

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.DOI

A Fast Lightweight 3D Separable Convolutional Neural Network with Multi-Input Multi-Output for Moving Object Detection

BINGXIN HOU¹, (Student Member, IEEE), YING LIU¹, (Member, IEEE), NAM LING¹, (Fellow, IEEE), LINGZHI LIU², (Senior Member, IEEE), and YONGXIONG REN², (Member, IEEE)

¹Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA 95053 USA

²Kwai, Inc., Palo Alto, CA 94306 USA

Corresponding author: Nam Ling (e-mail: nling@scu.edu).

This work was supported in part by Kwai, Inc. under the Kwai, Inc. grant.

ABSTRACT Advances in moving object detection have been driven by the active application of deep learning methods. However, many existing models render superior detection accuracy at the cost of high computational complexity and slow inference speed. This fact has hindered the development of such models in mobile and embedded vision tasks, which need to be carried out in a timely fashion on a computationally limited platform. In this paper, we propose a super-fast (inference speed-154 fps) and lightweight (model size-1.45 MB) end-to-end 3D separable convolutional neural network with a multi-input multi-output (MIMO) strategy named “3DS_MM” for moving object detection. To improve detection accuracy, the proposed model adopts 3D convolution which is more suitable to extract both spatial and temporal information in video data than 2D convolution. To reduce model size and computational complexity, the standard 3D convolution is decomposed into depthwise and pointwise convolutions. Besides, we proposed a MIMO strategy to increase inference speed, which can take multiple frames as the network input and output multiple frames of detection results. Further, we conducted the scene dependent evaluation (SDE) and scene independent evaluation (SIE) on the benchmark CDnet2014 and DAVIS2016 datasets. Compared to state-of-the-art approaches, our proposed method significantly increases the inference speed, reduces the model size, meanwhile achieving the highest detection accuracy in the SDE setup and maintaining a competitive detection accuracy in the SIE setup.

INDEX TERMS Convolutional neural network, depthwise convolution, moving object detection, multi-input multi-output, pointwise convolution, scene independent evaluation, 3D separable convolution, unseen videos, video analytics, video surveillance.

I. INTRODUCTION

WITH the increasing amount of network cameras, produced visual data and Internet users, it becomes quite challenging and crucial to process a large amount of video data at a fast speed. Moving object detection (MOD) is the process of extracting dynamic foreground content from the video frames, such as moving vehicles or pedestrians, while discarding the non-moving background. It plays an essential role in many real-world applications [1], such as intelligent video surveillance [2], medical diagnostics [3], anomaly de-

tection [4], human tracking and action recognition [5], [6].

Traditional methods [7]–[29] are unsupervised which do not require labeled ground truth for algorithm development. They usually include two steps: background modeling and pixel classification. However, these traditional methods meet difficulties when applied in complex scenarios, such as videos with illumination changes, shadows, night scenes, and dynamic backgrounds.

With the availability of a huge amount of data and the development of powerful computational infrastructure, deep

neural networks (DNNs) [30]–[32] have shown remarkable improvements in MOD problems and are developed to replace either background modeling or pixel classification in traditional methods or to combine these two steps into an end-to-end network. Existing DNN models are mostly supervised approaches based on 2D convolutional neural networks (CNNs) [33]–[50], 3D CNNs [51]–[56], 2D separable CNNs [57], or generative adversarial networks (GANs) [58]–[63]. Besides, unsupervised GANs [64], [65] and semi-supervised networks are also proposed [66]–[73]. It demonstrates that the DNNs can automatically extract spatial low-, mid-, and high-level features as well as temporal features, which turn out to be very helpful in MOD problems.

While existing DNN models offer superior moving object detection accuracy, they suffer from computationally expensive and memory-intensive issues. In particular, the architecture change in 3D CNNs leads to a huge increase in model size and computational complexity compared to 2D CNNs, making it challenging to apply those models to real-world scenarios, such as robotics, self-driving cars, and augmented reality. These tasks are usually deployed on mobile and embedded devices, which have limited memory and computing resources. Besides, these tasks are delay-sensitive and need to be carried out in a timely manner, which cannot be achieved by high-complexity deep learning models. Thus, we aim to design a deep moving object detection model suitable for mobile and embedded environment, that can achieve faster inference speed and smaller model size while maintaining high detection accuracy.

In this paper, we propose an efficient 3D separable convolutional neural network with a multi-input multi-output strategy called “3DS_MM”. This model is tailored for computation-resource-limited and delay-sensitive applications. Compared to state-of-the-art models, it significantly increases inference speed and reduces model size, meanwhile increasing detection accuracy or maintaining a competitive detection accuracy. Our key contributions are as follows:

- We propose a new 3D separable CNN for moving object detection. The proposed network adopts 3D convolution to explore spatio-temporal information in the video data and to improve detection accuracy. To reduce computational complexity and model size, the 3D convolution is decomposed into a depthwise convolution and a pointwise convolution. While existing 3D separable CNN schemes all addressed other problems such as gesture recognition, force prediction, 3D object classification or reconstruction, our work applied it to the moving object detection task for the first time in the literature.
- We propose a multi-input multi-output (MIMO) strategy. While existing networks are single-input single-output, multi-input single output, or two-input two-output, our MIMO network can take multiple input frames and output multiple binary masks using temporal-dimension in each sample. This MIMO embedded in 3D separable CNN can further increase model inference speed significantly and maintain high detec-

tion accuracy. To the best of our knowledge, this is the first time in the literature that such kind of MIMO scheme is used in the MOD task.

- We demonstrate that the proposed 3DS_MM offers overwhelmingly high inference speed in frames per second (154 fps) and extremely small model size (1.45 MB), while achieving the best detection accuracy in terms of F-measure, S-measure, E-measure, and MAE among all models in scene dependent evaluation (SDE) setup and achieving the best detection accuracy among the models with inference speeds exceeding 65 fps in scene independent evaluation (SIE) setup. The SDE setup is widely used to tune and test the model on a specific video as the training and test sets are from the same video. The SIE setup originally raised in [50] is specifically designed to assess the generalization capability of the model on completely unseen videos.

The rest of the paper is organized as follows. In Section II, we introduce existing algorithms for moving object detection. In Section III, we explain the principles of the 3D separable convolution which lays the foundation for our proposed 3DS_MM. In Section IV, we elaborate on our proposed network in detail. Section V explains the training and evaluation setup of the experiments. Section VI describes our experimental results compared to the state-of-the-art models. Section VII concludes the paper.

II. RELATED WORKS

The methods for MOD problems have been extensively studied and improved over the years. These methods can be broadly categorized into: (1) traditional methods (unsupervised learning), and (2) deep learning methods (supervised and semi-supervised learning).

Traditional methods [7]–[29] are unsupervised which do not require labeled ground truth. They basically consist of two components: (1) background modeling which initializes the background scene and updates it over time, and (2) classification which classifies each pixel to be foreground or background. There are many background modeling schemes, such as the temporal or adaptive filters being applied to build the background like running average background [10], temporal median filtering [11], and Kalman filtering [12]. Another way for background modeling is to statistically represent the background using parametric probability density functions such as a single Gaussian or a mixture of Gaussians [13]. On the other hand, non-parametric methods directly rely on observed data to model the background such as IUTIS-5 [14], WeSamBE [15], SemanticBGS [16], and kernel density estimation [17]. Sample consensus is another non-parametric strategy used in PAWCS [18], ViBe [19] and SuBSENSE [20]. In particular, SuBSENSE uses a feedback system to automatically adjust the background model based on the local binary similarity pattern (LBSP) features and pixel intensities [21]. Eigen-background based on principal-component analysis (PCA) [22]–[24] is also used in back-

ground modeling. Further, background subtraction based on robust principal-component analysis (RPCA) [25]–[29] solves camera motion and reduces the curse of dimensionality and scale. However, it is quite difficult for traditional methods to perform object detection in complex scenarios, such as videos with illumination changes, shadows, night scenes, and dynamic backgrounds.

Deep learning-based methods are mostly supervised and have been recently proposed for MOD problems [30]–[32], [42], [44]. The first work based on CNNs is ConvNet-GT [33], which replaces the pixel classification component with a well-defined network structure. The background is estimated by a temporal median filter, then the estimated backgrounds are stacked with the original video frames to form the input of the CNN that outputs the binary masks of detected objects. DeepBS [40] utilizes SuBSENSE [20] algorithm to generate background image and multiple layers CNN for segmentation. Also, a spatial-median filter is used for post-processing to perform smoothing. Wang *et al.* [34] proposed a multi-scale patch-wise method with a cascade CNN architecture called MSCNN+Cascade [34]. Although it achieves good detection performance, the patch-wise processing is very time consuming. Other multi-scale feature learning-based models such as Guided Multi-scale CNN [35], MCSCNN [36], MsEDNet [37] and VGG-16 [74] based networks FgSegNet_M [38] and FgSegNet_v2 [39] were also proposed. FgSegNet_S [38] is a 2D CNN that takes each video frame at its original resolution scale as the input, while its extended version FgSegNet_M [38] takes each video frame at three different resolution scales in parallel as the input of the encoding network. FgSegNet_v2 is the best-performing FgSegNet model in CDnet2014 [75] challenge. Another example, MSFgNet [41], has a motion-saliency network (MSNet) that estimates the background and subtracts it from the original frames, followed by a foreground extraction network (FgNet) that detects the moving objects.

3D convolution is applied to MOD problems to utilize spatial-temporal information in visual data. In [52], 3D CNN and a fully connected layer are adopted in a patch-wise method. 3D-CNN-BGS [53] uses 3D convolution to track temporal changes in video sequences. This approach performs 3D convolution on 10 consecutive frames of the video, and upsamples the low-, mid-, and high-level feature layers of the network in a multi-scale approach to enhance segmentation accuracy. 3DAtrous [54] captures long-term temporal information in the video data. It is trained based on a long short-term memory (LSTM) network with focal loss to tackle the class imbalance problem commonly seen in background subtraction. Another LSTM-based example is the autoencoder-based 3D CNN-LSTM [55] combining 3D CNNs and LSTM networks. In this work, time-varying video sequences are handled by 3D convolution to capture short temporal motions, while the long short-term temporal motions are captured by 2D LSTMs. Although these 3D convolution-based methods offer accurate detection results, they have high computational complexity.

Recently, the concept of generative adversarial networks (GAN) is adopted in MOD problems, such as BScGAN [58], BSGAN [59], BSPVGAN [60], FgGAN [61], BSIsGAN [62], and RMS-GAN [63]. BScGAN is based on conditional generative adversarial network (cGAN) that consists of two networks: generator and discriminator. BSGAN [59] and BSPVGAN [60] are based on Bayesian GANs. They use median filter for background modeling and Bayesian GANs for pixel classification. The use of Bayesian GANs can address the issues of sudden and slow illumination changes, non-stationary background, and ghost. In addition, BSPVGAN [60] exploits parallel vision to improve results in complex scenes. In [64], [65], adversarial learning is proposed to generate dynamic background information in an unsupervised manner.

However, the performance of all the aforementioned deep learning-based moving object detection methods comes at a high computational cost and a slow inference speed due to complex network structures and intense convolution operations. To reduce the amount of calculation, our previous work [57] proposed to use 2D separable CNN which splits the standard 2D convolution into a depthwise convolution and a pointwise convolution. It dramatically increases the inference speed and maintains high detection accuracy. However, this 2D separable CNN-based network does not exploit the temporal information in the video input.

In this work, we extend the 2D separable CNN to a 3D separable CNN, which reduces the computational complexity compared to standard 3D CNN. Although some existing works [76]–[79] adopt 3D separable CNN to extract high-dimensional features, none of them applied it to the problem of moving object detection. For example, the 3D separable CNN in [76] is for hand-gesture recognition, in which the last two layers of the network are fully connected layers that output class labels. The 3D separable CNN in [77] is used for two tasks: 3D object classification and reconstruction. Neither task utilizes temporal data, hence no temporal convolution is involved. The 3D separable CNN in [78] is to predict interactive force between two objects, hence its network output is a scalar representing the predicted force value. This problem essentially is a regression problem. Besides, the way that the 3D convolution is separated in [78], [79] is different from our proposed method. It first conducts channel-wise 2D convolution for each independent frame and channel, then conducts joint temporal-channel-wise convolution. In contrast, our proposed 3D separable CNN performs spatial-temporal convolution first, then performs pointwise convolution along the channel direction.

Another factor that limits the inference speed is the input-output relationship. The input-output relationship of existing moving object detection networks has two types: (1) single-input single-output (SISO), which is widely exploited in 2D CNNs such as FgSegNet_S [38] and 2D separable CNN [57]; and (2) multi-input single-output (MISO) which can be found in 3D CNNs such as 3D-CNN-BGS [53], 3DAtrous [54], and DMFC3D [51]. The disadvantage of SISO and MISO is that

they result in a slow inference speed because only one frame output is predicted in every forward pass. Recently, the X-Net [80] adopts a two-input two-output network structure, which takes two adjacent video frames as the network input and generates the corresponding two binary masks. Although it can track temporal changes, the network structure is inflexible and the temporal correlation it utilizes is limited. In this work, we propose a multi-input multi-output (MIMO) strategy, which can take multiple input frames and output multiple frames of binary masks in each sample. It explores temporal correlations on a larger time span and significantly increases the inference speed when embedded in 3D separable CNN.

Another issue for supervised methods is the generalization capability of the trained models on completely unseen videos. Several moving object detection models were designed and evaluated over completely unseen videos, such as BMN-BSN [47], BSUV-Net [48], BSUV-Net 2.0 [49], BSUV-Net+SemBGS [48], ChangeDet [50], and 3DCD [56]. Besides, semi-supervised networks were also designed to be extended to unseen videos. For example, GraphBGS [66] and GraphBGS-TV [67] are based on the reconstruction of graph signals and semi-supervised learning algorithm, MSK [68] is based on a combination of offline and online learning strategies, and HEGNet [71] combines propagation-based and matching-based methods for semi-supervised video moving object detection.

In this paper, we devise a new lightweight 3D separable CNN specifically for moving object detection in computation-resource-limited and delay-sensitive scenarios. It has an efficient end-to-end encoder-decoder structure with a multi-input multi-output (MIMO) strategy, named as the “3DS_MM”. The proposed 3DS_MM does not require explicit background modeling. We evaluate the model over CDnet2014 [75] dataset in an SDE framework with other state-of-the-art models, and we also assess the generalization capability of the model over CDnet2014 and DAVIS2016 [81] datasets in SIE setups over completely unseen videos.

The proposed 3DS_MM significantly increases the inference speed, reduces the trainable parameters, computational complexity and model size, meanwhile achieving the highest detection accuracy in SDE setup and maintaining a competitive detection accuracy in SIE setup.

III. 3D SEPARABLE CONVOLUTION

In this section, we elaborate on the rationale of the 3D separable convolution operation, which is the building block of our proposed 3DS_MM. In the following sections, we use the default data format “NLHWC” in Tensorflow to represent data, which denotes the batch size N , the temporal length L , the height of the image H , the width of the image W , and the number of channels C .

A. 2D CONVOLUTION VS. 3D CONVOLUTION

As shown in Fig. 1(a) [82], an ordinary 2D convolution takes a 3D tensor of size $H \times W \times C_i$ as the input, where H and W are the height and width of feature maps, and C_i is the

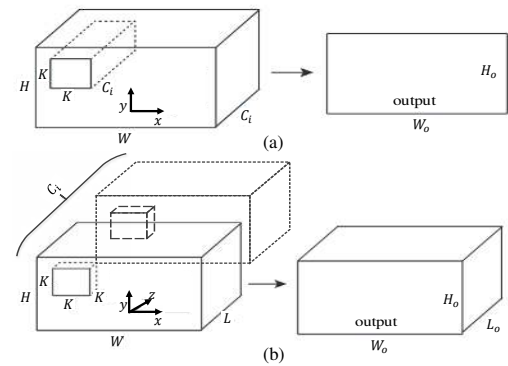


FIGURE 1. Illustration of (a) the 2D convolution with 3D input and (b) the 3D convolution with 4D input.

number of input channels. In this case, the filter is a 3D filter in a shape of $K \times K \times C_i$ moving in two directions (y, x) to calculate a 2D convolution. The output is a 2D matrix of size $H_o \times W_o$. If the filter number is C_o , the output shape will be $H_o \times W_o \times C_o$. The mathematical expression of such 2D convolution is given by

$$Out[h, w] = \sum_{j=0}^{K-1} \sum_{i=0}^{K-1} \sum_{c=0}^{C_i-1} f[j, i, c] \times In[h - j, w - i, c] \quad (1)$$

where In represents the 3D input to be convolved with the 3D filter f to result in a 2D output feature map Out . Here, h, w and c are the height, width, and channel coordinates of the 3D input, while j, i and c are those of the 3D filter.

However, for video signal the 2D convolution in Fig. 1(a) does not leverage the temporal information among adjacent frames. 3D convolution addresses this issue using 4D convolutional filters with 3D convolution operation, as illustrated in Fig. 1(b). In a 3D convolution, the “input” becomes C_i channels of 3D tensors of size $L \times H \times W$, where L is the temporal length (i.e. the number of successive video frames). Hence, the input is 4D and is of size $L \times H \times W \times C_i$. A 4D convolutional filter of size $K \times K \times K \times C_i$ moves in 3 directions (z, y, x) to calculate convolutions, where z, y , and x align with the temporal length, height, and width axes of the 4D input. The output shape is $L_o \times H_o \times W_o$. If the filter number is C_o , the output shape will be $L_o \times H_o \times W_o \times C_o$. The mathematical expression of the 3D convolution with a 4D input is given by

$$Out[l, h, w] = \sum_{k=0}^{K-1} \sum_{j=0}^{K-1} \sum_{i=0}^{K-1} \sum_{c=0}^{C_i-1} f[k, j, i, c] \times In[l - k, h - j, w - i, c] \quad (2)$$

where In represents the 4D input to be convolved with the 4D filter f to result in a 3D output Out . Here, l, h, w , and c are the temporal length, height, width, and channel coordinates of the 4D input, while k, j, i and c are those of the 4D filter. If the size of the filter is $K \times K \times K \times C_i$, then the indices k, j, i range from 0 to $K - 1$, and c ranges from 0 to $C_i - 1$.

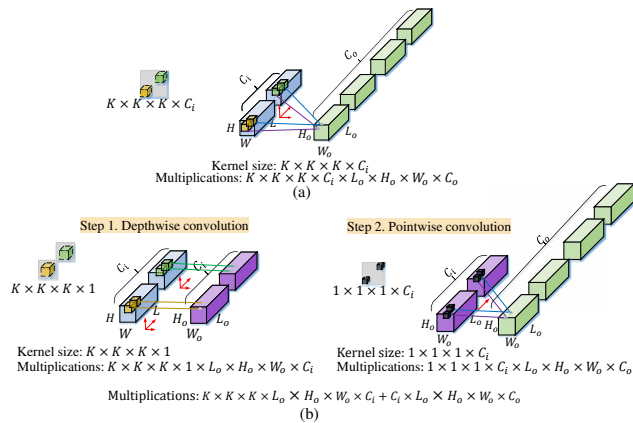


FIGURE 2. Illustration of (a) the standard 3D convolution and (b) the 3D separable convolution. Red arrows point to effective directions of the convolution calculation of the 3D filters.

The ability to leverage the temporal context improves moving object detection accuracy. However, 3D CNN is rarely used in practice because it suffers from a high computational cost due to the increased amount of computation used by 3D convolutions, especially when the dataset scale goes larger and the neural network model goes deeper. Thus, in order to make use of the temporal features, a low-complexity 3D CNN must be developed.

B. 3D CONVOLUTION VS. 3D SEPARABLE CONVOLUTION

2D separable convolution splits traditional 2D convolution into a depthwise convolution and a pointwise convolution, which drastically reduces computational complexity [57], [83]–[85].

In order to utilize temporal features in video data, the idea of separable convolution can be applied to the standard 3D convolution. As shown in Fig. 2 (a), in the standard 3D convolution, the 4D input of size $L \times H \times W \times C_i$, is convolved with C_o filters of size $K \times K \times K \times C_i$, resulting in a 4D output of size $L_o \times H_o \times W_o \times C_o$. The filters calculate the 3D convolution by moving in the directions of length, height, and width as shown by the red arrows. The computational complexity of such standard 3D convolution is $K \times K \times K \times C_i \times L_o \times H_o \times W_o \times C_o$.

To simplify the 3D convolution, we decompose it into a 3D depthwise convolution and a 1D pointwise convolution. As shown in Fig. 2 (b) Step 1, the 3D depthwise convolution adopts C_i independent filters of size $K \times K \times K \times 1$ to perform a 3D convolution on each input channel. This procedure is described in (3). The required multiplications of such 3D depthwise convolution is $K \times K \times K \times 1 \times L_o \times H_o \times W_o \times C_i$.

$$Out[l, h, w, c] = \sum_{k=0}^{K-1} \sum_{j=0}^{K-1} \sum_{i=0}^{K-1} f[k, j, i, c] \times In[l - k, h - j, w - i, c], c = 1, 2, \dots, C_i. \quad (3)$$

Afterwards, the output of Fig. 2 (b) Step 1 is used as the input of Fig. 2 (b) Step 2, where the pointwise convolution adopts a filter of size $1 \times 1 \times 1 \times C_i$, performs a linear projection along the channel axis as shown by the red arrow, and outputs a 3D tensor of size $L_o \times H_o \times W_o$. This procedure is described in (4). Using C_o such filters outputs C_o 3D tensors. The required multiplications of such 1D pointwise convolution is $1 \times 1 \times 1 \times C_i \times L_o \times H_o \times W_o \times C_o$.

$$Out[l, h, w] = \sum_{s=0}^{C_i-1} f[s] \times In[l, h, w, c - s]. \quad (4)$$

The combination of the 3D depthwise convolution and the 1D pointwise convolution, called 3D separable convolution, achieves a reduction in computational complexity of

$$\begin{aligned} ratio &= \frac{3D \text{ separable convolution}}{3D \text{ convolution}} \\ &= \frac{K \times K \times K \times L_o \times H_o \times W_o \times C_i + C_i \times L_o \times H_o \times W_o \times C_o}{K \times K \times K \times C_i \times L_o \times H_o \times W_o \times C_o} \quad (5) \\ &= \frac{1}{C_o} + \frac{1}{K^3}. \end{aligned}$$

With $K = 3$ and a large C_o , the computational complexity can be reduced by roughly 27 times compared to the standard 3D convolution.

This work adopts such 3D separable convolution in a moving object detection network for the first time. It substantially reduces the amount of computation, meanwhile extracting temporal features in the video sequence.

IV. PROPOSED 3DS_MM NETWORK

The proposed deep moving object detection network shown in Fig. 3 is based on two major designs: (1) the encoder-decoder-based 3D separable CNN and (2) the multi-input multi-output (MIMO) strategy. This section describes the proposed approach in detail.

A. ENCODER-DECODER-BASED 3D SEPARABLE CNN

As shown in Fig. 3, the proposed network is an encoder-decoder-based CNN utilizing the 3D separable convolution as described in Section III. The network involves six blocks in the encoder network and three blocks in the decoder network. These block numbers are selected to provide a good trade-off between the inference speed and the detection accuracy empirically. Table 1 shows the details of the network and the shape of the input and output in each layer.

1) The Encoder Network

For each training sample, the input to the encoder network is a set of video frames in a 4D shape of $9 \times H \times W \times 3$ without background frame needed, where 9 is the number of video frames, H and W are the height and width of the video frames, and 3 is the RGB color channels. In Fig. 3, $t_0, t_1, t_2, t_3, t_4 \dots$ represent different time slots. In the first step, the standard 3D convolution described in Fig. 2(a) is

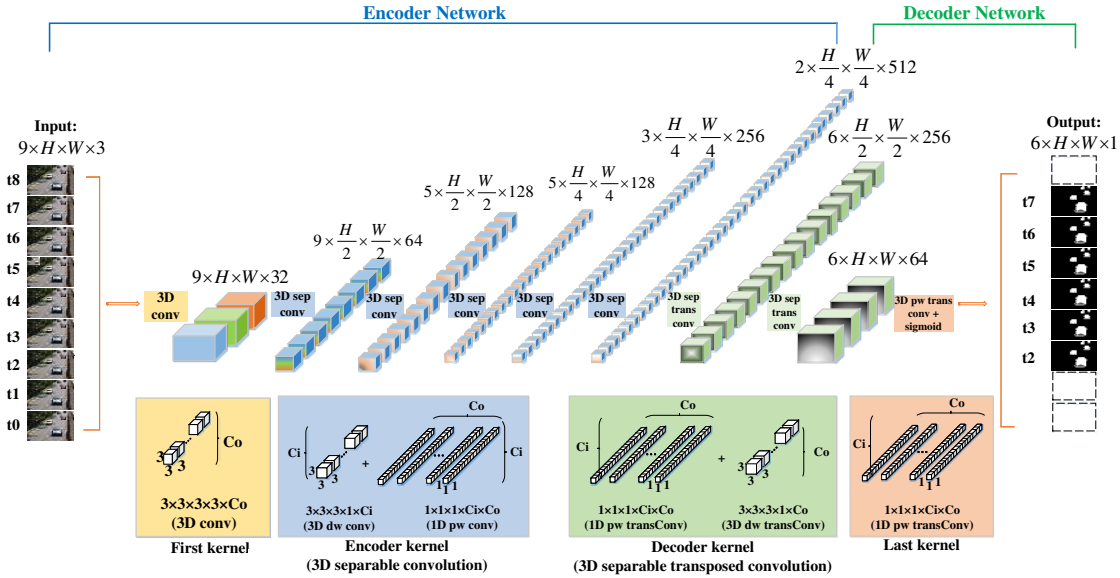


FIGURE 3. The architecture of the proposed 3DS_MM.

TABLE 1. The proposed network configuration. The encoder consists of blocks 0 to 5, and the decoder consists of blocks 6 to 8.

	Layer Type / Stride	(Filter Shape) × Filters	Output Shape	
Encoder	block 0	Conv3D / s=[1,1,1]	$9 \times H \times W \times 3$ $9 \times H \times W \times 32$	
	block 1	Conv3D dw / s=[1,2,2]	$(3 \times 3 \times 3 \times 1) \times 32$	$9 \times \frac{H}{2} \times \frac{W}{2} \times 32$
		Conv3D pw / s=[1,1,1]	$(1 \times 1 \times 1 \times 32) \times 64$	$9 \times \frac{H}{2} \times \frac{W}{2} \times 64$
	block 2	Conv3D dw / s=[2,1,1]	$(3 \times 3 \times 3 \times 1) \times 64$	$5 \times \frac{H}{2} \times \frac{W}{2} \times 64$
		Conv3D pw / s=[1,1,1]	$(1 \times 1 \times 1 \times 64) \times 128$	$5 \times \frac{H}{2} \times \frac{W}{2} \times 128$
	block 3	Conv3D dw / s=[1,2,2]	$(3 \times 3 \times 3 \times 1) \times 128$	$5 \times \frac{H}{4} \times \frac{W}{4} \times 128$
		Conv3D pw / s=[1,1,1]	$(1 \times 1 \times 1 \times 128) \times 128$	$5 \times \frac{H}{4} \times \frac{W}{4} \times 128$
	block 4	Conv3D dw / s=[2,1,1]	$(3 \times 3 \times 3 \times 1) \times 128$	$3 \times \frac{H}{4} \times \frac{W}{4} \times 128$
		Conv3D pw / s=[1,1,1]	$(1 \times 1 \times 1 \times 128) \times 256$	$3 \times \frac{H}{4} \times \frac{W}{4} \times 256$
	block 5	Conv3D dw / s=[2,1,1]	$(3 \times 3 \times 3 \times 1) \times 256$	$2 \times \frac{H}{4} \times \frac{W}{4} \times 256$
Conv3D pw / s=[1,1,1]		$(1 \times 1 \times 1 \times 256) \times 512$	$2 \times \frac{H}{4} \times \frac{W}{4} \times 512$	
Decoder	block 6	Conv3DTrans pw/s=[3,2,2]	$(1 \times 1 \times 1 \times 512) \times 256$	$6 \times \frac{H}{2} \times \frac{W}{2} \times 256$
		Conv3D dw/s=[1,1,1]	$(3 \times 3 \times 3 \times 1) \times 256$	$6 \times \frac{H}{2} \times \frac{W}{2} \times 256$
	block 7	Conv3DTrans pw/s=[1,2,2]	$(1 \times 1 \times 1 \times 256) \times 64$	$6 \times H \times W \times 64$
		Conv3D dw / s=[1,1,1]	$(3 \times 3 \times 3 \times 1) \times 64$	$6 \times H \times W \times 64$
	block 8	Conv3DTrans pw/s=[1,1,1]	$(1 \times 1 \times 1 \times 64) \times 1$	$6 \times H \times W \times 1$
		Sigmoid Activation		$6 \times H \times W \times 1$

The output shape is in data format “LHWC”, where L is the temporal length, H is the height, W is the width, C is the number of channels, dw represents “depthwise convolution”, pw represents “pointwise convolution”, and s represents the strides in temporal length, height, and width.

adopted with 32 filters of size $3 \times 3 \times 3 \times 3$ to calculate the convolution on nine input frames. The input video frames are transformed to 32 feature maps in a shape of $9 \times H \times W \times 32$

at the output. In the following blocks, each of the output feature maps of each layer is convolved with an independent filter of size $3 \times 3 \times 3 \times 1$ with strides $[1, 2, 2]$ (in the direction of temporal length, height, width) for depthwise convolution, and then convolved with C_o filters of size $1 \times 1 \times 1 \times C_i$ with strides $[1, 1, 1]$ for pointwise convolution.

2) The Decoder Network

The output of the encoder network is fed to the decoder network for decoding to produce the binary masks of the moving objects.

Each layer of the decoder network adopts a transposed convolution, which spatially upsamples the encoded features and finally generates the binary masks at the same resolution as the input video frames.

The standard transposed convolution is split into a 1D pointwise transposed convolution and a 3D depthwise transposed convolution. These operations are defined similarly to the 1D pointwise convolution and the 3D depthwise convolution in the encoder network. In block 6 shown in Table 1, the encoder output of size $2 \times \frac{H}{4} \times \frac{W}{4} \times 512$ is converted to a tensor of size $6 \times \frac{H}{2} \times \frac{W}{2} \times 256$ using the 1D pointwise transposed convolution with 256 filters of size $1 \times 1 \times 1 \times 512$. By setting strides to be $[3, 2, 2]$ for the temporal length, height and width in the pointwise transposed convolution, the feature maps are up-scaled by 3 times from 2 to 6 in the temporal length and enlarged by 2 times in height and width. Then followed by a 3D depthwise transposed convolution with 256 filters of size $3 \times 3 \times 3 \times 1$ and strides $[1, 1, 1]$, the feature maps are projected to a tensor of size $6 \times \frac{H}{2} \times \frac{W}{2} \times 256$ at the output of block 6. Block 7 is similarly defined. In the final block, the feature maps are projected to a 4D output of size $6 \times H \times W \times 1$, and a sigmoid activation function is appended to generate the probability masks for 6

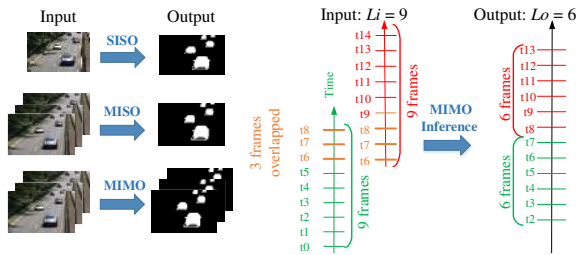


FIGURE 4. Left: Difference between Single-Input Single-Output (SISO), Multi-Input Single-Output (MISO), and Multi-Input Multi-Output (MIMO). Right: The proposed MIMO strategy used in the inference process.

successive frames. A threshold of 0.5 is applied to convert the probability masks to binary masks that indicate the detected moving objects.

B. MIMO STRATEGY

Fig. 4 illustrates our proposed MIMO strategy and how it is different from SISO and MISO. The temporal-dimension L of a 4D input or output of size $L \times H \times W \times C$ is redefined as the number of input frames L_i and the number of output masks L_o . By applying different padding and stride values in the convolutions in the neural network, different number of output masks L_o can be predicted. In our study, we set L_i as 9 and L_o as 6. As shown in Fig. 4 (right), in the inference process, two groups of 9 input frames with 3 frames overlapped can output two successive groups of 6 binary masks.

We also analyze how computational complexity can be reduced from MISO to this MIMO scheme. Let us consider our proposed network in Table 1. With the proposed MIMO scheme, the output layer in block 8 is of size $L_o \times H_o \times W_o \times (C_o = 1)$. Since block 8 mainly requires a pointwise convolution, the multiplications required to generate such output layer is $1 \times 1 \times 1 \times C_i \times L_o \times H_o \times W_o \times (C_o = 1) = C_i \times L_o \times H_o \times W_o$. Denote the total multiplications from block 0 to block 7 as M_{0-7} , then the overall complexity of generating L_o binary masks is

$$M_{0-7} + C_i \times L_o \times H_o \times W_o. \quad (6)$$

With the same network structure, if we adopt a MISO scheme, then the output layer is of size $(L_o = 1) \times H_o \times W_o \times (C_o = 1)$. The multiplications involved in block 8 to generate such output layer is $1 \times 1 \times 1 \times C_i \times (L_o = 1) \times H_o \times W_o \times (C_o = 1) = C_i \times H_o \times W_o$. To generate L_o output binary masks, the overall complexity is

$$(M_{0-7} + C_i \times H_o \times W_o) \times L_o = M_{0-7} \times L_o + C_i \times L_o \times H_o \times W_o. \quad (7)$$

Therefore, to output the same number of binary masks, MISO requires $(7) - (6) = (L_o - 1) \times M_{0-7}$ more multiplications than MIMO.

V. TRAINING AND EVALUATION OF THE PROPOSED MODEL

To analyze how the proposed model performs, we conducted three experiments illustrated in Table 2: (1) video-optimized SDE setup on CDnet2014 dataset, (2) category-wise SIE setup on CDnet2014 dataset, and (3) complete-wise SIE setup on DAVIS2016 dataset. In SDE [50], frames in training and test sets were from the same video, whereas, in SIE [50], completely unseen videos were used for testing. Further, in category-wise SIE, the training and testing were done per category over CDnet2014, whereas, in complete-wise SIE, training and testing were done over the complete DAVIS2016 dataset.

All the experiments were carried out on an Intel Xeon with an 8-core 3GHz CPU and an Nvidia Titan RTX 24G GPU. The following sections present the details of the training and evaluation processes and performance evaluation metrics.

A. VIDEO-OPTIMIZED SDE SETUP ON CDNET2014 DATASET















The CDnet2014 dataset [75] was used in the experiment. It contains 11 video categories: baseline, badWeather, shadow, and so on. Each category has four to six videos, resulting in a total of 53 videos (e.g., the baseline category has sequences highway, office, pedestrians, and PETS2006). A video contains 900 to 7,000 frames. The spatial resolution of the video frames varies from 240×320 to 576×720 pixels. In our experiments, we excluded the PTZ (pan-tilt-zoom) category since the camera has excessive motion.

We trained deep learning-based methods DeepBS [40], MSFgNet [41], VGG-PSL-CRF [42], BSPVGAN [60], RMS-GAN [63], MSCNN+Cascade [34], MsEDNet [37], FgSegNet_S [38], FgSegNet_M [38], FgSegNet_v2 [39], 2D_Separable CNN [57] and our proposed 3DS_MM in the same video-optimized SDE setup, in which a specific model was trained for each video.

From each video, we selected the first 50% of frames as the training set and the last 50% of frames as the test set. The SISO-based networks and the proposed MIMO-based 3DS_MM were using exactly the same frames for training. Suppose that one video contained 100 frames, then for the SISO-based networks, the first 50 frames $t_0 \sim t_{49}$ were used for training, and the last 50 frames $t_{50} \sim t_{99}$ were used for testing. For our proposed 3DS_MM, a 9-frame window slid over the same first 50% of frames, such as $t_0 \sim t_8$, $t_1 \sim t_9$, $t_2 \sim t_{10}$, ..., $t_{41} \sim t_{49}$ to form the training set if the stride was 1, and $t_{50} \sim t_{99}$ frames were for testing. In this way, all the deep-learning-based models were using the same frames for training. The only difference was that for the proposed network, the first 50% of frames were repeatedly utilized through the sliding operation. The traditional unsupervised methods WeSamBE [15], SemanticBGS [16], PAWCS [18], and SuBSENSE [20] were also tested on the same last 50% frames for performance comparison.

We used the RMSprop optimizer with binary cross-entropy loss function and trained each model for 30 epochs with batch

TABLE 2. Different data division schemes of scene dependent evaluation (SDE) and scene independent evaluation (SIE).

	Training and evaluation methodology		In this paper	Train	Test
SDE (Scene Dependent Evaluation)	Video-optimized		✓	Video1: 	
				Video2: 	
SIE (Scene Independent Evaluation)	Video-agnostic	Category-wise	✓	Category1:   	
		Complete-wise	✓	    	

size 1. The learning rate was initialized at 1×10^{-3} and was reduced by a factor of 10 if the validation loss did not decrease for 5 successive epochs.

B. CATEGORY-WISE SIE SETUP ON CDNET2014 DATASET

In order to evaluate the generalization capability of the proposed 3DS_MM, we also run experiments for the SIE setup. Compared to SDE, in SIE the training and test sets contain a completely different set of videos. In the category-wise SIE setup, the training and evaluation were conducted per category. A leave-one-video-out (LOVO) strategy originally raised in [50] was applied to divide videos in each category into training and test sets for CDnet2014 dataset. For example, the baseline category contains four videos, then three videos (highway, office, PETS2006) were used for training, and the 4th video (pedestrians) was for testing. This SIE setup was carried out on seven categories, so for each method in comparison, seven models were trained totally from scratch.

The traditional unsupervised methods WeSamBE [15], PAWCS [18], and SuSENSE [20] were compared in the category-wise SIE setup. We also compared our proposed 3DS_MM with the other DNN-based networks such as BMN-BSN [47], BSUV-Net [48], BSUV-Net 2.0 [49], and ChangeDet [50] which were demonstrated to have great performance on unseen videos.

We used the RMSprop optimizer with binary cross-entropy loss function and trained the model for 30 epochs with batch size 5. The learning rate was initialized at 1×10^{-3} and was reduced by a factor of 10 if the validation loss did not decrease for five successive epochs.

C. COMPLETE-WISE SIE SETUP ON DAVIS2016 DATASET

We also conducted an experiment in complete-wise SIE setup on DAVIS2016 dataset. Different from the category-wise setup on CDnet2014, the complete-wise setup on DAVIS2016 refers to the training and evaluation on the whole dataset. In our experiment, 30 videos in DAVIS2016 dataset

were used in training, and 10 completely unseen videos were used for testing. For each method in comparison, only one unified model was trained from scratch without using any pre-trained model data.

Semi-supervised deep learning-based methods such as MSK [68], CTN [69], SIAMMASK [70], PLM [73], and HEGNet [71], as well as FgSegNet_S [38], FgSegNet_M [38], FgSegNet_v2 [39], and 2D_Separable CNN [57] were trained and tested in the same SIE setup as our proposed 3DS_MM. We used the same training configuration parameters (optimizer, loss function, epochs, batch size, learning rate, etc.) as those in Section V-B.

D. EVALUATION METRICS

1) Efficiency

To evaluate the efficiency of our proposed model, the inference speed is measured in frames per second (fps), the model size is measured in megabytes (MB), the number of trainable parameters is measured in millions (M), and the computational complexity is measured in floating point operations (FLOPs).

2) Detection Accuracy

To measure the detection accuracy, we adopt four metrics: the region-based F-measure, the structure measure (S-measure) [86], the enhanced alignment measure (E-measure) [87], and the mean absolute error (MAE) [88].

The F-measure is defined as:

$$F\text{-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (8)$$

where $\text{precision} = \frac{TP}{TP+FP}$, $\text{recall} = \frac{TP}{TP+FN}$, given the true positive (TP), false positive (FP), true negative (TN), and false negative (FN).

The S-measure [86] combines the region-aware structural similarity S_r and object-aware structural similarity S_o , which is more sensitive to structures in scenes:

$$S\text{-measure} = \alpha \times S_o + (1 - \alpha) \times S_r, \quad (9)$$

where $\alpha = 0.5$ is the balance parameter.

The E-measure is recently proposed [87] based on cognitive vision studies and combines local pixel values with the image-level mean value in one term, jointly capturing image-level statistics and local pixel matching information.

We also evaluate the MAE [88] between the predicted output and the binary ground-truth mask as:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |Pred_i - GT_i|, \quad (10)$$

where $Pred_i$ is the predicted value of the i -th pixel, GT_i is the ground-truth binary label of the i -th pixel, and N is the total number of pixels.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

A. ABLATION STUDY

We first investigated the influence of different components of our proposed 3DS_MM through ablation experiments. In order to quantify the effect of two components “3D separable CNN” and “MIMO” in 3DS_MM, we conducted four experiments over 10 categories of CDnet2014 dataset in SDE setup. The results are shown in Table 3. We began with the standard 3D CNN and a MISO strategy, namely “3D CNN + MISO”. It has an F-measure of 0.9532, a very low inference speed of 26 fps, approximately 9.13 M trainable parameters, and a computational complexity of 693.31 GFLOPs, which generates 1 output binary mask. To generate 6 output masks, the GFLOPs need to be multiplied by 6 ($\times 6$). We then replaced the standard 3D CNN by the 3D separable CNN, while the MISO strategy was retained. For a fair comparison, the 3D CNN and the 3D separable CNN structures adopted the same number of network layers, and their intermediate layers have the same output sizes. The resultant “3D separable CNN + MISO” method has a slightly reduced F-measure, but the inference speed increased from 26 fps to 31 fps. More importantly, the parameters and FLOPs were drastically reduced, due to the separable convolution operations. On the other hand, we retained the standard 3D CNN but replaced MISO by MIMO. In particular, we kept the front part of the network the same and only modify the last layer to output 6 binary masks instead of a single mask. The resultant method “3D CNN + MIMO” significantly increased the inference speed (144 fps) compared to “3D CNN + MISO”.

Finally, the proposed “3D separable CNN + MIMO” method has a superior inference speed (154 fps) due to the MIMO strategy, as well as the fewest trainable parameters (~ 0.36 M) and FLOPs (~ 28.43 G) due to 3D separable convolutions. The above results have justified the effectiveness of our proposed model design.

B. OBJECTIVE PERFORMANCE EVALUATION

1) Objective Results in Video-Optimized SDE Setup on CDnet2014

The accuracy comparison of various methods in SDE setup in each video category is shown in Table 4. Each row lists

TABLE 3. Ablation study of the proposed 3DS_MM.

Methods	Accuracy \uparrow (F-measure)	Inference Speed \uparrow (fps)	# Param \downarrow (M)	FLOPs \downarrow (G)
3D CNN + MISO	0.9532	26	~ 9.13	$\sim 693.31 (\times 6)$
3D separable CNN + MISO	0.9521	31	~ 0.36	$\sim 28.40 (\times 6)$
3D CNN + MIMO	0.9522	144	~ 9.13	~ 693.97
3D separable CNN + MIMO	0.9517	154	~ 0.36	~ 28.43

#Param: Number of trainable parameters; M: millions; FLOPs: floating point operations, G: gigaflops; ($\times 6$): six times the FLOPs in order to generate the same number of output masks as the ‘MIMO’ strategy.

the inference speed, F-measure, S-measure, E-measure and MAE values for a specific method, each column lists the algorithm category, learning type (supervised or unsupervised learning), input-output relationship (SISO, MISO or MIMO), inference speed, GPU type, and F-measure values averaged on test frames from a certain video category, while the last four columns show the average F-measure, S-measure, E-measure and MAE values across all video categories. The first four classical methods are traditional non-deep learning-based methods. These traditional models are tested on the same last 50% of frames as the other compared models. In the subsequent rows, the results of deep learning-based models, including our proposed model are obtained by training and testing in exactly the same SDE setup as introduced in Section V-A. In Table 4, we highlight the best value in each column in bold. We observe that our proposed 3DS_MM model achieves the highest inference speed at 154 fps, and performs best in BDW-badWeather, DBG-dynamicBackground, IOM-intermittentObjectMotion, LFR-lowFramerate, and Turbulance categories in F-measure. It improved the average F-measure by 1.1% and 1.4% compared to methods with the second and third highest average F-measure values in Table 4. It also offers the highest average S-measure, E-measure, and the lowest average MAE values among all methods.

2) Objective Results in Category-Wise SIE Setup on CDnet2014

Table 5 lists the comparison results in category-wise SIE setup. Each column lists the inference speed and accuracy metrics values calculated on the unseen video being left out from each category for testing in the LOVO strategy. The models FgSegNet_S [38], FgSegNet_M [38], FgSegNet_v2 [39], BMN-BSN [47], BSUV-Net [48], BSUV-Net 2.0 [49], and ChangeDet [50] were trained and evaluated in the same SIE setup introduced in Section V-B as our proposed 3DS_MM. Our proposed 3DS_MM (with an inference speed at 154 fps, an F-measure of 0.8499, an S-measure of 0.8632, an E-measure of 0.9445, and an MAE of 0.0545) outperforms all the other listed methods in inference speed, while maintaining high detection accuracy by outperforming FgSegNet_S, FgSegNet_M, FgSegNet_v2, BMN-BSN, BSUV-Net, and BSUV-Net 2.0 by 26.6%, 34.8%, 24.9%, 7.2%, 2.7%, and 3.9% in F-measure, respectively. It achieves similar superiority in terms of S-measure, E-measure and MAE as well. Although ChangeDet [50] offers relatively

TABLE 4. Comparative F-measure, S-measure, E-measure and MAE performance in video-optimized SDE setup on CDnet2014 dataset.

Method	Algorithms (Learning type, Input-Output)	Inference Speed ↑ (fps)	GPU	Accuracy													
				F-measure ↑								S-measure ↑	E-measure ↑	MAE ↓			
				BDW	BSL	CJT	DBG	IOM	NVD	LFR	SHD	THM	TBL	Avg	Avg	Avg	Avg
WeSamBE [15]	Traditional Methods (unSV)	2	CPU i5	0.8530	0.9293	0.7830	0.7274	0.7256	0.5801	0.6532	0.8492	0.7768	0.7667	0.7644	0.7835	0.8536	0.1423
SemanticBGS [16]		7	Titan	0.8190	0.9488	0.8332	0.9326	0.7742	0.4886	0.7818	0.9050	0.8025	0.6851	0.7971	0.8094	0.8935	0.1002
PAWCS [18]		27	CPU i5	0.8072	0.9277	0.7996	0.8772	0.7628	0.4024	0.6518	0.8719	0.8130	0.6350	0.7549	0.7644	0.8478	0.1453
SubSENSE [20]		30	CPU i5	0.8539	0.9383	0.8006	0.8011	0.6433	0.5471	0.6375	0.8797	0.7977	0.7722	0.7671	0.7790	0.8598	0.1394
VGG-PSL-CRF [42]		4.9	Titan	0.8869	0.9474	0.9276	0.7190	0.7405	0.7398	0.6105	0.8890	0.8352	0.9137	0.8210	0.8398	0.9157	0.0801
DeepBS [40]	Deep CNNs (SV, MISO)	10	Titan	0.8221	0.9460	0.8844	0.8593	0.5962	0.5777	0.5932	0.9116	0.7389	0.8385	0.7768	0.7956	0.8712	0.1184
MSFNet [41]		83.8	Titan	0.8424	0.9091	0.8167	0.8348	0.7669	0.7973	0.8352	0.9151	0.7822	0.8572	0.8357	0.8545	0.9266	0.0613
BSPVGAN[60]		10	Titan	0.9564	0.9717	0.9747	0.9683	0.9230	0.8813	0.8448	0.9732	0.9570	0.9240	0.9380	0.9466	0.9856	0.0123
RMS-GAN [63]		50	Titan	0.9490	0.9658	0.9624	0.9612	0.9342	0.8813	0.9333	0.9262	0.9510	0.9434	0.9407	0.9490	0.9825	0.0155
MSEDNet [37]		13.6	Titan	0.8975	0.9248	0.9027	0.8902	0.8051	-	-	0.9002	0.8621	-	0.8832	0.8897	0.9766	0.0204
MSCNN-Cascade [34]	Multiscale CNNs (SV, MISO)	50	Titan	0.9351	0.9666	0.9612	0.9492	0.8358	0.8837	0.8312	0.9227	0.8764	0.9038	0.9066	0.9190	0.9568	0.0413
FgSegNet_M [38]		69	Titan	0.9307	0.9528	0.9403	0.9136	0.8943	0.8830	0.8897	0.9153	0.9160	0.7964	0.9032	0.9166	0.9789	0.0224
FgSegNet_S [38]		82	Titan	0.9331	0.9608	0.9407	0.9223	0.9045	0.8871	0.9123	0.9197	0.9152	0.7980	0.9095	0.9236	0.9758	0.0241
FgSegNet_v2 [39]		89	Titan	0.9396	0.9680	0.9475	0.9143	0.8985	0.8736	0.9247	0.9152	0.9196	0.8179	0.9119	0.9184	0.9876	0.0112
2D_Separable CNN [57]		2D Separable (SV, SISO)	149	Titan	0.9165	0.9552	0.9401	0.9324	0.9352	0.8459	0.9255	0.9030	0.9067	0.8936	0.9154	0.9304	0.9858
Proposed 3DS_MM	3D Separable (SV, MIMO)	154	Titan	0.9571	0.9704	0.9417	0.9686	0.9637	0.8848	0.9736	0.9432	0.9516	0.9621	0.9517	0.9687	0.9945	0.0067

unSV: unsupervised learning, SV: supervised learning, SISO: single-input single-output, MISO: multi-input single-output, MIMO: multi-input multi-output. The best value in each column is highlighted in bold. ↑ Larger value of the metric denotes better performance. ↓ Smaller value of the metric denotes better performance.

TABLE 5. Comparative F-measure, S-measure, E-measure and MAE performance in category-wise SIE setup for unseen videos on CDnet2014 dataset.

Method	Learning Type, Input-Output	Inference Speed ↑ (fps)	GPU	Accuracy										
				F-measure ↑								S-measure ↑	E-measure ↑	MAE ↓
				blizzard-BDW	pedestrians-BSL	boats-DBG	turnpike5fips-LFR	winterStreet-NVD	busStation-SHD	corridor-THM	Avg	Avg	Avg	Avg
WeSamBE [15]	unSV	2	CPU i5	0.8584	0.9569	0.6401	0.9130	0.5900	0.8628	0.8944	0.8165	0.8198	0.9112	0.0723
PAWCS [18]	unSV	27	CPU i5	0.6612	0.9511	0.8820	0.9072	0.4610	0.8583	0.6489	0.7671	0.7746	0.8003	0.1789
SubSENSE [20]	unSV	30	CPU i5	0.8501	0.9500	0.6893	0.8531	0.4469	0.8577	0.9129	0.7943	0.7990	0.8432	0.1432
BSUV-Net [48]	SV, MISO	6	Titan	0.8195	0.9765	0.9004	0.6802	0.6100	0.9398	0.8350	0.8231	0.8342	0.9109	0.0691
BSUV-Net 2.0 [49]	SV, MISO	29	Titan	0.8310	0.9630	0.8750	0.7077	0.6170	0.8012	0.8743	0.8100	0.8301	0.9032	0.0910
BMN-BSN [47]	SV, MISO	48	Titan	0.8401	0.9523	0.6400	0.6893	0.6122	0.9211	0.7933	0.7783	0.7894	0.8712	0.1213
ChangeDet [50]	SV, MISO	58.8	Titan	0.9484	0.9490	0.9182	0.8492	0.7699	0.7801	0.8350	0.8643	0.8798	0.9484	0.0466
FgSegNet_M [38]	SV, MISO	69	Titan	0.5511	0.7209	0.6857	0.2233	0.4200	0.6051	0.3104	0.5024	0.5232	0.6043	0.3812
FgSegNet_S [38]	SV, SISO	82	Titan	0.7412	0.6478	0.4045	0.5767	0.4500	0.5244	0.7435	0.5840	0.5987	0.6543	0.3778
FgSegNet_v2 [39]	SV, SISO	89	Titan	0.6990	0.6310	0.6189	0.5290	0.4300	0.5415	0.7590	0.6012	0.6281	0.7223	0.2712
Proposed 3DS_MM	SV, MIMO	154	Titan	0.8942	0.9165	0.7998	0.9147	0.7856	0.7978	0.8409	0.8499	0.8632	0.9445	0.0545

unSV: unsupervised learning, SV: supervised learning, SISO: single-input single-output, MISO: multi-input single-output, MIMO: multi-input multi-output. The best value in each column is highlighted in bold. The second best average accuracy values are also highlighted. ↑ Larger value of the metric denotes better performance. ↓ Smaller value of the metric denotes better performance.

TABLE 6. Comparative F-measure, S-measure, E-measure and MAE performance in complete-wise SIE setup for unseen videos on DAVIS2016 dataset.

Method	Learning Type, Input-Output	Inference Speed ↑ (fps)	GPU	Accuracy													
				F-measure ↑								S-measure ↑	E-measure ↑	MAE ↓			
				camel	car-roundab	car-roundab shadow	cows	goat	horsejump-high	kite-surf	paragliding launch	parkour	soapbox	Avg	Avg	Avg	Avg
MSK [68]	semi-SV, MISO	0.5	Titan	0.7350	0.9260	0.9480	0.8120	0.8140	0.8510	0.4380	0.2290	0.8740	0.8420	0.7469	0.7598	0.8068	0.1900
CTN [69]	semi-SV, MISO	4.5	Titan	0.7250	0.7750	0.8670	0.7750	0.7460	0.8660	0.4600	0.2270	0.8820	0.7440	0.7067	0.7123	0.7855	0.2102
PLM [73]	semi-SV, MISO	9.5	Titan	0.6130	0.7140	0.7310	0.7410	0.6940	0.7860	0.4560	0.1810	0.8120	0.6300	0.6358	0.6436	0.6975	0.2890
HEGNet [71]	semi-SV, MISO	12.5	Titan	0.7490	0.7892	0.7798	0.7792	0.7312	0.7402	0.6843	0.7392	0.8029	0.6500	0.7304	0.7489	0.7837	0.2110
SIAMMASK [70]	semi-SV, MISO	78	Titan	0.7480	0.8720	0.9780	0.7720	0.7210	0.6880	0.3260	0.1910	0.8290	0.5470	0.6672	0.6703	0.7182	0.2701
FgSegNet_M [38]	SV, MISO	69	Titan	0.6047	0.4892	0.8704	0.5620	0.4009	0.6199	0.6308	0.8639	0.5190	0.5835	0.6144	0.6265	0.7034	0.2803
FgSegNet_S [38]	SV, SISO	82	Titan	0.6163	0.5194	0.8940	0.5356	0.4063	0.6273	0.6904	0.8738	0.5345	0.5902	0.6288	0.6398	0.7134	0.2511
FgSegNet_v2 [39]	SV, SISO	89	Titan	0.6201	0.5120	0.8744	0.5309	0.4509	0.5940	0.6820	0.8729	0.5029	0.6194	0.6260	0.6379	0.7201	0.2710
2D_Separable CNN [57]	SV, SISO	149	Titan	0.5235	0.5286	0.8304	0.5387	0.4701	0.3815	0.4729	0.8163	0.4818	0.6209	0.5665	0.5934	0.6235	0.3723
Proposed 3DS_MM	SV, MIMO	154	Titan	0.7495	0.7103	0.7849	0.7039	0.7290	0.6103	0.7012	0.8749	0.7693	0.6835	0.7317	0.7492	0.8024	0.2089

semi-SV: semi-supervised learning, SV: supervised learning, SISO: single-input single-output, MISO: multi-input single-output, MIMO: multi-input multi-output. The best value in each column is highlighted in bold. The second best average accuracy values are also highlighted. ↑ Larger value of the metric denotes better performance. ↓ Smaller value of the metric denotes better performance.

better detection accuracy than our model, the inference speed of our model is 2.6 times that of ChangeDet.

3) Objective Results in Complete-Wise SIE Setup on DAVIS2016

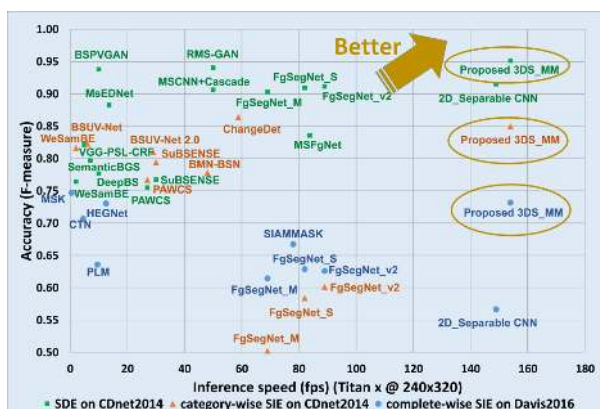
All the models listed in Table 6 were trained and evaluated in the same complete-wise SIE setup as described in Section V-C. It is more challenging for a model to perform well in such SIE setup on DAVIS2016 dataset, because (1) the complete-wise SIE setup mixes 30 different

kinds of videos from the real-world together for training, and (2) the content complexity of DAVIS2016 dataset is high. We compared our proposed model 3DS_MM (with an inference speed at 154 fps and an average F-measure of 0.7317, S-measure of 0.7492, E-measure of 0.8024 and MAE of 0.2089 over 10 test videos) to the state-of-the-art semi-supervised deep learning-based models MSK [68], CTN [69], SIAMMASK [70], HEGNet [71], and PLM [73]. It turns out that our proposed model is superior over these

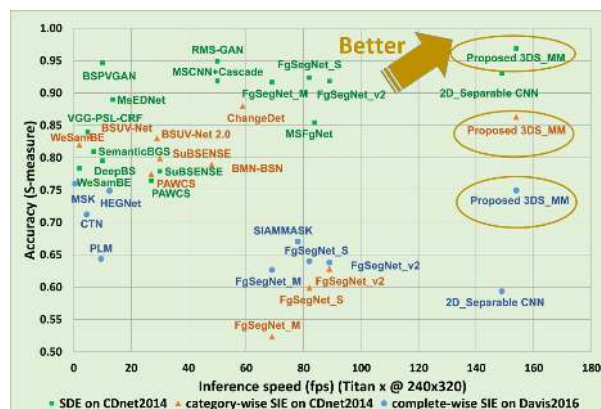
TABLE 7. The comparison between our proposed method and other deep learning-based methods for speed, trainable parameters, computational complexity, model size, and accuracy metrics values. The Table is sorted in ascending order of the inference speed.

Method	Inference Speed (fps)	# Param (M)	FLOPs (G)	Model Size (MB)	SDE				SIE (category-wise)				SIE (complete-wise)			
					F ↑	S ↑	E ↑	MAE ↓	F ↑	S ↑	E ↑	MAE ↓	F ↑	S ↑	E ↑	MAE ↓
MSK [68]	0.5	-	-	-	-	-	-	-	-	-	-	-	0.7469	0.7598	0.8068	0.1900
CTN [69]	4.5	-	-	-	-	-	-	-	-	-	-	-	0.7067	0.7123	0.7855	0.2102
VGG-PSL-CRF [42]	4.9	~ 48.72	~3270 G	127	0.8210	0.8398	0.9157	0.0801	-	-	-	-	-	-	-	-
BSUV-Net [48]	6.0	-	-	116	-	-	-	-	0.8231	0.8342	0.9109	0.0691	-	-	-	-
PLM [73]	9.5	-	-	-	-	-	-	-	-	-	-	-	0.6358	0.6436	0.6975	0.2890
DeepBS [40]	10.0	~ 3.15	~1750 G	28.46	0.7768	0.7956	0.8712	0.1184	-	-	-	-	-	-	-	-
BSPVGAN [60]	10.0	-	-	-	0.9380	0.9466	0.9856	0.0123	-	-	-	-	-	-	-	-
HEGNet [71]	12.5	-	-	-	-	-	-	-	-	-	-	-	0.7304	0.7489	0.7837	0.2110
MsEDNet [37]	13.6	~ 23.29	~1120 G	95	0.8832	0.8897	0.9766	0.0204	-	-	-	-	-	-	-	-
BSUV-Net 2.0 [49]	29.0	~15.90	~540 G	110	-	-	-	-	0.8100	0.8301	0.9032	0.0910	-	-	-	-
BMN-BSN [47]	48.0	-	-	-	-	-	-	-	0.7783	0.7894	0.8712	0.1213	-	-	-	-
MSCNN+Cascade [34]	50.0	~ 10.30	~318 G	76.35	0.9066	0.9190	0.9568	0.0413	-	-	-	-	-	-	-	-
RMS-GAN [63]	50.0	-	-	-	0.9407	0.9490	0.9825	0.0155	-	-	-	-	-	-	-	-
ChangeDet [50]	58.8	~ 0.13	~262 G	1.59	-	-	-	-	0.8643	0.8798	0.9484	0.0466	-	-	-	-
FgSegNet_M [38]	69.0	~ 15.83	~220 G	60.40	0.9032	0.9166	0.9789	0.0224	0.5024	0.5232	0.6043	0.3812	0.6144	0.6265	0.7034	0.2803
SIAMMASK [70]	78.0	-	-	-	-	-	-	-	-	-	-	-	0.6672	0.6703	0.7182	0.2701
FgSegNet_S [38]	82.0	~ 8.16	~199 G	31.20	0.9095	0.9236	0.9758	0.0241	0.5840	0.5987	0.6543	0.3778	0.6288	0.6398	0.7134	0.2511
MSFgNet [41]	83.8	~ 0.29	~193 G	1.48	0.8357	0.8545	0.9266	0.0613	-	-	-	-	-	-	-	-
FgSegNet_v2 [39]	89.0	~ 7.49	~181 G	29.80	0.9119	0.9184	0.9876	0.0112	0.6012	0.6281	0.7223	0.2712	0.6260	0.6379	0.7201	0.2710
Proposed 3DS_MM	154.0	~ 0.36	~ 28.43 G	1.45	0.9517	0.9687	0.9945	0.0067	0.8499	0.8632	0.9445	0.0545	0.7317	0.7492	0.8024	0.2089

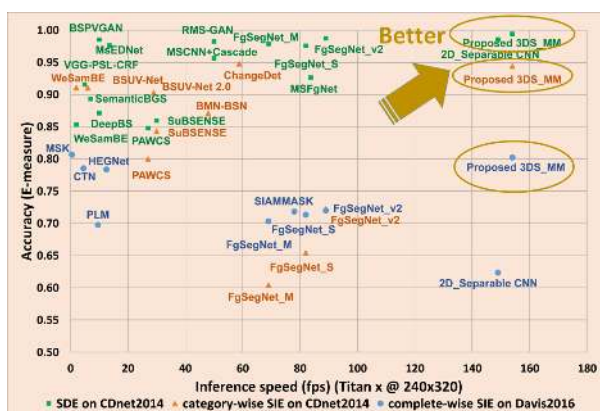
#Param: Number of trainable parameters; M: millions; G: gigaflops; F: F-measure; S: S-measure; E: E-measure; MAE: mean absolute error. The best value in each column is highlighted in bold. The second best accuracy values are also highlighted. ↑ Larger value of the metric denotes better performance. ↓ Smaller value of the metric denotes better performance.



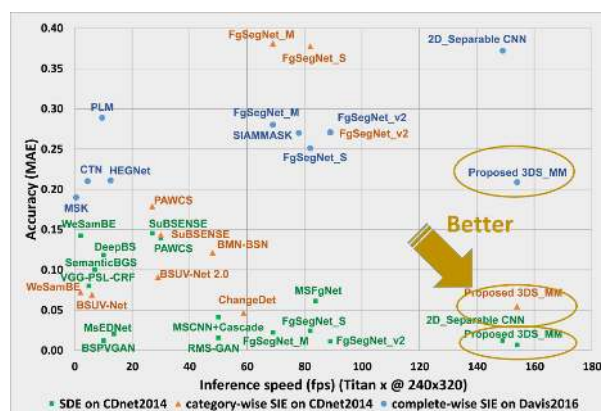
(a) F-measure vs. Inference speed



(b) S-measure vs. Inference speed



(c) E-measure vs. Inference speed



(d) MAE vs. Inference speed

FIGURE 5. Accuracy vs. inference speed (in fps) on an NVIDIA Titan GPU of our proposed model and other compared models in the three experiments (in SDE, category-wise SIE, and complete-wise SIE setup).

models in the inference speed. Besides, our model improved the F-measure by 2.5%, 9.6% and 6.5% compared to CTN,

PLM and SIAMMASK, respectively, and its F-measure is on par with HEGNet. Although MSK offers 1.5% higher F-

measure than ours, its inference speed is extremely low. Our proposed model also outperforms the supervised learning-based models FgSegNet_S [38], FgSegNet_M [38], FgSegNet_v2 [39], and 2D_Separable CNN [57] in F-measure by 10.3%, 11.7%, 10.6%, and 16.5%, respectively. Our proposed method demonstrates a similar superiority in S-measure, E-measure and MAE values. Although there are other models in DAVIS Challenge website with higher detection accuracy than our proposed model, those models are far less efficient and their inference speed is too slow to be applied in delay-sensitive scenarios.

C. ACCURACY, SPEED, MEMORY, AND COMPUTATIONAL COMPLEXITY ANALYSIS

Fig. 5 displays the detection accuracy metrics in F-measure, S-measure, E-measure and MAE versus the inference speed of all the compared models in the SDE setup, category-wise SIE setup, and complete-wise SIE setup. Since we aim at delay-sensitive applications, we expect our proposed 3DS_MM to offer overwhelmingly high inference speed, and a superior detection accuracy among models with high inference speeds. In Fig. 5, we observe that our proposed 3DS_MM surpasses all the other schemes in inference speed in all three experiment setups. In terms of the F-measure, S-measure, E-measure and MAE, in the SDE setup our method is the best among all models, while in both the category-wise and complete-wise SIE setups our method is the best among all models with an inference speed above 65 fps.

In Table 7, we summarize the overall performance including inference speed, trainable parameters, computational complexity, model size, and detection accuracy of our proposed 3DS_MM and other methods. The table is sorted in an ascending order of the inference speed. It is evident that the proposed 3DS_MM outperforms all the other listed methods with the highest inference speed at 154 fps, which is increased by 1.7 times and 1.8 times respectively, compared to the second and third fastest methods in Table 7. The computational complexity and the model size of our proposed method are 28.43 GFLOPs and 1.45 MB, smaller than all the other models in Table 7, due to our proposed 3D separable convolution.

In terms of detection accuracy (F-measure, S-measure, E-measure, and MAE), our proposed model outperforms all other models in SDE setup. In category-wise SIE setup, our proposed method offers the second best accuracy scores. Although it is slightly worse than ChangeDet [50], its inference speed (154 fps) is 2.6 times that of ChangeDet (58.8 fps). In complete-wise SIE setup, although our model offers slightly worse accuracy scores than MSK [68], it offers overwhelming superiority in terms of inference speed. The extremely low inference speed of MSK (0.5 fps) hinders the practical use of this model for delay-sensitive applications.

The number of trainable parameters of our proposed model (~ 0.36 million) is much less than most of the models in comparison. The reason that ChangeDet [50] (~ 0.13 million) and MSFgNet [41] (~ 0.29 million) have fewer trainable

parameters than ours is because they use 2D filters and they are shallower networks with fewer convolutional layers, while our proposed 3DS_MM uses 3D filter and a deeper network. Nevertheless, the inference speeds of ChangeDet and MSFgNet are much slower than ours since they are both MISO networks. In contrast, our 3DS_MM is able to significantly increase the inference speed due to the proposed MIMO strategy and 3D separable convolution.

D. SUBJECTIVE PERFORMANCE EVALUATION

In addition to objective performance, we also provide visual quality comparison as shown in Fig. 6¹, Fig. 7, and Fig. 8.

1) Subjective Results in Video-Optimized SDE setup on CDnet2014

In Fig. 6, we randomly picked a sample test frame from categories BSL-baseline, BDW-badWeather, NVD-nightVideos, and IOM-intermittentObjectMotion. We observe that (1) the proposed 3DS_MM provides more details and clearer edges in the detected foreground objects, such as the car mirrors in “BSL” and “BDW”, and (2) the proposed method detects more contiguous objects such as the bus in “NVD” and the walking man in “IOM”. In contrast, the detected binary masks of other methods in comparison have either blurry edges or missing parts.

2) Subjective Results in Category-Wise SIE setup on CDnet2014

In Fig. 7, we randomly select a sample frame from each of the four categories (BSL-baseline, BDW-badWeather, LFR-lowFramerate, SHD-shadow) of CDnet2014 test results to show the visual quality of the models in Category-Wise SIE setup. Our proposed model has a better generalization capability compared to other models. It shows that our proposed model detects clearer shapes of the persons in BSL and SHD, and detects more details of person legs in SHD. The results of other methods, however, are either noisy, blurry, or have missing parts. In addition, the proposed model performs better in BDW and LFR categories with clear and correct shapes, while other models detect excessive or non-contiguous content.

3) Subjective Results in Complete-Wise SIE setup on DAVIS2016

In Fig. 8, we randomly select four videos (camel, horsejump-high, paragliding-launch, and kite-surf) from the results of DAVIS2016. Our proposed model detects the shapes of objects consistently well for all four videos, while the detection results of 2D_Separable [57], FgSegNet_S [38], FgSegNet_v2 [39], and SIAMMASK [70] are either noisy or incomplete. Besides, the detection results of CTN [69], MSK [68], and PLM [73] for the kite-surf video are less accurate than the proposed model.

¹There are some non-ROI (non-region-of-interest) areas shown as gray color regions in the ground truth images, which were not considered in the training.

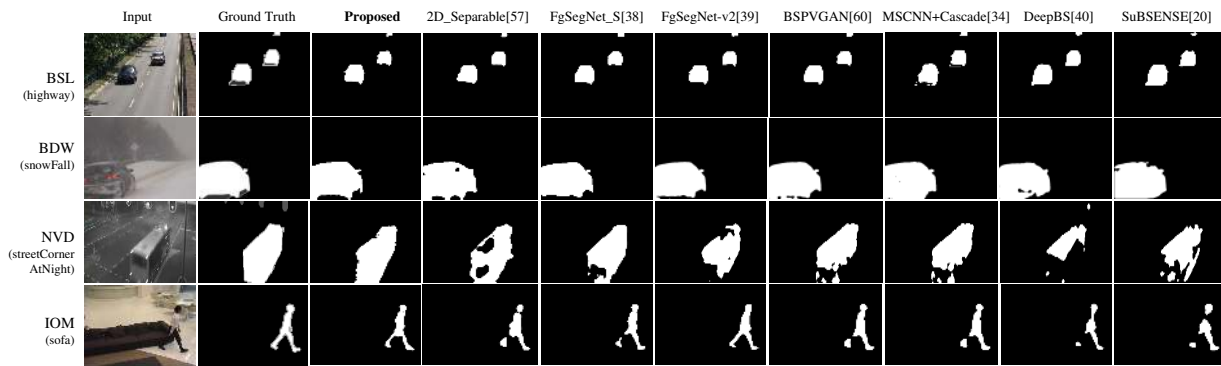


FIGURE 6. Visual comparison of sample results from CDnet2014 dataset in video-optimized SDE setup. BSL: baseline, BDW: badWeather, NVD: nightVideo, IOM: intermittentObjectMotion.

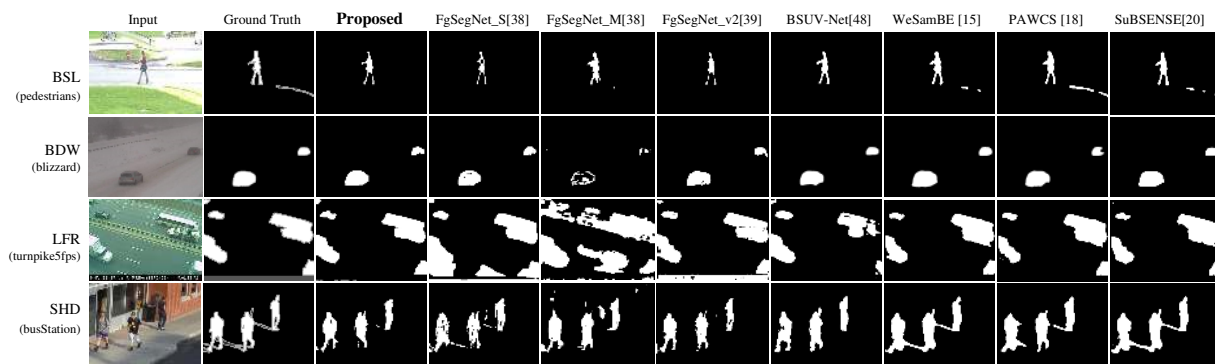


FIGURE 7. Visual comparison of unseen sample results from CDnet2014 dataset in category-wise SIE setup. BSL: baseline, BDW: badWeather, LFR: lowFramerate, SHD: shadow.

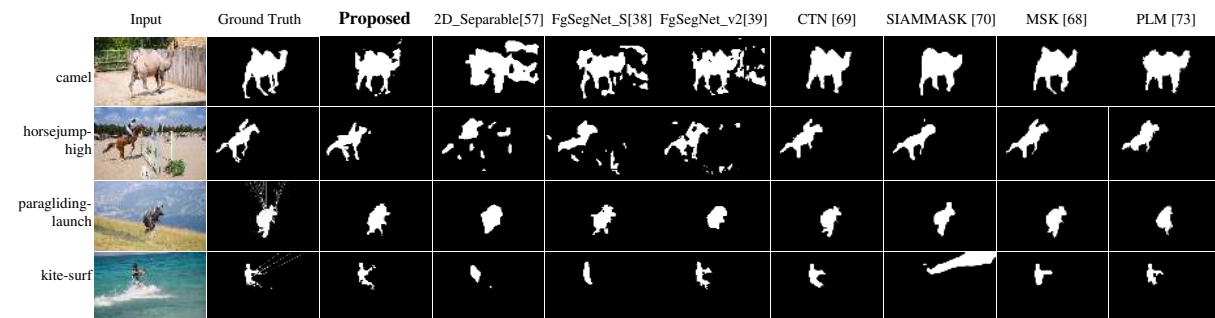


FIGURE 8. Visual comparison of unseen sample results from DAVIS2016 dataset in complete-wise SIE setup.

VII. CONCLUSION

In this paper, we propose the 3DS_MM model for moving object detection. Our model is designed specifically for memory- and computation-resource-limited environments and for delay-sensitive tasks. Our model utilizes spatial-temporal information in the video data via 3D convolution. The proposed 3D depthwise and pointwise convolutions with the MIMO strategy effectively reduce computational complexity and significantly enhance the inference speed. In addition, the 3D separable convolution leads to very few trainable parameters and a small model size. Finally,

the defined SDE and SIE experiments demonstrate that our proposed model achieves superior detection accuracy among all compared models with high inference speeds suitable for low-latency vision applications.

In terms of future study, we plan to use data-augmentation technique to improve the robustness of the proposed model and to further improve the model generalization capability on unseen videos. We will also investigate the potential of feature fusion to improve moving object detection accuracy without reducing the efficiency. Further, we plan to extend the work to semantic segmentation tasks.

REFERENCES

- [1] T. Bouwmans and B. García, "Background subtraction in real applications: Challenges, current models and future directions," *Comput. Sci. Rev.*, vol. 35, p. 100204, 2020.
- [2] S. Brutzer, B. Höferlin, and G. Heidemann, "Evaluation of background subtraction techniques for video surveillance," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1937–1944.
- [3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241.
- [4] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [5] F. Porikli and O. Tuzel, "Human body tracking by adaptive background models and mean-shift analysis," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, Mar 2003, pp. 1–9.
- [6] S. Zhu and L. Xia, "Human action recognition based on fusion features extraction of adaptive background subtraction and optical flow model," *Mathematical Problems in Engineering*, Aug 2015.
- [7] T. Bouwmans, "Recent advanced statistical background modeling for foreground detection: A systematic survey," *Recent Patents on Computer Science*, vol. 4, pp. 147–176, 09 2011.
- [8] S. Javed, P. Narayanamurthy, T. Bouwmans, and N. Vaswani, "Robust PCA and robust subspace tracking: A comparative evaluation," in *IEEE Statistical Signal Processing Workshop (SSP)*, 2018, pp. 836–840.
- [9] S. Bianco, G. Ciocca, and R. Schettini, "How far can you get by combining change detection algorithms?" in *Image Analysis and Processing (ICIAP)*. Springer, 2017, pp. 96–107.
- [10] Y. Zheng and L. Fan, "Moving object detection based on running average background and temporal difference," in *IEEE International Conference on Intelligent Systems and Knowledge Engineering*, 2010, pp. 270–272.
- [11] Q. Zhou and J. Aggarwal, "Tracking and classifying moving objects using single or multiple cameras," in *Handbook of Pattern Recognition and Computer Vision*, Jan 2005, pp. 499–524.
- [12] S. C. Sen-ching and K. Chandrika, "Robust techniques for background subtraction in urban traffic video," in *Visual Communications and Image Processing 2004*, vol. 5308, 2004, pp. 881–892.
- [13] T. S. F. Haines and T. Xiang, "Background subtraction with dirichlet process mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 4, pp. 670–683, 2014.
- [14] S. Bianco, G. Ciocca, and R. Schettini, "Combination of video change detection algorithms by genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 6, pp. 914–928, 2017.
- [15] S. Jiang and X. Lu, "WeSamBE: A weight-sample-based method for background subtraction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2105–2115, 2018.
- [16] M. Braham, S. Piérard, and M. Van Droogenbroeck, "Semantic background subtraction," in *IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 4552–4556.
- [17] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proceedings of the 6th European Conference on Computer Vision-Part II*, ser. ECCV '00, D. Vernon, Ed., pp. 751–767.
- [18] P. St-Charles, G. Bilodeau, and R. Bergevin, "A self-adjusting approach to change detection based on background word consensus," in *IEEE Winter Conference on Applications of Computer Vision*, 2015, pp. 990–997.
- [19] O. Barnich and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709–1724, 2011.
- [20] P. St-Charles, G. Bilodeau, and R. Bergevin, "SuBSENSE: A universal change detection method with local adaptive sensitivity," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 359–373, 2015.
- [21] G. Bilodeau, J. Jodoin, and N. Saunier, "Change detection in feature space using local binary similarity patterns," in *International Conference on Computer and Robot Vision*, 2013, pp. 106–112.
- [22] Y. Liu and D. A. Pados, "Compressed-sensed-domain L1-PCA video surveillance," *IEEE Transactions on Multimedia*, vol. 18, no. 3, pp. 351–363, 2016.
- [23] Y. Liu, Z. Bellay, P. Bradsky, G. Chandler, and B. Craig, "Edge-to-fog computing for color-assisted moving object detection," in *Big Data: Learning, Analytics, and Applications*, F. Ahmad, Ed., vol. 10989, International Society for Optics and Photonics. SPIE, 2019, pp. 9–17.
- [24] Y. Liu, K. Tountas, D. A. Pados, S. N. Batalama, and M. J. Medley, "L1-subspace tracking for streaming data," *Pattern Recognition*, vol. 97, p. 106992, 2020.
- [25] S. E. Ebadi, V. G. Ones, and E. Izquierdo, "Dynamic tree-structured sparse RPCA via column subset selection for background modeling and foreground detection," in *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3972–3976.
- [26] S. Javed, A. Mahmood, S. Al-Maadeed, T. Bouwmans, and S. K. Jung, "Moving object detection in complex scene using spatiotemporal structured-sparse RPCA," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 1007–1022, 2019.
- [27] S. Javed, T. Bouwmans, and S. K. Jung, "Improving OR-PCA via smoothed spatially-consistent low-rank modeling for background subtraction," in *Proceedings of the Symposium on Applied Computing*, 04 2017.
- [28] P. Narayanamurthy and N. Vaswani, "A fast and memory-efficient algorithm for robust PCA (MEROP)," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4684–4688.
- [29] P. Rodríguez and B. Wohlberg, "Translational and rotational jitter invariant incremental principal component pursuit for video background modeling," in *IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 537–541.
- [30] T. Minematsu, A. Shimada, H. Uchiyama, and R. Taniguchi, "Analytics of deep neural network-based background subtraction," *Journal of Imaging*, vol. 4, p. 78, 06 2018.
- [31] T. Minematsu, A. Shimada, and R. Taniguchi, "Rethinking background and foreground in deep neural network-based background subtraction," in *IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 3229–3233.
- [32] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, "Deep neural network concepts for background subtraction: A systematic review and comparative evaluation," *Neural Networks*, vol. 117, pp. 8–66, 2019.
- [33] M. Braham and M. Van Droogenbroeck, "Deep background subtraction with scene-specific convolutional neural networks," in *International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2016, pp. 1–4.
- [34] Y. Wang, Z. Luo, and P.-M. Jodoin, "Interactive deep learning method for segmenting moving objects," *Pattern Recognition Letters*, vol. 96, pp. 66–75, 2017.
- [35] X. Liang, S. Liao, X. Wang, W. Liu, Y. Chen, and S. Z. Li, "Deep background subtraction with guided learning," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2018, pp. 1–6.
- [36] J. Liao, G. Guo, Y. Yan, and H. Wang, *Multiscale Cascaded Scene-Specific Convolutional Neural Networks for Background Subtraction: 19th Pacific-Rim Conference on Multimedia, Hefei, China, September 21-22, 2018, Proceedings, Part I*, 09 2018, pp. 524–533.
- [37] P. W. Patil, S. Murala, A. Dhall, and S. Chaudhary, "MsEDNet: Multi-scale deep saliency learning for moving object detection," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2018, pp. 1670–1675.
- [38] L. A. Lim and H. Yalim Keles, "Foreground segmentation using convolutional neural networks for multiscale feature encoding," *Pattern Recognition Letters*, vol. 112, pp. 256–262, 2018.
- [39] L. A. Lim and H. Yalim Keles, "Learning multi-scale features for foreground segmentation," *Pattern Analysis and Applications*, vol. 23, no. 3, pp. 1369–1380, Aug 2019.
- [40] M. Babaee, D. T. Dinh, and G. Rigoll, "A deep convolutional neural network for background subtraction," *Pattern Recognition*, Sep 2017.
- [41] P. W. Patil and S. Murala, "MSFgNet: A novel compact end-to-end deep network for moving object detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 11, pp. 4066–4077, 2019.
- [42] Y. Chen, J. Wang, B. Zhu, M. Tang, and H. Lu, "Pixelwise deep sequence learning for moving object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 9, pp. 2567–2579, 2019.
- [43] C. Zhao, T. Cham, X. Ren, J. Cai, and H. Zhu, "Background subtraction based on deep pixel distribution learning," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2018, pp. 1–6.
- [44] C. Zhao and A. Basu, "Dynamic deep pixel distribution learning for background subtraction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4192–4206, 2020.
- [45] K. Lim, W. Jang, and C. Kim, "Background subtraction using encoder-decoder structured convolutional neural network," in *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2017, pp. 1–6.

- [46] S. Choo, W. Seo, D. Jeong, and N. I. Cho, "Multi-scale recurrent encoder-decoder network for dense temporal classification," in *24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 103–108.
- [47] V. Mondéjar-Guerra, J. Rouco, J. Novo, and M. Ortega, "An end-to-end deep learning approach for simultaneous background modeling and subtraction," in *British Machine Vision Conference*, 2019, p. 266.
- [48] M. O. Tezcan, P. Ishwar, and J. Konrad, "BSUV-Net: A fully-convolutional neural network for background subtraction of unseen videos," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 2763–2772.
- [49] M. O. Tezcan, P. Ishwar, and J. Konrad, "BSUV-Net 2.0: Spatio-temporal data augmentations for video-agnostic supervised background subtraction," *IEEE Access*, vol. 9, pp. 53 849–53 860, 2021.
- [50] M. Mandal and S. K. Vipparthi, "Scene independency matters: An empirical study of scene dependent and scene independent evaluation for cnn-based change detection," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.
- [51] Y. Wang, Z. Yu, and L. Zhu, "Foreground detection with deeply learned multi-scale spatial-temporal features," *Sensors*, vol. 18, p. 4269, 12 2018.
- [52] Y. Gao, H. Cai, X. Zhang, L. Lan, and Z. Luo, "Background subtraction via 3D convolutional neural networks," in *International Conference on Pattern Recognition (ICPR)*, 2018, pp. 1271–1276.
- [53] D. Sakkos, H. Liu, J. Han, and L. Shao, "End-to-end video background subtraction with 3D convolutional neural networks," *Multimedia Tools and Applications*, vol. 77, pp. 23 023–23 041, 2017.
- [54] Z. Hu, T. Turki, N. Phan, and J. T. L. Wang, "A 3D Atrous convolutional long short-term memory network for background subtraction," *IEEE Access*, vol. 6, pp. 43 450–43 459, 2018.
- [55] T. Akilan, Q. J. Wu, A. Safaei, J. Huo, and Y. Yang, "A 3D CNN-LSTM-Based image-to-image foreground segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 959–971, 2020.
- [56] M. Mandal, V. Dhar, A. Mishra, S. K. Vipparthi, and M. Abdel-Mottaleb, "3DCD: Scene independent end-to-end spatiotemporal feature learning framework for change detection in unseen videos," *IEEE Transactions on Image Processing*, vol. 30, pp. 546–558, 2021.
- [57] B. Hou, Y. Liu, and N. Ling, "A super-fast deep network for moving object detection," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.
- [58] M. C. Bakkay, H. A. Rashwan, H. Salmane, L. Khoudour, D. Puig, and Y. Ruichek, "BScGAN: Deep background subtraction with conditional generative adversarial networks," in *IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 4018–4022.
- [59] W. Zheng and K. Wang, "Background subtraction algorithm with bayesian generative adversarial networks," *Zidonghua Xuebao/Acta Automatica Sinica*, vol. 44, 05 2018.
- [60] W. Zheng, K. Wang, and F.-Y. Wang, "A novel background subtraction algorithm based on parallel vision and Bayesian GANs," *Neurocomputing*, vol. 394, pp. 178–200, 2020.
- [61] P. Patil and S. Murala, "FgGAN: A cascaded unpaired learning for background estimation and foreground segmentation," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019, pp. 1770–1778.
- [62] M. Sultana, A. Mahmood, T. Bouwmans, and S. K. Jung, "Dynamic background subtraction using least square adversarial learning," in *IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 3204–3208.
- [63] P. W. Patil, A. Dudhane, and S. Murala, "End-to-end recurrent generative adversarial network for traffic and surveillance applications," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14 550–14 562, 2020.
- [64] M. Sultana, A. Mahmood, T. Bouwmans and S. K. Jung, "Unsupervised adversarial learning for dynamic background modeling," in *International Workshop on Frontiers of Computer Vision*, 04 2020, pp. 248–261.
- [65] M. Sultana, A. Mahmood, S. Javed, and S. K. Jung, "Unsupervised deep context prediction for background foreground separation," *Machine Vision and Applications*, vol. 30, pp. 375–395, 2018.
- [66] J. H. Giraldo and T. Bouwmans, "GraphBGS: Background subtraction via recovery of graph signals," 2020. [Online]. Available: arXiv: 2001.06404
- [67] J. H. Giraldo and T. Bouwmans, "Semi-supervised background subtraction of unseen videos: Minimization of the total variation of graph signals," in *IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 3224–3228.
- [68] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, "Learning video object segmentation from static images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3491–3500.
- [69] W. Jang and C. Kim, "Online video object segmentation via convolutional trident network," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7474–7483.
- [70] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1328–1338.
- [71] C.-H. Shih and W.-J. Tsai, "Hierarchical embedding guided network for video object segmentation," in *IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 1124–1128.
- [72] X. Lu, W. Wang, J. Shen, Y. Tai, D. J. Crandall, and S. C. H. Hoi, "Learning video object segmentation from unlabeled videos," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8957–8967.
- [73] J. S. Yoon, F. Rameau, J. Kim, S. Lee, S. Shin, and I. Kweon, "Pixel-level matching for video object segmentation using convolutional neural networks," in *IEEE International Conference on Computer Vision (ICCV)*, 10 2017.
- [74] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
- [75] Y. Wang, P. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, "CDNet 2014: An expanded change detection benchmark dataset," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 393–400.
- [76] Z. Hu, H. Youmin, J. Liu, b. wu, D. Han, and T. Kurfess, "3D separable convolutional neural network for dynamic hand gesture recognition," *Neurocomputing*, vol. 318, 08 2018.
- [77] R. Ye, F. Liu, and L. Zhang, "3D depthwise convolution: Reducing model parameters in 3D vision tasks," 2018. [Online]. Available: arXiv:1808.01556
- [78] D. Kim, H. Cho, H. Shin, S.-C. Lim, and W. Hwang, "An efficient three-dimensional convolutional neural network for inferring physical interaction force from video," *Sensors*, vol. 19, p. 3579, 08 2019.
- [79] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5534–5542.
- [80] J. Zhang, Y. Li, F. Chen, Z. Pan, X. Zhou, Y. Li, and S. Jiao, "X-Net: A binocular summation network for foreground segmentation," *IEEE Access*, vol. 7, pp. 71 412–71 422, 2019.
- [81] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 724–732.
- [82] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4489–4497.
- [83] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017. [Online]. Available: arXiv: 1704.04861
- [84] A. Ignatov, R. Timofte, S. Ko, S. Kim, K. Uhm, S. Ji, S. Cho, J. Hong, K. Mei, J. Li et al., "AIM 2019 challenge on RAW to RGB mapping: Methods and results," in *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, pp. 3584–3590.
- [85] A. Ignatov, R. Timofte, Z. Zhang, M. Liu, H. Wang, W. Zuo, J. Zhang, R. Zhang, Z. Peng, S. Ren et al., "AIM 2020 challenge on learned image signal processing pipeline," 2020. [Online]. Available: arXiv: 2011.04994
- [86] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [87] D.-P. Fan, C. Gong, Y. Cao, B. Ren, M.-M. Cheng, and A. Borji, "Enhanced-alignment measure for binary foreground map evaluation," in *International Joint Conferences on Artificial Intelligence*, 2018, pp. 698–704.
- [88] D.-P. Fan, Z. Lin, Z. Zhang, M. Zhu, and M.-M. Cheng, "Rethinking RGB-D salient object detection: Models, data sets, and large-scale benchmarks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, p. 2075–2089, May 2021.



BINGXIN HOU (S'20) received the M.S. degree in Imaging Science from the Rochester Institute of Technology in Rochester, New York in 2010. She worked with advisor Dr. Roy. S. Berns in the Munsell Color Science Laboratory. She worked in Hewlett Packard Company (HP) as Color Imaging Scientist from 2011 to 2017. Currently, she is a Ph.D. candidate in Computer Science and Engineering at Santa Clara University. She is working with advisors Dr. Nam Ling and Dr. Ying Liu

on efficient deep network for moving object detection, and video coding for machine vision. Her research interests include deep learning, computer vision, video compression and camera image processing.



YING LIU (S'11-M'13) received the B.S. degree in communications engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2006, the M.S. and Ph.D. degrees in Electrical Engineering from The State University of New York at Buffalo, NY, USA, in 2008 and 2012, respectively. She currently is an Assistant Professor in the Department of Computer Science and Engineering at Santa Clara University, Santa Clara, CA, USA. Her general areas of expertise are

computer vision, machine learning, and signal processing.



NAM LING (S'88-M'90-SM'99-F'08) received the B.Eng. degree from the National University of Singapore, Singapore, in 1981, and the M.S. and Ph.D. degrees from the University of Louisiana at Lafayette, Lafayette, LA, USA, in 1985 and 1989, respectively. From 2002 to 2010, he was an Associate Dean with the School of Engineering, Santa Clara University, Santa Clara, CA, USA. He was the Sanfilippo Family Chair Professor, and is currently the Wilmot J. Nicholson Family

Chair Professor and the Chair with the Department of Computer Science and Engineering, Santa Clara University. He is/was also a Consulting Professor with the National University of Singapore, a Guest Professor with Tianjin University, Tianjin, China, a Guest Professor with Shanghai Jiao Tong University, Shanghai, China, a Cuiying Chair Professor with Lanzhou University, Lanzhou, China, a Chair Professor and Minjiang Scholar with Fuzhou University, Fuzhou, China, a Distinguished Professor with the Xi'an University of Posts and Telecommunications, Xi'an, China, a Guest Professor with the Zhongyuan University of Technology, Zhengzhou, China, and an Outstanding Overseas Scholar with the Shanghai University of Electric Power, Shanghai, China. He has authored or coauthored over 220 publications and seven adopted standard contributions. He has been granted nearly 20 U.S. patents so far. Dr. Ling is an IEEE Fellow due to his contributions to video coding algorithms and architectures. He is also an IET Fellow. He was named as an IEEE Distinguished Lecturer twice and was also an APSIPA Distinguished Lecturer. He was a recipient of the IEEE ICCE Best Paper Award (First Place) and the Umedia Best/Excellent Paper Award three times, six awards from Santa Clara University, four at the University level (Outstanding Achievement, Recent Achievement in Scholarship, President's Recognition, and Sustained Excellence in Scholarship), and two at the School/College level (Researcher of the Year and Teaching Excellence). He was a Keynote Speaker for IEEE APCCAS, VCVF (twice), JCPC, IEEE ICAST, IEEE ICIEA, IET FC Umedia, IEEE Umedia, IEEE ICCIT, and Workshop at XUPT (twice), and a Distinguished Speaker for IEEE ICIEA.

He has served as a General Chair/Co-Chair for IEEE Hot Chips, VCVF (twice), IEEE ICME, Umedia (seven times), and IEEE SiPS. He was an Honorary Co-Chair for IEEE Umedia 2017. He has also served as a Technical Program Co-Chair for IEEE ISCAS, APSIPA ASC, IEEE APCCAS, IEEE SiPS (twice), DCV, and IEEE VCIP. He was a Technical Committee Chair for IEEE CASCOM TC and IEEE TCMM, and has served as a Guest Editor or an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I:REGULAR PAPERS, the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, IEEE ACCESS, Springer JSPS, and Springer MSSP.



LINGZHI LIU (M'07-SM'09) is the Location Manager of US R&D Center, the Head and Chief Architect of Heterogeneous Computing Group of Kuaishou Technology in Palo Alto, CA since 2018. Before joining Kuaishou, he held several manager and professional positions in Alibaba-inc, Realtek USA, Intel Corp. and Futurewei Technologies around the Silicon Valley, California. He also worked in Fujitsu 2005 and Midea Corp from 1998 to 2000. He was a Postdoctoral Researcher in

EE Dept. of University of Washington from 2005 to 2008. He was an adjunct Professor of Wuhan University, China from 2015 to 2018. He received the B.S. degree from Xi'an Jiaotong University and the Ph.D. degree from Shanghai Jiaotong University, China, in 1998 and 2004 respectively. His general interests include neural network algorithm and architecture, multi-media algorithm and implementation, VLSI system, ASIC and FPGA design and channel coding theory. Dr. Liu has published more than 80 patents, 40 papers and 100 proposals to international standards. Dr. Liu was the Panel/Keynote Chair of ICME2013, Track Chair of APSIPA ASC 2010, Review Committee member of ISCAS2010, Session Chair of ASICOM 2007. He served the TPC of conferences including VLSI-SOC 2014, APSIPA ASC 2011&2010, PICOM 2009 and etc. He is a Senior Member of IEEE since 2008.



YONGXIONG REN (S'11-M'16) obtained his Ph.D. (2016) degree in Electrical Engineering at the University of Southern California, Los Angeles. He received his Master (2011) and Bachelor (2008) degrees from Peking University, China and University of Posts and Telecommunications, China, respectively. He is currently a Video Algorithm Architect at Kwai Inc, Palo Alto, focusing on optimization and acceleration of deep learning models. He has more than 130 publications with

over 9000 Google scholar citations. His publication lists contain two book chapters, 5 U.S. patents, more than 60 journal papers, and more than 70 conference papers and invited presentations. He is a recipient of Best Paper Award at IEEE GLOBECOM 2014. His research interests include deep learning, digital signal processing, image processing, and hardware architecture.

...