Deep Atrous Spatial Features based Supervised Foreground Detection Algorithm for Industrial Surveillance Systems

Ajmal Shahbaz, *Member, IEEE*, and Kang-Hyun Jo, *Senior Member, IEEE*

***Abstract*—Camera-based surveillance systems largely perform an intrusion detection task for sensitive areas. The task may seem trivial but is quite challenging due to environmental changes and object behaviors such as those due to night-time, sunlight, IR camera, camouflage, and static foreground objects, etc. Convolutional Neural Network (CNN)-based algorithms have shown promise in dealing with these challenges. However, they are exclusively focused on accuracy. This paper proposes an efficient supervised foreground detection (SFDNet) algorithm based on atrous deep spatial features. The features are extracted using atrous convolution kernels to enlarge the field-of-view (FOV) of a kernel mask, thereby encoding rich context features without increasing the number of parameters. The network further benefits from a residual dense block strategy that mixes the mid and high-level features to retain the foreground information lost in low-resolution high-level features. The extracted features are expanded using a novel pyramid upsampling network. The feature maps are upsampled using bilinear interpolation and pass through a 3x3 convolutional kernel. The expanded feature maps are concatenated with the corresponding mid and low-level feature maps from an atrous feature extractor to further refine the expanded feature maps. The SFDNet showed better performance than high-ranked foreground detection algorithms on the three standard databases. The testing demo can be found at https://drive.google.com/file/d/1z_zEj9Yp7GZeM2gSIwYKvSzQlxMAiarw/view?usp=sharing.**

***Index Terms*—Intelligent surveillance systems, dual-camera sensors, IR camera, Convolutional Neural Networks.**

I. Introduction

SMART cities are employing camera-based surveillance systems for public safety. According to statistics, some smart cities have 20-100 cameras per 1000 people. Earlier surveillance systems require authorized personnel to closely monitor the footage for a possible security breach. Thus, such systems require a high level of focus from security personnel. Thus, it is important to automate said surveillance systems.

Intelligent surveillance systems are taking over their conventional counterparts. They are autonomous due to the implementation of computer vision algorithms. They detect anomalies and alert security personnel. Such systems are widely implemented at borders and in industrial complexes to monitor restricted areas. These camera-based surveillance systems implement a foreground detection algorithm as a building block of high-level tasks, like intrusion detection. Thus, foreground detection directly affects the overall performance of the system.

Foreground detection algorithms aim to segment desired information (termed as foreground) from the background in a scene. Background subtraction and foreground segmentation are used interchangeably. The latter usually refers to CNN based supervised algorithms. The foreground is defined according to the application. It could be a human entering a prohibited area [1], an unattended bag left in an airport [2], or a car parked on a street [3], [4], etc. In other words, it could be any object of interest that requires segmentation from the background.

Foreground detection algorithms are broadly classified as unsupervised and supervised [5]–[7]. The unsupervised algorithms build a concrete background model from the initial frames of a video sequence using low-level features such as intensity, color, or texture. Gaussian Mixture Models (GMM) is a popular choice among unsupervised algorithms. They use a Gaussian to model the background based on mean and covariance [8].

Self-Balanced SENsitivity (SuBSENSE) [9] relies on the observation of pixel values for modeling the background. It exploits the spatial region of the pixel to model the background using a local binary pattern. Pixel-based Adaptive Word Consensus Segmenter (PAWCS) [10], an improved version of SuBSENSE, incorporates an adaptive threshold scheme. SuBSENSE was further improved by introducing a reward/penalty weighing strategy in Weight Sample Background Extractor (WeSamBE) [11].

In Unity There Is Strength (IUTIS) [12] proposes a genetic algorithm that handpicks the best algorithm for the particular video sequence at hand. Multi-Layer Robust Principal Component Analysis (ML-RPCA) [13] exploits low-rank recovery by extracting information using multi-dimensional arrays. These unsupervised algorithms perform well on benchmark datasets, e.g., Change Detection dataset (CDNet) [22]. However, they suffer in practical challenges posed by camera-based surveillance systems [5] such as night time, use of IR camera, and camouflage effect.

Supervised algorithms are trained offline with ground-truth

Manuscript received Month xx, 20xx; accepted Month xx, 20xx. Date of publication Month, xx, 20xx; date of current version Month, xx, 20xx. This work has been supported by the National Research Foundation of Korea (NRF) grant funded by the government (MSIT) (No. 2020R1A2C2008972). (Corresponding author: Kang-Hyun Jo.)

Ajmal Shahbaz is with the School of Electrical and Computer Engineering, University of Ulsan, Ulsan, 44610 South Korea (e-mail: ajmal@islab.ulsan.ac.kr).

Kang-Hyun Jo is with the Department of Electrical and Computer Engineering, University of Ulsan, Ulsan, 44610 South Korea (e-mail: acejo@ulsan.ac.kr).

using Convolutional Neural Networks (CNNs) as the feature extractor [14]–[30]. DBS [14] labels a patch of an image as a foreground or background. The network is trained with a background image as a ground-truth. It uses 50% of a video sequence as training and the remainder to test. The algorithm provides promising prospects and a possible solution to the drawbacks inherent to unsupervised algorithms.

DeepBS [16] trains multiple CNNs with multiscale input images. It trains only one model with 5% of all the frames of the video sequence from the change detection dataset (CDNet). DBS and DeepBS train a CNN with a background model as ground-truth. Thus, such algorithms fail to deal with cluttered scenes. Additionally, CNN weights are learned from scratch, which is time inefficient.

CascadeCNN [16] takes input images in three different scales and feeds them into three CNNs using foreground/background labels as ground-truth. The feature maps are upsampled using bilinear interpolation. It also trains CNN weights from scratch. RT-BGS [18] builds probability-based background and foreground models using a pretrained CNN model.

Several algorithms tried to learn long-term foreground dependencies using spatio-temporal features. M2DC4 [19] proposes a dilated CNN with convolutional long-term short-term memory networks (ConvLSTM). MS-ST [20] employed multi-scale spatio-temporal features using a pretrained CNN and ConvLSTM. MvRF-CNN [28], similar to M2DC4, employed dilated CNN with ConvLSTM for traffic surveillance applications.

These ConvLSTM based algorithms require sequential input with labeled ground-truths for training (70%). For instance, MS-ST used 14 sequences of images, the most that could fit into NVIDIA 1080Ti GPU memory [20]. 3D CNN-LSTM [29] tried to decrease the number of sequential frames to 4 with grayscale input. Pretrained model and 3D convolutional filters were used. Still, it takes 80-120 minutes to train a single model. While pretrained models might be fast to train, the network might not properly learn the foreground object in a particular video. Their deployment into real-time systems is challenging due to the high computational complexity, a large amount of data needed for training, and a high-end hardware requirement.

This paper proposes an efficient supervised foreground detection algorithm based on CNN (SFDNet) and contributes in four folds:

- **Network Design**: SFDNet proposes a new atrous spatial feature extractor (ASFE) and pyramid upsampling network (PUPN). ASFE is designed using standard and atrous convolutional layers to enlarge field-of-view (FOV) without increasing number of parameters. The mid and high-level features are intermixed via residual-dense blocks strategy (RD) to build a global context and retain the foreground information. PUPN is designed as a sandwich of 3x3 convolutions and bilinear interpolation, which ease the gradient flow during training. The mid and low-level features of ASFE are also propagated to PUPN to improve foreground extraction. SFDNet is trained via a hybrid training strategy.
- **Extensive Experiments**: SFDNet is tested on three datasets with more than 180,000 image sequences. The choice of network design is further supported by ablation studies to validate the effectiveness of network design.
- **New Dataset**: A dataset of HD videos in an industrial setting is developed. It is unique as it also provides videos shot with an IR camera at night. The dataset is supported by manually labeled ground-truth images for training the supervised algorithms. The dataset would be made public for the research community.
- **Application Side**: Industrial surveillance systems are crucial for securing sensitive areas. The proposed algorithm adds-on with efficiency and efficacy and can be implemented on low-end hardware. However, the scope of the SFDNet is not limited to the industrial domain. It can be extended to other applications such as airport ground surveillance, etc.

The remainder of the paper is organized as follows: Section II describes the proposed algorithm. Section III proves the effectiveness of the proposed algorithm. The paper is concluded with prospects.

II. Proposed Algorithm

The SFDNet takes an input image of size WxHx3, where W, H, 3 are respectively width, height, and depth/channels of an image as shown in Figure 1. SFDNet modifies the Visual Geometry Group (VGG-16) CNN [23]. There are 13 standard convolutional layers (five blocks) and three fully-connected layers in VGG-16. The number of convolution kernels employed is 64, 128, 256, and 512 in each block. The input is zero padded to maintain spatial dimensions. Rectified Linear Unit (ReLU) is applied after each convolution layer to remove negative values in a feature map.

A. Atrous Spatial Feature Extractor (ASFE)

The mainstream feature extractors for foreground detection can be summarized into three different groups as shown in Figure 2. Each block represents stacked convolutional layers. Figure 2a shows a feature extractor composed of stacked standard convolutional layers. Figures 2b and 2c show the spatio-temporal feature extractors with convLSTM blocks. These extractors employ either a standard or atrous convolutional layers. The difference between mainstream feature extractors and proposed ASFE is evident from Figure 2d. The proposed ASFE employs a combination of stacked standard convolutional layers (SSCL) and stacked atrous convolutional layers (SACL) to increase the FOV. Residual-dense (RD) blocks strategy is applied to intermix the mid and high-level features.

1) Stacked Standard Convolution Layers (SSCL): Like VGG-16, the first two blocks of SFDNet are SSCL with two convolutional layers and max-pooling layers each (Fig. 1). The convolutional layers are stacked to define a block to increase field-of-view (FOV) without increasing trainable parameters. The FOV of a 3x3 convolution kernel is the same as its size. However, the FOV of two stacked 3x3 convolution kernels would be effectively equal to that of a 5x5 convolution

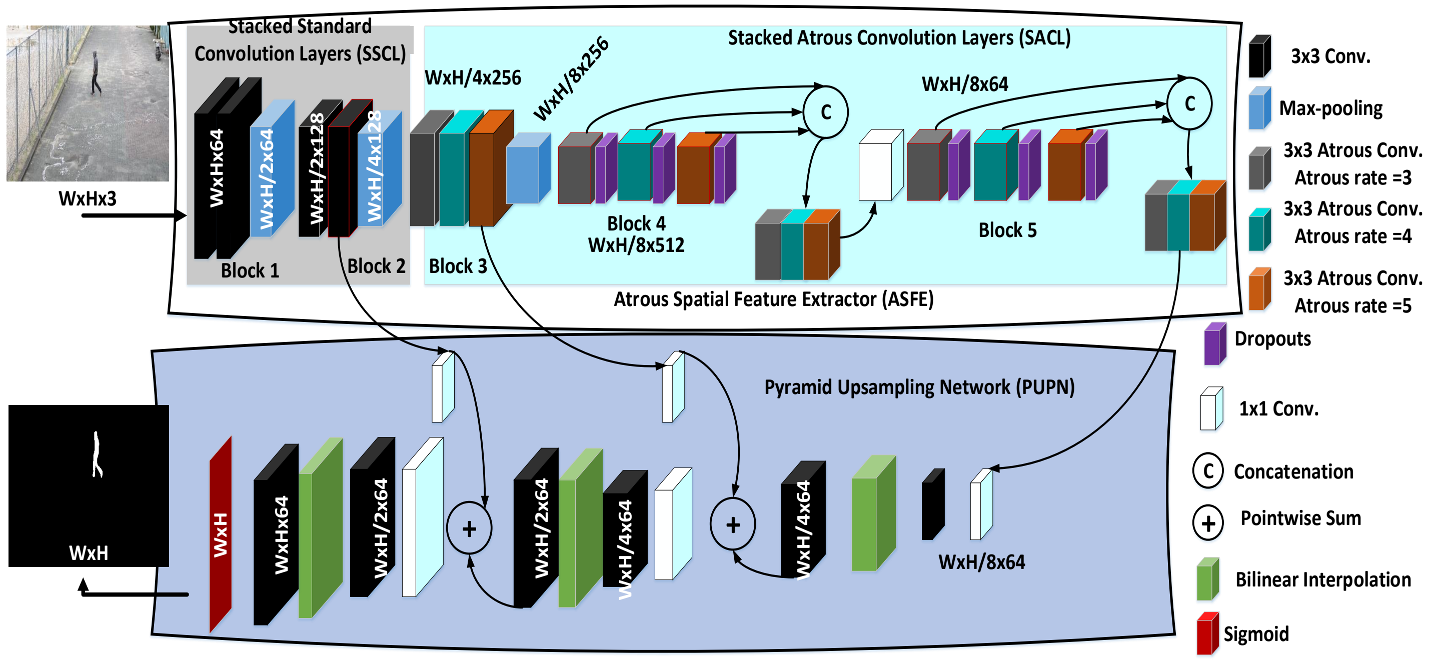
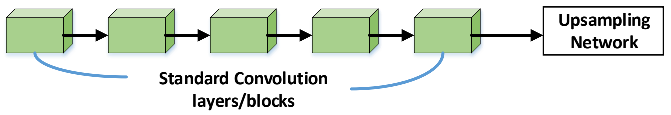
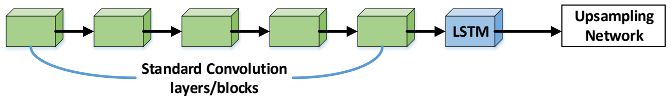

Stacked Standard Convolution Layers (SSCL)
Stacked Atrous Convolution Layers (SACL)
WxHx3
WxHx64
WxH/2x64
WxH/2x128
WxH/4x128
Block 1
Block 2
Block 3
WxH/4x256
WxH/8x256
Block 4
WxH/8x512
WxH/8x64
Block 5
C
Atrous Spatial Feature Extractor (ASFE)
Pyramid Upsampling Network (PUPN)
WxH
WxH/4x64
WxH/8x64
3x3 Conv.
Max-pooling
3x3 Atrous Conv. Atrous rate =3
3x3 Atrous Conv. Atrous rate =4
3x3 Atrous Conv. Atrous rate =5
Dropouts
1x1 Conv.
Concatenation
Pointwise Sum
Bilinear Interpolation
Sigmoid

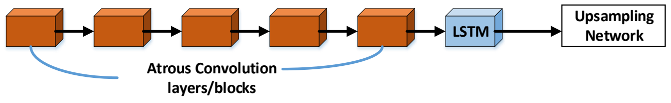
Figure 1. SFDNet with an atrous spatial feature extractor (ASFE) and pyramid upsampling network (PUPN).



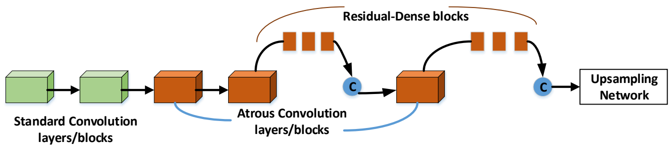
(a) Spatial Feature Extractor [16], [17]



(b) Spatial Feature Extractor with ConvLSTM [20], [29]



(c) Atrous Spatial Feature Extractor with ConvLSTM [19], [28]


Residual-Dense blocks
Upsampling Network
Standard Convolution layers/blocks
Atrous Convolution layers/blocks
C

(d) Proposed Atrous Spatial Feature Extractor (ASFE)

Figure 2. The difference between mainstream feature extractors and the proposed ASFE.

kernel [23]. This decreases the number of trainable parameters appreciably. For example, a 5x5 convolution filter has 25 trainable parameters. But, a stack of two 3x3 convolution filters has 18 parameters [23]. Hence, a feature from the same FOV can be extracted with fewer trainable parameters.

2) Stacked Atrous Convolution Layers (SACL): The function of the 3rd-5th blocks is the same as before, i.e., increasing FOV without increasing trainable parameters. The FOV of three 3x3 convolution kernels is effectively equal to that of a 7x7 convolution kernel [23]. A 7x7 convolution filter has 49 trainable parameters, while, a stack of three 3x3 convolution filters have 27 parameters (45% fewer parameters).

Unlike VGG-16, the 3rd-5th blocks of proposed ASFE are three SACL with varied atrous rates a. The number of convolution kernels is 256, 512, and 64. Motivated from [24, 25], an atrous convolution kernel (ACK) is employed to extract the features. ACK is expanded by inserting zeros in the appropriate positions of the kernel mask. The ACK increases the FOV of a kernel mask without increasing the number of parameters.

The feature maps produced by the ACK are the same size as the input. But, each neuron in the feature map has rich global context information due to a larger FOV [24]. An increase in the FOV of an ACK can be formulated as $[(a\text{-}1)\times(K\text{-}1) + K]$, where K is kernel size and a is atrous rate [25]. Thus, the FOV of a 3x3 ACK with a= 3 is 7x7. Varied atrous rates (a= 3, 4, and 5) are applied on each ACK to increase the FOV to 7x7, 9x9, and 11x11. Atrous rates a were chosen after extensive experimentation. As objects often have various scales in an image, the varied atrous rates help to obtain feature maps with multi-scales forming a feature pyramid [23].

Similarly, the FOV of three stacked convolution kernels can be written as $(K_1+K_2+K_3\text{-}2)$, where K_1, K_2, K_3 are respective atrous kernel sizes [25]. Thus, stacking three atrous convolution layers with FOV 7x7, 9x9, and 11x11 will result in the FOV of 25x25 as compared to the FOV of 7x7 in VGG-16, without an increase in the trainable parameters.

3) Residual-Dense Blocks (RD): Inspired by [26, 28], the atrous convolutional layers in the 4th block are further concatenated channel-wise to form residual-dense block and fed into the 5th block. For example, feature maps of block4conv1 are 1-512. Then, feature maps of block4conv2 would be from 513-1024, and so on. This residual-dense blocks (RD) strategy helps to aggregate a global context and retain foreground information lost due to several convolutional operations on low-resolution high-level feature maps [25]. Furthermore, concatenating features extracted from different FOVs helps to build a better global context via a feature pyramid. The feature maps are squeezed back from 1536 to 64 using 1x1 pointwise convolutions and fed in the 5th block.

Similarly, atrous convolutional layers of 5th block are also concatenated channel-wise and fed into the pyramid upsampling network (PUPN). The low resolution of high-level features often results in decreased pixel-level prediction. Unlike VGG-16, SFDNet removes the max-pooling layers in the 4th and 5th blocks and reduces the input size by 8 times. The spatial dropouts [27] are added after each convolution layer of the 4th and 5th block. It helps to generalize the network (avoid over-fitting) and further decrease trainable parameters by zeroing out the whole feature map.

B. Pyramid Upsampling Network (PUPN)

The final feature maps from the ASFE are expanded to the original size to get the pixel-wise prediction. The design of an upsampling network is crucial for better foreground object detail extraction. Bilinear interpolation (BI) has been widely employed to expand the feature maps for pixel-wise prediction [14]-[17]. BI can be regarded as a simple approach and cannot guarantee the recovery of the foreground object's detail. Another drawback is that it utilizes fixed weights, which cannot be learned during training. This is problematic during back-propagation as the gradient does not flow through an upsampling network.

Later works have utilized transposed convolution to expand feature maps [18, 19, 20, 28, 29]. The feature maps are zero-padded and convolved with a transposed convolution kernel. The weights of the kernel are tuned during training, but this approach results in a checkerboard effect on the expanded feature maps [7].

SFDNet proposes a new pyramid upsampling network (PUPN). The PUPN is designed as BI sandwiched between 3x3 convolutional layers (conv.→BI→conv.). There are three such sandwiched structures in PUPN as shown in Figure 2. BI is applied by a factor of 2 to expand the feature maps. The expanded feature maps are then refined using a 3x3 convolutional kernel. Such a strategy allows a gradient flow during the back-propagation process and mitigates the checkerboard effect. After the first BI, the spatial size increases from $\frac{1}{8}$ to $\frac{1}{4}$. The expanded feature maps are added with the corresponding mid-level features (block3conv3). This helps in two folds. First, the expanded features which have more global features representation from ASFE are mixed with the locally extracted mid-level features. Secondly, it helps to retain the lost foreground information after the application of several convolution layers. The depth of feature maps is again squeezed to 64 via 1×1 pointwise convolution. Then, the spatial dimension is expanded to $\frac{1}{2}$ using BI.

The expanded feature maps are again added with corresponding low-level feature maps (block2conv2) and pass through BI to get the original input size of an image. The depth of feature maps is always squeezed to 64 using a 1×1 pointwise convolution. This helps to control the model size (number of parameters). The ablation study is performed in Section III. C to demonstrate the effectiveness of network design (Table II).

The final layer of the PUPN outputs the probability of a pixel being a foreground using the sigmoid function. The threshold (Th=0.9) is applied to the final layer to get the foreground and background pixels. The threshold is chosen after extensive experimentation. Binary cross entropy loss L_i is employed during training (Equation 1). It compares probabilities of a pixel being a foreground or background with the ground-truth defined as:

$$L_i = \frac{-1}{M}\sum_{j=1}^{M}[y_j^i \log(p_j^i) + (1-y_j^i)\log(1-p_j^i)], \quad (1)$$

where y_j^i is the ground-truth label and p_j^i is the predicted value of the pixel i at location j. M is the total number of pixels in an image.

C. Hybrid Training Strategy

The SFDNet is trained using a hybrid training strategy. The weights of the first two blocks of the ASFE were borrowed from the pretrained VGG model. Such a scheme reduces the number of weights to be learned. It gives the SFDNet a head start and the network trains faster. The intuition is that the initial convolutional layers extract low-level features such as edges, color, corners, etc. [24]. The weights of the 3rd-5th blocks are calibrated via fine-tuning with the specific video or dataset, e.g., CDNet. Such a strategy helps the network to learn deep atrous spatial features according to the specific video or dataset.

III. Experimental Analysis and Results

SFDNet is compared with high-ranked foreground detection algorithms. The unsupervised algorithms used include SuBSENSE [9], PAWCS [10], WeSamBE [11], IUTIS-5 [12], and ML-RPCA [13], whereas DeepBS [16], CascadeCNN [17], RT-BGS [18], M2DC4 [19], MS-ST [20], MvRF-CNN [28] and 3D CNN-LSTM [29] are supervised algorithms.

A. Datasets Description

The SFDNet is tested on three different datasets: the change detection dataset [22], the i-LIDS dataset [21], and Our dataset. The datasets pose practical challenges an industrial surveillance system may face such as illumination changes, dynamic backgrounds, shadows, bad weather, thermal camera, moving camera, camera jitter, infra-red (IR) camera, camouflaged foreground object, static foreground object, scale and speed-variance of an object.

1) Change Detection Dataset (CDNet): CDNet [22] is the standard benchmark for foreground detection. There are 54 videos, each consisting of 900-8000 frames. The image size varies from 320×240-720×480. All the videos accumulate to 150,000 frames with manually labeled ground-truths. There are 11 categories and each category has 4-6 videos. Therefore, the dataset is a good benchmark to test the endurance of the foreground detection algorithms.

Table I
DATASETS DESCRIPTION.

Video	# Training Frames	# Testing	Scenario
i-LIDS dataset			
1	50	950	Day, Normal walk
2	50	950	Day, Running
3	-	1000	Day, Crawling
4	-	1000	Day, Slow walk
5	-	1000	Day, Walking fast
6	50	950	Night, Walking away
7	-	1000	Night, Walking slowly
8	-	1000	Night, Far from camera
9	-	1000	Night, Camouflage intruder
10	-	1000	Night, Camouflage intruder
ISL-ISZM dataset			
11	50	2250	Day, Normal walk
12	-	2300	Day, Walking fast
13	-	2300	Day, Running
14	-	2300	Day, Slow walking
15	-	2300	Day, Multiple intruders
16	50	2250	Day, Normal walk
17	-	2300	Day, Running
18	-	2300	Day, Slow walking
19	-	2300	Day, Dynamic background
20	-	2300	Day, Dynamic background
21	50	950	Night, Walking fast
22	-	1000	Night, Camouflage intruder
23	-	1100	Night, Camouflage intruder
24	-	1100	Night, Camouflage intruder
25	50	1050	Night, Multiple intruder

2) i-LIDS Dataset: The Imagery Library for Intelligent Detection Systems (i-LIDS) dataset [21] is the standard benchmark for video surveillance systems. There are 10 videos with 1000 frames each, five for the day, and five for the night. The image size is 390×220. The scenario is an intruder entering a restricted area and trying to bypass the fence.

3) ISL-ISZM Dataset: Intelligent Systems Laboratory Dataset for Industrial Sterile Zone Monitoring (ISL-ISZM) has 15 videos, 10 for the day, and 5 for the night with 1000-2300 frames. The image size is 720×480. The videos were constructed by mimicking the i-LIDS dataset challenges. The scenario consists of an intruder entering a restricted area in an industrial setting. The dataset and training frames can be found at https://drive.google.com/file/d/1QWCZBa6DIbIK8pOqjsDrqr-lmy0kq7-C/view?usp=sharing. Table I shows a detailed description of the i-LIDS and Our datasets with challenges.

B. Implementation Details

SFDNet is trained on Intel Core i5 Hardware with 8 GB RAM with a low-end NVIDIA GTX570 GPU. It is implemented in Keras [27]. SFDNet was trained for 50 epochs on 50 frames with 20% validation frames, i.e., 10 from 50 training frames. It has 11.2 Million parameters in total with

Table II
ABLATION STUDY USING CDNET WHERE $SSCL$, $SACL$, AND RD ARE STACKED STANDARD CONVOLUTIONAL LAYER, STACKED ATROUS CONVOLUTIONAL LAYER, AND RESIDUAL-DENSE BLOCKS STRATEGY. 1, 2, 3, 4, AND 5 REFER TO THE BLOCK NUMBER.

Atrous Spatial Feature Extractor ($ASFE$)			
$SSCL$	$SACL$	RD	F
1,2,3,4,5	-	X	0.8941
-	1,2,3,4,5	✓	0.9224
1,2,3,4	5	✓	0.9264
1,2,3	4,5	✓	0.9388
1,2	3,4,5	✓	**0.9541**
1	2,3,4,5	✓	0.9407

Pyramid Upsampling Network ($PUPN$)		
Upsampling	Pyramid	F
BIConv	X	0.9541
BI	✓	0.9328
TConv	✓	0.9534
BIConv	✓	**0.9747**

7.6 Million as learnable. The total number of Giga Floating Points Operations (GFLOPs) is 0.66. Regularization method RMSProp and 0.001 learning rate was used. A spatial dropout of 0.4 was applied.

The SFDNet was trained for each video sequence of the CDNet. The training frames and respective ground-truths for the CDNet were provided by CascadeCNN authors[1]. CascadeCNN provided subsets of 50 and 200 training frames. The training and testing details for i-LIDS and our dataset is shown in Table I. The ground-truths (50 training frames) for both datasets were made via the Interactive Segmentation tool[2]. It is assumed that all the possible foreground objects appear in the limited training frames.

C. Ablation Study

Extensive experiments were performed to demonstrate the effectiveness of the network design of the SFDNet using CDNet as shown in Table II. The experiments are split into two levels, i.e., atrous spatial feature extractor (ASFE) and pyramid upsampling network (PUPN). In the ASFE, the network design of stacked standard convolution layers (SSCL), stacked atrous convolution layers (SACL), and residual-dense blocks strategy (RD) were evaluated. In the PUPN design, however, the effectiveness of the upsampling technique and pyramid structure was demonstrated.

Initially, pretrained VGG-16 (1st row) with 5 SSCL blocks were employed. Then, all the VGG-16 blocks were replaced by SACL blocks (2nd row). Later, experiments were performed by supplementing SSCL and SACL blocks together with hybrid training. SFDNet benefits from SSCL, SACL, CT, and hybrid training (5th row). Its F-measure was 5% more than the only SSCL model (1st row).

The pyramid upsampling network (PUPN) was switched with other upsampling techniques (UP), i.e., bilinear interpolation BI (2nd row) and transposed convolution TConv (3rd

[1]https://github.com/zhimingluo/MovingObjectSegmentation
[2]http://www.cs.cmu.edu/ mohitg/segmentation.html

Table III
QUANTITATIVE ANALYSIS ON ALL THE CATEGORIES OF THE CDNET. THE PERFORMANCE METRICS ARE RECALL R, SPECIFICITY Sp, FALSE POSITIVE RATE FPR, FALSE NEGATIVE RATE FNR, PERCENTAGE OF WRONG CLASSIFICATIONS PWC, PRECISION P, AND F-MEASURE F.

Algorithm	# Training Frames	R	Sp	FPR	FNR	PWC	P	F
SFDNet (ours)	50	0.9448	0.9997	0.0002	0.0551	0.1248	0.9733	0.9580
SFDNet (ours)	200	0.9698	**0.9999**	**0.0001**	0.0301	**0.0653**	**0.9807**	**0.9747**
MS-ST	630-5600	0.9650	0.9995	0.0005	0.0350	0.1123	0.9657	0.9671
M2DC4	630-5600	**0.9701**	0.9991	0.00012	**0.0224**	0.246	0.9661	0.9615
3D CNN-LSTM	630-5600	-	-	-	-	-	-	0.94.02
MvRF-CNN	630-5600	-	-	-	-	-	-	0.9489
CascadeCNN	200	0.9506	0.9968	0.0032	0.0494	0.4052	0.8997	0.9209
RT-BGS	-	0.7890	0.9961	0.0039	0.2110	1.0722	0.8305	0.7892
DeepBS	200	0.7545	0.9905	0.0095	0.2455	1.9920	0.8332	0.7458
SuBSENSE	200	0.8124	0.9904	0.0096	0.1876	1.678	0.7509	0.7408
IUTIS-5	200	0.7849	0.9948	0.0052	0.2151	1.1986	0.7717	0.8087

Table IV
F-MEASURE OF EACH CATEGORY OF THE CDNET SUCH AS BASELINE B, BAD WEATHER BW, DYNAMIC BACKGROUND DB, CAMERA JITTER CJ, LOW FRAME RATE LFR, NIGHT VIDEOS NV, PAN TILT ZOOM PTZ, THERMAL Th, SHADOW Sh, INTERMITTENT OBJECT MOTION IOM, AND TURBULENCE T.

Algorithm	B	CJ	DB	IOM	Sh	Th	BW	LF	NV	PTZ	T	Average
SFDNet (ours)	**0.9938**	**0.9857**	**0.9888**	**0.9897**	**0.9938**	0.9808	**0.9879**	**0.9294**	**0.9659**	**0.9817**	0.9273	**0.9747**
MS-ST	0.9895	0.9802	0.9791	0.9893	0.9874	**0.9840**	0.9846	0.9013	0.9390	0.9314	**0.9568**	0.9657
M2DC4	0.9897	0.9645	0.9789	0.9637	0.9813	0.9833	0.9609	0.8994	0.9489	0.9582	0.9488	0.9615
3D CNN-LSTM	0.9470	0.9525	0.9502	-	0.9446	0.8870	0.9583	0.9660	-	-	0.9624	0.9402
MvRF-CNN	0.9632	0.9507	0.9590	0.9660	0.9579	-	0.9480	-	0.8963	-	-	0.9489
CascadeCNN	0.9786	0.9758	0.9658	0.8505	0.9593	0.8958	0.9431	0.8370	0.8965	0.9168	0.9108	0.9209
RT-BGS	0.9604	0.8388	0.9489	0.7878	0.9478	0.8219	0.8260	0.7888	0.5014	0.5673	0.6921	0.7892
DeepBS	0.9580	0.8990	0.8761	0.6098	0.9304	0.7583	0.8301	0.6002	0.5835	0.3133	0.8455	0.7458
SuBSENSE	0.9503	0.8152	0.8177	0.6569	0.8986	0.8171	0.8619	0.6445	0.559	0.3476	0.7792	0.7408
IUTIS-5	0.9567	0.8332	0.8902	0.7296	0.8766	0.8303	0.8248	0.7743	0.5290	0.4282	0.7836	0.7717

Table V
QUANTITATIVE ANALYSIS USING RECALL R, PRECISION P AND F-MEASURE F.

	i-LIDS dataset			Our dataset		
Algorithm	R	P	F	R	P	F
SFDNet (ours)	1.0	1.0	1.0	1.0	1.0	1.0
SuBSENSE	0.80	0.80	0.80	0.73	0.73	0.73
ML-RPCA	1.00	0.50	0.66	1.00	0.50	0.66
DeepBS	0.70	0.70	0.70	0.66	0.66	0.66
RT-BGS	0.80	0.80	0.80	0.86	0.86	0.86

row). The proposed PUPN (4th row) performed 2-5% better than BI and TConv.

D. Quantitative Analysis

1) Change Detection Dataset (CDNet): Table III shows the quantitative analysis of SFDNet and baselines on the CDNet using seven performance metrics. CDNet defines strict pixel-wise performance evaluation as compared to frame-level. Thus, CDNet is better in bench-marking the foreground detection algorithms.

Most supervised algorithms were trained for 200 frames except MS-ST and M2DC4, which were trained for significantly more frames (630-5600). This is due to the use of temporal data for ConvLSTM. Each video in CDNet comprises of 900-8000 frames. Hence, the number of test frames varies from 700-7800. For example, the highway sequence in the baseline category consists of 900 frames. Thus, 200 frames were used for training and 700 frames for testing.

SFDNet gets an average of F of 0.9747 and PWC of 0.0653, which is within the error margin of human annotation accuracy [17]. SFDNet outperformed the baselines in 5 out of 7 performance metrics. However, the difference is quite low, e.g., recall R of M2DC4 (0.9701) is 0.003 more than SFDNet (0.9698). False Negative Rate FNR of M4DC4 (0.0224) is 0.0077 less than SFDNet (0.0301). Despite this, SFDNet performed significantly better than baselines in other categories such as F-measure and precision. 3D CNN-LSTM and MvRF-CNN only reported F-measure.

Table IV shows the category-wise F-measure F of SFDNet and the baselines. SFDNet is significantly better than the baselines on 9 categories of CDNet. SFDNet achieved more than 98% F-measure on the 8 categories. It achieved 96% F-measure on the night category, which is considered the most challenging for video surveillance systems. The turbulence T category is also challenging due to the very small foreground object. 3D CNN-LSTM and MvRF-CNN did not test all the categories of CDNet.

2) i-LIDS Dataset: Table V shows the quantitative analysis of SFDNet and baselines on the i-LIDS dataset. Unlike CDNet, the i-LIDS dataset employs soft frame-level evaluation as compared to pixel-wise prediction. An intruder must be detected for atleast 75% of a video to be marked as successful [21]. For instance, each video weighs 10% of the final F-measure. SuBSENSE detected an intruder in 8 videos. Thus, its F=80%. ML-RPCA detected an intruder in all videos. However, it gave false positives, decreasing its overall performance.

Supervised algorithms suffered due to a strong camouflage

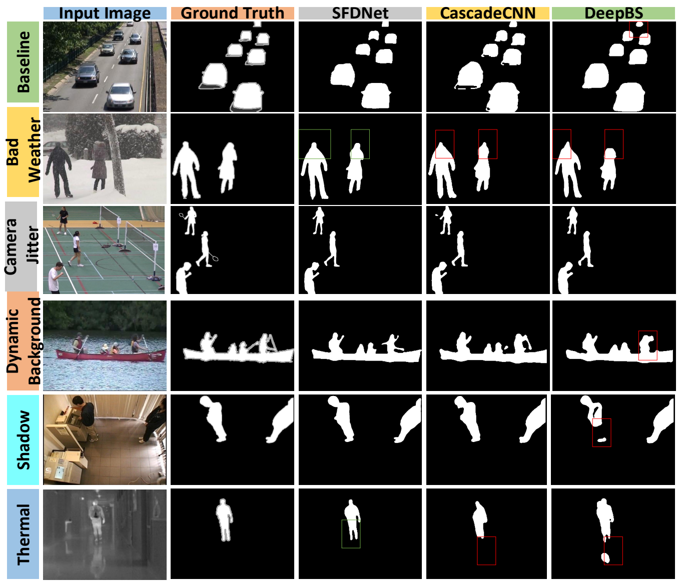


Figure 3. The qualitative comparison of SFDNet and the supervised baselines on the CDNet. The columns show input, ground-truth, and foreground masks of the SFDNet, CascadeCNN, and DeepBS, respectively. Each row shows the different categories of the CDNet such as baseline, bad weather, camera jitters, dynamic background, and shadows, respectively. The red boxes refer to false positives or false negatives from baselines. Green boxes refer to true positives from SFDNet.

effect in night-time videos. RT-BGS and DeepBS detected an intruder in, respectively, 8 and 7 videos. MS-ST and M2DC4 were not evaluated due to their high requirement of labeled training data. SFDNet detected an intruder in all the videos without any false positives.

3) Our Dataset: Table V shows quantitative analysis on our dataset using the i-LIDS dataset criterion. It is more challenging than the i-LIDS dataset. Each video weighs 6.66% of the final F-measure. SuBSENSE detected an intruder in all day-time videos with illumination noise and shadows. But, it detected an intruder in one night video only due to a strong camouflage effect. Thus, its F=73%.

Although ML-RPCA detected the intruder in all day and night videos, it suffered due to illumination changes and the shadow of the intruder, severely decreasing its overall performance. RT-BGS and DeepBS detected the intruder in 13 and 10 sequences, respectively. The SFDNet detected an intruder in all the videos.

E. Qualitative Analysis

1) Change Detection Dataset (CDNet): It is evident from Figure 3 that the SFDNet (3rd column) foreground masks are comparable with the ground-truths. It was able to detect the foreground object precisely. The detection of foreground objects in challenging situations shows promise. The baselines were unable to detect the foreground objects precisely. It is worth mentioning that SFDNet was only trained for 50 training frames, while the baselines might be trained for at least 200 training frames.

CascadeCNN (4th column) detected foreground objects partially in thermal sequences. It was unable to segment the object geometry precisely, e.g., in bad weather (2nd row) and thermal sequence (6th row). DeepBS (5th column) did well on some challenges of the CDNet like baseline and dynamic background categories. It could not extract foreground details. It partially detected the foreground object in shadows (5th row) and thermal (6th row) sequences. It even gave false positives in the thermal sequence (6th row).

2) i-LIDS Dataset: Figure 4a shows the foreground masks from SFDNet and the baselines. Each column depicts different video sequences from day and night. The background setting offers the challenge of illumination changes and dynamic background, while the foreground object comes with the challenge of speed-variance, scale-variance, shadows, camouflage, and static intruder. The IR camera videos are also challenging for baselines. This is because currently available datasets like CDNet do not test algorithms against the IR camera.

The baselines showed similar trends towards the challenges of illumination changes, shadows, and camouflaged intruders. DeepBS (2nd row) performed well in the day sequences. It suffered from the camouflage effect in IR videos. It detected a distinct part of an intruder. ML-RPCA (3rd row) performed well among baselines by detecting the intruders in both datasets. However, it gave false positives owing to illumination changes shadows. RT-BGS (4th row) performed well on the day sequences. It missed the intruders in the night sequences. SFDNet detected intruder with precise object details.

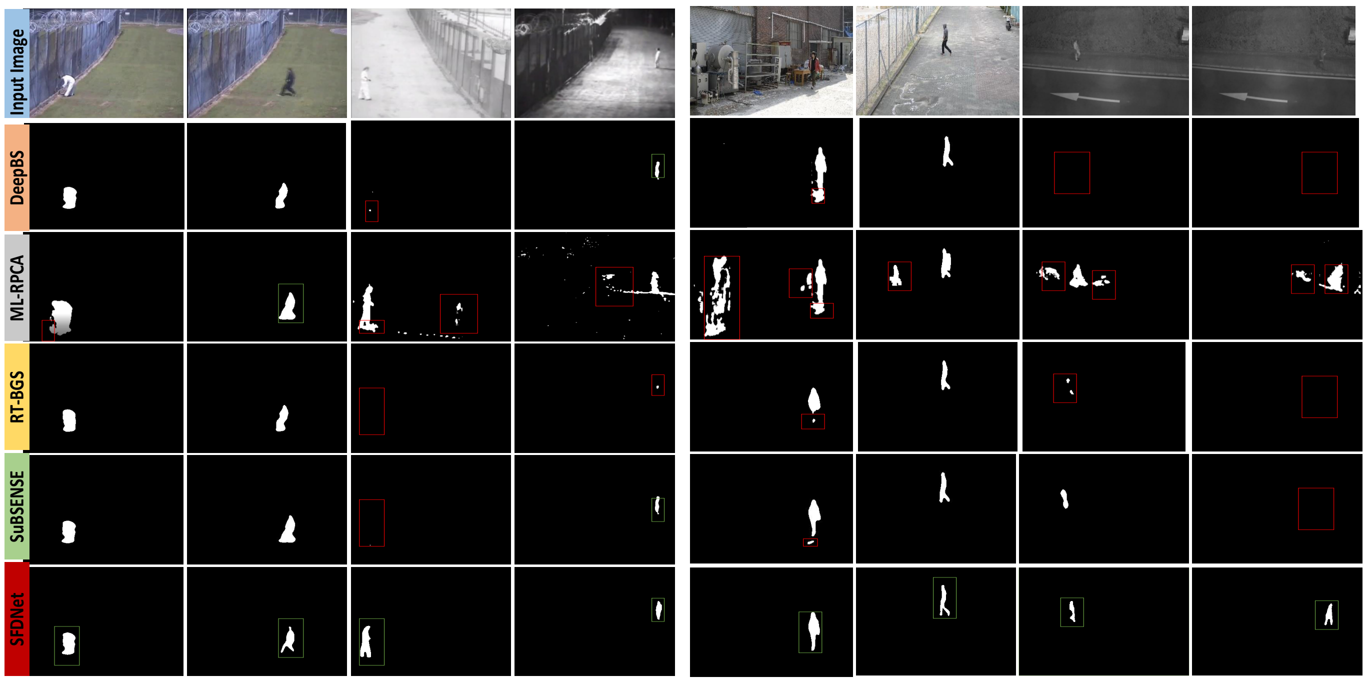
3) Our Dataset: Our dataset offers strong challenges of illumination changes, shadows, and camouflaged effect. Like the i-LIDS dataset, baselines showed similar trends on our dataset as shown in Figure 4b. DeepBS (2nd row) could not cope with shadows and camouflaged effect. It detected the shadow of an intruder in day sequences. It was unable to detect an intruder in night videos due to the camouflaged effect. Like the i-LIDS dataset, ML-RPCA (3rd row) performed well with false positives due to illumination and shadows.

RT-BGS (4th row) only detected distinct parts of the intruder from a background as shown in the red box. Similarly, in the day sequence of our dataset, it missed detecting the intruder's legs shown as a red box. DeepBS and RT-BGS suffering from the camouflage effect at night sequences may be due to the use of a background model for training the CNN model.

SuBSENSE (5th row) followed similar trends to other baselines and detected the shadow of the intruder. It was challenged by the night videos as it only segmented distinct parts of the camouflaged intruder from the background. The SFDNet was able to detect an intruder precisely (6th row) and differentiate between its shadow as well. Similarly, it was able to detect the camouflaged intruder in the night videos. Additionally, it did not give any false positives. The video results can be accessed at https://drive.google.com/open?id=1PjFPQs6a2Y6qzymBSVcTB49vl3Ze512O.

F. Computational Complexity

Table VI shows the computational complexity of SFDNet and baselines in terms of training time, number of parameters, and testing speed in frames per second (fps). The analysis is performed on the NVIDIA GTX1080Ti GPU. It is evident



(a) i-LIDS Dataset (b) Our Dataset

Figure 4. Qualitative results of the SFDNet (6th row) with baselines such as DeepBS (2nd row), ML-RPCA (3rd row), RT-BGS (4th row), and SuBSENSE (5th row). Red boxes refer to false positives or false negatives from baselines. Green boxes refer to true positives from SFDNet.

Table VI
COMPUTATIONAL COMPLEXITY ON 320×240 IMAGE SIZE.

Method	Training time	# Parameters	Testing speed
SFDNet (ours)	12 minutes	7.3 Million	32 fps
MS-ST	90 minutes	>14 Million	11 fps
M2DC4	120 minutes	>11 Million	17 fps
3D CNN-LSTM	80 minutes	2.9 Million	24 fps
MvRF-CNN	>90 minutes	8.6 Million	25 fps
DeepBS	90 minutes	-	22 fps
RT-BGS	-	-	24 fps
SuBSENSE	-	-	2 fps
ML-RPCA	-	-	0.3 fps

that SFDNet is better in training time (12 minutes) and testing speed (32 fps) as compared to baselines. However, 3D CNN-LSTM has fewer parameters owing to the pretrained network and 3D convolution operation. Still, SFDNet is significantly better than 3D CNN-LSTM in training and testing time. SFDNet is suitable for real-time systems. Depending on image size (720×640-320×240), it performs at 8-15 fps on low-end NVIDIA GTX570 GPU which is within the real-time performance requirement of 10 fps [21].

IV. CONCLUSION

This paper proposes efficient SFDNet to tackle the challenges of camera-based surveillance systems. SFDNet benefits from atrous-convolved feature maps that extract rich encoded semantic features from the input, without increasing computational complexity. Qualitative, quantitative, and computational results with top-ranked algorithms on three standard datasets are presented to demonstrate the effectiveness of SFDNet. The trained models are scene specific, i.e., a model can only perform better in the specific trained background setting. The algorithm requires a minimum low-end GPU to perform real-time. The authors wish to integrate SFDNet into other high-level tasks of video surveillance systems such as abandoned object detection and illegally parked vehicle detection in the future.

REFERENCES

[1] A. Shahbaz, V. Hoang, and K. Jo, “Convolutional neural network based foreground segmentation for video surveillance systems,” in *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1, Oct 2019, pp. 86–89.

[2] Wahyono and K. Jo, “Cumulative dual foreground differences for illegally parked vehicles detection,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2464–2473, 2017.

[3] Wahyono, A. Filonenko, and K. Jo, “Unattended object identification for intelligent surveillance systems using sequence of dual background difference,” *IEEE Transactions on Industrial Informatics*, vol. 12, no. 6, pp. 2247–2255, 2016.

[4] T. Zhang, S. Liu, C. Xu, and H. Lu, “Mining semantic context information for intelligent video surveillance of traffic scenes,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 149–160, 2013.

[5] A. Shahbaz, L. Kurnianggoro, Wahyono, and K.-H. Jo, “Recent Advances in the Field of Foreground Detection: An Overview,” in *Advanced Topics in Intelligent Information and Database Systems*. Springer International Publishing, 2017, pp. 261–269.

[6] A. Shahbaz, J. Hariyono, and K. H. Jo, “Evaluation of background subtraction algorithms for video surveillance,” in *21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, Jan 2015, pp. 1–4.

[7] 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.

[8] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” in *Pattern Recognition. ICPR. Proceedings of the 17th International Conference on*, vol. 2, Aug 2004, pp. 28–31 Vol.2.

[9] P. L. St-Charles, G. A. 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, Jan 2015.

[10] P. St-Charles, G. Bilodeau, and R. Bergevin, "Universal background subtraction using word consensus models," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4768–4781, Oct 2016.

[11] 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, Sept 2018.

[12] 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.

[13] J. Giraldo-Zuluaga, A. Gómez, A. Salazar, and A. Diaz-Pulido, "Camera-trap images segmentation using multi-layer robust principal component analysis," *CoRR*, vol. abs/1701.08180, 2017.

[14] 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, no. 17, pp. 23 023–23 041, Sep 2018.

[15] M. Braham and M. V. Droogenbroeck, "Deep background subtraction with scene-specific convolutional neural networks," in *International Conference on Systems, Signals and Image Processing (IWSSIP)*, May 2016, pp. 1–4.

[16] M. Babaee, D. T. Dinh, and G. Rigoll, "A deep convolutional neural network for video sequence background subtraction," *Pattern Recogn.*, vol. 76, no. C, pp. 635–649, Apr. 2018.

[17] Y. Wang, Z. Luo, and P.-M. Jodoin, "Interactive deep learning method for segmenting moving objects," *Pattern Recogn. Lett.*, vol. 96, no. C, pp. 66–75, Sep. 2017.

[18] A. Cioppa, M. V. Droogenbroeck, and M. Braham, "Real-time semantic background subtraction," *CoRR*, vol. abs/1409.1556, 2020.

[19] 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.

[20] Y. Yang, T. Zhang, J. Hu, D. Xu, and G. Xie, "End-to-end background subtraction via a multi-scale spatio-temporal model," *IEEE Access*, vol. 7, pp. 97 949–97 958, 2019.

[21] H. O. S. D. Branch, "Imagery library for intelligent detection systems (i-lids)," in *Crime and Security. The Institution of Engineering and Technology Conference on*, June 2006, pp. 445–448.

[22] Y. Wang, P. M. 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*, June 2014, pp. 393–400.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[24] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," *CoRR*, vol. abs/1802.02611, 2018. [Online]. Available: http://arxiv.org/abs/1802.02611

[25] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[27] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[28] T. Akilan, Q. M. J. Wu, and W. Zhang, "Video foreground extraction using multi-view receptive field and encoder–decoder dcnn for traffic and surveillance applications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9478–9493, 2019.

[29] 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.

[30] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

Ajmal Shahbaz (S'16–M'20) Ajmal Shahbaz received the Bachelors of Electronics Engineering from Islamia University of Bahawalpur, Bahawalpur, Pakistan in 2014. He is working towards a Ph.D. degree in Electrical Engineering, University of Ulsan, Ulsan, South Korea. His research interests include computer vision, image processing, and deep learning.

Kang-Hyun Jo (M'96-SM'16) received the Ph.D. degree in Computer Controlled Machinery from Osaka University, Japan, in 1997.

After a year of experience at ETRI as a post-doctoral research fellow, he joined the School of Electrical Engineering, University of Ulsan, Ulsan, Korea. He has served as a director or an AdCom member of the Institute of Control, Robotics and Systems, The Society of Instrument and Control Engineers, and IEEE IES Technical Committee on Human Factors Chair, AdCom member, and the Secretary until 2019. Currently, he is serving as Faculty Dean of School of Electrical Engineering, University of Ulsan. He has also been involved in organizing many international conferences such as International Workshop on Frontiers of Computer Vision, International Conference on Intelligent Computation, International Conference on Industrial Technology, International Conference on Human System Interactions, and Annual Conference of the IEEE Industrial Electronics Society. At present, he is an Editorial Board Member for international journals, such as the International Journal of Control, Automation, and Systems and the Transactions on Computational Collective Intelligence. His research interests include computer vision, robotics, autonomous vehicle, and ambient intelligence.