IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 25, 2020, accepted September 13, 2020, date of publication September 28, 2020, date of current version October 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3027260

FISHnet: Learning to Segment the Silhouettes of Swimmers

GUIDO ASCENSO[1], MOI HOON YAP[2], (Senior Member, IEEE), THOMAS BRUCE ALLEN[3], SIMON S. CHOPPIN[4], AND CARL PAYTON[1]

[1]Department of Sport and Exercise Sciences, Manchester Metropolitan University, Manchester M15 6BH, U.K.
[2]Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M15 6BH, U.K.
[3]Department of Engineering, Manchester Metropolitan University, Manchester M15 6BH, U.K.
[4]Centre for Sports Engineering Research, Sheffield Hallam University, Sheffield S1 1WB, U.K.

Corresponding author: Guido Ascenso (guido.ascenso@stu.mmu.ac.uk)

ABSTRACT We present a novel silhouette extraction algorithm designed for the binary segmentation of swimmers underwater. The intended use of this algorithm is within a 2D-to-3D pipeline for the markerless motion capture of swimmers, a task which has not been achieved satisfactorily, partly due to the absence of silhouette extraction methods that work well on images of swimmers. Our algorithm, FISHnet, was trained on the novel Scylla dataset, which contains 3,100 images (and corresponding hand-traced silhouettes) of swimmers underwater, and achieved a dice score of 0.9712 on its test data. Our algorithm uses a U-Net-like architecture and VGG16 as a backbone. It introduces two novel modules: a modified version of the Semantic Embedding Branch module from ExFuse, which increases the complexity of the features learned by the layers of the encoder; and the Spatial Resolution Enhancer module, which increases the spatial resolution of the features of the decoder before they are skip connected with the features of the encoder. The contribution of these two modules to the performance of our network was marginal, and we attribute this result to the lack of data on which our network was trained. Nevertheless, our model outperformed state-of-the-art silhouette extraction algorithms (namely DeepLabv3+) on Scylla, and it is the first algorithm developed specifically for the task of accurately segmenting the silhouettes of swimmers.

INDEX TERMS Biomechanics, CNN, deep learning, silhouette extraction, swimming.

I. INTRODUCTION

The ultimate goal of sports motion capture[1] is to accurately and automatically reconstruct the 3D locations of the joints of an athlete in motion just from one or more images (rather than by using sensors or markers attached to the body of the athlete) [2]. How are 3D joint coordinates inferred from images? As a first step, modern image-based 2D-to-3D methods (also referred to as *markerless motion capture systems*) extract from each recorded image one of two (or both) types of information: the silhouette of the athlete, and/or the location of the joints in image coordinates [3], [4]. The silhouette and/or the 2D joints are then used as inputs for either an optimisation algorithm (typically Iterative Closest Point [5], [6]) or a neural network (typically a deep convolutional neural network [7]–[9], more recently a generative adversarial network [10], [11]). The literature seems to agree on what the best options are to extract the 2D locations of the joints (which are collectively referred to as *2D pose*): they can be either digitised manually [4], or extracted automatically by a deep neural network, such as the Stacked Hourglass network [7], [12] or OpenPose [13]. What the best method is to extract silhouettes, on the other hand, is unclear. Whereas 2D joints can be manually digitised in a reasonable (albeit long) time (∼1 minute per frame to digitise 20 joints [14]), it would be impractical to manually trace the silhouette of an athlete (∼7 minutes per frame) for each frame that was to be digitised, especially if images from several cameras were to be used. Instead, authors who use silhouettes as inputs must rely on some type of algorithm to automatically extract them from the images. For controlled laboratory setups, in which the lighting is stable and the contrast between the subject and the background is sharp, silhouette extraction can be performed simply by recording a reference image of the background (i.e. without the subject present in the scene) and then subtracting that reference image from all the images in which the subject

The associate editor coordinating the review of this manuscript and approving it for publication was Kang Li.

[1]Describing the different types of motion capture is beyond the scope of this paper. For a review on this topic, please refer to [1].

is present [15]. This algorithm, known as *basic background subtraction*, is naive, as it assumes that the background remains unaltered throughout the recording session [15]. If even small changes of the background occur with the subject in the scene, basic background subtraction will treat the parts of the background that have changed as new objects that have appeared, thus lowering the accuracy of the segmented silhouette. When filming underwater swimming, turbulence and changing reflections and refractions void the assumption of a stable background, which means that 2D-to-3D methods that rely on background subtraction to obtain their silhouettes (for example, SIMI Shape [16]) are likely to be poor choices for swimming motion capture, because the silhouettes used as inputs will be inaccurate (see Figure 1). More recently, off-the-shelf convolutional neural networks (CNNs) have replaced background subtraction in 2D-to-3D pipelines [7], [9] and in 2D real-time segmentation of sports videos [17], leading to better and more generalisable results. Unfortunately, the generalisability of these more advanced silhouette extraction methods may not extend to images of underwater swimmers (see Figure 2), which contain a background so unique and in constant motion, and body configurations so unconventional and specific to swimming, that the datasets on which the CNNs were trained are unlikely to contain any similar images.

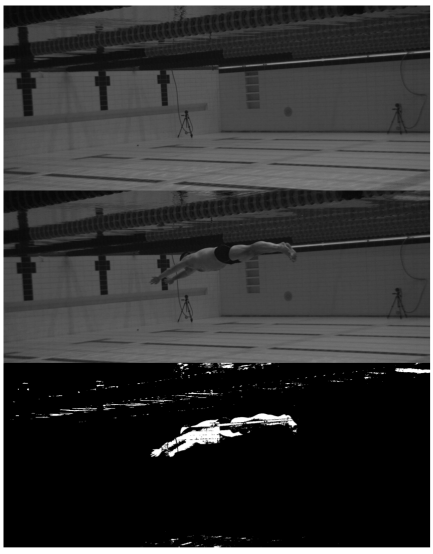

FIGURE 1. Top: background initialisation frame. Middle: foreground object inserted into the scene. Bottom: result of basic background subtraction.

The difficulty of performing markerless motion capture underwater is one of the main reasons why our understanding of swimming technique and biomechanics is less advanced than for other sports [18]. We believe that by providing authors with a robust silhouette extraction algorithm designed to work for images taken by underwater cameras, it may be possible in the future to construct 2D-to-3D systems that work well for swimming motion capture. Such systems, in turn, could lead to a better understanding of swimming technique.

FIGURE 2. Top: original image. Bottom: silhouette extracted using DeepLabv3+ (trained on PASCAL VOC).

The purpose of our research, then, was to develop a silhouette extraction method that would perform well specifically on images of swimmers underwater. The backbone of our proposed method combines the popular CNN architectures VGG16 [19] and U-Net [20], and it was trained on a novel dataset, Scylla [currently under submission], which was designed specifically for the task of silhouette extraction of images of swimmers and which is briefly described in the Method section. To allow for a fair evaluation of our algorithm, we also trained and tested popular segmentation algorithms (namely U-Net and DeepLabv3+ [21]) on the Scylla dataset and compared their results with ours.

II. RELATED WORK

Ceseracciu *et al.* [22] are the only authors who developed a markerless motion capture system for the analysis of swimmers. Their system—which uses the visual hull algorithm [23] to reconstruct 3D shapes from images—uses exclusively the silhouettes of the swimmers as inputs. The silhouettes they used were obtained by means of an algorithm similar to basic background subtraction. Instead of using an image of the background as a reference, their silhouette extraction algorithm considers n initialisation frames of empty background, and then constructs a Mixture of Gaussians in such a way that each Gaussian models a certain portion of the variability present in the initialisation frames [24]. While this method provides a background model that is more robust against noise and changes in lighting than basic background subtraction [24], it still cannot provide the level of silhouette quality required for markerless motion capture. Predictably, the results achieved by Ceseracciu *et al.* were unsatisfactory.

More recent works that use silhouette extraction algorithms in 2D-to-3D pipelines forego entirely the definition of a

background model, and instead rely on a neural network to internally estimate one [25]. For example, Lassner *et al.* [8] and Huang *et al.* [9] developed a 2D-to-3D system in which the silhouettes are automatically segmented using ResNet101 [26], whereas Pavlakos *et al.* [7] developed a CNN for silhouette extraction. Unfortunately, the quality of the silhouettes was not evaluated in these studies, in which the only metric of interest was the accuracy of the 2D-to-3D reconstruction, which could depend on factors other than the accuracy of the input silhouettes, such as the number of camera views and the quality of their calibration.

If the literature on 2D-to-3D methods offers little in terms of silhouette extraction algorithms that could perform well on images of underwater swimmers, the semantic segmentation literature offers some interesting insights and starting points. Though silhouette extraction is a problem of *binary* segmentation (the two classes being the background and the silhouette of the object), it is linked with the problem of *semantic* segmentation, in which the silhouettes of multiple objects present in a scene need to be segmented and labelled according to the class to which they belong; binary segmentation, then, can be viewed as a special case of semantic segmentation where the number of classes to be segmented is two (background and silhouette).

Most modern semantic segmentation algorithms employ CNNs arranged in an encoder-decoder architecture [21], [27], [28]: first, a series of convolutional and max pooling layers encode the input image into progressively higher-level features, which have progressively lower resolution and higher semantic content; then, the features of the deepest layer of the encoder are upsampled (or 'decoded') some number of times, so as to restore the original shape of the image. The features of the decoder are, therefore, the highest-level features of the encoder, upsampled once at each layer of the decoder. As the upsampling operation does not restore spatial resolution, the features of the decoder lack spatial resolution at all depths [27].

A well-known example of an encoder-decoder is U-Net [20], in which the encoder and decoder are symmetrical. Symmetry between the encoder and the decoder allows skip connections to be attached from each block of the encoder at depth n to the corresponding block of the decoder at depth n. The purpose of these skip connections is to pass some of the spatial resolution information present in low-level features of the encoder to the high-level features of the decoder. For example, the features at depth $d = 1$ of the encoder (maximum spatial resolution, minimum semantic meaning) are connected with the features at depth $d = 1$ of the decoder (maximum semantic meaning, little to no spatial resolution), which is the layer most responsible for the final segmentation and which, therefore, is most in need of fine-grained spatial resolution. The authors of ExFuse [27], one of the best-performing semantic segmentation algorithms available, evidenced that if, prior to the skip connection taking place, more semantic information is embedded in the low-level features and more spatial resolution is embedded in the high-level features, the effectiveness of the skip connection is magnified. To embed more semantic information in low-level features, they introduce auxiliary supervision on the shallow layers of the encoder, thus forcing their output to learn features closer in meaning to the desired output (performance improvement: 1.1%); and they introduce a new module, called Semantic Embedding Branch (SEB; performance improvement: 0.7%), which adds to each skip connection an additional connection from each layer of the encoder below the current layer. For example: if a skip connection is being performed between layers at depth n of the encoder and decoder, the outputs of the layers of the encoder at depth n - *1*, n - *2*, n - *3*, etc, are fused to the output of layer n of the encoder before being fused with layer n of the decoder (more on this in the following section). Additionally, ExFuse embeds, prior to the skip connection, more spatial resolution in the high-level features of the encoder by introducing two new modules: Densely Adjacent Prediction (DAP; performance improvement: 0.6%) and Explicit Channel Resolution Embedding (ECRE; performance improvement: 0.5%). A detailed description of these modules can be found in [27].

Another popular semantic segmentation algorithm, DeepLabv3+ [21] also uses an encoder-decoder structure, but unlike ExFuse it preserves spatial resolution in its decoder by replacing the last few downsampling layers of the encoder with atrous convolutions (from the French *à trous*, 'with holes'). By inserting 'holes' within the elements of the filter, atrous convolutions learn features with higher spatial resolution without needing extra parameters [29]. In DeepLabv3+, several convolutional layers with different atrous rates are concatenated into a module called Atrous Spatial Pyramid Pooling (ASPP), which produces high-level features that are able to detect objects at different scales. Though DeepLabv3+ is the best-performing semantic segmentation algorithm on the PASCAL VOC dataset [30], it struggles to segment the silhouettes of underwater swimmers (as illustrated by Figure 2).

III. METHOD

A. ALGORITHM

Our method uses a U-Net-like architecture in which the encoder consists of VGG16 and the decoder of a simple series of bilinear upsampling and convolutional layers (Figure 4). Additionally, some of the intuitions from ExFuse [27] and DeepLabv3+ [21] are implemented in our network. The main intuition behind ExFuse is that more semantic information needs to be introduced in the low-level features of the encoder and more spatial resolution needs to be introduced in the high-level features of the decoder. We now describe how we addressed both aspects.

Our solution to introducing more semantic information in the low-level features of the encoder is simple. Since it is the complexity of the features that determines the amount of semantic information they can encode, and since

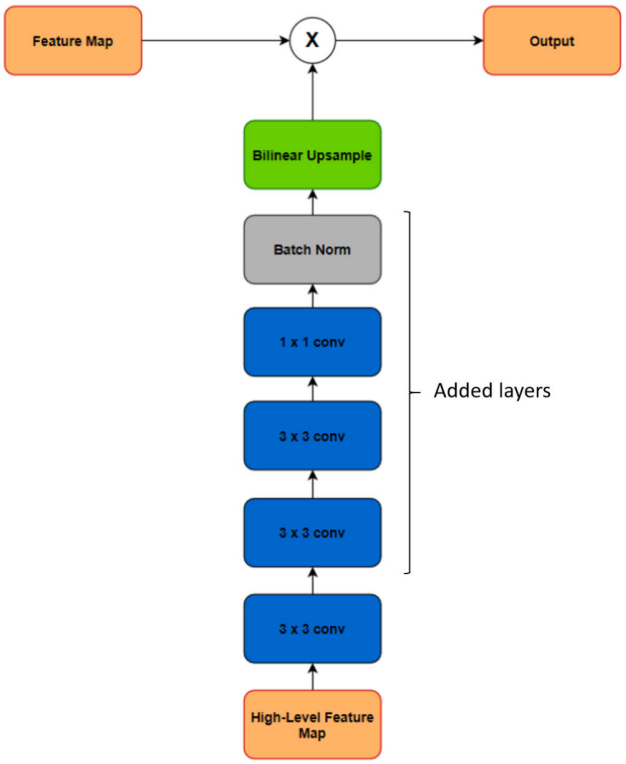

FIGURE 3. Our modified SEB module; compared with the SEB module proposed by the authors of ExFuse [27], our SEB module adds two more 3 × 3 convolutional layers, one 1 × 1 convolutional layer, and a batch normalisation layer; the 'x' sign represents element-wise multiplication. If multiple high-level feature maps are present (which only happens for depth $d < 4$), they are all multiplied together (element-wise). Our modifications to the SEB module allow it to encode much more complex features.

complexity in neural networks is determined by the degree of non-linearity present in the features [31], we have to introduce more non-linearity in the low-level features of the encoder before they are fused in the skip connections. To achieve this, we modify the SEB module of ExFuse and increase its complexity by adding to it two convolutional layers—each with a ReLU activation, to introduce the desired non-linearity—followed by a 1 × 1 convolutional layer and a batch normalisation layer (Figure 3). The module illustrated in Figure 3 encodes a function $f(x_d, x_{d+1}, x_{d+2}, \ldots, x_D)$ which fuses the output x_d of the encoder at depth d with the outputs of all the layers of the encoder at depth greater than d, up to the maximum depth D. The output of $f(\cdot)$ is then used for a skip connection with the layer y_d of the decoder at depth d. In other words, a normal skip connection can be represented by the equation:

$$y_d = [UpSampling(y_{d+1}),\ \ x_d] \tag{1}$$

whereas a skip connection using a SEB module [27] can be represented by the equation:

$$y_d = [UpSampling(y_{d+1}),\ \ f(x_d, x_{d+1}, \ldots, x_D)] \tag{2}$$

To introduce more spatial resolution in the high-level features of the decoder, we take inspiration from DeepLabv3+. In DeepLabv3+, the function of the ASPP module (whose structure and function are similar to the structure of TridentNet [32]) is to preserve the spatial resolution of the last few layers of the encoder [21]. However, the ASPP module could also be re-purposed to make it increase the spatial resolution of the layers of the decoder prior to the skip connections. In other words, our intuition is the following: before the skip connection takes place, the high-level features of the decoder are connected to an ASPP-like module, which we call Spatial Resolution Enhancer (SRE). The SRE consists of a variable number of parallel atrous convolutional layers, followed by a 1 × 1 convolutional layer (Figure 5). How many atrous convolutional layers should the SRE contain–and what their atrous rate should be–depends on the depth at which the skip connection is taking place. If the skip connection is taking place between the deepest layer of the encoder and that of the decoder (in the case of our network, the deepest possible skip connection is performed at $d = 4$), the features of the encoder do not contain much more spatial information than those of the decoder. Therefore, only a little spatial resolution needs to be injected into the features of the decoder for the skip connection to be effective.[2] Therefore, the SRE module does not need to be as large as the ASPP module of DeepLabv3+. Conversely, at $d = 1$ the features of the encoder have much higher spatial resolution than the features of the decoder, which means that the SRE module at $d = 1$ will need to embed much more spatial resolution than the SRE module at $d = 4$. This leads us to the formulation of a variable structure for the SRE module, summarised in Table 1. The variable complexity of the SRE module allows it to embed an amount of spatial information proportional to the expected spatial information of the features of the encoder where it will be connected. The SRE module (illustrated in Figure 5) encodes a function $g(y_d)$ which modifies the layer y of the encoder at depth d. Therefore, our skip connections are described by the following equation:

$$y_d = [g(UpSampling(y_{d+1})),\ \ f(x_d, x_{d+1}, \ldots, x_D)] \tag{3}$$

TABLE 1. Atrous rates and number of layers of the SRE modules at each depth of the decoder.

Depth	Number of Layers	Atrous Rates
4	1	4
3	2	4, 8
2	3	4, 8, 16
1	4	4, 8, 16, 32

Figure 6 illustrates the structure of the skip connections in our network and compares it with that of the skip connections in a normal U-Net architecture; for simplicity, the networks are illustrated only up to $d = 3$.

[2]Remember the assumption proposed by the authors of ExFuse [27] that the effectiveness of a skip connection increases if the elements to the left and to the right of it have similar semantic content and spatial resolution.

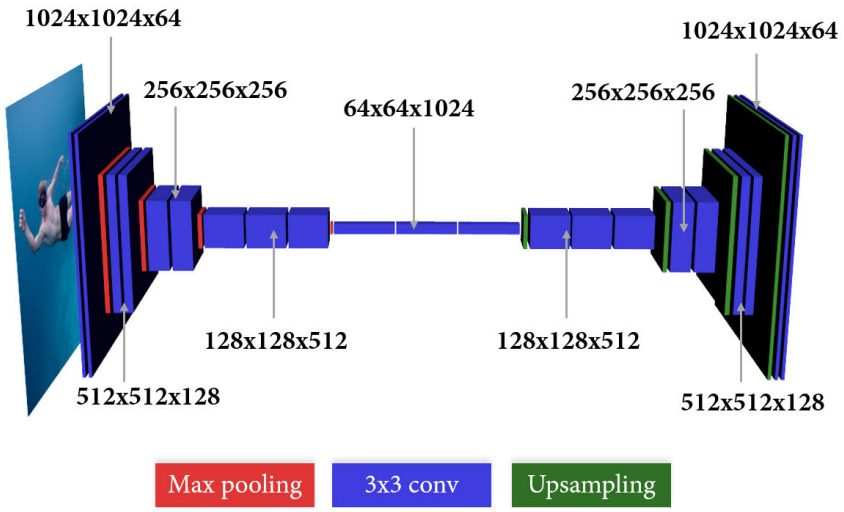

FIGURE 4. Structure of FISHnet; the backbone of the encoder is VGG16.

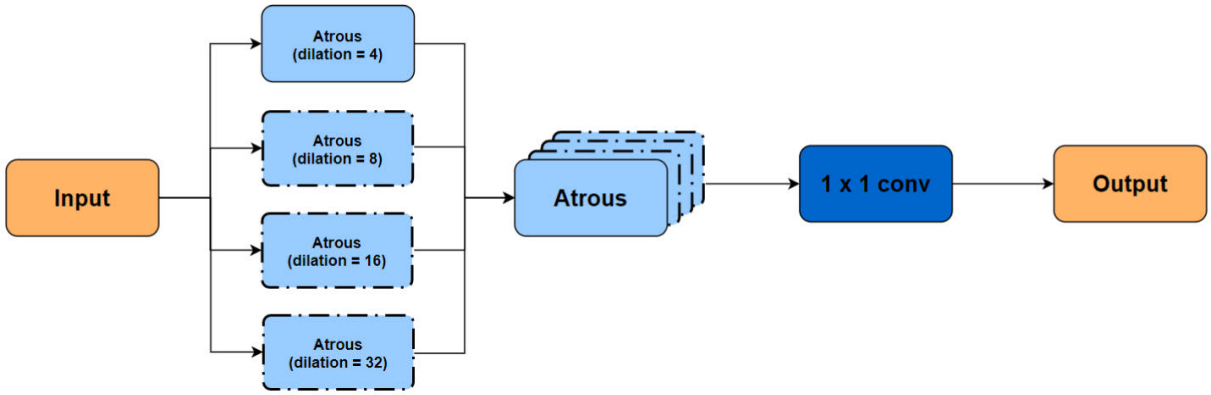

FIGURE 5. Our SRE module takes as input a layer y_d of the decoder (after upsampling) and passes it through one to four atrous convolutions in parallel. The outputs of the atrous convolutions are concatenated together and fed to a 1 × 1 convolutional layer, the output of which is the output of the SRE module. In this figure, the outline of most atrous convolutional layers is left dashed because those layers will only be active at certain depths (see Table 1).

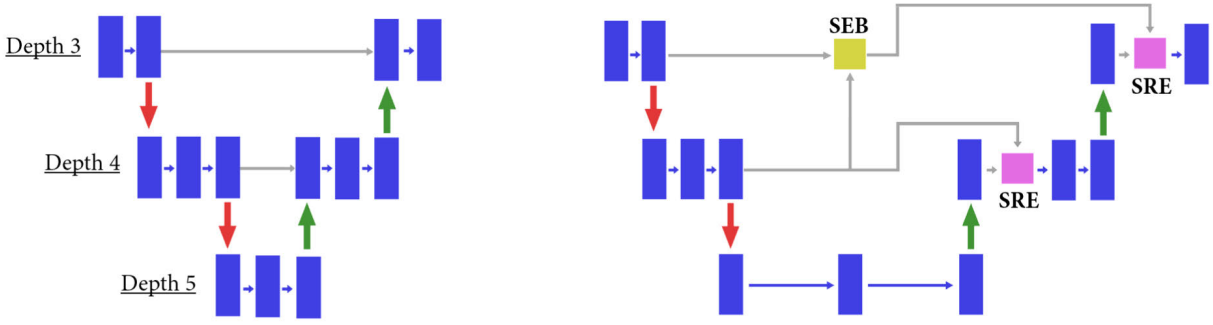

FIGURE 6. Left: skip connections in a normal U-Net; the output of the encoder at depth d is copied and concatenated with the first layer of the decoder at depth d. Right: skip connections in FISHnet; the yellow block represent our version of the SEB module, while the pink blocks represent our SRE modules.

B. DATASET

Training an algorithm on popular semantic segmentation datasets like COCO [33] or PASCAL VOC [30] is not pertinent if the purpose of the algorithm is to accurately segment the silhouettes of swimmers. This is evidenced by Figure 2, which shows how even DeepLabv3+, the most

accurate semantic segmentation algorithm available, can fail to segment the silhouette of a swimmer. Therefore, given that our algorithm was intended to perform well on the specific task of extracting the silhouettes of swimmers, and not on the generic task of semantic segmentation, we propose to train and test it on the novel Scylla dataset [awaiting publication]. Scylla was assembled by extracting frames from over 100 videos of 11 elite swimmers performing either underwater butterfly kicks or underwater breaststroke. The videos were recorded with either a one- or four-camera system, placed underwater in waterproof housings. The four-camera system was used to increase the number of views from which the swimmers were recorded: a camera was placed on either side of the swimmer (perpendicular to the direction of motion), one at the back, and one at the front. Each video was cropped so it began on the first, and ended on the last, frame that the swimmer was fully visible. Using a custom MATLAB (Mathworks) script, a number of frames were extracted and saved as JPEG images. The number of images extracted per video varied semi-randomly: at least 10 images per video were extracted, and a gap of at least X frames was kept between any two extracted images; X was randomly assigned a value between 10 and 30 after each image was extracted. This was done under the assumption that extracting frames at a regular interval (say, one frame every 10) may have introduced unwanted bias in the dataset due to the cyclicality of the movements performed by the swimmers; likewise, choosing $X < 10$ may have resulted in images that were too similar. This process led to the formation of a dataset of 3,100 images, which were split into 2,635 images for training and 465 images for testing. Swimmers who feature in the training set do not also feature in the test set. Since they came from different cameras, the images have different resolutions: 2,793 images have a resolution of 900×2048 pixels, 307 images have a resolution of 1080×1920 pixels. An experienced Photoshop user was recruited to manually trace in Photoshop the silhouette of each swimmer in each image. After the labelling was completed, each frame was inspected to ensure that no gross tracing mistakes had been made. To quantify the reliability of the labeller, three other experienced Photoshop users manually labelled 10 of the images, and the mean overlap between their traced silhouettes and those of the 'original' labeller was quantified using the Dice coefficient, which was averaged across the 10 images for each 'control' labeller. The evaluation metric used for Scylla is the Dice coefficient.

Since ours was the first algorithm to be tested on the Scylla dataset, we also trained on it the original U-Net and DeepLabv3+ networks, to allow for a fair evaluation of our algorithm.

C. IMPLEMENTATION DETAILS

Our network uses VGG16 (minus the fully-connected layers) pre-trained on ImageNet as its encoder. The network was trained with a batch size of 2, the RMSprop optimiser (starting learning rate: 0.0001), and a Dice loss function defined as:

$$f(x) = 1 + BCE - Dice \tag{4}$$

where BCE is the binary cross-entropy function. Of the 2,635 images available for training (85% of the total 3,100), 395 (15%) were used for validation. The network was allowed to train until the validation loss had not decreased by at least 0.0001 for five consecutive epochs. Due to the limited amount of memory available, during training all input images were resized to 1024×1024 (using padding when necessary) and the batch size was set to 2. The following data augmentation techniques were used: horizontal flip, random scale, random rotation. Our network was implemented entirely using Keras with TensorFlow backend. A single Nvidia Titan X was used for training.

To investigate the degree of overfitting (interpreted as the magnitude of the difference between train and test dice values) that our algorithm suffers from, we trained it using different subsets of Scylla's training data. We expected that as the size of the training set decreased, the accuracy on the training set would increase and the accuracy on the test set decrease (i.e., the degree of overfitting would increase). By plotting the amount of training data used against the difference between train and test dice scores, we can visualise the trend of overfitting: if the trend plateaus as it approaches 85% training data (the default amount of training data in Scylla), it would indicate that the Scylla dataset contains enough training examples to not cause overfitting. If, however, the trend does not plateau as it approaches 85%, but instead keeps decreasing linearly or close to it, it would indicate that the Scylla dataset would benefit from having more training examples.

IV. RESULTS

Training converged after 26 epochs (for a total of 19 hours of training), and led to a dice score on the test set of Scylla of 0.9712. After training, inference took 215 ms/image. Table 2 compares the accuracy of our model to that of DeepLabv3+ and different versions of U-Net. In Table 2, the number next to U-Net refers to the size of the inputs (and therefore of the outputs, since U-Net is symmetrical) of the network. For example, in U-Net 1024 all inputs are resized to 1024×1024 (which is the resolution that our network also uses as input).

TABLE 2. Results on the Scylla dataset.

Network	Dice score
U-Net 128	0.8727
U-Net 512	0.9399
U-Net 1024	0.9609
DeepLabv3+	0.9510
FISHnet (ours)	**0.9712**

To investigate how much each element of our network contributed to the overall accuracy achieved, we report in Table 3 an ablation study that shows the dice score achieved when a

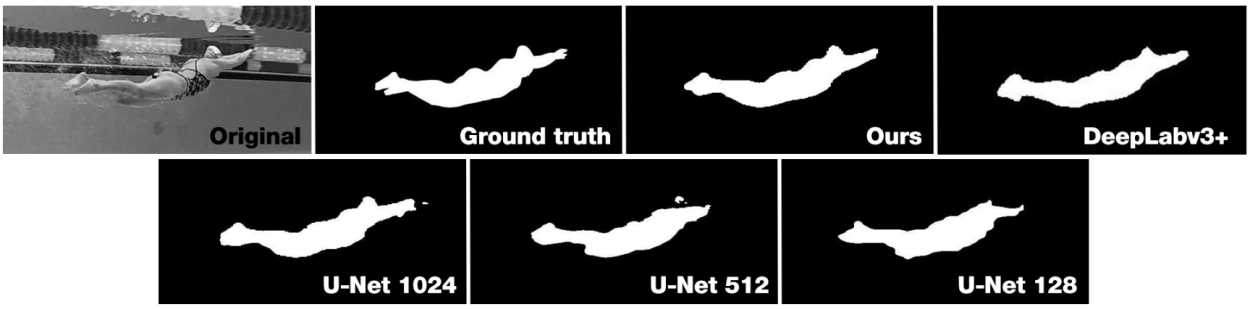

FIGURE 7. Comparison of the algorithms considered in this paper on a sample image from the Scylla dataset. All images were cropped around the figure of the swimmer, to make the fine details of each image more visible.

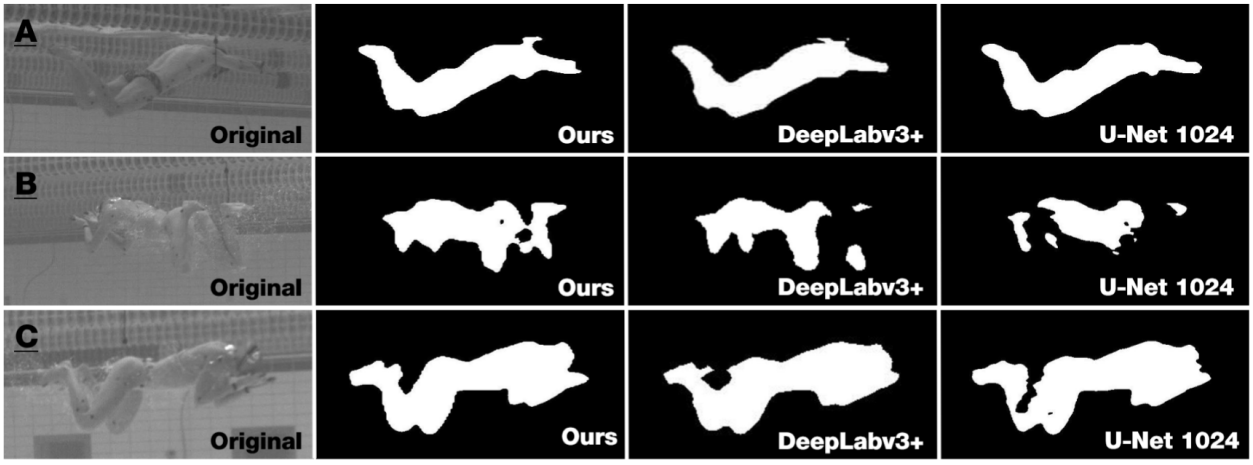

FIGURE 8. Three examples of our network making noticeable segmentation errors. The outputs of DeepLabv3 and U-Net 1024 are also reported here to understand if the images on which our network failed were inherently difficult.

TABLE 3. Ablation study. Each row represents a separate test; a '✓' symbol indicates that the component was active during that test.

Baseline	Pre-trained	SEB	Modified SEB	SRE	Dice
✓					0.9648
✓	✓				0.9701
✓	✓	✓			0.9709
✓	✓		✓		**0.9712**
✓	✓		✓	✓	0.9711

certain subset of elements was active. In Table 3, Baseline refers to a simple U-Net-like structure in which the encoder consists of VGG16, whereas Pretrained refers to whether or not the encoder of the baseline network was pre-trained on ImageNet.

Figure 7 shows the performance of the algorithms reported in Table 2 tested on a sample image from Scylla's test data, so that the differences between the algorithms' performance can be seen. To better understand the strengths and limitations of our algorithm, we also report in Figure 8 three of the images of the Scylla dataset on which our network performed the worst, along with the output of DeepLabv3+ and U-Net 1024 on the same images, for comparison. These images are discussed in the following section.

Finally, Figure 9 illustrates how our model's overfitting (interpreted as the difference between train and test Dice scores) varies as a function of the amount of training data used.

V. DISCUSSION

Models that take in higher-resolution inputs achieved better results (Table 2). This happens because if a network outputs a silhouette of resolution below 1088×2048, it needs to be upsampled to 1088×2048 to be tested against the ground truth present in the Scylla dataset. If the output resolution is below 1088×2048, details along the contour of the silhouette will be lost, resulting in low-resolution silhouettes. Figure 7 visually confirms this statement: the silhouette of U-Net 1024 has more detail than the silhouette of U-Net 128, and a smoother contour. As the output resolution moves closer to 1088×2048, more details of the silhouette will be preserved. Since DeepLabv3+ outputs images at a resolution that is 8 to 16 times (depending on the implementation) smaller than the input image, its accuracy is hampered when the test images are as large as those of Scylla. It is not entirely surprising, then, that DeepLabv3+, which is deeper and more

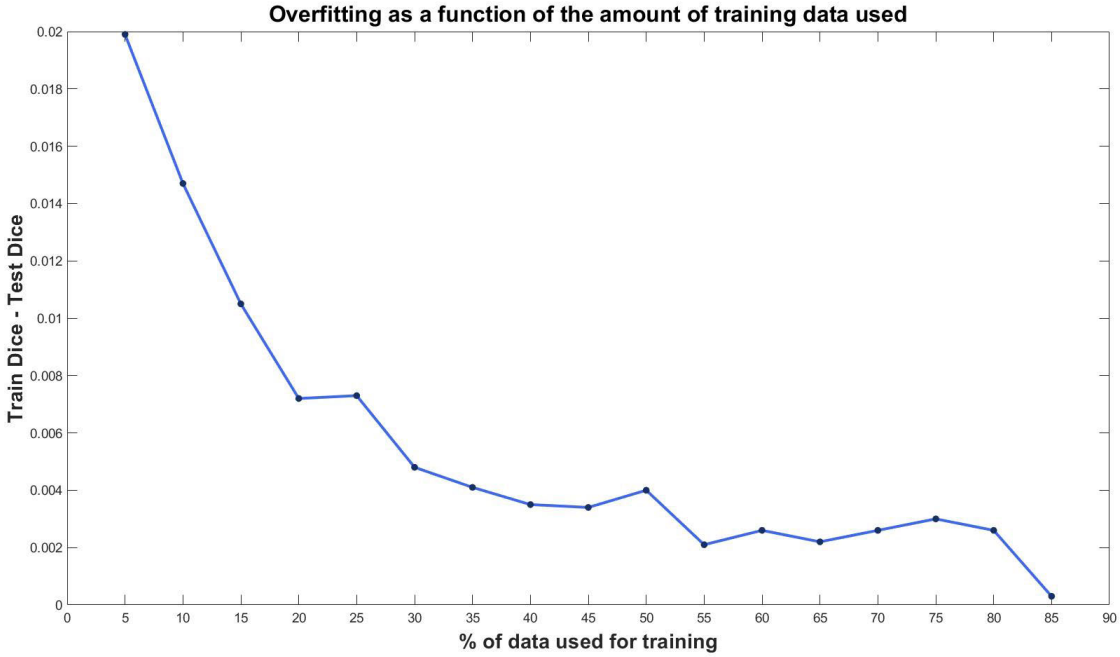

FIGURE 9. As more data is used to train our model, the degree of overfitting becomes lower and lower. Nevertheless, the negative trend towards the end of the graph (between 75% and 85%) indicates that expanding the training set of the Scylla dataset may reduce overfitting even further.

complex than our model, does not outperform our model despite having a more complex and deep architecture.

Figure 9 illustrates how overfitting changes as a function of the percentage of data used to train our model. As expected, overfitting decreases as more data is used for training. However, it is surprising to find that even with very small training sets the amount of overfitting is negligible: when training our network with just 5% of the total data available, the difference between the train and test Dice scores was only 0.02. This indicates that there is so much redundancy in the images used for training that only a few hundred of them are enough for our model to achieve satisfactory results. Nevertheless, using the highest percentage possible of training data (85% of the total 3,100 images of Scylla) resulted in ten times lower overfitting than using the lowest. Furthermore, between 75% and 85% training data there is a clear—albeit small—negative trend, which indicates that having had access to more training data would have perhaps reduced overfitting even futher.

The results in Table 3 are quite surprising, in that they seem to indicate that the modules described in the Method section of this paper only improve performance marginally over the baseline model's (0.11% for the modified SEB module; 0.10% for the SRE module). One interpretation of this result is that the SRE and the modified SEB modules are not effective and do not improve the effectiveness of the skip connections as intended. However, the regular SEB module also has a marginal contribution to overall performance (0.08%), whereas in ExFuse it had a contribution almost ten times as large (0.7%) [27]. Therefore, there must be some other explanation for why all additional modules that are attached to our network only have marginal effects. We believe the explanation could be found in the amount of training data available. The optimal capacity of a neural network depends on the complexity of the task at hand and on the amount of training data available [31]—which, in the case of the Scylla dataset, is only 2,635 images. This could mean that our model reaches optimal capacity roughly at baseline, and that adding capacity to the model (by introducing the SRE and modified SEB modules) past that point results in overfitting. Since our training paradigm mitigates overfitting by using early stopping (i.e. stopping training when the validation loss stops improving), all our models converge to similar values because the added capacity of the more complex models is not allowed to be trained. To test this theory, we re-trained our model without early stopping, and observed that convergence took 59 epochs (compared to 26 with early stopping) and reached a train dice of 0.9740 (compared to 0.9715 with early stopping). Furthermore, the test dice without early stopping was 0.9687 (compared to 0.9712 with early stopping). This finding suggests that our model has enough capacity to achieve higher dice scores, but is gated by the limited data available to it for training. In other words, if it expresses its full capacity it incurs into overfitting. Future studies may address this limitation either by pre-training our full network on large semantic segmentation datasets like PASCAL VOC and COCO and then using Scylla for fine-tuning only, or by collecting more swimming-specific training data.

Figure 8 shows three of the test images on which our algorithm performed the worst. The reason for the failure on image A of Figure 8 was because a background object (a floating pole) was segmented as part of the swimmer's head. The mistake is repeated by DeepLabv3, but, unexplainably, not by U-Net 1024. The pole is also visible in image B of Figure 8, but in this case it was not segmented as part of the swimmer's silhouette, likely because in that image the pole is of a colour that is not consistent with any part of the swimmer, whereas in image A the pole was of the same colour as the swimming cap of the swimmer, with which it was juxtaposed.

The reason for our network's failure on image B of Figure 8 is not as easily diagnosed, though it is clear that this image must have some feature that makes it difficult, as all algorithms failed on it. In this image, the colour camouflaging of the swimmer with the background is particularly pronounced. Indeed, even a human might struggle to detect the outline of the legs. However, many of the images in the Scylla dataset have a similar degree of colour camouflaging—for example, though to a lesser degree, image A in Figure 8. Another possible explanation for why this image was particularly challenging is the presence of bubbles around the legs (which are the area where the grossest segmentation mistakes are made): the algorithms might have learned from the training data to ignore bubbles, and therefore in this image ignore much of the legs. The negative effect of bubbles is somewhat confirmed by image C of Figure 8, in which all algorithms fail to correctly segment the left shank and right foot of the swimmer—the areas most enveloped by bubbles. However, the effect of the bubbles on this image was less drastic than in image B. We could conclude, then, that the reason all algorithms performed poorly on image B was the combination of bubbles and colour camouflaging. Since these two conditions overlap rarely in the images of the Scylla dataset, our algorithm commits few gross mistakes. The main source of error, then, is not to be found in easily observable mistakes as the ones shown in Figure 8, but in small-scale errors that occur along the contour of all silhouettes. Because our algorithm and U-Net 1024 should reconstruct the most accurate outlines (since their output resolution is the closest to the resolution of the images in the Scylla dataset), and since we just established that accuracy along the contour is likely to be indicative of overall performance on the test set of the Scylla dataset, we would expect in theory that our algorithm and U-Net 1024 should outperform the other algorithms; this result is confirmed empirically by the results reported in Table 2. In other words, we attribute the margin of our algorithm over DeepLabv3+, U-Net 512, and U-Net 128 to it outputting smoother silhouettes, which in turn is a consequence of the higher output resolution on which our model and U-Net 1024 operate.

What accounts for our algorithm's advantage over U-Net 1024 is less clear, since both algorithms operate on the same resolution and should, in theory, produce silhouettes with equally smooth outlines. A possible explanation is that our algorithm learns more complex features than U-Net 1024 due to its more dense skip connections, features that make our algorithm generalise better to harder images. For example, our algorithm may have learned a better way of modelling bubbles, thus failing less severely than U-Net 1024 in their presence (as shown in image B of Figure 8).

Finally, a remark on the scope of our research. In semantic segmentation it is common practice to validate an algorithm on PASCAL VOC or COCO, to show that the algorithm can generalise well to objects from multiple classes, and therefore, in theory, to many different tasks. For the specific case of the binary segmentation of underwater swimmers, however, it makes little sense to validate an algorithm on PASCAL VOC or COCO, for two reasons. First, the spatial resolution of the silhouettes has to be as high as possible, and the images in PASCAL VOC and COCO are small ($\sim$500 $\times$ 500 for PASCAL VOC, $\sim$640 $\times$ 640 for COCO). The spatial resolution has to be high because the intended use of these silhouettes is in a 2D-to-3D pipeline, in which having larger silhouettes means having access to a better constraint for where the 3D object is allowed to lie. Second, neither PASCAL VOC nor COCO contain any images of swimmers underwater. Therefore, if our model had been trained and tested on PASCAL VOC or COCO, the results would have told us only if it was a good algorithm for semantic segmentation, which was not the objective of this research.

VI. CONCLUSION

Accurate markerless motion capture of underwater swimmers has not been achieved yet, mainly because there is no algorithm that can accurately segment the silhouettes of swimmers underwater: traditional algorithms like background subtraction are too noisy, and off-the-shelf pre-trained models (like DeepLab) fail to generalise to images of swimmers. This paper presents the first algorithm to date designed specifically for the task of binary segmentation of swimmers underwater. This represents a paradigm shift in the field of computer vision: instead of testing our algorithm—which is designed for a specific task—on generic datasets like PASCAL VOC and MS COCO, we propose to test it on a novel dataset, Scylla, that was designed for this specific task. Scylla contains 3,100 images (and corresponding hand-drawn silhouettes) of underwater swimmers, and it is the first dataset of its kind in this field. The backbone of our algorithm is U-Net (with VGG16 pretrained on ImageNet for a decoder), and we add to it two novel modules: a modified version of the SEB module from ExFuse, which serves to increase the complexity of the features learned by the layers of the encoder; and the SRE module, which serves to increase the spatial resolution of the features of the decoder before they are skip connected with the features of the encoder. The contribution of these two modules to the overall performance of our network was marginal, but our ablation study shows that this may have been due to the limited data available for training. Nevertheless, we showed that on Scylla our model outperformed two state-of-the-art segmentation algorithms (U-Net and DeepLabv3+), achieving a dice score of 0.9712.

Further studies with access to more training data may achieve even higher dice scores using our model.

REFERENCES

[1] E. van der Kruk and M. M. Reijne, "Accuracy of human motion capture systems for sport applications; state-of-the-art review," *Eur. J. Sport Sci.*, vol. 18, no. 6, pp. 806–819, Jul. 2018.
[2] G. Ferrigno, N. A. Borghese, and A. Pedotti, "Pattern recognition in 3D automatic human motion analysis," *ISPRS J. Photogramm. Remote Sens.*, vol. 45, no. 4, pp. 227–246, Aug. 1990.
[3] X. Chen, K.-Y. Lin, W. Liu, C. Qian, and L. Lin, "Weakly-supervised discovery of geometry-aware representation for 3D human pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 750–767.
[4] S. L. Colyer, M. Evans, D. P. Cosker, and A. I. T. Salo, "A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system," *Sports Med. Open*, vol. 4, no. 1, p. 24, Dec. 2018.
[5] S. Corazza, L. Mündermann, E. Gambaretto, G. Ferrigno, and T. P. Andriacchi, "Markerless motion capture through visual hull, articulated icp and subject specific model generation," *Int. J. Comput. Vis.*, vol. 87, nos. 1–2, p. 156, 2010.
[6] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proc. 3rd Int. Conf. 3-D Digit. Imag. Modeling*, 2001, pp. 145–152.
[7] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis, "Learning to estimate 3D human pose and shape from a single color image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 459–468.
[8] C. Lassner, J. Romero, M. Kiefel, F. Bogo, M. J. Black, and P. V. Gehler, "Unite the people: Closing the loop between 3D and 2D human representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6050–6059.
[9] Y. Huang, F. Bogo, C. Lassner, A. Kanazawa, P. V. Gehler, J. Romero, I. Akhter, and M. J. Black, "Towards accurate marker-less human shape and pose estimation over time," in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 421–430.
[10] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7122–7131.
[11] D. Drover, C.-H. Chen, A. Agrawal, A. Tyagi, and C. P. Huynh, "Can 3D pose be learned from 2D projections alone?" in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 78–94.
[12] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 483–499.
[13] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," 2018, *arXiv:1812.08008*. [Online]. Available: http://arxiv.org/abs/1812.08008
[14] F. A. Magalhaes, Z. Sawacha, R. Di Michele, M. Cortesi, G. Gatta, and S. Fantozzi, "Effectiveness of an automatic tracking software in underwater motion analysis," *J. Sports Sci. Med.*, vol. 12, no. 4, p. 660, 2013.
[15] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Comput. Sci. Rev.*, vols. 11–12, pp. 31–66, May 2014.
[16] A. Henrichs, "Validation of 3D silhouette tracking as clinical movement analysis tool," Ph.D. dissertation, FH Münster, Münster, Steinfurt, 2016.
[17] A. Cioppa, A. Deliege, M. Istasse, C. De Vleeschouwer, and M. Van Droogenbroeck, "ARTHuS: Adaptive real-time human segmentation in sports through online distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019.
[18] R. Mooney, G. Corley, A. Godfrey, C. Osborough, J. Newell, L. R. Quinlan, and G. ÓLaighin, "Analysis of swimming performance: Perceptions and practices of US-based swimming coaches," *J. Sports Sci.*, vol. 34, no. 11, pp. 997–1005, Jun. 2016.
[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[20] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2015, pp. 234–241.
[21] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 801–818.
[22] E. Ceseracciu, Z. Sawacha, S. Fantozzi, M. Cortesi, G. Gatta, S. Corazza, and C. Cobelli, "Markerless analysis of front crawl swimming," *J. Biomech.*, vol. 44, no. 12, pp. 2236–2242, Aug. 2011.
[23] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 2, pp. 150–162, Feb. 1994.
[24] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Video-Based Surveillance Systems*. London, U.K.: Springer, 2002, pp. 135–144.
[25] G. Ascenso, M. H. Yap, T. Allen, S. S. Choppin, and C. Payton, "A review of silhouette extraction algorithms for use within visual hull pipelines," *Comput. Methods Biomech. Biomed. Eng., Imag. Vis.*, pp. 1–22, Jul. 2020.
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[27] Z. Zhang, X. Zhang, C. Peng, X. Xue, and J. Sun, "Exfuse: Enhancing feature fusion for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 269–284.
[28] D. Lin, Y. Ji, D. Lischinski, D. Cohen-Or, and H. Huang, "Multi-scale context intertwining for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 603–619.
[29] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: http://arxiv.org/abs/1706.05587
[30] D. Hoiem, S. K. Divvala, and J. H. Hays, "Pascal VOC 2008 challenge," in *Proc. PASCAL Challenge Workshop (ECCV)*. Graz, Austria: Citeseer, 2009.
[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[32] Y. Li, Y. Chen, N. Wang, and Z.-X. Zhang, "Scale-aware trident networks for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6054–6063.
[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.

GUIDO ASCENSO received the M.Sc. degree in sports biomechanics from Robert Gordon University, Aberdeen, U.K., in 2016. He is currently pursuing the Ph.D. degree with Manchester Metropolitan University. The topic of his Ph.D. degree is the development of a markerless motion capture systems for the biomechanical analysis of swimmers. His research interests include the applications of deep learning techniques to biomechanical problems.

MOI HOON YAP (Senior Member, IEEE) received the Ph.D. degree in computer science from Loughborough University, in 2009. She is a Reader (Associate Professor) in computer vision with Manchester Metropolitan University and a Royal Society Industry Fellow of Image Metrics Ltd. After the Ph.D. degree, she worked as a Postdoctoral Research Assistant with the Centre for Visual Computing, University of Bradford, from April 2009 to October 2011. Her research interests include computer vision and deep learning. She serves as an Associate Editor for the *Journal of Open Research Software* and a Reviewer for IEEE TRANSACTIONS/journals such as IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CYBERNETICS, and IEEE JOURNAL ON BIOMEDICAL AND HEALTH INFORMATICS.

THOMAS BRUCE ALLEN is a Senior Lecturer in mechanical engineering with Manchester Metropolitan University, where he specializes in sports engineering. As a Sports Engineer, he is interested in the effect of engineering and technology on sport, especially in terms of performance, participation, and injury risk. He enjoys collaborating with academic researchers as well as working with sports brands and other organizations. He is also a Board Member of the International Society for Snowsport Safety and a Fellow of the Institution of Mechanical Engineers. He is an active member of the International Sports Engineering Association and acts as the Editor-in-Chief for their journal *Sports Engineering*.

SIMON S. CHOPPIN received the M.Eng. degree in mechanical engineering with mathematics from The University of Nottingham, in 2004, and the Ph.D. degree in the analysis of elite tennis player movement from The University of Sheffield, in 2008. He is a Senior Research Fellow of the Centre for Sports Engineering Research, Sheffield Hallam University. His main research interests include the 3D analysis of human shape for applications in sports and health, and he is a member of the IEEE 3D Body Processing Group.

CARL PAYTON holds the position of Professor of Sports Biomechanics with the Musculoskeletal Science and Sports Medicine Research Centre, Manchester Metropolitan University, U.K. He received his Ph.D. degree in sports biomechanics from Manchester Metropolitan University in 1999. His main research interests include swimming biomechanics with a particular focus on the use of three-dimensional motion analysis to enhance the performance of elite swimmers.