Lightweight Deep Neural Network for Joint Learning of Underwater Object Detection and Color Conversion

Chia-Hung Yeh, *Senior Member, IEEE*, Chu-Han Lin, Li-Wei Kang, *Member, IEEE*, Chih-Hsiang Huang, Min-Hui Lin, Chuan-Yu Chang, *Senior Member, IEEE*, and Chua-Chin Wang, *Senior Member, IEEE*

***Abstract*— Underwater image processing has been shown to exhibit significant potential for exploring underwater environments. It has been applied to a wide variety of fields, such as underwater terrain scanning and autonomous underwater vehicles (AUVs)-driven applications, such as image-based underwater object detection. However, underwater images often suffer from degeneration due to attenuation, color distortion, and noise from artificial lighting sources as well as the effects of possibly low-end optical imaging devices. Thus, object detection performance would be degraded accordingly. To tackle this problem, in this article, a lightweight deep underwater object detection network is proposed. The key is to present a deep model for jointly learning color conversion and object detection for underwater images. The image color conversion module aims at transforming color images to the corresponding grayscale images to solve the problem of underwater color absorption to enhance the object detection performance with lower computational complexity. The presented experimental results with our implementation on the Raspberry pi platform have justified the effectiveness of the proposed lightweight jointly learning model for underwater object detection compared with the state-of-the-art approaches.**

***Index Terms*— Convolutional neural networks, deep learning, lightweight deep model, underwater image processing, underwater object detection.**

Manuscript received 13 March 2020; revised 1 October 2020 and 21 January 2021; accepted 30 March 2021. Date of publication 26 April 2021; date of current version 28 October 2022. This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant NSC 102-2221-E-110-032-MY3, Grant MOST 103-2221-E-110-045-MY3, Grant MOST 103-2221-E-003-034-MY3, Grant MOST 105-2221-E-003-030-MY3, Grant MOST 108-2221-E-003-027-MY3, Grant MOST 108-2218-E-003-002, Grant MOST 108-2218-E-110-002, Grant MOST 109-2218-E-110-007, Grant MOST 109-2224-E-110-001, and Grant MOST 109-2218-E-003-002; and in part by the Intelligent Recognition Industry Service Center, National Yunlin University of Science and Technology, Douliu, through the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan. *(Corresponding author: Li-Wei Kang.)*

Chia-Hung Yeh is with the Department of Electrical Engineering, National Taiwan Normal University, Taipei 106, Taiwan, and also with the Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung City 80021, Taiwan.

Chu-Han Lin, Chih-Hsiang Huang, Min-Hui Lin, and Chua-Chin Wang are with the Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung City 80021, Taiwan.

Li-Wei Kang is with the Department of Electrical Engineering, National Taiwan Normal University, Taipei 106, Taiwan (e-mail: lwkang@ntnu.edu.tw).

Chuan-Yu Chang is with the Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, Douliu 64002, Taiwan.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3072414.

Digital Object Identifier 10.1109/TNNLS.2021.3072414

I. Introduction

THE exploration of underwater environment has been popular recently based on the growing deficiency of natural resources and the development of global economy. In addition, several ocean engineering-related applications and studies have increasingly relied on underwater images captured by autonomous underwater vehicles (AUVs) [1]. However, acquiring underwater images based on optical imaging devices encounters more challenges than that in the atmosphere. More specifically, underwater images often suffer from degeneration due to attenuation, color distortion, and noise from artificial lighting sources, as well as the effects possibly induced by low-end optical imaging devices, that is, the scattering and absorption attenuate the direct transmission, resulting in surrounding scattered light. As a result, the attenuated direct transmission reduces the scene intensity and induces color distortion, while the surrounding scattered light distorts the scene appearance. Such degenerations seriously affect the related tasks in exploration of underwater environments, such as underwater object detection and recognition. Some research works have been presented in the literature for underwater image restoration or enhancement (see [2]–[9]). More specifically, in [3], underwater image enhancement was formulated as a haze removal problem and a dehazing method with minimum information loss and histogram distribution prior was presented. Moreover, for better representing underwater images, a revised underwater image formation model was proposed in [7]. This model used the oceanographic measurements to derive the physically valid space of backscatter, which would be beneficial to better correct complex underwater scenes. Furthermore, a color recovery method for underwater images based on the image formation model [7] using RGB-D images was presented in [8] and [9]. However, this article focuses on object detection from a single underwater image without needing to restore image quality in advance.

On the other hand, several underwater applications are associated with devices (e.g., cameras) equipped on an AUV [10]. However, for a battery-powered AUV, it is critical to embed low-complexity or low-power implementations of algorithms into it for performing related tasks (e.g., image acquisition or object detection/recognition) [11], that is, it would be beneficial to design a low-complexity model for the desired task performed underwater while maximizing the overall

system efficiency among the AUV lifetime and the task performance [12]. To achieve effective underwater image-based object detection with low-complexity computation, this article focuses on designing a lightweight deep model for object detection. It is expected that the proposed deep model would be suitable to be embedded into an AUV for several vision-based underwater applications.

A. Object Detection Based on Deep Learning

Object detection, as one of the most fundamental and challenging problems in the computer vision community, has received much attention in recent years [13]. Some classic object detection frameworks based on handcrafted feature engineering include Viola–Jones detector [14], histograms of oriented gradients (HoG) [15], and deformable part-based model (DPM) [16]. Moreover, a multiview-based parameter-free framework (MPF) was recently proposed to detect coherent groups for crowd behavior analysis in [17].

With the rapid development of deep learning techniques with great success in numerous perceptual tasks, see [18]–[21], several deep learning-based object detection frameworks [22]–[36] have been presented with better detection performance, compared with the state-of-the-art handcrafted feature-based methods (see [14]–[16]). Some of the deep learning-based object detectors include R-CNN [24], SPPNet [25], fast R-CNN [26], faster R-CNN [27], robust faster R-CNN [28], YOLO [29]–[32], SSD [33], feature pyramid networks (FPNs) [34], VSSA-NET [35], and Retina Net [36]. The core of the deep model-based object detection frameworks is the convolutional neural network (CNN), which automatically performs feature learning (or representation learning), incorporated with classification or regression task. To focus on underwater object detection relying on the deep learning techniques, some related research works were also presented recently, briefly introduced in Section I-B.

B. Underwater Object Detection Based on Deep Learning

Vision-based underwater object detection studies have been popular in [37]–[41] for the applications of exploring underwater resources and ocean environments. Similarly, with the rapid development of deep learning techniques, some deep learning-based underwater object detection techniques [42]–[51] have also been presented recently. For example, a fast R-CNN-based method was proposed in [44], aiming at the detection and recognition of fish species from underwater images. In addition, in [46], the deep residual network (ResNet) model [52] was used for classifying underwater images of plankton objects. Moreover, a lightweight deep neural network was presented in [47] for fish detection by using some building blocks, including concatenated ReLU [53], inception [54], and HyperNet [55]. Furthermore, a single-shot feature aggregation network for underwater object detection was proposed in [49] by introducing multiscale features and complementary context information. On the other hand, an underwater data set with annotated image sequences of fish, crabs, and starfish captured in brackish water with varying visibility was presented in [51]. The YOLOv2 [30] and YOLOv3 [31] deep models were fine-tuned and tested on the Brackish data set [51].

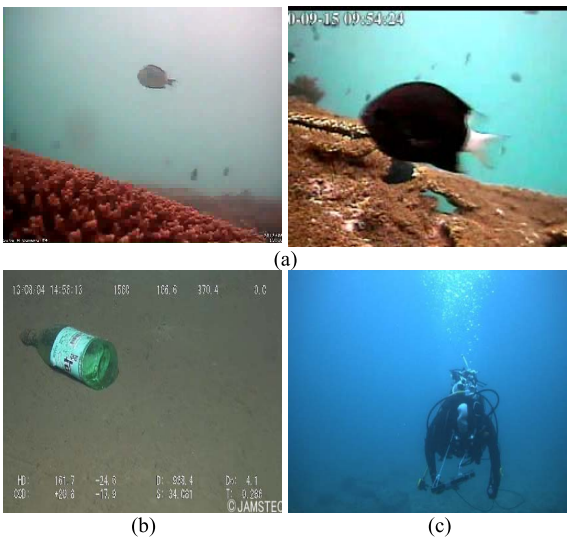

Fig. 1. Samples of underwater objects mainly considered to be detected in this article. (a) Fishes, (b) debris, and (c) diver (extracted from the underwater videos from [59]).

C. Major Contributions and Novelties of the Proposed Lightweight Deep Underwater Object Detection Network

In this article, a novel lightweight deep neural network model for underwater object detection is proposed. The major novelties and contributions of this article are threefold.

1) For the applications of battery-powered AUVs, this article presents a lightweight deep model for underwater object detection. Relying on our multiscale feature learning network, the proposed deep network has shown to be lightweight without significantly sacrificing object detection accuracy implemented on the Raspberry pi platform.
2) Different from the state-of-the-art underwater object detection frameworks, this article presents to jointly learn the process for image color conversion and object detection to solve the problem of underwater color distortion and scattering effects for improving detection performance.
3) Based on the fact that training samples for underwater images are hard to collect, this article proposes to generate training underwater images for well training the proposed deep object detection model.

The rest of this article is organized as follows. Section II presents the proposed method for generating training samples of underwater images for object detection deep model learning. In Section III, the proposed lightweight deep underwater object detection network is addressed. Section IV demonstrates the experimental results. Finally, Section V provides the conclusion.

II. Generation of Training Samples for Underwater Images

Based on the fact that it is not easy to find a suitable data set of underwater images [56]–[58], especially for the objects of fishes, debris, and divers (shown in Fig. 1), this

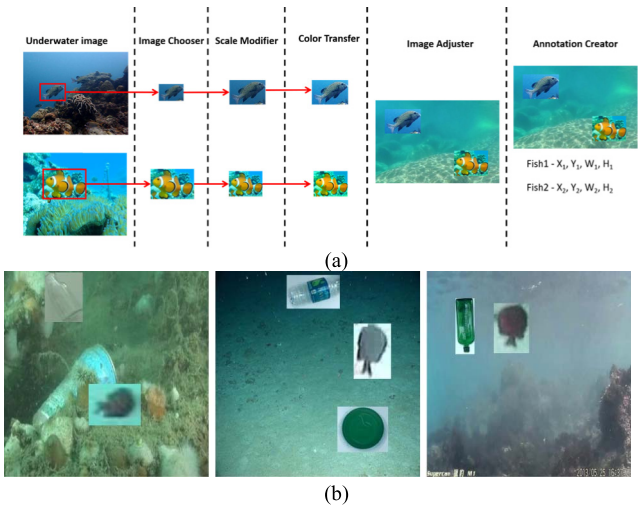

Fig. 2. (a) Proposed generation process for generating underwater training images with objects included for our object detection model learning. (b) Some generated underwater images based on the proposed generation process.

article proposes to generate the related sample images for our deep model learning. However, to evaluate the performance of the proposed method by comparing the state-of-the-art methods, more sample images (usually extracted from underwater videos) used in the literature are also included, as addressed in Section IV. A similar issue has also been mentioned in recent studies on underwater object detection or tracking (see [56]–[58]) that no such benchmark or data set exists so far for underwater object detection/tracking. Although some related data sets have been presented recently, the focuses of these data sets and our goal are different. For example, the data set presented in [56] is mainly collected for underwater object tracking for moving objects, while we also consider static objects (e.g., debris). In addition, the data set provided in [57] consists of the three categories of seacucumber, seaurchin, and scallop. The data set [57] is mainly collected for underwater robot picking, while we consider more diverse objects. Moreover, the data set proposed by Chen *et al.* [58] is mainly used for underwater image enhancement and evaluation of the enhancement performance based on object detection before and after enhancing images. The categories of objects in this data set were not clearly addressed in [58], and therefore, the data set cannot fit our purpose. Therefore, it is needed to create our own data set for underwater object detection.

Inspired by the idea that patches are cut and pasted among training images presented in [60], our training sample generation method for underwater image object detection is described as follows. As shown in Fig. 2(a), we collected several underwater images from the Internet to form the backgrounds of the images to be generated. The Image Chooser module randomly and manually picks out the objects of fishes, debris, or divers from the annotated image set [61] to be the object components (or foregrounds) for being embedded onto the background underwater images. To consider various scales of objects, the Scale Modifier module aims at modifying the scales and aspect ratios to form different versions (with different sizes) for each object.

Moreover, inspired by Jeong *et al.* [62], to naturally fuse the objects into the underwater background images, the color transfer module first transforms all of the object and background images from the RGB (red, green, and blue) color space to the HSI (hue, saturation, and intensity) color space [63]. Then, the hue values of the object images are adjusted to be similar to those of their targeting underwater background images, shown as follows. For an object image O to be fused into an underwater background image B, we just assign the average hue value ($\mathcal{H}_B$) of the hue values for all the pixels of B to the hue value of each ith pixel ($\mathcal{H}_O^i$) of O, i.e., we set $\mathcal{H}_O^i = \mathcal{H}_B$ for all i. The saturation and intensity values of O are kept unchanged. As a result, the object images can be fused with the background images. In addition, the Image Adjuster module is used to randomly adjust the locations in the background image for the objects (possibly rotated with some angles) to be pasted. Finally, the Annotation Creator is used to annotate each fused image with attached information, including the coordinates (X_i and Y_i) of the center point, the width (W_i), and the height (H_i) of each object i. Fig. 2(b) shows some generated underwater images obtained by the proposed generation process [see Fig. 2(a)] for our deep model learning.

In the proposed underwater training sample generation framework, each of the extracted objects is placed in the randomly selected position of the corresponding background image. Based on the fact that each underwater object can be viewed as a floater, it is reasonable to randomly place them in an underwater background image. In particular, several selected background images include coral reefs or seabeds. With randomly placed objects on them, it would be helpful to simulate complex underwater scenes. In addition, the main goal of the proposed training sample generation framework is to augment training samples for object detection based on YOLO-like object detectors, that is, the detection process relies on using a bounding box to describe each target object location. Even though the target box background of a placed object in a generated image does not blend well with the background image, the generated image with bounding box-bounded target object(s) still fit our training process for object detection. Moreover, based on our experiments (described in Section IV), the proposed data augmentation process indeed benefits the final detection results.

III. Proposed Lightweight Deep Underwater Object Detection Network

As shown in Fig. 3, the proposed lightweight deep neural network for underwater object detection mainly consists of the image color conversion network and the underwater object detection network. The main motivation and guideline for designing the network architecture of the proposed framework is addressed as follows. Since underwater images usually suffer from color distortion, the proposed color conversion network module is specifically designed to correct the color information of underwater images for better object detection. Moreover, the proposed object detection network mainly follows the general architecture of the FPNs presented in [34].

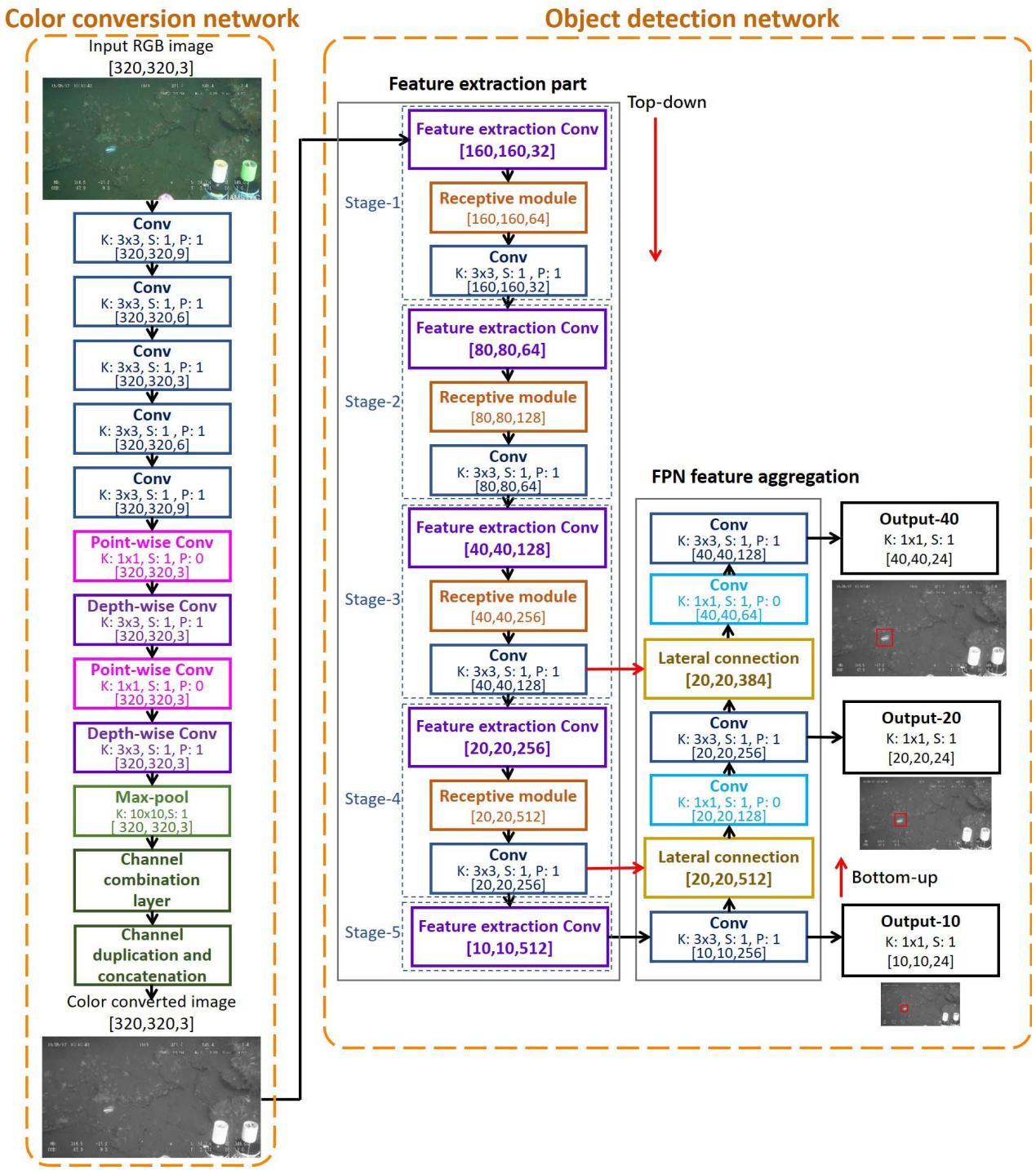

Fig. 3. Proposed lightweight deep neural network for underwater object detection, where the left and right parts are the color conversion network and the underwater object detection network, respectively. For each layer, $[W, H, C]$ means the size, including the width (W), the height (H), and the number of channels (C), of the output feature map from this layer. In addition, Conv, Max-pool, K, S, and P mean the convolution operation, the max-pooling operation, the kernel size, the stride size, and the padding size, respectively.

FPN has been shown to be a generic solution for building feature pyramids inside any deep CNNs for object detection. To meet the two requirements that the proposed architecture should be lightweight for the application of AUVs and the considered underwater scenario may include several small objects, we adopted our lightweight implementation by integrating the idea of FPN for multiscale prediction to form our object detection network. In addition, to embed underwater image characteristics into our method, we jointly learn the proposed color conversion network and the proposed object detection network for adjusting the color information of underwater images while achieving better object

detection performance. The two network modules shall be elaborated in Sections III-A and III-B.

A. Color Conversion Network Module in the Proposed Deep Model

The proposed image color conversion network aims at transforming an input color image to the corresponding grayscale image. We have shown that this conversion is beneficial to solve the problem of underwater color absorption to enhance the object detection performance with lower computational complexity.

As shown in the left part of Fig. 3, the image color conversion network takes an RGB underwater image as input and outputs the three parameters for color conversion. The three parameters (p_α, p_β, and p_γ) are used to transform the input RGB to its corresponding grayscale $_{\text{gray}}$, which is expressed by

$$I_{\text{Gray}} = \frac{p_\alpha}{p_\alpha + p_\beta + p_\gamma} \times I_R + \frac{p_\beta}{p_\alpha + p_\beta + p_\gamma} \times I_G + \frac{p_\gamma}{p_\alpha + p_\beta + p_\gamma} \times I_B \quad (1)$$

where I_R, I_G, and I_B, denote the red, green, and blue channels of the input, respectively, and the three parameters p_α, p_β, and p_γ, are learned by the proposed color conversion network.

More specifically, the proposed color conversion network is composed of ten lightweight convolution layers. The first five convolution layers aim at extracting features from the input image. The following convolution layers are designed based on the idea of the MobileNets relying on the depth-wise separable convolutions presented in [64]. The MobileNet model [64] relies on the depth-wise separable convolutions that factorize a standard convolution into a depth-wise convolution and a 1×1 convolution called a point-wise convolution. This factorization drastically reduces the computation complexity and the model size. The depth-wise convolution applies a single filter per each input channel, whereas the point-wise convolution uses a filter of kernel in size of 1×1 to nonlinearly combine the outputs of the depth-wise convolution. However, in our color conversion deep network, different from [64], the depth-wise separable convolution used here first applies a point-wise convolution of kernel in size of 1×1 to reduce the dimension of the output from the previous layer. Then, by applying a depth-wise convolution of kernel in size of 3×3, the feature extraction is performed for generating the transformation parameter of each color channel.

The depth-wise separable convolution layers are then followed by a max-pooling layer and a channel combination layer to produce the transformed grayscale image. The main goal for applying a max-pooling layer in the end of the proposed color conversion CNN module is to avoid possible noises in the feature learning process. In this layer, the kernel size and the stride size are set to 10×10 and 1, respectively. The max-pooling layer is useful for stabilizing the color conversion results. The channel combination layer is designed to perform the color transformation defined by (1). Then, as shown in Fig. 3, the "Channel duplication and concatenation" block duplicates the channel of the transformed grayscale image three times and concatenates them to form the output grayscale image of three channels. The grayscale image transformed from the input image is then fed into the object detection network module described in Section III-B.

To train the proposed color conversion network, the three different loss functions were adopted described as follows. Please note that the proposed color conversion network and the proposed object detection network (described later) are jointly trained. Based on the fact that no ground truth grayscale images are available for color images, the loss functions used here rely on the self-correlation of a generated grayscale image and the preservations of image feature and style for the generated grayscale image, compared with its color version. More specifically, to encourage the spatial smoothness in the generated image, the total variation (TV) loss function [65] is employed, which is defined by

$$\mathcal{L}_{\text{TV}}(\hat{y}) = \sum_{i,j} \sqrt{\left(\hat{y}_{i+1,j} - \hat{y}_{i,j}\right)^2 + \left(\hat{y}_{i,j+1} - \hat{y}_{i,j}\right)^2} \quad (2)$$

where $\hat{y}_{i,j}$ denotes the value of the pixel at the location (i, j) in the generated grayscale image $\hat{y}$.

Moreover, inspired by Johnson *et al.* [66], two perceptual loss functions are also employed. The first one is the feature reconstruction loss, which is defined by

$$\mathcal{L}_{\text{feature}}(y, \hat{y}) = \frac{1}{C \times H \times W} \|\phi(y) - \phi(\hat{y})\|_2^2 \quad (3)$$

where y and $\hat{y}$, respectively, denote the input color image and the generated grayscale image, where both of them are in size of $C \times H \times W$. C, H, and W represent the number of channels, the height, and the width of an image, respectively. For a generated grayscale image, the number of channels is also set to 3 by stacking the grayscale channel three times to be consistence with that of the input color image. $\phi(y)$ denotes a feature extractor for extracting the feature representation from the image y. Based on the suggestion mentioned in [66], the function ϕ is realized by the 16-layer VGG network [67] pretrained on the ImageNet data set [68]. The feature extractor ϕ is mainly used for image feature extraction included in the loss function [described in (5)] for training the proposed color conversion network. Based on the diversity of the ImageNet [68] for pretraining VGG-16 [67], used as the backbone of ϕ and our experiments (described in Section IV), the selection of ϕ is effective enough for jointly learning our deep model.

The second perceptual loss function used here is the style reconstruction loss based on the Gram matrix defined in [69], which is expressed by

$$\mathcal{L}_{\text{style}}(y, \hat{y}) = \|G(y) - G(\hat{y})\|_F^2 \quad (4)$$

where y and $\hat{y}$, respectively, denote the input color image and the generated grayscale image, $G(y)$ is the Gram matrix of y, and $\|\cdot\|_F^2$ represents the squared Frobenius norm. The Gram matrix has been shown to provide an efficient style representation for an image in [66]. As a result, the final loss function for training the proposed color conversion network is

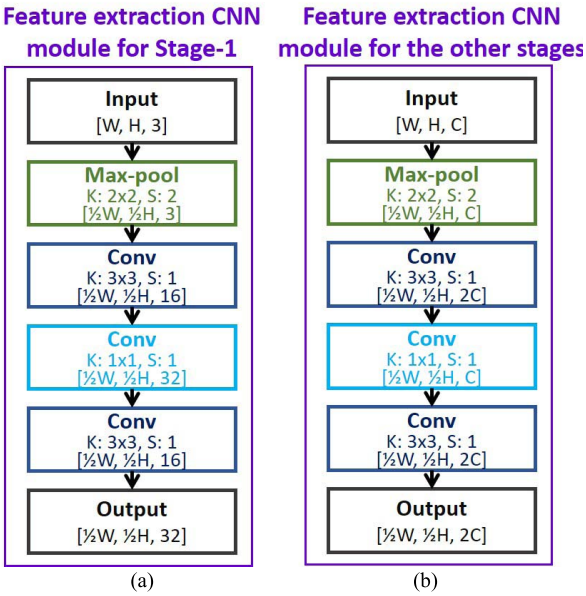

Fig. 4. Feature extraction CNN modules in the proposed feature extraction network (the "Feature extraction part" in Fig. 3). For each layer, $[W, H, C]$ means the size, including the width (W), the height (H), and the number of channels (C), of the output feature map from this layer. In addition, Conv, Max-pool, K, and S mean the convolution operation, the max-pooling operation, the kernel size, and the stride size, respectively. (a) Feature extraction CNN module used at Stage-1 of the "Feature extraction part" in Fig. 3. (b) Feature extraction CNN module used at the other stages of the "Feature extraction part" in Fig. 3. In this figure, different colors are used to represent convolution blocks of different kernel sizes, where deep blue and light blue, respectively, denote the kernel sizes of 3×3 and 1×1.

defined as

$$\begin{aligned}\mathcal{L}_{\text{Color\_Conversion}}(y, \hat{y}) = {} & \lambda_{\text{TV}} \times \mathcal{L}_{\text{TV}}(\hat{y}) \\ & + \lambda_{\text{feature}} \times \mathcal{L}_{\text{feature}}(y, \hat{y}) \\ & + \lambda_{\text{style}} \times \mathcal{L}_{\text{style}}(y, \hat{y}) \end{aligned} \tag{5}$$

where λ_{TV}, λ_{feature}, and λ_{style} are the weighting coefficients for the TV, feature reconstruction loss, and style reconstruction loss, respectively.

B. Object Detection Network Module in the Proposed Deep Model

In the proposed deep model, the object detection network module consists of the two submodules, named by the feature extraction network module and the feature aggregation network module, respectively. The two modules are addressed as follows.

1) Feature Extraction Network Module: The proposed feature extraction network module is designed to perform feature extraction from the output image of the color conversion network module for achieving object detection. As shown in the right of Fig. 3, the feature extraction network module is composed of the cascading of the feature extraction CNN modules (denoted by the "Feature extraction Conv" in Fig. 3), the receptive modules, and the convolution layers, elaborated as follows.

As shown in Fig. 4, the feature extraction CNN module consists of an input layer, followed by a max-pooling layer, three convolution layers, and an output layer. The feature extraction CNN module is mainly used to extract initial features with halving the spatial size of the input feature map. To significantly reduce the computational burden of the convolutional operations, only a few feature maps are used in our feature extraction CNN module. Moreover, as shown in Fig. 4, the feature extraction CNN module of the first stage adopts the 1×1 convolution operation with kernel in size of $1 \times 1 \times 32$ to double the channel size. The main goal is to increase the feature maps in the initial stage for the succeeding feature extraction process. On the other hand, for each feature extraction CNN module used in the following stages, the 1×1 convolution operation is used to halve the channel size to decrease the feature sizes and reduce the computational complexity. For example, the 1×1 convolution layer of the second stage adopts the convolution operation with kernel in size of $1 \times 1 \times 64$ to halve the feature channels obtained from the previous layer. To compensate for the possible problem of insufficiently extracted features, the receptive module is proposed to further refine the features from the previous feature extraction CNN module. The receptive module is used to efficiently extract multiscale features for the input feature map. The main goal is to keep the feature representational power in low computational complexity, detailed as follows. As shown in Fig. 5, the proposed receptive module consists of an expand layer, a squeeze layer, and a shortcut connection. The expand layer is based on the inception v3 architecture presented in [70]. The main idea of the inception v3 module is to expand the network scale relying on the added computation as efficiently as possible by suitably factorized convolutions, that is, spatial aggregation can be achieved over lower dimensional embeddings without significant loss in representational power. For example, in Fig. 5, the cascaded two 3×3 convolution operations are originally performed by a 5×5 convolution operation in the previous inception module [54]. It has been shown that such technique of factorization into smaller convolutions would be efficient to the reduction of computational complexity, that is, in our expand layer, we mainly borrowed the idea of factorization of convolution operations from the inception v3 [70]. The architecture of our expand layer is

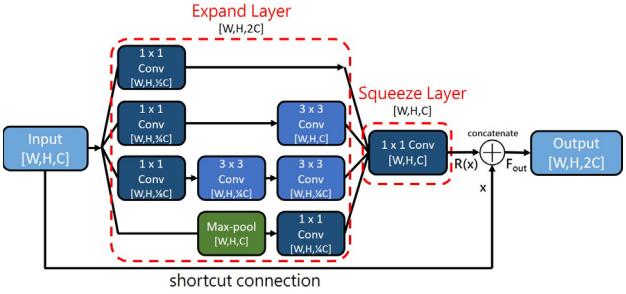

Fig. 5. Receptive module in the proposed feature extraction network module. For each layer, $[W, H, C]$ means the size, including the width (W), the height (H), and the number of channels (C), of the output feature map from this layer. In addition, Conv and Max-pool mean the convolution operation and the max-pooling operation, respectively. Moreover, x and $R(x)$ denote the input feature map and the output of the Squeeze layer for x, respectively. Finally, $F_{\text{out}} = x \oplus R(x)$, where $\oplus$ denotes the concatenation operation.

different from and simpler than that of the inception v3 (with more convolution layers of larger kernel sizes). In addition, expanding the network with multiscale processing is beneficial for extracting multiscale features, which would be useful for the detection of larger or smaller objects in an image. The expand layer is followed by a squeeze layer, which just performs one 1×1 convolution operation for decreasing the number of feature channels, inspired by Iandola *et al.* [71]. On the other hand, to preserve the input feature map of a receptive module and avoid the vanishing gradient problem, a shortcut connection is used to directly transmit the input to be concatenated with the output of the squeeze layer to form the output of the receptive module. Using the concatenation operation to form the output, instead of the element-wise addition used in ResNet [52], is mainly inspired by densely connected convolutional networks (DenseNets) [72], which has been shown to outperform ResNet in feature preservation. The output of the receptive module is then fed into a convolution operation with kernel in size of 3×3 and stride size of 1.

In the feature extraction network module, there are five cascaded stages. Each of the first four stages is composed of a feature extraction CNN, a receptive module, and a convolution layer. The fifth stage consists of only one feature extraction CNN. The outputs of the latter three stages will be fed into the feature aggregation network module, described as follows.

2) Feature Aggregation Network Module: The feature aggregation network module is designed based on the idea of the FPN architecture presented in [34] for connecting our feature extraction network module. The main idea of FPN is to use the inherently multiscale pyramidal hierarchy of deep CNN to establish feature pyramids with insignificant extra cost. The key property is to develop a top-down architecture with lateral connections for extracting high-level semantic feature maps at all scales. As shown in Fig. 3, in our feature aggregation network module, there are three connection points connected with our feature extraction network (of a top-down architecture). The feature aggregation network module is designed as a bottom-up architecture, consisting of two lateral connection blocks (as shown in Fig. 6) and several convolution layers. The lateral connection block is designed for merging the feature maps of the same spatial resolution from the bottom-up path (the feature aggregation network module) and the top-down path (the feature extraction network module).

As shown in Fig. 6, the lateral connection block consists of an unsampling layer, a concatenation layer for concatenating the upscaled feature map and the feature map from the feature extraction network module, and a convolution layer. In our implementation, the third, fourth, and fifth stages of the feature extraction network module will output the feature maps to the feature aggregation network module. The feature aggregation network will finally produce the three outputs at different levels for object detections of different scales, that is, based on the learned feature pyramid, the object predictions can be made independently at all levels.

3) Loss Function for Object Detection: The size of the output at each scale (total three scales) in our feature aggregation network module is denoted by $S \times S \times [(5 + N_C) \times N_B]$. In this

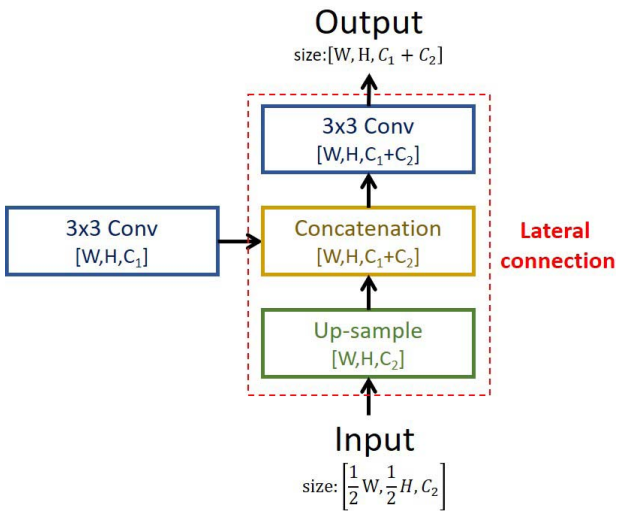

Fig. 6. Lateral connection block in the proposed feature aggregation network module. For each layer, $[W, H, C]$ means the size, including the width (W), the height (H), and the number of channels (C), of the output feature map from this layer. In addition, Conv denotes the convolution operation.

expression, $S \times S$ denotes the spatial size of the output feature map (or the number of grid cells used in YOLOv3 [31]), N_C denotes the number of considered underwater object classes, and N_B denotes the number of evaluated bounding boxes in each grid. In our experiments, N_C is set to 3 for the three object categories to be detected, and N_B is set to 3 based on YOLOv3 [31]. The term "5" denotes the four values of bounding box offsets and 1 score value of object confidence [31].

To train the proposed object detection network, the loss function used in YOLOv3 [31] is directly used, denoted by $\mathcal{L}_{\text{detection}}$. The loss function is designed subject to the four losses: 1) the loss of the four values of bounding box offsets: the MSE (mean squared error) for the coordinates of the center point, the width, and the height of each bounding box and the corresponding ground truths; 2) the loss of object confidence score of a bounding box; 3) the loss of nonobject confidence score of a bounding box; and 4) the loss of multiclass predictions of a bounding box. The latter three items are calculated by the binary cross-entropy loss, instead of the MSE. The loss function $\mathcal{L}_{\text{detection}}$ can be mathematically expressed by

$$\begin{aligned}
&\mathcal{L}_{\text{detection}} \\
&= \lambda_{\text{coord}} \sum_{i}^{S^2} \sum_{j}^{B} \left\{ 1_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right.\right. \\
&\qquad \left.\left. + (w_i - \hat{w}_i)^2 + \left(h_i - \hat{h}_i\right)^2 \right] \right\} \\
&\quad + \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{obj} \text{BCE}(c_i, \hat{c}_i) \\
&\quad + \lambda_{\text{noobj}} \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{\text{noobj}} BCE(c_i, \hat{c}_i) \\
&\quad + \sum_{i}^{S^2} 1_{i}^{\text{obj}} \sum_{c}^{C} \text{BCE}\left(P_i(c), \hat{P}_i(c)\right)
\end{aligned} \tag{6}$$

TABLE I
COLLECTED UNDERWATER IMAGE DATA SET USED FOR DEEP MODEL TRAINING, VALIDATION, AND TESTING IN THIS ARTICLE

Object Types	Source Datasets	Number of Images
Fishes	LifeCLEF 2014 (videos) [61]	5,200
Underwater debris	Deep-sea Debris Database (videos) [73]	1,000
Divers	CADDY Underwater Stereo-Vision Dataset (images) [74]	600
	ImageNet (images) [68]	1,600
Fishes / Underwater debris / Divers	Dataset generated by the proposed generation process (images, only for model training and validation, not for testing)	4,000
Total number of images		**12,400**
Training / validation		8,000 / 2,000
Testing		2,400

where λ_{coord} denotes the parameter controlling the weight of the coordinate loss of the bounding box, S^2 is the number of grid cells, B is the number of bounding boxes, 1_{ij}^{obj} denotes that the object is detected by the jth bounding box in the ith cell, (x_i, y_i) and $(\hat{x}_i, \hat{y}_i)$ are the ground truth of the center point and the predicted one, respectively, w_i and h_i and $\hat{w}_i$ and $\hat{h}_i$ are the ground truths of the width and the height of the bounding box and the correspondingly predicted ones, respectively, BCE denotes the binary cross-entropy function, expressed by $\text{BCE}(c_i, \hat{c}_i) = -c_i \log(\hat{c}_i) + (1 - c_i)\log(1 - \hat{c}_i)$, c_i and $\hat{c}_i$ are the confidence score of the object included in the ith cell and the correspondingly predicted one, respectively, λ_{noobj} denotes the parameter controlling the weight of the loss of nonobject confidence, 1_{ij}^{noobj} denotes that the object is not detected by the jth bounding box in the ith cell, 1_i^{obj} denotes that the object is detected in the ith cell, C is the total number of object classes, and $P_i(c)$ and $\hat{P}_i(c)$ are the ground truth of the probability for the object in the ith cell belonging to the cth class and the correspondingly predicted one, respectively.

Finally, to jointly train the proposed color conversion network and the proposed object detection network, the total loss function is expressed as

$$\mathcal{L}_{\text{Total}} = \lambda_{\text{TV}} \times \mathcal{L}_{\text{TV}} + \lambda_{\text{feature}} \times \mathcal{L}_{\text{feature}} + \lambda_{\text{style}} \times \mathcal{L}_{\text{style}} + \lambda_{\text{detection}} \times \mathcal{L}_{\text{detection}} \quad (7)$$

where the former three items are defined in (5) and the four weighting coefficients, λ_{TV}, λ_{feature}, λ_{style}, and $\lambda_{\text{detection}}$, are empirically set to 10, 1, 1, and 1, respectively. The selection of the weighting coefficient λ_{TV} is described in Section IV-E.

IV. EXPERIMENTAL RESULTS

A. Network Training and Parameter Settings

For training, validating, and testing the proposed lightweight deep underwater object detection network, other than our generated image data set, we also collected several underwater images, as summarized in Table I. In Table I, only video data can be available from some data sets, where some related video frames were selected to be included in our data set.

To create our training data set, we selected 8000 images for training and 2000 images for validation, as shown in Table I. In the training/validating phase, only the training/validating data from the respective data source were included in our training/validating set to ensure that the training/validating data and the testing data are split. Moreover, the class split of our data set for the categories of fishes, debris, and divers is approximately 20%, 50%, and 30% of the total number of images, respectively. Each image patch is in size of 320 × 320 and the batch size is set to 32. The learning rate in our experiments is initially set to 0.001 and decayed by 0.1 if the loss is not further reduced for each period of ten epochs. The early stop is triggered if the loss is not further reduced for a period of 20 epochs. In addition, the activation function used in the proposed deep model is the leaky rectified linear unit (Leaky ReLU) function [75], and the model parameters were initialized based on the weight initialization technique presented in [76]. Moreover, the loss function was minimized using the Adam optimization algorithm [77]. Based on our implementation with the above settings using the Keras [78] and the PyTorch [79] platforms, through 80 epochs, a well-trained converged deep network can be obtained. In addition, no pretrained model was used in our experiments.

B. Quantitative Results

The proposed method was implemented on a personal computer equipped with Intel Core i7-8700k processor, 32-GB memory, and NVIDIA GeForce GTX 1080 GPU. The processing speed of the proposed detection method achieves the frame rate of 150–170 images (in size of 320 × 320 for each) per second, which is extremely fast for object detection purposes.

However, the main goal of this article is to design a lightweight object detector, equipped on a battery-powered AUV, used in the underwater environment. To simulate this scenario, we also implemented the proposed deep learning-based object detector on the Raspberry Pi 3 model B+ with ARM Cortex-A53, 1.4 GHz CPU, Dual Core Multimedia Co-Processor GPU, 1-GB RAM, and 5.661-W maximum working power. To quantitatively evaluate the performance of the proposed lightweight underwater object detection deep network, the four state-of-the-art deep learning-based object detection frameworks were used for comparisons. They are the Faster R-CNN [27], SSD [33], YOLO [29], and Tiny-YOLO [29]. The backbones used for Faster R-CNN, SSD, YOLOv2, and YOLOv3 frameworks in our experiments are VGG-16 [67], ResNet-101 [52], Darknet-19 [80], and Darknet-53 [80], respectively. The same data augmentation technique (by the proposed training sample generation method) was used for training all the evaluated deep models for object detection. The

TABLE II

QUANTITATIVE PERFORMANCE EVALUATIONS CONDUCTED ON THE RASPBERRY PI IN TERMS OF THE mAP FOR OBJECT DETECTIONS, GFLOPS, PROCESSING TIME (IN SECONDS) PER IMAGE, AND FPS FOR THE FOUR STATE-OF-THE-ART METHODS AND THE PROPOSED METHOD ON OUR COLLECTED DATA SET DESCRIBED IN TABLE I, WHERE "-" MEANS THAT THE CORRESPONDING EVALUATED MODEL IS TOO COMPLEX TO BE EMBEDDED INTO THE USED RASPBERRY PI FOR PROCESSING MULTIPLE SUCCESSIVE FRAMES

Evaluated Methods	Faster R-CNN [27]	SSD [33]	YOLO [29]	Tiny-YOLO [29]	Proposed
mAP	78.62%	82.34%	89.99%	89.81%	89.56 %
GFLOPs	43.82	34.334	38.22	8.225	**5.06**
Processing Time per Image (in seconds)	1055.87	55.73	65.22	30.88	**3.02**
FPS	-	-	-	-	**0.5~1**

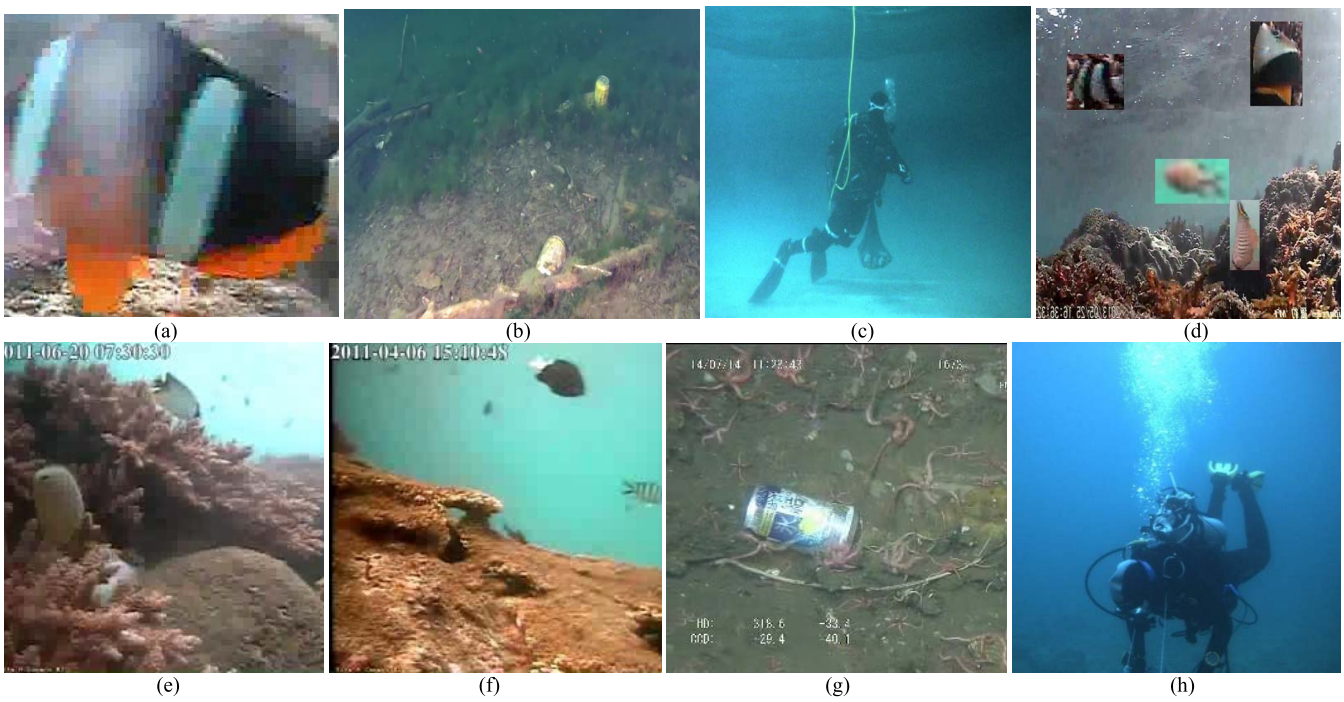

Fig. 7. Samples of training/testing underwater images containing objects of fishes, debris, and divers, described in Table I. (a)–(d) Training images, where (d) is generated by our image generator just for augmenting training samples and (e)–(h) testing images.

four compared methods were also implemented on the same platform and evaluated with the proposed method, in terms of the mean of average precision (mAP) for object detections, required (giga floating-point operations (GFLOPs) [81], average processing time (in seconds) per image, and FPS (average number of processed frames per second, i.e., frame rate), for system performance assessment. The mAP metric used in our experiments is calculated by setting the intersection over union (IoU) threshold to 0.5. This setting follows the primary metric used in the evaluation of the PASCAL VOC data set [82] and also used as another metric in evaluating the Brackish data set [51]. Table II shows the experimental results conducted on the 2400 testing underwater images (in size of 320 × 320 for each) containing the three classes of objects, i.e., fishes, underwater debris, and divers (described in Table I, as some samples shown in Fig. 7). As shown in Table II, the proposed method outperforms or is comparable with the four methods [27], [29], [33] used for comparisons in terms of mAP (the most popular metric in assessing the accuracy of object detectors). However, in the viewpoint of computational complexity, the proposed method significantly outperforms the four state-of-the-art methods in terms of the GFLOPs, processing time per image, and FPS metrics. It should be noted that the Faster R-CNN [27], SSD [33], YOLO [29], and Tiny-YOLO [29] network models are too complex to be embedded into the used Raspberry Pi platform for processing multiple successive frames based on that loading such complex model will consume most of the system resources. They can usually process one frame and the system would be out of resource when processing the next frame. Therefore, it is hard to report the FPS of the compared models in Table II based on the used Raspberry Pi platform.

Moreover, the main reason for that SSD [33] outperforms Faster R-CNN [27] in our experiments (based on Table II)

TABLE III
QUANTITATIVE PERFORMANCE EVALUATIONS CONDUCTED ON THE RASPBERRY PI IN TERMS OF THE mAP FOR OBJECT DETECTIONS, GFLOPS, AND PROCESSING TIME (IN SECONDS) PER IMAGE FOR THE THREE STATE-OF-THE-ART METHODS AND THE PROPOSED METHOD ON THE BRACKISH DATA SET [51]

Evaluated Methods	Faster R-CNN [27]	YOLOv2 [30]	YOLOv3 [31]	Proposed
mAP	75.69%	33.55%	82.03%	80.12%
GFLOPs	44.01	17.22	38.45	**5.06**
Processing Time per Image (in seconds)	> 1000	35.86	67.08	**3.28**

should be related to the used underwater image data set. The collected underwater image data set includes many images containing small objects to be detected. Based on our experiments conducted on our underwater image data set, SSD outperforms Faster R-CNN in terms of mAP. Similar results were also found in the literature, such as [28] and [33]. On the other hand, YOLOv3 achieves the best performance in the experiments based on its multiscale prediction mechanism. It was also shown that YOLOv3 exhibits better performance for detecting smaller objects in the literature, see [31], [32]. The proposed method also applies multiscale prediction for object detection. Since the proposed framework is designed in a lightweight manner, the performance is therefore slightly worse than those of YOLO and Tiny-YOLO in Table II and slightly worse than that of YOLOv3 in Table III. However, the proposed framework is with significantly lower network complexity than these compared architectures.

In addition, we also compared the network complexity of the proposed deep model with other lightweight deep model designed for fish detection (see [47]). The lightweight deep model presented in [47] relies on Faster R-CNN [27]. It was reported in [47] that the run time per image (0.089 s) of this network is just slightly faster than that (0.102 s) of faster R-CNN on the platform equipped with two high-end cards of NVIDIA Tesla K20. However, the main goal of this article is to design a lightweight deep model to be embedded into an AUV. Based on Table II, on the (low-end) Raspberry Pi platform, the run time of faster R-CNN is significantly higher than that of the proposed method. Therefore, it is expected that the network complexity of the proposed deep model should be lower than that of the faster R-CNN-based model presented in [47].

Moreover, other than the reported GFLOPs and processing time of the proposed method, the lightweightness of the proposed deep model mainly relies on the specific design described as follows. In the feature extraction CNN module of the proposed object detection network, the max-pooling operation is used to reduce the spatial dimension of feature maps. In addition, based on the stage-by-stage manner of the feature extraction part (see Fig. 3), the feature map size can be greatly reduced. Moreover, for a larger feature map, a smaller number of channels are used for output. On the other hand, the receptive module (also used in the feature extraction part) applies the squeeze layer to reduce the channel size of the expand layer output for efficient network computation.

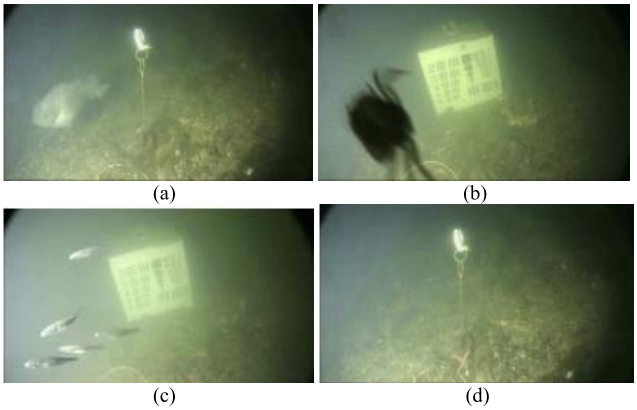

Fig. 8. Samples of underwater objects from the Brackish data set released by Pedersen *et al*. [51] used in our experiment described in Table III. (a) Big fish, (b) crab, (c) small fish, and (d) starfish.

On the other hand, the Brackish data set with the object detection results using YOLOv2 [30] and YOLOv3 [31] was presented in [51]. The Brackish data set (also employed for performance evaluation in this study) consists of annotated image sequences of fish, crabs, and starfish captured in brackish water with varying visibility. More specifically, the videos in the Brackish data set were categorized and manually annotated, resulting in a total of 14 518 frames with 25 613 annotations of the six classes, i.e., big fish, small fish, crab, jellyfish, shrimp, and starfish. Based on the fact that the numbers of the images, including the objects of jellyfish and shrimp, are too few, the two classes are ignored in our experiment. Some samples of the four categories considered in this experiment are shown in Fig. 8. Following the experimental settings in [51], the data set is split into 80% data for training, 10% data for validation, and 10% data for testing. The data split of the Brackish data set is provided by the original paper [51] and the data released from [83]. In our experiments, the input image size is set to 320×320. Table III reports the quantitative results conducted on the Brackish data set, where the Faster R-CNN [27], YOLOv2 [30], and YOLOv3 [31], and proposed methods were evaluated on the Raspberry Pi 3 model B+ (described earlier). As revealed in Table III, the proposed method still outperforms or is comparable with the three methods [27], [30], [31] used for comparisons in terms of mAP. The mAP performances obtained by our experiments for the YOLOv2 [30] and YOLOv3 [31] are also consistent

TABLE IV

QUANTITATIVE RESULTS IN TERMS OF THE mAP, GFLOPS, AND THE NUMBER OF TRAINING EPOCHS FOR THE ABLATION STUDY BY EVALUATING THE APPROACHES OF THE PROPOSED W/O COLOR CONVERSION, THE PROPOSED WITH STANDARD COLOR CONVERSION, AND THE COMPLETE PROPOSED FRAMEWORK

Evaluated Methods	Proposed w/o Color Conversion	Proposed with Standard Color Conversion	Proposed
mAP	87.8	84.43 %	**89.56 %**
GFLOPs	4.88	5.02	5.06
Number of Training Epochs	107	81	**76**

with those reported in [51]. In addition, the proposed method significantly outperforms the three state-of-the-art methods in terms of the GFLOPs and processing time per image.

C. Ablation Study for Evaluation of Color Conversion

To evaluate the contribution of the proposed color conversion network to the overall object detection performance, an ablation study was conducted as follows. In this study, the following three approaches were evaluated. The first is to only use the proposed object detection network without incorporating the proposed color conversion network (denoted by proposed w/o color conversion). The second is to employ the color conversion relying on the standard color space [84] as a preprocessing stage to transform an input color image to the corresponding grayscale version (denoted by proposed with standard color conversion). The employed standard color conversion is based on the standard red green blue (sRGB) color space [84] and the gamma expansion (linearization) to transform the image to a linear RGB space. Thus, the appropriate weighting coefficients can be applied to the linear color components, R, G, and B, to calculate the linear luminance (grayscale) component [85]. Moreover, the third approach is the complete proposed deep framework, including joint learning of color conversion and object detection (denoted by proposed). Table IV shows the quantitative results (also based on the data set described in Table I) of the three evaluated approaches in terms of the mAP, GFLOPs, and the number of training epochs. The number of epochs for each evaluated method was obtained when researching the minimum validation error. As observed from Table IV, the complete proposed deep model (with a fewer number of training epochs and slightly higher required GFLOPs) outperforms the approaches of the proposed method without color conversion and the proposed method with standard color conversion in terms of the mAP for object detection. The main reason is that the proposed method (with our color conversion) simultaneously learning the color conversion task and the underwater object detection task would obtain better conversed grayscale images than those obtained by the standard color conversion method. The better transformed grayscale images with less possible noises and better grayscale values (e.g., better contrast) would benefit the feature learning for the object detection process. In addition, the GFLOPs of the proposed color conversion network are only 0.18.

Moreover, some examples of converted grayscale images (with highlighted regions) obtained by the standard color conversion method [85] and the proposed color conversion network are shown in Fig. 9. As a result, the proposed deep model for joint learning of color conversion and object detection is indeed beneficial to underwater object detection.

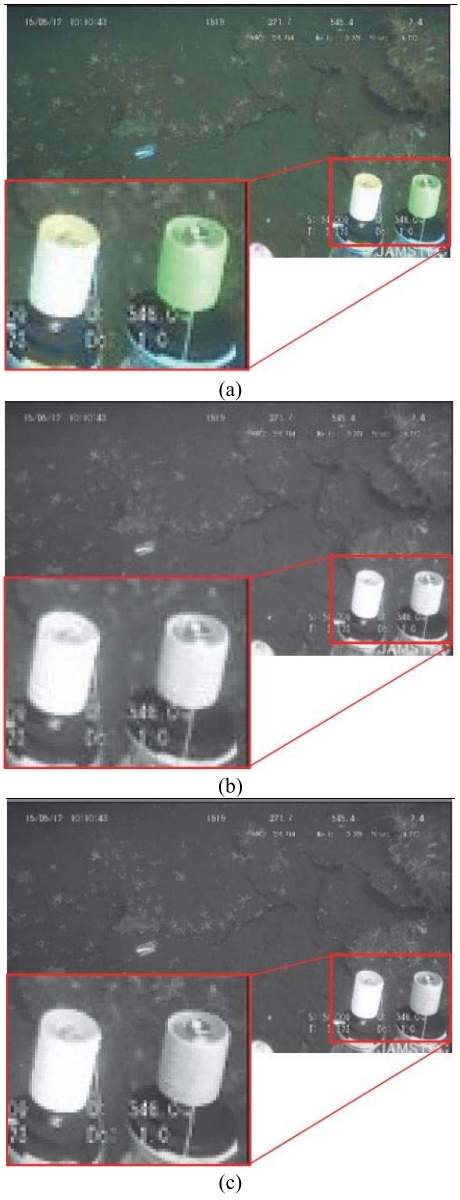

Fig. 9. Examples of color conversion (a) original color image and the converted grayscale images of (a) obtained by (b) standard color conversion method [85] and (c) proposed color conversion network.

TABLE V
ABLATION STUDY RESULTS IN TERMS OF THE mAP FOR OBJECT DETECTION BY EVALUATING THE APPROACHES OF THE PROPOSED w/o GENERATED SAMPLES AND THE COMPLETE PROPOSED FRAMEWORK

Evaluated Methods	Proposed w/o Generated Samples	Proposed
mAP	86.13 %	**89.56 %**

TABLE VI
ABLATION STUDY RESULTS IN TERMS OF THE mAP FOR OBJECT DETECTION BASED ON DIFFERENT WEIGHTING VALUES OF λ_{TV} USED IN THE LOSS FUNCTION OF THE PROPOSED METHOD

Values of λ_{TV}	mAP
1	86.04%
5	87.69%
10	**89.56%**
50	86.23%
100	84.66%

D. Ablation Study for Evaluation of Training Sample Generation

To evaluate the effectiveness of the proposed training sample generation framework, we also conducted an ablation study described as follows. We trained another object detector (denoted by proposed w/o generated samples) by using only the training samples described in Table I, excluding the generated training images. To train the deep model without the generated training samples, we followed the same parameter settings as the proposed deep model with generated training samples described in Section IV-A. Table V presents the mAP performances for object detection obtained by the approaches of Proposed w/o generated samples and the complete proposed framework (with generated training samples). It can be observed from Table V that artificially augmenting training samples indeed result in better object detection performance. The main reason can be described as follows. In the proposed training sample generation process, many objects of the three categories (e.g., fishes, debris, and divers) mainly considered in our object detection process are selected with different underwater scenes (backgrounds) to form our training images. Based on this data augmentation process, the proposed deep model would be more adaptable to diverse underwater scenes for accurately detecting the interested objects than other models without using the data augmentation process.

E. Selection of Weighting Coefficient for Total Variation Term in the Loss Function

To empirically decide the best weighting coefficient λ_{TV} for the TV term used in the loss function defined by (7), we conducted the following ablation study based on our collected data set described in Table I. We used different values for λ_{TV} with fixing all the values of λ_{feature}, λ_{style}, and $\lambda_{\text{detection}}$ to 1 and reported the corresponding detection accuracies in Table VI. Based on Table VI, λ_{TV} is empirically set to 10 in all the experiments of this study.

V. CONCLUSION

In this article, we have proposed an end-to-end lightweight underwater object detection deep neural network based on joint learning of color conversion and object detection. The learned image color conversion module aims at transforming color images to the corresponding grayscale images to solve the problem of underwater color absorption to enhance the object detection performance with lower computational complexity. The presented ablation study has shown the effectiveness of the proposed color conversion network, contributing to the overall object detection performance. Experimental results conducted on the low-power Raspberry Pi device have justified the effectiveness of the proposed lightweight jointly learning model for underwater object detection compared with the state-of-the-art approaches.

REFERENCES

[1] M. Han, Z. Lyu, T. Qiu, and M. Xu, "A review on intelligence dehazing and color restoration for underwater images," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 5, pp. 1820–1832, May 2020.

[2] J. Y. Chiang and Y.-C. Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1756–1769, Apr. 2012.

[3] C.-Y. Li, J.-C. Guo, R.-M. Cong, Y.-W. Pang, and B. Wang, "Underwater image enhancement by dehazing with minimum information loss and histogram distribution prior," *IEEE Trans. Image Process.*, vol. 25, no. 12, pp. 5664–5677, Dec. 2016.

[4] Y.-T. Peng and P. C. Cosman, "Underwater image restoration based on image blurriness and light absorption," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1579–1594, Apr. 2017.

[5] C. O. Ancuti, C. Ancuti, C. De Vleeschouwer, and P. Bekaert, "Color balance and fusion for underwater image enhancement," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 379–393, Jan. 2018.

[6] S.-B. Gao, M. Zhang, Q. Zhao, X.-S. Zhang, and Y.-J. Li, "Underwater image enhancement using adaptive retinal mechanisms," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5580–5595, Nov. 2019.

[7] D. Akkaynak and T. Treibitz, "A revised underwater image formation model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 6723–6732.

[8] D. Akkaynak and T. Treibitz, "Sea-thru: A method for removing water from underwater images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 1682–1691.

[9] D. Akkaynak and T. Treibitz, "First drain, then train: Unleash the full potential of AI on large underwater datasets (after removing the water from them)," in *Proc. Ocean Sci. Meeting*, San Diego, CA, USA, Feb. 2020. [Online]. Available: https://www.agu.org/-/media/Files/OSM2020/AGU_OSM_20_Thursday_program.pdf

[10] R. B. Wynn *et al.*, "Autonomous underwater vehicles (AUVs): Their past, present and future contributions to the advancement of marine geoscience," *Mar. Geol.*, vol. 352, pp. 451–468, Jun. 2014.

[11] A. B. Phillips, M. Haroutunian, A. J. Murphy, S. W. Boyd, J. I. R. Blake, and G. Griffiths, "Understanding the power requirements of autonomous underwater systems, Part I: An analytical model for optimum swimming speeds and cost of transport," *Ocean Eng.*, vol. 133, pp. 271–279, Mar. 2017.

[12] A. Bereketli, I. Yazgi, B. Yeni, and M. Koseoglu, "Battery-powered AUV network lifetime under energy constraints," in *Proc. OCEANS-Aberdeen*, Aberdeen, U.K., Jun. 2017, pp. 1–5.

[13] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," 2019, *arXiv:1905.05055*. [Online]. Available: http://arxiv.org/abs/1905.05055

[14] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, May 2004.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.

[16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[17] Q. Wang, M. Chen, F. Nie, and X. Li, "Detecting coherent groups in crowd scenes by multiview clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 1, pp. 46–58, Jan. 2020.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[19] C.-H. Yeh, C.-H. Huang, and L.-W. Kang, "Multi-scale deep residual learning-based single image haze removal via image decomposition," *IEEE Trans. Image Process.*, vol. 29, pp. 3153–3167, 2020.

[20] C.-H. Yeh, M.-H. Lin, P.-C. Chang, and L.-W. Kang, "Enhanced visual attention-guided deep neural networks for image classification," *IEEE Access*, vol. 8, pp. 163447–163457, 2020.

[21] C.-Y. Lin, Z. Tao, A.-S. Xu, L.-W. Kang, and F. Akhyar, "Sequential dual attention network for rain streak removal in a single image," *IEEE Trans. Image Process.*, vol. 29, pp. 9250–9265, 2020.

[22] L. Liu *et al.*, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Oct. 2020.

[23] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *Proc. Eur. Conf. Comput. Vis.* Heidelberg, Germany: Springer, 2014, pp. 346–361.

[26] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[27] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[28] T. Zhou, Z. Li, and C. Zhang, "Enhance the recognition ability to occlusions and small objects with robust faster R-CNN," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 11, pp. 3155–3166, Nov. 2019.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[30] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[31] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[32] Q.-C. Mao, H.-M. Sun, Y.-B. Liu, and R.-S. Jia, "Mini-YOLOv3: Real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133529–133538, 2019.

[33] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Heidelberg, Germany: Springer, 2016, pp. 21–37.

[34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[35] Y. Yuan, Z. Xiong, and Q. Wang, "VSSA-NET: Vertical spatial sequence attention network for traffic sign detection," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3423–3434, Jul. 2019.

[36] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[37] D. Walther, D. R. Edgington, and C. Koch, "Detection and tracking of objects in underwater video," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun./Jul. 2004, pp. 544–549.

[38] D. Lee, G. Kim, D. Kim, H. Myung, and H.-T. Choi, "Vision-based object detection and tracking for autonomous navigation of underwater robots," *Ocean Eng.*, vol. 48, pp. 59–68, Jul. 2012.

[39] S.-H. Cho, H.-K. Jung, H. Lee, H. Rim, and S. K. Lee, "Real-time underwater object detection based on DC resistivity method," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6833–6842, Nov. 2016.

[40] H. Ghafoor and Y. Noh, "An overview of next-generation underwater target detection and tracking: An integrated underwater architecture," *IEEE Access*, vol. 7, pp. 98841–98853, Jul. 2019.

[41] Z. Chen, H. Gao, Z. Zhang, H. Zhou, X. Wang, and Y. Tian, "Underwater salient object detection by combining 2D and 3D visual features," *Neurocomputing*, vol. 391, pp. 249–259, May 2020.

[42] M. Moniruzzaman, S. M. S. Islam, M. Bennamoun, and P. Lavery, *Deep Learning on Underwater Marine Object Detection: A Survey* (Lecture Notes in Computer Science), vol. 10617. Heidelberg, Germany: Springer, 2017.

[43] H. Qin, X. Li, Z. Yang, and M. Shang, "When underwater imagery analysis meets deep learning: A solution at the age of big visual data," in *Proc. OCEANS-MTS/IEEE Washington*, Washington, DC, USA, Oct. 2015, pp. 1–5.

[44] X. Li, M. Shang, H. Qin, and L. Chen, "Fast accurate fish detection and recognition of underwater images with fast R-CNN," in *Proc. OCEANS-MTS/IEEE Washington*, Washington, DC, USA, Oct. 2015, pp. 1–5.

[45] X. Li, M. Shang, J. Hao, and Z. Yang, "Accelerating fish detection and recognition by sharing CNNs with objectness learning," in *Proc. OCEANS-Shanghai*, Shanghai, China, Apr. 2016, pp. 1–5.

[46] X. Li and Z. Cui, "Deep residual networks for plankton classification," in *Proc. OCEANS MTS/IEEE Monterey*, Monterey, CA, USA, Sep. 2016, pp. 1–4.

[47] X. Li, Y. Tang, and T. Gao, "Deep but lightweight neural networks for fish detection," in *Proc. OCEANS-Aberdeen*, Aberdeen, U.K., Jun. 2017, pp. 1–5.

[48] S. Pelletier, A. Montacir, H. Zakari, and M. Akhloufi, "Deep learning for marine resources classification in non-structured scenarios: Training vs. Transfer learning," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, Quebec City, QC, Canada, May 2018, pp. 1–4.

[49] L. Zhang, X. Yang, Z. Liu, L. Qi, H. Zhou, and C. Chiu, "Single shot feature aggregation network for underwater object detection," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Beijing, China, Aug. 2018, pp. 1906–1911.

[50] P. W. Patil, O. Thawakar, A. Dudhane, and S. Murala, "Motion saliency based generative adversarial network for underwater moving object segmentation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 1565–1569.

[51] M. Pedersen, J. B. Haurum, R. Gade, T. B. Moeslund, and N. Madsen, "Detection of marine animals in a new underwater dataset with varying visibility," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Jun. 2019, pp. 18–26.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[53] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *Proc. 33rd Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 2217–2225.

[54] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[55] T. Kong, A. Yao, Y. Chen, and F. Sun, "HyperNet: Towards accurate region proposal generation and joint object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 845–853.

[56] L. Kezebou, V. Oludare, K. Panetta, and S. S. Agaian, "Underwater object tracking benchmark and dataset," in *Proc. IEEE Int. Symp. Technol. Homeland Secur. (HST)*, Woburn, MA, USA, USA, Nov. 2019.

[57] Z. Wang *et al.*, "UDD: An underwater open-sea farm object detection dataset for underwater robot picking," 2020, *arXiv:2003.01446*. [Online]. Available: http://arxiv.org/abs/2003.01446

[58] L. Chen *et al.*, "A benchmark dataset for both underwater image enhancement and underwater object detection," 2020, *arXiv:2006.15789*. [Online]. Available: http://arxiv.org/abs/2006.15789

[59] R. B. Fisher, Y.-H. Chen-Burger, D. Giordano, L. Hardman, and F.-P. Lin, *Fish4Knowledge: Collecting and Analyzing Massive Coral Reef Fish Video Data*. Cham, Switzerland: Springer, 2016.

[60] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proc. IEEE Int. Conf. Comput. Vis.*, Seoul, South Korea, Oct. 2019, pp. 6022–6031.

[61] A. Joly *et al.*, *LifeCLEF 2014: Multimedia Life Species Identification Challenges* (Lecture Notes in Computer Science), vol. 8685. Cham, Switzerland: Springer, 2014.
[62] H.-J. Jeong, K.-S. Park, and Y.-G. Ha, "Image preprocessing for efficient training of YOLO deep learning networks," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Jan. 2018, pp. 635–637.
[63] A. R. Weeks and G. E. Hague, "Color segmentation in the HSI color space using the K-means algorithm," *Proc. SPIE*, vol. 3026, Apr. 1997, Art. no. 271117.
[64] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861
[65] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5188–5196.
[66] J. Johnson, A. Alahi, and F.-F. Li, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, vol. 2, 2016, pp. 694–711.
[67] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015. [Online]. Available: https://dblp.org/rec/journals/corr/SimonyanZ14a.html
[68] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[69] L. A. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Proc. Neural Inf. Process. Syst.*, 2015, pp. 262–270.
[70] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826.
[71] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360
[72] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.
[73] *JAMSTEC Deep-Sea Debris Database*. Accessed: Jan. 24, 2021. [Online]. Available: http://www.godac.jamstec.go.jp/catalog/dsdebris/e/
[74] C. A. Gomez, A. Ranieri, D. Chiarella, E. Zereik, A. Babić, and A. Birk, "CADDY underwater stereo-vision dataset for human–robot interaction (HRI) in the context of diver activities," *J. Mar. Sci. Eng.*, vol. 7, no. 1, p. 16, Jan. 2019.
[75] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. Int. Conf. Mach. Learn.*, Atlanta, GA, USA, Jun. 2013, pp. 1–6.
[76] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1026–1034.
[77] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learning Represent.*, May 2015. [Online]. Available: https://dblp.org/rec/journals/corr/KingmaB14.html
[78] F. Chollet. *Keras: The Python Deep Learning Library*. Accessed: Jun. 25, 2020. [Online]. Available: https://keras.io
[79] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. Annu. Conf. Neural Inf. Process. Syst. Workshop*, 2017. [Online]. Available: https://www.bibsonomy.org/bibtex/21530dd0202e55d3eb1ada151e09c499
[80] J. Redmon. (2016). *Darknet: Open Source Neural Networks in C*. [Online]. Available: http://pjreddie.com/darknet/
[81] M. Cloutier, C. Paradis, and V. Weaver, "A raspberry pi cluster instrumented for fine-grained power measurement," *Electronics*, vol. 5, no. 4, p. 61, Sep. 2016.
[82] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015.
[83] (2020). *The Brackish Dataset-Bounding Box Annotated Underwater Image Dataset*. [Online]. Available: https://www.kaggle.com/aalborguniversity/brackish-dataset
[84] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta. (Nov. 1996). *A Standard Default Color Space for the Internet–sRGB*. Version 1.10. [Online]. Available: https://www.w3.org/Graphics/Color/sRGB
[85] *Parameter Values for the HDTV Standards for Production and International Programme Exchange*, document ITU-R BT.709-6, Jun. 2015.

Chia-Hung Yeh (Senior Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Electrical Engineering, National Chung Cheng University, Chiayi, Taiwan, in 1997 and 2002, respectively.

He was an Assistant Professor from 2007 to 2010, an Associate Professor from 2010 to 2013, and a Professor from 2013 to 2017 with the Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung City, Taiwan. He is currently a Distinguished Professor with National Taiwan Normal University, Taipei, Taiwan. He has coauthored more than 250 technical international conferences and journal articles and held 47 patents in USA, Taiwan, and China. His research interests include deep learning, video coding, 3-D reconstruction, and image/video processing.

Dr. Yeh became a fellow of IET in 2017. He was a recipient of the 2013 IEEE MMSP Top 10% Paper Award, the 2014 IEEE GCCE Outstanding Poster Award, the 2015 APSIPA Distinguished Lecturer, the 2017 IEEE SPS Tainan Section Chair, the 2017 Distinguished Professor Award of NTNU, and the IEEE Outstanding Technical Achievement Award (IEEE Tainan Section). He was the Associate Editor of the *Journal of Visual Communication and Image Representation*, *EURASIP Journal on Advances in Signal Processing*, and *International Journal of Pattern Recognition and Artificial Intelligence.*

Chu-Han Lin received the B.S. degree from the Department of Electrical Engineering, Feng Chia University, Taichung, Taiwan, in 2018, and the M.S. degree from the Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan, in 2020.

His current research interests include deep learning for computer vision, deep learning algorithm acceleration, and multimedia applications.

Li-Wei Kang (Member, IEEE) received the Ph.D. degree in computer science from National Chung Cheng University, Chiayi, Taiwan, in 2005.

He was an Associate Professor from 2016 to 2019 and an Assistant Professor from 2013 to 2016 with the Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, Douliu, Taiwan. Since 2019, he has been an Associate Professor with the Department of Electrical Engineering, National Taiwan Normal University, Taipei, Taiwan. His research interests include multimedia content analysis and multimedia communications.

Dr. Kang was a member of the Multimedia Systems and Applications Technical Committee and the IEEE CAS Society. He received the Top 10% Paper Award from the IEEE MMSP 2013, the Best Performance Award from the Social Media Prediction Challenge of the ACM MM 2019, and the Best Paper Award of the APSIPA ASC 2020.

Chih-Hsiang Huang received the B.S. and M.S. degrees from the Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung City, Taiwan, in 2016 and 2018, respectively.

His research interests are deep learning for image processing, computer vision, and multimedia applications.

Min-Hui Lin received the B.S. degree in electrical engineering from National Kaohsiung University, Kaohsiung City, Taiwan, in 2016, and the Ph.D. degree in electrical engineering from National Sun Yat-sen University, Kaohsiung City, in 2021.

She is currently a Senior Engineer with Qualcomm Inc., Hsinchu, Taiwan. Her research interests are in the areas of deep learning for computer vision, 3-D reconstruction, and multimedia applications.

Dr. Lin received the Best Oral Presentation Award from the 2018 IEEE Asia-Pacific Conference on Circuits and Systems and the Second Best Paper Award from the 2020 International Computer Symposium.

Chuan-Yu Chang (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from National Cheng Kung University, Tainan, Taiwan, in 2000.

He is currently the Deputy General Director of the Service Systems Technology Center, Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan. He is also a Distinguished Professor with the Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology (YunTech), Douliu, Taiwan. His current research interests include computational intelligence and their applications to medical image processing, automated optical inspection, emotion recognition, and pattern recognition. In the above areas, he has more than 200 publications in journals and conference proceedings.

Dr. Chang is a fellow of IET and a Life Member of IPPR and TAAI. From 2015 to 2017, he was the Chair of the IEEE Signal Processing Society Tainan Chapter and the Representative for Region 10 of IEEE SPS Chapters Committee. He is currently the President of the Chinese Image Processing and Pattern Recognition Society.

Chua-Chin Wang (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the State University of New York (SUNY) at Stony Brook, Stony Brook, NY, USA, in 1992.

He was a CEO of the Operation Center of Industry—University Cooperation, National Sun Yat-sen University (NSYSU), from 2012 to 2014. He has been the Director of Underwater Vehicle Research and Development Center, Kaohsiung City, Taiwan, since 2018. His research interests include memory and logic circuit design, communication circuit design, biomedical circuits, and particularly interfacing I/O circuits.

Dr. Wang became a fellow of IET in 2012. He was appointed to be a VP of the Office of Industrial Collaboration and Continuing Education Affairs, from 2014 to 2015. He was elected as the Dean of the Engineering College during 2014–2017. He was honored to be a Distinguished Professor of NSYSU. In 2012, he won the "Distinguished Engineering Professor" Award of the Chinese Institute of Engineers and the Outstanding Research Award of NSYSU. He was named as the ASE Chair Professor in 2013 for the recognition of his achievement. He won the Outstanding Technical Achievement Award of the IEEE Tainan Section in 2018. He was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS from 2010 to 2013 and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS from 2010 to 2011. He was the General Chair of the 2012 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS) and the Flagship Conference of IEEE CASS Region 10. He was assigned as a Distinguished Lecturer of the IEEE CASS from 2019 to 2021.