, esta prova de conceito apresenta a execução das principais funções do Kraken de maneira correta.

Figura 25 – Resultado do Classificador

(a) Resultado - Vídeo 1



(b) Resultado - Vídeo 2



Fonte Própria

5.2.2 Análise do módulo de contagem e estimativa de tamanho

Com a amostra de filmagens limitada a cinco vídeos, foi adotado o teste estatístico de Mann Whitney (NACHAR et al., 2008), para comparar tendências centrais de duas amostras independentes de tamanhos iguais, de dois grupos não pareados, a fim de verificar se pertencem ou não à mesma população. Para resolver a terceira questão (**QP3**) proposta na Seção 5.1, foi definido um experimento, que consiste na utilização dos dados representados na Tabela 3, composto por três etapas: (1) contagem humana dos vídeos representados pela Figura 26; (2) análise dos resultados do classificador do Kraken apresentado na Figura 27; (3) aplicação do teste estatístico de Mann Whitney sobre os resultados obtidos em (1) e (2).

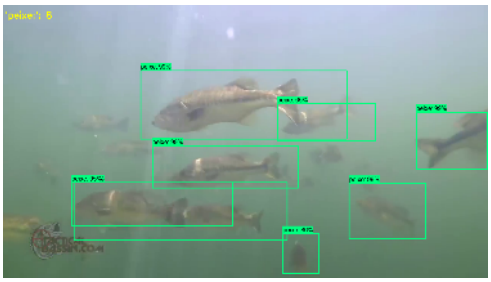
Tabela 3 – Quantidade contabilizada pelos experimentos.

	Quantidade de peixes contabilizados por humano	**Quantidade de peixes contabilizados pelo Kraken**
Vídeo 1	9	7
Vídeo 2	4	4
Vídeo 3	3	3
Vídeo 4	10	8
Vídeo 5	6	3

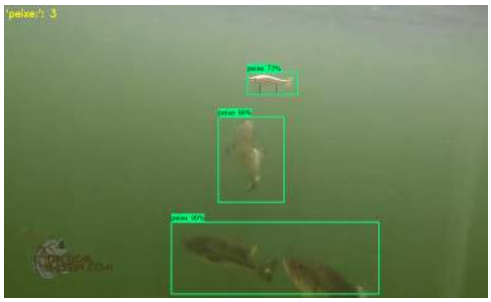
Fonte Própria

Figura 26 – Contagem Humana.

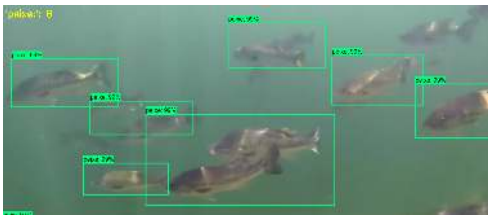
(a) Contagem Humana - Vídeo 1



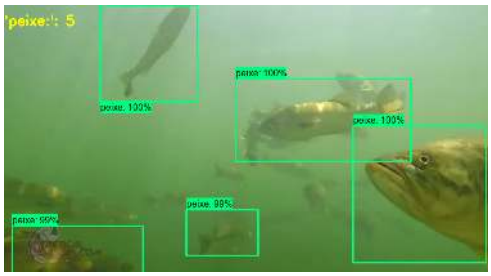
(b) Contagem Humana - Vídeo 2



(c) Contagem Humana - Vídeo 3



(d) Contagem Humana - Vídeo 4



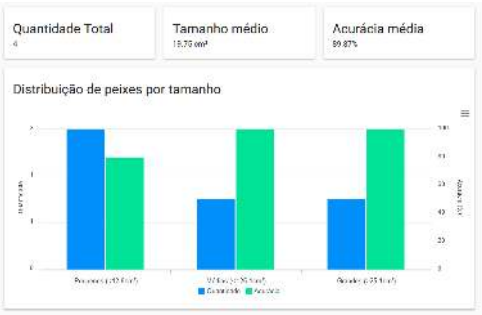
Fonte Própria

Após executar o teste proposto, aliado ao teste estatístico de Mann Whitney , pôde-se obter o valor-p de 0.22. O teste define que valores acima de um valor $alpha$ 0.05 determinam que as amostras têm a mesma tendência, pertencendo assim ao mesmo grupo, como pode ser visto na Figura 28, em que **CH** refere-se ao grupo de **Contagem Humana** e **CK** ao resultado do **Kraken**, com isso a terceira questão (**QP3**) pôde ser respondida: o módulo de contagem e classificação são funcionais.

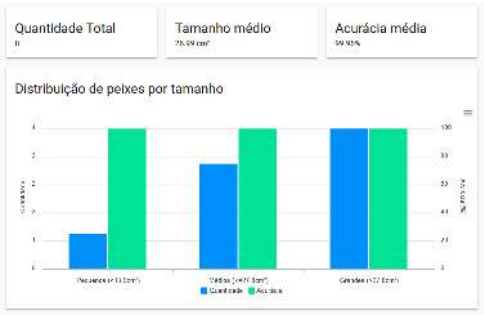
Figura 27 – Resultado do classificador.

(a) Resultado Classificador - Vídeo 1

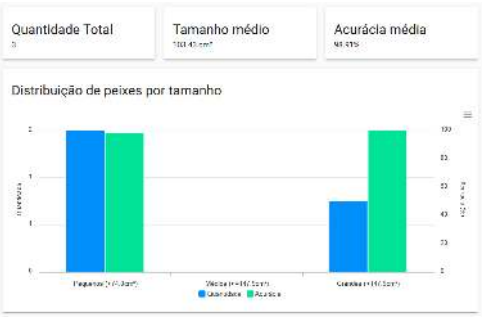
(b) Resultado Classificador - Vídeo 2



(c) Resultado Classificador - Vídeo 3

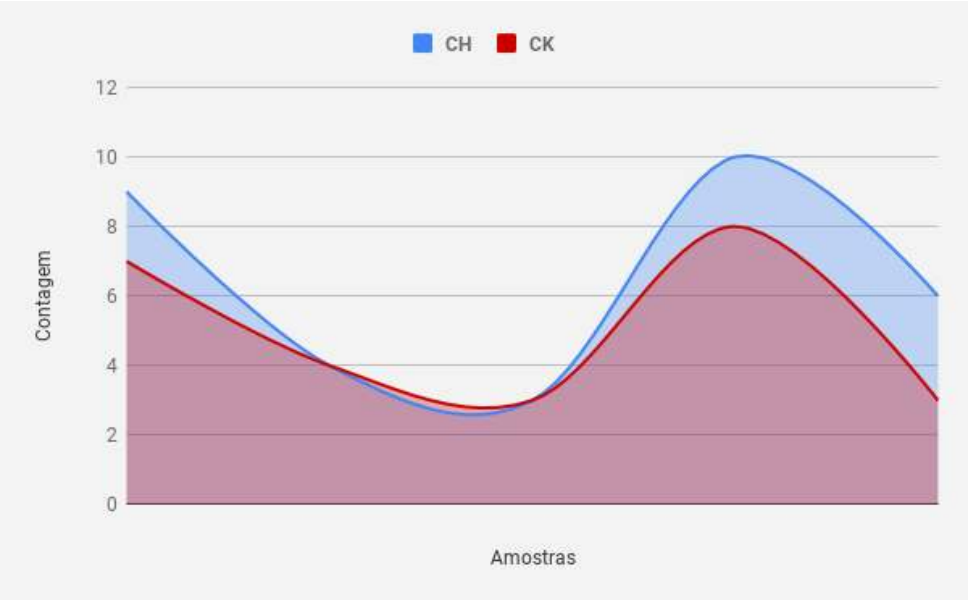


(d) Resultado Classificador - Vídeo 4



Fonte Própria

Figura 28 – Gráfico de normalidade.



Fonte Própria

5.2.3 Análise do funcionamento completo do sistema

Para responder a primeira questão de pesquisa (**QP1**) proposta neste capítulo, foram analisados os dados da Seção 5.2, em que as soluções foram satisfatórias para eficiência do sistema proposto, **Kraken**. Analisando os dados coletados pôde-se perceber que a acurácia em ambos os experimentos foi acima de 90%, tanto em ambiente controlado, como em trechos coletados de vídeos de pesca sub-aquática.

Considerando os resultados encontrados no Capítulo 5, pode-se destacar as vantagens e as desvantagens da utilização do sistema proposto:

- **Vantagens**:
  - Automatizar o trabalho da contagem de peixe em açudes, diminuindo mão de obra para tal tarefa, facilitando o trabalho de separar os grupos de peixes, exemplos, prontos para consumo.
  - Resultados satisfatórios durante a classificação, contagem e estimativa de tamanho dos peixes.
  - Baixo custo, comparado a mão de obra e tempo gastos para retirar os peixes, medir e contar.
- **Desvantagens**:
  - Limitação de distância para calcular o tamanho;
  - Tempo de processamento elevado, caso o seu uso seja feito com uma central de desempenho igual ou menor que a média de mercado;

6 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este trabalho abordou o desenvolvimento de um sistema computacional que consiste em uma boia que utiliza um computador de placa única ou *smartphone* para detecção de peixes (utilizando aprendizado de máquina) dentro de ambientes fluviais com alta turbidez, para auxiliar na atividade de piscicultura.

Ao longo do desenvolvimento do projeto foram observados trabalhos semelhantes que ajudaram a construir um método proposto (Kraken) para solucionar o problema proposto nesse trabalho de conclusão de curso.

Durante os experimentos foi possível responder as questões propostas para a execução do sistema proposto, Kraken. Os resultados obtidos foram satisfatórios, mostrando ótima precisão e acurácia enquanto classificando os peixes e estimando seus respectivos tamanhos, em contrapartida, constatou-se que o Kraken requer um grande poder computacional, necessitando de um *hardware* superior ou similar aos usados para a experimentação, detalhada na Capítulo 5.

Parte do objetivo do sistema computacional proposto é estimar o tamanho dos peixes detectados e relatar para o usuário. O Kraken possui grande potencial para adição de sensores no futuro, sensores esses como um oxímetro, ou motores para locomoção e estabilização. A Boia será projetada para suportar tais modificações de forma que os sensores sejam acoplados com facilidade, evitando desgaste no corpo da Boia.

Como trabalhos futuros planeja-se melhorias nos módulos, aumentando a eficiência para casos reais, pesquisa com proprietários de açudes e profissionais pisciários, para calcular o nível de interesse para utilização do Kraken, melhorias ou funções a serem adicionadas para melhorar a eficiência desse ramo de trabalho voltado a piscicultura.

7 APÊNDICE 1: REVISÃO SISTEMÁTICA DA LITERATURA

7.1 Revisão sistemática sobre detecção de peixes em ambiente parcialmente observáveis submersos

A revisão sistemática da literatura, foi parcialmente executada e foi o ponto de partida do projeto proposto, para encontrar artigos e publicações acadêmicas e industriais relevantes para extração de informações e métodos para identificação e detecção de peixes. A revisão sistemática da literatura utiliza as palavras-chaves relacionadas ao tema e artigos similares ao tema, aliando a formulação de questões de pesquisa para direcionar o estudo da revisão sistemática Kitchenham et al. (2009). Após a seleção das palavras-chaves, uma string de busca foi criada e executada no mecanismo de busca online da editora Elsevier, Scopus. Os resultados foram então processados e filtrados para encontrar artigos relevantes ao tema estudado, por meio de critério de seleção criados para avaliar e selecionar as publicações retornadas. Assim, foram identificados trabalhos correlatos e extraídos dados relacionados aos métodos propostos nestes trabalhos.

7.1.1 Estruturação das questões de pesquisa

A *string* de busca foi estruturada seguindo o padrão PICO (*population, intervention, comparison, outcome*) proposto por Kitchenham e Charters (2007). Onde:

- População: Trabalhos publicados em conferências e periódicos que relacionem estudos em ambientes aquáticos submersos.
- Intervenção: Relacionam a detecção e reconhecimento de peixes em ambientes aquáticos parcialmente observáveis.
- Comparação: Análise dos métodos e abordagens utilizados para detecção e reconhecimento de peixes nos ambientes estudados.
- Resultados: Dado os dados coletados através dos relatos encontrados sobre detecção e reconhecimento de peixes, pretende-se verificar os métodos para avaliar as abordagens encontradas.

A *string* de busca foi desenvolvida em bases nas seguintes características:

População: publicações que fazem referências à estudos em ambientes aquáticos submersos.

- **Palavra-Chave:** "underwater objects"OR “underwater vision” OR “underwater archaeology"OR“underwater images” OR “underwaterimaging” OR "submerged objects"OR “accurate underwater” OR “underwatersites” OR “sonar image enhancement"

Intervenção: publicações que relacionam detecção e reconhecimento de peixes em ambientes aquáticos parcialmente observáveis.

- **Palavra-Chave:** “computervisionphotogrammetry” OR “scene contrast” OR"object detection"OR "object tracking"OR “object classification” OR “detection and tracking of underwater” OR "object recognition"OR computer vision recognition” OR "object identification"OR “hidden-object detection” OR "hearlike view"

Para a aplicação da revisão sistemática serão definidas alguns itens de planejamento, sendo esses as questões de pesquisas estudadas e o ambiente utilizado. Visando responder as seguintes perguntas:

- Q1: Quais são os métodos de detecção de peixes em ambiente submersos com baixa visibilidade?
- Q1.1: Foi desenvolvido e está disponível alguma ferramenta para a aplicação do método?
- Q1.2: O método proposto foi gerado a partir da integração com outros?
- Q1.3: Qual é a estratégia de execução do método?
- Q1.4: Quais foram os resultados positivos ou negativos da validação/experimentação do método?
- Q1.5: Quais as limitações do método proposto?

7.2 Procedimentos de Seleção e Critérios

A estratégia de busca será aplicada por um pesquisador para identificar as publicações em potencial. A seleção das publicações dar-se-á em 4 etapas:

1. **Seleção e catalogação preliminar dos dados coletados:**A seleção preliminar das publicações será feita a partir da aplicação da expressão de busca às fontes selecionadas. Cada publicação será catalogada em um banco de dados criado especificamente para este fim e armazenada em um repositório para análise posterior;
2. **Seleção dos dados relevantes - [1 filtro]:** A seleção preliminar com o uso da expressão de busca não garante que todo o material coletado seja útil no contexto da pesquisa, pois a aplicação das expressões de busca é restrita ao aspecto sintático. Dessa forma, após

a identificação das publicações através dos mecanismos de buscas, deve-se ler o título, os resumos/abstracts e as palavras-chave e analisá-los seguindo os critérios de inclusão e exclusão identificados a seguir. Neste momento, poder-se-ia classificar as publicações apenas quanto aos critérios de exclusão, entretanto, para facilitar a análise e reduzir o número de publicações das quais se possam ter dúvidas sobre sua aceitação, deve-se também classificá-las quanto aos critérios de inclusão. Devem ser excluídas as publicações contidas no conjunto preliminar que:

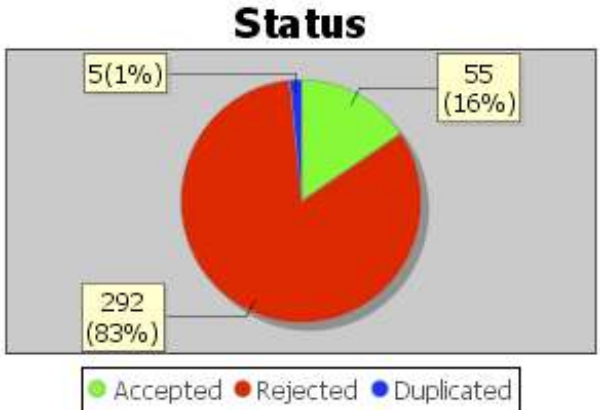
- **CE1-01:** Não serão selecionadas publicações em que as palavras-chave da busca não apareçam no título, resumo e/ou texto da publicação (excluem-se os seguintes campos: as seções de agradecimentos, biografia dos autores, referências bibliográficas e anexas).
- **CE1-02:** Não serão selecionadas publicações em que descrevam e/ou apresentam 'keynote speeches', tutoriais, cursos e similares.
- **CE1-03:**Não serão selecionadas publicações em que o contexto das palavras-chave utilizadas no artigo leve a crer que a publicação não cita uma abordagem para identificação de peixes submersos em água com baixa visibilidade.
- **CE1-04:** Não serão selecionadas publicações em que o contexto das palavras-chave utilizadas no artigo leve a crer que a publicação não cita uma abordagem para identificação de peixes submersos em água com baixa visibilidade.

Podem ser incluídas apenas as publicações contidas no conjunto preliminar que:

- **CI1-01:**Podem ser selecionadas publicações em que o contexto das palavras-chave utilizadas no artigo leve a crer que a publicação cita uma abordagem para identificação de peixes submersos em água com baixa visibilidade.
- **CI1-02:**Podem ser selecionadas publicações em que o contexto das palavras-chave utilizadas no artigo leve a crer que a publicação cita recomendações de melhoria na utilização de abordagens na identificação de peixes submersos em água com baixa visibilidade

Os resultados do primeiro filtro da revisão (parcial) da literatura estão ilustrados pelo gráfico na Figura 29 onde a taxa de aceitação e rejeição dos artigos.

Figura 29 – Resultados do primeiro filtro.



Fonte: própria

REFERÊNCIAS

ARDUINO UNO REV3. Arduino, 2018. Disponível em: <https://store.arduino.cc/usa/arduino-uno-rev3>.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. *Computer networks*, Elsevier, v. 54, n. 15, p. 2787–2805, 2010.

BAIER, C.; KATOEN, J.-P. *Principles of Model Checking (Representation and Mind Series)*. [S.l.]: The MIT Press, 2008. ISBN 026202649X, 9780262026499.

BARTHEM, R. B.; FABRÉ, N. N. Biologia e diversidade dos recursos pesqueiros da Amaz{ô}nia. *A pesca e os recursos pesqueiros na Amaz{ô}nia brasileira*, IBAMAPROV{Á}RZEA, v. 1, p. 17–62, 2004.

BAZEILLE, S. et al. Automatic underwater image pre-processing. In: *CMM'06*. [S.l.: s.n.], 2006. p. xx.

BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

BRADSKI, G.; KAEHLER, A. *Learning OpenCV: Computer vision with the OpenCV library*. [S.l.]: "O'Reilly Media, Inc.", 2008.

CHUANG, M. C.; HWANG, J. N.; WILLIAMS, K. A feature learning and object recognition framework for underwater fish images. *IEEE Transactions on Image Processing*, v. 25, n. 4, p. 1862–1872, 2016. ISSN 10577149.

CORTES, C.; VAPNIK, V. Support vector machine. *Machine learning*, v. 20, n. 3, p. 273–297, 1995.

CUNHA, E. et al. Formal Verification of UML Sequence Diagrams in the Embedded Systems Context. *2011 Brazilian Symposium on Computing System Engineering*, v. 1, p. 39–45, 2011. ISSN 2324-7886.

DAMME, T. V. Computer vision photogrammetry for underwater archaeological site recording in a low-visibility environment. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, Copernicus GmbH, v. 40, n. 5, p. 231, 2015.

FORESTI, G. L.; GENTILI, S. a Vision Based System for Object Detection in Underwater Images. *International Journal of Pattern Recognition and Artificial Intelligence*, v. 14, n. 02, p. 167–188, 2000. ISSN 0218-0014. Disponível em: <http://www.worldscientific.com/doi/abs/10.1142/S021800140000012X>.

GENTILI, G. F.; S. A Vision Based System for Object Detection in Underwater Images . n. March 2000, 2000.

GONZALEZ, R. C.; WOODS, R. E.; others. *Digital image processing*. [S.l.]: Addison-wesley Reading, 1992.

GUBBI, J. et al. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, v. 29, n. 7, p. 1645–1660, 2013. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X13000241>.

HEATH, S. *Embedded Systems Design*. Elsevier Science, 2002. ISBN 9780080477565. Disponível em: <https://books.google.com.br/books?id=BjNZXwH7HlkC273000/83f14e519542c81360f3d38c68d7c888>.

IMAGE RECOGNITION. TensorFlow, 2018. Disponível em: <https://www.tensorflow.org/tutorials/image_recognition>. Acesso em: 29 jun. 2018.

JAIN, A. K. *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989. v. 46. 400 p. ISSN 0734189X. ISBN 0133361659. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/0734189X89900418>.

JAIN, S.; VAIBHAV, A.; GOYAL, L. Raspberry Pi based interactive home automation system through E-mail. In: IEEE. *Optimization, Reliabilty, and Information Technology (ICROIT), 2014 International Conference on*. [S.l.], 2014. p. 277–280.

KITCHENHAM, B. et al. Systematic literature reviews in software engineering - A systematic literature review. *Information and Software Technology*, Elsevier B.V., v. 51, n. 1, p. 7–15, 2009. ISSN 09505849. Disponível em: <http://dx.doi.org/10.1016/j.infsof.2008.09.009>.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing Systematic Literature reviews in Software Engineering Version 2.3. *Engineering*, v. 45, n. 4ve, p. 1051, 2007. ISSN 00010782. Disponível em: <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Guidelines+for+performing+Systematic+Literature+Reviews+in+Software+Engineering#0%5Cnhttp://www.dur.ac.uk/ebse/resources/Systematic-reviews-5-8.pdf>.

LIONS, J.-L. et al. *Ariane 5 flight 501 failure report by the inquiry board*. [S.l.]: European space agency Paris, 1996.

LU, H.; LI, Y.; SERIKAWA, S. Computer vision for ocean observing. *Studies in Computational Intelligence*, v. 672, p. 1–16, 2017. ISSN 1860949X.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à visão computacional usando opencv. *Revista de Informática Teórica e Aplicada*, v. 16, n. 1, p. 125–160, 2009.

Martin˜Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Disponível em: <https://www.tensorflow.org/>.

MARWEDEL, P. *Embedded system design: Embedded systems foundations of cyber-physical systems*. [S.l.]: Springer Science & Business Media, 2010.

NACHAR, N. et al. The mann-whitney u: A test for assessing whether two independent samples come from the same distribution. *Tutorials in quantitative Methods for Psychology*, v. 4, n. 1, p. 13–20, 2008.

OPENCV. *OpenCV*. 2019. Disponível em: <https://opencv.org>.

PORTAL BRASIL. *Rios e bacias do Brasil formam uma das maiores redes fluviais do mundo*. Portal Brasil, 2009. Disponível em: <http://www.brasil.gov.br/meio-ambiente/2009/10/rios-e-bacias-do-brasil-formam-uma-das-maiores-redes-fluviais-do-mundo>.

QIN, H. et al. DeepFish: Accurate underwater live fish recognition with a deep architecture. *Neurocomputing*, v. 187, p. 49–58, 2016. ISSN 18728286.

RASPBERRY PI. *DATASHEET Raspberry Pi Compute Module (CM1) Raspberry Pi Compute Module 3 (CM3) Raspberry Pi Compute Module 3 Lite (CM3L)*. Raspberry Pi, 2016. 0–21 p. Disponível em: <https://www.raspberrypi.org/documentation/hardware/computemodule/RPI-CM-DATASHEET-V1_0.pdf%0Ahttps://www.sparkfun.com/products/13825>.

RUMBAUGH, J.; BOOCH, G.; JACOBSON, I. *The unified modeling language reference manual*. [S.l.]: Addison Wesley, 2017.

SANTOS, G. M. d.; SANTOS, A. C. M. d. Sustentabilidade da pesca na Amaz{ô}nia. *Estudos avancados*, SciELO Brasil, v. 19, n. 54, p. 165–182, 2005.

SCHLETT, M. Trends in embedded-microprocessor design. *Computer*, v. 31, n. 8, p. 44–49, 1998. ISSN 0018-9162.

SIEKKINEN, M. et al. How low energy is bluetooth low energy? comparative measurements with zigbee/802.15. 4. In: IEEE. *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*. [S.l.], 2012. p. 232–237.

SIOLI, H. *The Amazon: limnology and landscape ecology of a mighty tropical river and its basin*. [S.l.]: Springer Science & Business Media, 2012. v. 56.

SZEGEDY, C. et al. Rethinking the Inception Architecture for Computer Vision. *CoRR*, abs/1512.0, 2015. Disponível em: <http://arxiv.org/abs/1512.00567>.

SZELISKI, R. *Computer Vision : Algorithms and Applications*. Springer Science & Business Media, 2010. v. 5. 832 p. ISSN 10636919. ISBN 1848829345. Disponível em: <http://research.microsoft.com/en-us/um/people/szeliski/book/drafts/szelski_20080330am_draft.pdf>.

TENSORFLOW. *Premade Estimators*. 2018. Disponível em: <https://www.tensorflow.org/get_started/premade_estimators>. Acesso em: 3 jul. 2018.

VAHID, F.; GIVARGIS, T. *Embedded system design - a unified hardware/software introduction.* [S.l.: s.n.], 2002. I–XXI, 1–324 p. ISBN 978-0-471-45303-1.

WATSON, D. L. et al. A comparison of temperate reef fish assemblages recorded by three underwater stereo-video techniques. *Marine Biology*, Springer, v. 148, n. 2, p. 415–425, 2005.

WHITE, E. *Making Embedded Systems: Design Patterns for Great Software*. O'Reilly Media, 2011. v. 2011. 328 p. ISBN 1449320589. Disponível em: <http://books.google.com/books?id=VCOTy1xWZmQC%7B&%7Dpgis=1http://gen.lib.rus.ec/book/index.php?md5=198A2D91EB3D7296D85BBF1D099F5C84>.

XIA, F. et al. Internet of things. *International Journal of Communication Systems*, v. 25, n. 9, p. 1101–1102, 2012. ISSN 10745351.

ÖZLü, A. *TensorFlow Object Counting API*. 2018. Disponível em: <https://github.com/ahmetozlu/tensorflow_object_counting_api>.