Análise e monitoramento da biodiversidade em ecossistemas marinhos utilizando técnicas de processamento de imagem

Pedro H. F. Fernandes

[1]Universidade Federal do Cariri (UFCA)
CEP 63.048-080 — Juazeiro do Norte — CE — Brasil

pedro.fernandes@aluno.ufca.edu.br

Abstract. *Over the years, marine ecosystems have suffered from various threats, such as pollution, climate change and overfishing. Monitoring biodiversity in these environments can contribute to their conservation, but this is often done manually and inefficiently. The use of underwater cameras and image processing techniques can help automate species identification and counting, contributing to subsequent studies on ecology and conservation. In this work, developed for the Image Processing discipline, we propose a method to identify marine animals in underwater images, using various pre-processing, segmentation, counting and performance evaluation techniques to achieve this.*

Resumo. *Ao longo dos anos, os ecossistemas marinhos vem sofrendo com diversas ameaças, como poluição, mudanças climáticas e sobrepesca. Monitorar a biodiversidade nesses ambientes pode contribuir para a conservação, mas muitas vezes isso é feito de maneira manual e ineficiente. O uso de câmeras subaquáticas e técnicas de processamento de imagens pode ajudar a automatizar a identificação e contagem de espécies, contribuindo para subsequente estudos de ecologia e conservação. Nesse trabalho, desenvolvido para a disciplina de Processamento de Imagem, propomos um método para identificar animais marinhos em imagens subaquáticas, onde para realizá-lo foram utilizadas de várias técnicas de pré-processamento, segmentação, contagem e avaliação de desempenho.*

1. Introdução

O petróleo é uma das fontes de combustível mais importante atualmente, mas apesar de sua relevância, a exploração petrolífera tem como consequência diversos impactos desde a extração até a utilização. A importância e sensibilidade ecológica do ambiente marinho faz com que haja riscos em qualquer alteração não natural do mesmo, especialmente se tratando de atividades de grande magnitude como é a exploração e produção de petróleo.

A compreensão desses impactos possibilita uma conscientização para o desenvolvimento de novas tecnologias e aplicação de melhores e mais eficazes técnicas de controle, prevenção e remediação. Nascimento et al. alertou em [Nascimento et al. 2021] sobre o impacto ambiental da extração do petróleo em ambientes marinhos.

Contudo, o petróleo não é o único fator que vem contribuindo com esse impacto. Vários fatores como poluição, mudanças climáticas e sobrepesca também contribuem muito para a degradação dos ecossistemas marinhos ao longo das últimas décadas. Isso acabar por afetar também a biodiversidade marinha, a qual não podemos monitorar de formas eficaz, eficiente e automática em vários lugares do mundo.

Uma possível solução para esse problema, seria, por exemplo, o uso de câmeras embarcadas subaquáticas combinadas com técnicas de processamento de imagem para automatizar a identificação e contagem de espécies, o que acabaria por contribuir para o desenvolvimento de novos estudos com o foco em conservar a biodiversidade desses ambientes aquáticos.

Com isso, o objetivo desse trabalho, que foi desenvolvido para a disciplina de Processamento de Imagem, é propor um método para identificar animais marinhos em imagens subaquáticas, utilizando-se de várias técnicas simples de pré-processamento, segmentação, contagem e avaliação de desempenho.

2. Trabalhos Relacionados

[Marini et al. 2018] foi um dos trabalhos relacionados usados como inspiração para este. Neste estudo, foi desenvolvida uma nova metodologia para reconhecimento e contagem automatizada de peixes em um vídeo-observatório cabeado, que permitiu levar em conta uma variedade de circunstâncias operacionais que incluíram grandes variações na intensidade da luz, turbidez da água, crescimento de incrustações e densas assembleias de peixes.

A metodologia de análise e reconhecimento de imagens proposta neste trabalho combinou um processo de segmentação de imagens e um processo de extração de características de imagem, juntamente com uma abordagem de aprendizado de máquina supervisionado. Isso é acoplado a uma estrutura de validação cruzada K-fold. O processo de segmentação proposto visava reduzir os efeitos das mudanças na difusão da luz, os efeitos da água turva e os efeitos da presença de bioincrustação na câmera.

O algoritmo de reconhecimento de imagem baseado em conteúdo para contagem de peixes foi aprendido e testado usando o conjunto de dados de imagens fornecido pelo Observatório Expansível SEAfoor do Mediterrâneo Ocidental (OBSEA) nos anos de 2012 e 2013, com frequência de 30 minutos, continuamente durante o dia e a noite. Os resultados encontrados nesse trabalho, mostram que o desempenho do reconhecimento é afetado principalmente pela bioincrustação e pelos erros do sistema. Na verdade, as mudanças na radiação luminosa e a aglomeração de peixes são onipresentes no conjunto de dados de imagem e seus efeitos combinados no desempenho de reconhecimento são marginais e podem ser ignorados.

3. Método Proposto

A metodologia que foi escolhida para ser aplicada nesse trabalho, tem o objetivo de realizar a detecção e contagem de animais marinhos em imagens de ecossistemas subaquáticos, de preferência em imagens com muito pouca iluminação natural, utilizando técnicas de processamento de imagem computacional.

Foram implementadas para isso, etapas de pre-processamento, segmentação, extração de regiões e avaliação de desempenho, usando para validação dos resultados a localização real dos animais de cada imagem provida junto com a base de dados utilizada. A escolha das técnicas utilizadas em cada etapa foram baseados em [Raveendran et al. 2021, Jian et al. 2021], trabalhos focados em fazer um apanhado de técnicas de processamento de imagens, especificamente para imagens em baixo d'água. A Figura 1 ilustra o método proposto:

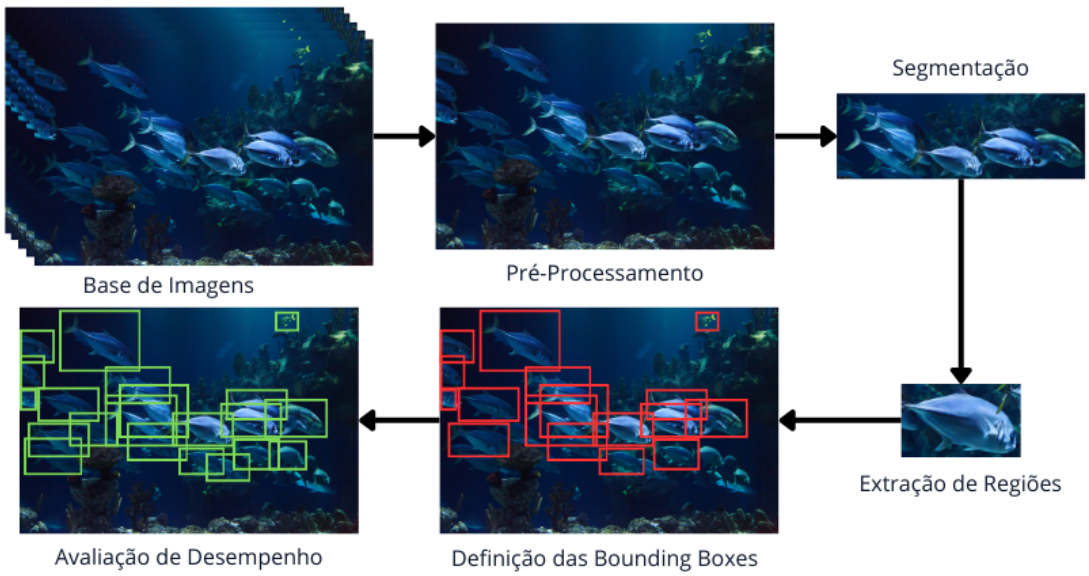

Figura 1. Etapas do método proposto.

3.1. Base de Dados

A base de dados usada para validar o método proposto foi a ”Underwater Object Detection Dataset”do usuário Slavko Prytula encontrada no site Kaggle. Ela inclui 638 imagens de diferentes animais marinhos, todas em formato JPG, com o conjunto de dados já dividido em conjuntos de treinamento, validação e teste. Ela também inclui arquivos contendo as localizações das *bounding boxes* de cada animal para todas as imagens. A Figura 2 mostra uma das imagens incluídas na base dados, que abrangem desde imagens parecidas a imagens bem diferentes como de pinguins fora da água.

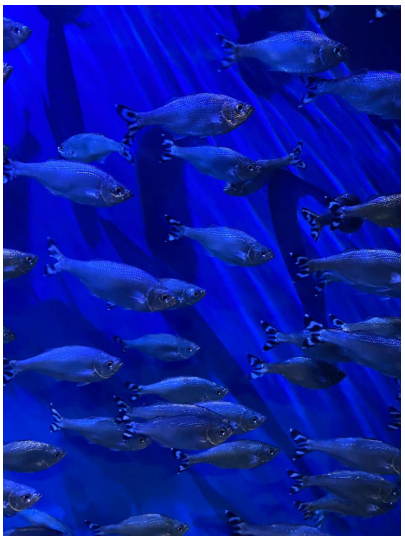

Figura 2. Imagem do banco de dados.

3.2. Pré-Processamento

O pré-processamento foi realizado antes da segmentação para melhorar a qualidade das imagens, com o objetivo de facilitar os passos subsequentes do algoritmo. As etapas executadas incluíram:

Conversão para Escala de Cinza: Primeiro, a imagem foi convertida para escala de cinza com a função `cv2.cvtColor` da biblioteca do *OpenCV*. Essa conversão tem a função de melhorar a iluminação da imagem, problema comum em imagens em baixo d'água.

Desfoque Gaussiano: Depois, aplicamos um filtro de desfoque gaussiano `cv2.GaussianBlur` de tamanho 3x3 pixels, para remover ruídos e deixar a imagem mais suave, facilitando assim a identificação dos objetos.

Melhoramento de CLAHE: Por fim, aplicamos o algoritmo de CLAHE (Contrast Limited Adaptive Histogram Equalization) `clahe.apply`, que é um algoritmo de equalização de histograma que particiona a imagem em regiões contextuais e aplica a equalização do histograma a cada um. Como visto no trabalho [Rai et al. 2012], isso acaba por equilibrar a distribuição dos valores de cinza usados e, assim, faz características ocultas da imagem mais visíveis.

A Figura 3 ilustra o resultado do pré-processamento na imagem mostrada na figura 2.

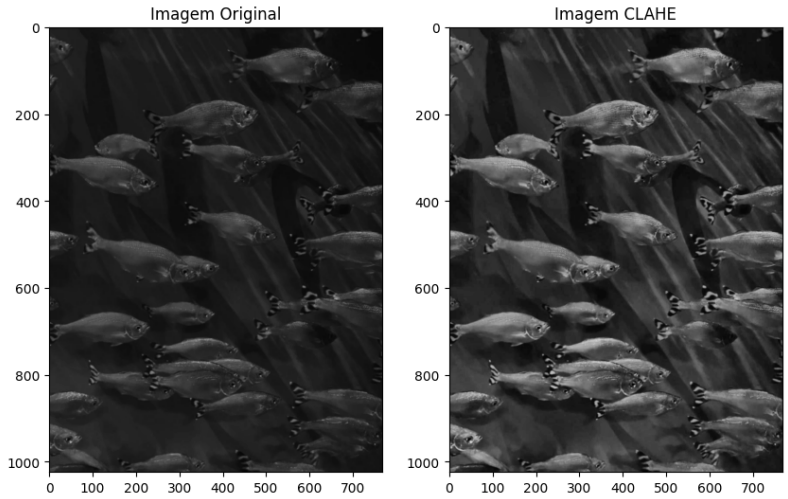

Figura 3. Imagem após o pré-processamento.

3.3. Segmentação

Após a etapa de pré-processamento, realizamos a segmentação da imagem, onde geramos uma máscara binária a partir da imagem para realizar a extração das componentes conexas dela logo em seguida. As etapas executadas incluíram:

Limiarização: Primeiro, aplicamos novamente um desfoque gaussiano, seguido diretamente por uma *Binarização de Otsu* `cv2.threshold`, para gerarmos uma máscara inicial que seja mais provável de conter os animais da imagem.

Transformação Morfológica: Depois, aplicamos uma transformação morfológica de fechamento `cv2.morphologyEx(cv2.MORPH-CLOSE)` usando um *kernel* elíptico de tamanho 50x50 pixels, para tentar manter a forma completa dos animais.

Segunda Passagem: Então, combinamos a máscara gerada com a imagem inicial e realizamos uma segunda leva de operações compostas por: um desfoque gaussiano, seguido por outra binarização de Otsu, e finalizando com duas transformações morfológicas de abertura `cv2.morphologyEx(cv2.MORPH-OPEN)` seguidas, usando dois kernels elípticos de tamanho 20x5 e 25x25 pixels respectivamente.

A Figura 4 ilustra o resultado das etapas de segmentação na imagem mostrada na figura 3.

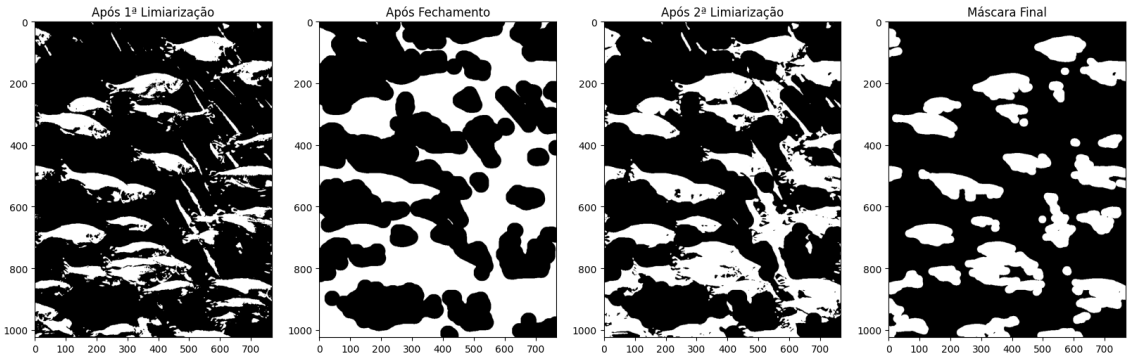

Figura 4. Imagem após as etapas de segmentação.

3.4. Extração de Regiões

Apos a etapa da segmentação, as regiões conectadas foram identificadas utilizando a função `label` do modulo `skimage.measure`. E então, a função `regionprops` foi usada para calcular as caixas delimitadoras (*bounding boxes*) de cada animal detectado, o que permite a identificação de todos os animais na imagem. Além disso, todas as caixas encontradas foram expandidas em uma razão igual a um parâmetro regulável `fator-aumento`, que utilizamos para fazer vários testes diferentes mudando ele.

A Figura 5 ilustra as *bounding boxes* encontradas usado a máscara encontrada na segmentação que foi mostrada na figura 4.

3.5. Avaliação de Desempenho

Como etapa final, usamos a métrica *Intersection over Union* (IoU) para realizar a avaliação do desempenho do modelo. Também usamos o Classificador KNN `KNeighborsClassifier` do modulo `sklearn.neighbors` para mapearmos cada caixa encontrada pelo modelo com a caixa equivalente das reais.

Com isso, conseguimos o valor do IoU para todas as caixas encontradas, para as caixas equivalentes reais, e calculamos a média de todos esses valores. Além disso, multiplicamos essa média por um valor modificador, que inicialmente tem valor 1 e diminui em $0,01$ para caixa encontrada a mais ou menos do número real de caixas esperado. O valor encontrado é a **acurácia** obtida para aquela imagem.

A Figura 5 mostra a comparação das *bounding boxes* encontradas com as reais, onde o cálculo nos deu uma acurácia de $41,57\%$.

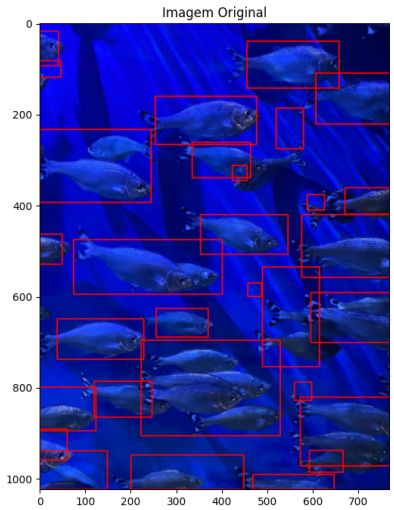

Figura 5. Imagem após a adição das caixas delimitadoras. (fator-aumento = 1,2)

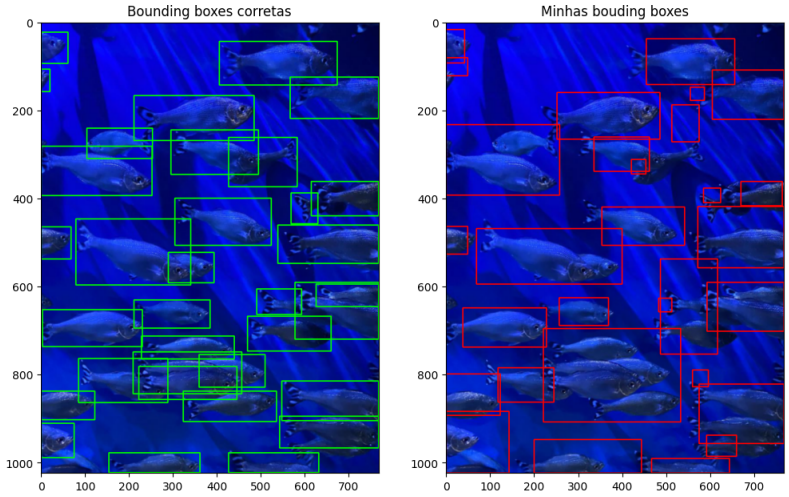

Figura 6. Comparação da imagem com as caixas delimitadoras reais e as caixas delimitadoras encontradas, onde foi obtido acurácia de 41,57%.

Por fim, calculamos a média das acurácias de todas as imagens do conjunto e obtemos a média de todas esses valores, para obter a acurácia geral do modelo para aquele conjunto de dados.

4. Resultados

Nessa seção, mostraremos as configurações utilizadas no experimentos, quais foram os valores dos parâmetros utilizados nos experimentos bem como o resultado das acurácias obtidas em cada um dos testes. Não foram feitos nenhum procedimento de preparação na base de imagens que não foram mencionados na Seção 3.2.

4.1. Configuração Experimental

Os testes foram realizados em um ambiente virtual do *Visual Studio Code*, utilizando um *Jupyter Notebook*. As especificações de *hardware* incluíram GPU Nvidia GeForce RTX

3050 6GB. O desenvolvimento foi conduzido em linguagem Python, usando bibliotecas como OpenCV e Scikit-Learn.

A base de imagens usada (Underwater Object Detection Dataset) já veio dividida em 3 conjuntos, *train*, *valid* e *test*, com 448, 127 e 63 imagens respectivamente. A base também possui arquivos das listas de localizações das *bounding boxes* corretas para todas as imagens. Além disso, nós criamos um quarto conjunto de teste *new* com 184 imagens, selecionando a mão imagens dos outros conjuntos para testar como a variância da base afetou o método.

Por fim, para avaliar a precisão do método, utilizamos a métrica *Intersection over Union* (IoU). Para cada imagem, calculamos o IoU das caixas delimitadoras similares entre as obtidas pelo método e as verdadeiras. Fizemos então a média dos valores de IoU calculados a partir de todas as caixas encontradas pelo método. E por fim, multiplicamos essa média por um modificador que começa em 1 e diminui em $0,01$ para cada caixa obtida que faz o número de caixas encontradas se afastar do número de caixas real.

4.2. Resultados Experimentais

Para validar o método proposto, foram realizados os seguintes experimentos:

Para cada um dos conjuntos de imagens usados (*train*, *valid* e *test*, *new*), o modelo foi testado usando 4 valores diferentes do parâmetro `fator-aumento` para considerar condições diferentes para o sistema. A Tabela 1 mostra os resultados das acurácias obtidas para cada valor do parâmetro em cada conjunto:

Tabela de Acurácia				
fator_aumento	train	valid	test	new
1	7,00%	6,10%	8,99%	12,21%
1,2	7,53%	6,53%	9,39%	12,99%
1,5	7,37%	6,42%	8,95%	12,30%
2	6,78%	6,04%	8,02%	10,60%

Tabela 1. Tabela de resultados da acurácia do método para valores diferentes de fator_aumento em todos os conjuntos da base de imagens.

Como pode ser visto na Tabela 1, o valor do `fator-aumento` que conseguiu melhores resultados em todos os conjuntos foi $1,2$. Além disso, podemos ver que o conjunto `new`, que foi construído com base nos outros conjuntos, conseguiu desempenho consideravelmente melhor que os outros para todos os valores do parâmetro. Isso mostra que, muito provavelmente, a alta variância da base de imagens afetou negativamente os resultados do modelo.

Por via de comparação, no trabalho [Marini et al. 2018], o desempenho de reconhecimento obtido na fase de validação resultou em uma precisão média de 92% com desvio padrão (std) igual a $0,02$, uma taxa de verdadeiros positivos (TPR) igual a 95% com std igual a $0,03$ e uma taxa de falsos positivos (FPR) que equivale a 12% com std

igual a $0,04$. A Tabela 2 resume os dados utilizados para o treinamento do classificador de imagens e também o desempenho de validação correspondente. Em comparação, percebe-se que o nosso resultado performou muito abaixo do esperado.

Validation Performance	Accuracy (std)	92% (0.02)
	True Positive Rate (std)	95% (0.03)
	False Positive Rate (std)	12% (0.04)

Tabela 2. Resumo dos dados adquiridos no ano de 2012 que foram utilizados para treinar e validar o classificador de imagens binárias.

5. Conclusão

Como foi visto na Seção 4, os resultados obtidos a partir da aplicação do modelo proposto para segmentação e identificação de animais marinhos em imagens de ecossistemas subaquáticos, de preferência em imagens com muito pouca iluminação natural, não foram muito satisfatórios, conseguindo acurácias calculadas entre 0,06 e 0,13 dependendo do conjunto de imagens. Como essa acurácia foi calculada com base na comparação entre as *bounding boxes* encontradas e as verdadeiras, o baixo valor conseguido mostra as falhas dessa abordagem, principalmente no que se diz a segmentação dos objetos.

Como possuíam muitos tipos de animais diferentes, tornou-se difícil conseguir segmentar todos eles do fundo usando apenas técnicas de processamento de imagem, o que acabou impactando negativamente no desempenho. Como é muito comum na literatura, uma abordagem melhor para esse problema provavelmente envolveria técnicas de aprendizado de máquina, como foi visto no trabalho [Zhao et al. 2019].

Além disso, a alta variância das imagens da base de dados acabou de se tornando um empecilho, como indica o fato do conjunto `new` construído para o problema ter conseguido um resultado consideravelmente melhor que o dos outros conjuntos. Isso se deu pelo fato da base possuir muitos tipos de imagens com composições de cores e luminosidade completamente diferentes, o que acabou tornando inviável de se desenvolver um método para separar os animais do fundo utilizando apenas técnicas de segmentação simples de forma precisa. Novamente, esse problema poderia ser melhor resolvido utilizando técnicas mais robustas de visão computacional.

Agora em relação aos pontos positivos desse trabalho, além de conseguirmos entender melhor as limitações das ferramentas de processamento de imagem, pudermos notar como o algoritmo de CLAHE se mostrou eficiente no que diz respeito a aumento de contraste, especialmente em imagem com baixo nível de luminosidade.

E, apesar de tudo, considerando todos os desafios, limitações e escopo do problema, o resultado obtido foi aceitável, visto que nas imagens que mais se aproximavam do cenário ideal, o modelo conseguia se aproximar bem mais do resultado real. Dito isso, ângulos futuros a se seguir seriam melhorar o modelo usando técnicas diferentes, que não foram usadas nesse trabalho, diminuir o escopo de modo a obter um ambiente mais controlado que tornasse o trabalho de identificação mais receptível ou então aumentar a escala do modelo de modo a incluir as técnicas mais robustas de aprendizado de máquina e *deep learning*, por exemplo.

6. Agradecimentos

Agradeço a professora Luana Batista por ministrar essa disciplina de Processamento de Imagens, bem como os meus colegas que compartilharam a experiência de participar dessa cadeira.

Referências

Jian, M., Liu, X., Luo, H., Lu, X., Yu, H., and Dong, J. (2021). Underwater image processing and analysis: A review. *Signal Processing: Image Communication*, 91:116088.

Marini, S., Fanelli, E., Sbragaglia, V., Azzurro, E., del Rio, J., and Aguzzi, J. (2018). Tracking fish abundance by underwater image recognition. *Scientific Reports*, 8.

Nascimento, N. R., Vital, G. V., Plaza, A. S., and Souza, G. L. M. (2021). Atividade petrolífera offshore e sua relação com os impactos ambientais nos ecossistemas marinhos. *Meio Ambiente (Brasil)*, 3(5).

Rai, R., Gour, P., and Singh, B. (2012). Underwater image segmentation using clahe enhancement and thresholding. *International Journal of Emerging Technology and Advanced Engineering*, 2:118–123.

Raveendran, S., Patil, M., and Birajdar, G. (2021). Underwater image enhancement: a comprehensive review, recent trends, challenges and applications. *Artificial Intelligence Review*, 54.

Zhao, M., Hu, C., Wei, F., Wang, K., Wang, C., and Jiang, Y. (2019). Real-time underwater image recognition with fpga embedded system for convolutional neural network. *Sensors*, 19(2).