SecureCam: Selective Detection and Encryption Enabled Application for Dynamic Camera Surveillance Videos

Ifeoluwapo Aribilola, Mamoona Naveed Asghar, *Senior Member, IEEE*, Nadia Kanwal, *Senior Member, IEEE*, Martin Fleury, *Member, IEEE*, and Brian Lee

***Abstract*—Using dynamic surveillance cameras for security has significantly increased the privacy concerns for captured individuals. Malicious users may misuse these videos by performing Replay and/or Man-in-the-Middle attacks during storage or recording over the network. Considering these risks, this paper proposes an effective security application *SecureCam* based on selective detection (focused moving objects) and protection using encryption. For object detection, this paper implements a novel low computational unsupervised learning algorithm, i.e., *Motion-Fusion (MF)* for more precise motion detection in the mobile camera videos. After that, selective encryption (SE) is applied by the lightweight Chacha20 cipher to the detected video parts. Proposed *SecureCam* is extensively evaluated based on performance analysis, security analysis and computational complexity. For object detection, the comparative evaluation shows that the *MF* algorithm outperforms traditional state-of-the-art dense optical flow (DOF) algorithm with an average (mean) difference increase: in the accuracy of 54%; and in the precision of 42% making it computationally effective for such videos. The visual results with 21% encryption space ratio (ESR) indicate that the videos are sufficiently protected against identification. Overall comparative evaluation with existing approaches also affirm the significance and utility of proposed *SecureCam* for Internet of Multimedia Things (IoMT) environment.**

***Index Terms*—Chacha20, dense optical flow (DOF), encryption space ratio (ESR), Internet of Multimedia Things (IoMT), Man-in-the-middle attack, Region-of-interest (ROI), replay attack, structural similarity index (SSIM).**

I. Introduction

MOBILE camera devices, which monitor activities in at-risk environments, have generated a need to protect the identity of captured people, and their associated objects as a part of sensitive information. As dynamic cameras, pan-tilt-zoom (PTZ), dashboard cams, and unmanned aerial vehicles (UAV) (like drones) rotate and/or move as they capture videos, which also change the background with their movement. In the 20^{th} century, drones were only used for the military operations. However, they have been commercially available since the 21^{st} century. Consumer drones [1] are capable of taking photos, recording videos, and delivering packages at previously unreachable heights and distances. They are offered in a variety of forms and accessories for customizing personal drone experiences in smart cities [2]. Furthermore, with the advent of the Internet of Things (IoT) infrastructure [3], these drones are also utilised for entertainment purposes [4], and for monitoring smart agricultural operations [5] and smart construction sites [6]. Hence, due to the sensitive visual information embedded and being exchanged, these devices must be protected from intruders.

Additionally, due to the limited internal storage of these devices, broadcasting of recorded videos from one device to another on the network without securing the video content, can give an attacker: firstly, the ability to launch a Man-in-the-Middle attack to monitor identifiable persons and/or objects in the video. Secondly, an attacker can add a valid replayed scene, one which has possibly occurred years ago, to masquerade as a current scene, which is referred to as a Replay Attack [7]. Thirdly, an attacker can: hack or hijack these devices in some way; and tamper the videos on those devices [8]. In each of these cases, the attacker could demand a ransom or perpetrate a fraud (aka scam) to the individuals recognized in the video [9], clearly violating the European general data protection regulation (EU-GDPR) [10].

Object identification techniques are widely implemented in computer vision either for static or dynamic camera video devices to separate the region-of-interest (ROIs). The ROIs (in this study) are the foreground (FG) consisting of the non-stationary objects and the background (BG) containing the immovable objects in the video. Static camera devices are always fixed to a position, making it easy to detect the FG information in the form of motion by utilizing background subtraction [11], [12], [13] technique. However, dynamic camera devices resulting in a complex situation, as the BG changes along with the FG information. There are various methods that could identify the objects in these videos [14] but optical

Manuscript received 8 June 2022; revised 19 August 2022 and 12 October 2022; accepted 8 December 2022. Date of publication 13 December 2022; date of current version 25 April 2023. This work was supported in part by the Presidential Doctoral Scheme (PDS) of the Technological University of the Shannon: Midlands Midwest (Athlone Campus), Ireland. *(Ifeoluwapo Aribilola and Mamoona Naveed Asghar contributed equally to this work.) (Corresponding author: Ifeoluwapo Aribilola.)*

Ifeoluwapo Aribilola and Brian Lee are with the Software Research Institute, Technological University of the Shannon: Midlands Midwest (Athlone Campus), Athlone, N37 HD68 Ireland (e-mail: i.aribilola@research.ait.ie; blee@ait.ie).

Mamoona Naveed Asghar is with the School of Computer Science, College of Science and Engineering, National University of Ireland, Galway, H91 TK33 Ireland (e-mail: mamoona.asghar@nuigalway.ie).

Nadia Kanwal is with the Department of Computing and Mathematics, University of Keele, ST5 5BG Keele, U.K. (e-mail: n.kanwal@keele.ac.uk).

Martin Fleury is with the School of Science, Technology, and Engineering, University of Suffolk, IP4 1QJ Ipswich, U.K. (e-mail: fleury.martin55@gmail.com).

Digital Object Identifier 10.1109/TCE.2022.3228679

flow (OF) methods are broadly used in existing research. OF is the visible estimated motion of objects between two successive video frames, produced by the camera and/or object movement. OF is divided into two types: sparse OF, which enables motion detection based on some features of the object, and DOF which performs motion detection based on all the features of an object. Pixel segmentation using global threshold [15], [16] is generally employed to separate the FG pixels from the BG pixels.

To comply with the EU-GDPR, data-protection-by-design solutions for protecting visual information are advisable [17]. Encryption is the only reversible data protection safeguard suggested by the regulation. Data encryption can be applied on complete video frames in the form of nave encryption (NE), that is encryption of the full video payload and header information, or encryption of specific parts/regions-of-interest (ROIs) within frames using selective encryption (SE). NE is impenetrable, but it is computationally slow and suffers from high memory consumption. Hence, NE is not advisable for real-time applications [18], [19], especially when dynamic, battery-based devices, are likely to be faced with high energy consumption costs during memory access. In contrast, SE has minimal memory consumption and computational cost, which makes it suitable for efficient encryption of real-time applications [20], [21]. The research [22] also revealed that SE provides satisfactory security against attacks.

Chacha20 [23] is a stream cipher that utilizes a symmetric (single) key of length 256-bits, and a nonce of either 64-bits or 96-bits. The implementation of Chacha20 is hardware-independent and it is computationally fast for real-time encryption on dynamic cameras with 50-volt batteries [24], which are considered as extra-low voltage devices [25]. Due to its lightweight computation nature, Chacha20 was initially developed for securing Internet of Things (IoT) devices.

By keeping in view the computational and security challenges of surveillance videos captured with dynamic camera devices, the following research questions (RQs) were identified, as a way of targeting the research of this paper:

RQ1: Does video segmentation into FG and BG information help to prevent Replay attacks?

RQ2: What would be the impact of joint detection and protection scheme on the avoidance of Replay, tampering and Man-in-the-Middle attacks on surveillance videos during their transmission and storage?

To answer the aforementioned RQs, this paper proposes a Data-Protection-by-Design solution for videos captured with dynamic cameras. (A Protection-by-Design solution is one in which protection is first introduced at the design stage of a proposal.) The research contributions are as follows:

- Prevention of Replay and Man-in-the-Middle attacks on dynamic camera videos by segmenting the videos into parts.
- By implementing the newly proposed *Motion-Fusion (MF)* algorithm, objects are accurately detected and segmented into FG and BG, for maximum security against their content identification.
- Content protection is achieved by encryption algorithm Chacha20 (initially proposed for IoT devices) on the segmented parts of the videos.
- Evaluation of *SecureCam* application demonstrated the highly accurate selective detection and protection of the tested videos. Thus, this application fits well as a part of IoMT environment.

The remainder of this paper is structured as follows: Section II describes related studies on object identification techniques, pixel segmentation with thresholding and encryption algorithms. Section III describes the methodological workflow of proposed *SecureCam* application related to selective detection and encryption. Section IV demonstrates the evaluation of *SecureCam* over publicly available dynamic camera datasets. This section also includes the performance analysis, security analysis and comparative analysis. Finally, Section V concludes this research study.

II. Related Studies

This section reviews the recent studies in the field of object detection, pixel segmentation and encryption for surveillance videos.

A. Object Identification in Cameras

Cameras are classified into two main groups called: (1) static cameras; and (2) dynamic cameras. The static cameras are fixed to a position within a public area, such as a bank, supermarket, or bus station, recording activities in that location. Closed Circuit Television (CCTV) is a common example of a static surveillance camera. By considering motion as a foreground (FG) object, the detection of the FG, consisting of moving objects, by these cameras is easily achieved by means of frame differencing [26], [27], [28] and background subtraction [11], [12], [13] methods. The background subtraction technique can be implemented either as an unsupervised learning [29], [30], a semi-supervised [31], [32], or a supervised learning [33], [34].

Dynamic/mobile cameras are devices that are in motion when recording. Because their positions are not fixed, as the camera moves the background (BG) tends to change hence giving a complex situation. Examples of these cameras are the car dashboard-cams, PTZ and UAV security cameras. Object identification by these cameras can be achieved by supervised learning with deep neural network [35], [36], YOLO [37], Convolutional neural network(CNN) [38] but this will require high computation for the training these models. While for unsupervised learning algorithms pre-training of detection models are not required.

The study [39] implemented a background subtraction for freely moving cameras but the quality of the result was affected by the accuracy of the detection making background subtraction unsuitable for object detection in dynamic camera videos. Thus, different techniques such as a motion compensation method [40], [41], trajectory classification [42], [43], [44], and optical flow (OF) [45] are often prescribed to separate the FG objects from the moving BG.

These techniques cannot be implemented in isolation from other algorithms. Taking Dense Optical Flow (DOF) as an example, the study [46] introduced an OS-Flow method that combines DOF and SOF. The research [47] implemented a computation method for DOF and texture features. Also, the authors of [48] proposed a DOF-based background subtraction technique using a homography matrix, with single Gaussian and DOF. Likewise the authors of [49] described a movement detection method applying DOF and a fundamental matrix.

Also, the studies [42], [50] discussed different unsupervised methods to detect the moving objects in a moving cameras like panoramic background subtraction, motion compensation, motion segmentation, subspace segmentation, sparse matrix decomposition, trajectory classification. However, this study implemented motion segmentation.

B. Pixel Segmentation With Thresholding

Pixel segmentation, which is also referred to as intensity based segmentation [51], is the grouping of a grey-scaled image into two classes of either light or dark pixels [52]. This is achieved by applying a threshold to these images. The threshold might either be locally or globally applied [22]. In local thresholding, an individual object has its own threshold value, resulting in: multiple thresholding, as described in [53], and in a high computational load. Alternatively, setting a global threshold value and comparing if a pixel is above or below that value results in a diminished computational burden. In fact, the current study applies global thresholding, which differentiate the foreground from the background after object detection [54].

C. Privacy Protection With Encryption Algorithms

As already discussed, encryption can be performed as Nave (or Full) encryption (NE) and Selective Encryption (SE). In NE, the whole video content is encrypted, normally including the video header [18], [55]. This converts a 2D image to 1D before encryption [55]. This type of encryption offers good protection against attack but requires considerable computational time and memory consumption [55]. Alternatively, SE can be performed before or after video compression and simultaneously [55]. Frequently, the crucial areas in a video are of small pixel extent compared to the entire video content. Thus, SE uses a low computational time and memory space and, for many practical purposes, provides sufficient protection against attacks [19].

Asymmetric encryption uses two different keys (one public and one private key) for its encryption and decryption while symmetric encryption uses a single key for both encryption and decryption process [18], [21]. Symmetric encryption can further be divided into block ciphers and stream ciphers. Chacha20, a stream cipher in the family variant of Salsa [56] was created by Daniel Bernstein [23]. In fact, some have claimed it to be the lightest and fastest encryption algorithms [57]. As the name implies, Chacha20 performs 20-rounds of encryption, which are equivalent to 80-quarter rounds of XORing, additions and rotations in a cipher round [23]. The study [56] performed a cryptanalysis of Chacha20, arriving at a conclusion that correlation attacks do not pose a threat to this algorithm because it does not use a look-up table. Thus, it is not vulnerable to cache-timing attacks. Currently, Chacha20 has been implemented for text (messages) [23] and IoT devices, but not for the protection of multimedia, i.e., audios and videos.

All the existing research studies reviewed herein show that DOF is not sufficient to accurately detect FG and BG information from dynamic camera videos. Therefore, DOF is implemented in combination with other algorithms. However, the combined implementation of algorithms for object detection increase the computational cost, making them unsuitable for small battery-operated camera electronics as discussed in [58].

It is also worth noting that the existing literature is not focused on privacy-protection of the detected objects in dynamic camera videos, which constitutes a research-gap in the literature. In contrast, this paper proposes a *SecureCam* application, which jointly detects and encrypts the detected FG and BG video parts using *MF* algorithm and Chacha20 algorithm for dynamic camera videos. The reason of implementing Chacha20 is its low computational and hardware-independent nature, making it suitable for low-voltage IoT devices. By this time, only Google organization uses Chacha20 for its Transport Layer System (TLS) [59], and to the best of the authors' knowledge, Chacha20 has not been utilized for the privacy protection of the mobile camera surveillance videos to facilitate the IoMT infrastructure.

III. The Proposed Method

The process flow of the proposed *SecureCam* application is described in "Fig. 1" with five (05) stages. In the First stage, the video frames are loaded from a recording output of a dynamic camera device. Secondly, *MF* algorithm (explained in Section III-A) was applied on consecutive frames to detect the objects in motion (FG). Thirdly, pixel segmentation was applied on the retrieved output, using global thresholding to separate the FG pixels from the BG pixels. The FG and BG pixels are encrypted separately with the Chacha20 algorithm by randomly generating a key and a nonce value at stage four. The key and nonce are securely stored in a hardware wallet, so that their security cannot be compromised. Finally, the encrypted FG and BG pixels (ROIs) are also stored separately. The storage and transmission of these ROIs will help in preventing Replay attacks, because the video parts are not together as a file for easy identification by the attacker. By accessing the FG stored file, the attacker cannot identify how the BG looks and vice versa. "Table I" describes the frequently used acronyms in this paper.

A. Motion Detection With MF

"Fig. 1" (column 2) represents the Motion detection stage of the *SecureCam* application. *MF* algorithm was implemented for precise ROIs detection in the video in order to preserve the device's battery when applying the proposed *SecureCam* to real-time videos from dynamic camera devices. *MF* considered

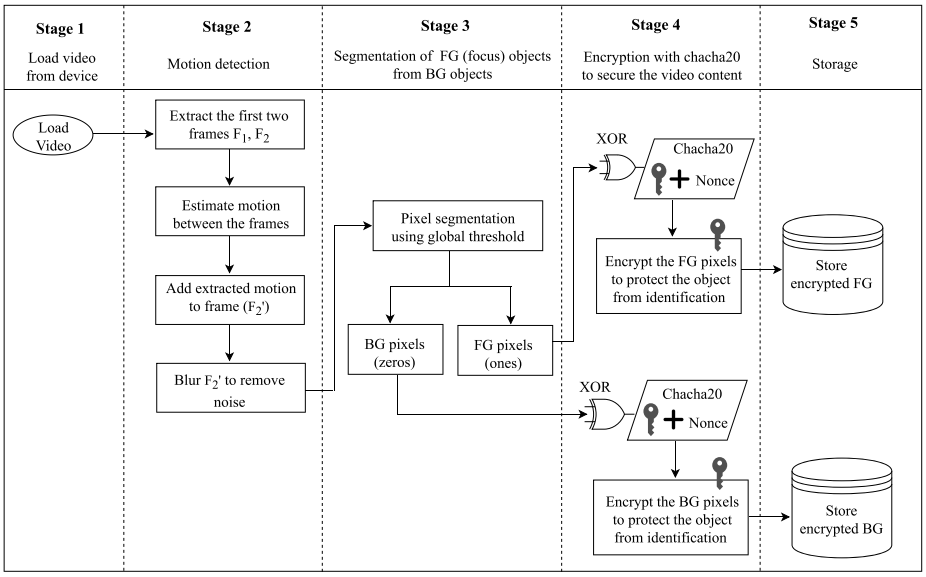


Fig. 1. The process flow of *SecureCam* Application.

TABLE I
ACRONYMS USED FOR *SecureCam* APPLICATION

Acronyms	Description
BG	Background (non moving objects))
DOF	Dense optical flow
ESR	Encryption Space Ratio
FG	Foreground (focused/moving objects)
IoMT	Internet of Multimedia Thing
MF	Motion Fusion
ROI	Region-of-Interest
SE	Selective encryption
SSIM	Structural similarity index

moving objects as the FG and static objects as the BG within videos.

The *MF* algorithm firstly select first two consecutive frame F_1 and F_2 from the video. The pixel intensity in this frame is constant as given in "(1)", where I = intensity, a = horizontal axis, b = vertical axis and t = time

$$I(a, b, t) \tag{1}$$

Secondly, perform a motion estimation on F_1 and F_2 by extracting the vectors coordinate of the motion between the frames using Farneback algorithm [45]. The change in the intensity of the pixels with time from the previous frame to the current frame is been identified as the movement of the object as analysed in "(2)".

$$I(a, b, t) = I(a + \delta a, b + \delta b, t + \delta t) \tag{2}$$

Taylor's approximation series is applied on the right hand side (RHS) of "(2)" and hence divided by the time change giving the result in "(3)", where u = $\frac{\delta a}{\delta t}$ and v = $\frac{\delta b}{\delta t}$.

$$\begin{aligned} &\approx \frac{\delta I}{\delta a} \cdot \delta a + \frac{\delta I}{\delta b} \cdot \delta b + \frac{\delta I}{\delta t} \cdot \delta t = 0, \\ &\approx \frac{\delta I}{\delta a} \cdot \frac{\delta a}{\delta t} + \frac{\delta I}{\delta b} \cdot \frac{\delta b}{\delta t} + \frac{\delta I}{\delta t} \cdot \frac{\delta t}{\delta t} = 0, \\ &\approx \frac{\delta I}{\delta a} \cdot \frac{\delta a}{\delta t} + \frac{\delta I}{\delta b} \cdot \frac{\delta b}{\delta t} + \frac{\delta I}{\delta t} = 0, \\ &\approx \frac{\delta I}{\delta a} \cdot u + \frac{\delta I}{\delta b} \cdot v + \frac{\delta I}{\delta t} = 0 \end{aligned} \tag{3}$$

Lastly, this motion is extracted and infused back to the frame to identify the actual moving object (FG) in the frame. The infusion was achieved using "(4)" where the α value varies, β value is constant at 0.5 and the γ value is 0.

$$F_2' = \alpha \cdot F_2 + \beta \cdot \left(\frac{\delta I}{\delta a} \cdot u + \frac{\delta I}{\delta b} \cdot v + \frac{\delta I}{\delta t} \right) + \gamma \tag{4}$$

This process was repeated on the remaining frames in the video resulting in "(5)"

$$F_n' = F_n + F_{n-1} \tag{5}$$

B. Pixel Segmentation With Global Thresholding

Pixel segmentation is the stage 3 of the *SecureCam* process flow ("Fig. 1" - column 3). Pixel segmentation was performed by setting a global threshold on the grey-scaled result of the *MF* algorithm. This global threshold separates the detected moving object pixels from the static object pixels in the frame.

C. Encryption With Chacha20 Algorithm

At stage 4 ("Fig. 1" - column 4), encryption is applied using Chacha20 algorithm. The segmented FG and BG pixels are been XORed separately with a key and nonce generated randomly from Chacha20 algorithm and also simultaneously securing the key and nonce in a hardware wallet. To restore

the original frame, decryption was performed by XORing the cipher-pixels with the same securely stored key and nonce.

A complete round for Chacha20 is a 10 column-rounds and 10 diagonal-rounds which include:

- Four 4-byte constants (giving a total of 16-bytes (128-bits)).
- A random 32-byte (256-bits in all) long key.
- A 4-byte (32-bits in all) block counter.
- A 12-byte (96-bits in all) or 8-bytes (64-bits) nonce [60].

A single round for Chacha20 contains four quarter-rounds for a column-round and four quarter-rounds for a diagonal-round. A single quarter-round involves three arithmetical operations that are:

- Integer Addition Modulo [w+=x]
- Bitwise Exclusive OR (XOR) [w^ =x]
- N-bit left rotation [<<< n] [60].

A complete round of Chacha20 with single rounds arithmetic operation, makes chacha20 resistance to attack.

D. Storing of the Encrypted Video Segments (ROIs)

The FG encrypted pixels are stored separately from the BG encrypted pixels as shown in ("Fig. 1" - column 5). If an attacker access the FG encrypted pixels, it will be difficult to identify the BG pixels of the frame due to the encryption of the BG of the captured videos. This makes it hard to perform a Replay attack on the video.

E. Computational Complexity of SecureCam

The pseudo-code of the proposed *SecureCam* application is presented in the "Algorithm 1". The *SecureCam* algorithm has multiple stages, stage 1 loads the video for processing therefore, consumes $Big(O) = n$ for n number of video frames. Stage 2 performs object detection which takes $Big(O) = n$ for n number of video frames. Stage 3 loops through every pixel of the frame to classify the pixels into FG and BG pixels and generate two frames of the same size, hence the time complexity of this stage is $Big(O) = Frame\_Size$ or $Big(O)frame\_width \times frame\_height$. Similarly, in stage 4 the algorithm loops through every pixel of the two frames generated in stage 3 to apply chacha20 encryption, which results in $Big(O)2 \times Frame\_Size$. Stage 5 stores the FG and BG objects which takes $Big(O) = n$ for n number of video frames. This sums up to a $Big(O) = 3n(3 \times Frame\_Size)$ which is approximately a linear time complexity.

$$
\begin{aligned}
&Stage1\text{: } Big(O) = n \text{ // } n \text{ times of frames}\\
&Stage2\text{: } Big(O) = n \text{ // } n \text{ times of frames}\\
&Stage3\text{: } Big(O) = Frame\_Size \text{ // frame size}\\
&Stage4\text{: } Big(O) = 2 * Frame\_Size \text{ // frame size}\\
&Stage5\text{: } Big(O) = n \text{ // } n \text{ times of frames}\\
&T_{sum} = \big[stage1 + stage2 + stage3 + stage4 + stage5\big]\\
&\qquad = [n + n + Frame\_Size + 2 \times Frame\_Size + n]\\
&Big(O) = 3n(3 \times Frame\_Size)
\end{aligned}
$$

For the ease of readers,"Table II" describes the summary of mathematical notations with their respective meanings.

Algorithm 1 Pseudo-code of *SecureCam* Application

```
Input: Dynamic Camera Video
Output: Video with Chacha20 encryption
Data: Load Video from path

/* SecureCam Stage 1 & 2: Loading Data and Motion
   Detection                                        */
while video == True do
    frame ← video.read()
    grey ← frame change to grey
    flow ← calcOpticalFlowFarneback()
    mag, ang ← flow's magnitude and angle
    motion ← normalize mag, ang
    blend ← add motion to frame
    b_grey ← convert blend to grey
    blur ← remove noise
    /* SecureCam Stage 3: FG and BG Segmentation
       */
    ret, thresh ← apply global thresholding
    dilated ← smooth the result
    pixelpoints ← (separate pixels to zeros and ones)
    FG_pixels ← (pixels with zeros)
    BG_pixels ← (pixels with one)
    /* SecureCam Stage 4: Encryption            */
    Encrypt_FG ← chacha20_encrypt(FG_pixels)
    Encrypt_BG ← chacha20_encrypt(BG_pixels)
    /* SecureCam Stage 5: Storage               */
    Secure_Folder_FG ← Encrypt_FG
    Secure_Folder_BG ← Encrypt_BG

    if cv2.waitKey(27) & 0xFF == ord ('q') then
        break
cv2.destoryAllWindows()
video.release()
Output: Separately stored FG and BG Chacha20 encrypted video

/* Chacha-2020 function                         */
def chacha20_encrypt(seg_pixels):
    pixels ← seg_pixels.tobytes()
    k ← get_random_bytes(32)
    nonce ← get_random_bytes(12)
    cipher ← ChaCha20.new(key = k, nonce = nonce)
    video ← cipher.encrypt(pixels)
    nonce ← b64encode(cipher.nonce)
    cvid ← b64encode(video)
    result ← {'key':k, 'nonce':nonce, 'video':cvid}
    return result
```

IV. Evaluation

For the development of proposed *SecureCam* application, both *MF* and Chacha20 algorithms were implemented in python programming language with the `OpenCV` vision library for *MF*, and the `Base64` and `Crypto` algorithms for Chacha20. The testbed setup specs for the experiments are provided in "Table III". The experiments were executed on the dataset of five (05) publicly available dynamic camera videos, [61], [62], [63] each with differing characteristics, i.e., in terms of colours, motion activity, and spatial information. The properties of these test videos are described in "Table IV", where frame per second (FPS) represents the frame rate of the videos.

A. Performance Analysis of SecureCam

As described earlier, newly proposed *MF* algorithm is implemented for selective detection of mobile camera videos

TABLE II
SUMMARY OF THE MATHEMATICAL NOTATION

Symbol	Description
I , δI	Intensity , change in intensity
a , δa	Horizontal axis , change in horizontal axis
b , δb	Vertical axis , change in vertical axis
t , δt	Time , change in time
$\frac{\delta a}{\delta t}$ = u	Change in horizontal axis divided by change in time
$\frac{\delta b}{\delta t}$ = v	Change in vertical axis divided by change in time
α	Weight for the first frame
β	Weight for the second frame
γ	Constant with value zero
$Big(O)$	Algorithm complexity

TABLE III
EXPERIMENTAL SET UP FOR *SecureCam*

System / Device	Specification
Processor	Intel(R) Core (TM) i7-10510U CPU @ 1.80GHz 2.30GHz
Installed RAM	16GB RAM
System type	64-bit operating system, x64-based
Operating Systems	Windows 10
Graphics	Intel(R) UHD graphics

TABLE IV
PROPERTIES OF THE DATASETS USED FOR THE EXPERIMENTS

File	Size	Resolution	Time	Rate	Count
Horse Moving	4.29	860 x 484	5	23	126
MOT16-12	2.35	960 x 540	6	30	180
Car Moving	2.27	1280 x 720	6	30	180
Dash Cam	6.61	1280 x 720	6	24	144
Mountain Hiking	2.58	1366 x 720	6	24	144

in this paper. We have analysed the performance evaluation of *MF* and the state-of-the-art (SOTA) DOF algorithms in this section using different parameters along with visual results.

1) Visual Analysis: The comparative visual results given in "Table V" shows the the increased accuracy in object detection with *MF* algorithm in comparison with DOF. The object detection illustrations on column 2 of "Table V" enables a comparison between the *MF* and DOF algorithms. These visual results demonstrate a clearer and more accurate FG object detection by *MF* in comparison with DOF, according to findings from the test videos.

After detecting FG objects with *MF* and DOF in the test videos, the next stage in our procedure is to apply privacy protection to the extracted FG and BG ROIs. The visual results of SE implemented with Chacha20 are shown in columns 3 and 4 of "Table V". The results, taken after encryption of FG and the BG segregated frames ("Table V" - columns 3 and 4), shows that the DOF method [45] is not suitable for SE on videos taken by dynamic cameras. However, the *MF* results demonstrate an effective and accurate implementation of SE. This signifies that the *MF* can efficiently contribute to privacy protection of FG and BG ROIs by virtue of their accurate detection.

2) Accuracy and Precision: Results for accuracy and precision were analysed in "Table VI". The accuracy was calculated as the ratio, given as a percentage, of the correctly predicted objects versus the total objects in the video, while the precision was calculated as the ratio (given as a percentage) of the correctly detected objects to the total of detected objects in the video.

The accuracy and precision results presented in "Table VI" indicate that *MF* has a greater accuracy and precision ratio for each of the tested videos compared to DOF, thus verifying the better performance of *MF* in object detection.

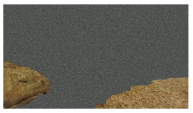
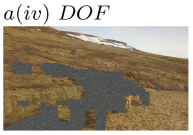
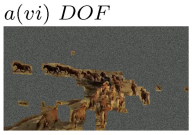
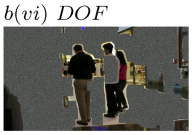
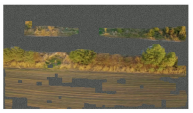
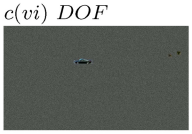
3) Encryption Space Ratio (ESR): The ESR indicates the amount of data encrypted in terms of percentages. ESR is directly proportional to the computational cost of encryption. Thus the smaller ESR indicate the lower encryption cost and higher the efficiency of the encryption scheme over the streaming data. ESR is directly proportional to the pixel space rate (PSR). The ESR for the FG of the implemented *MF* method was compared with the DOF method. "Fig. 2(a)" confirms that the *MF* produced a lower PSR/ESR value in comparison to DOF. The lower value indicates that fewer objects as part of the FG are encrypted. The higher value of ESR for the FG after application of DOF shows that DOF wrongly detect some BG objects as being part of the FG. Thus, *MF* leads to more efficient encryption because it selectively encrypt fewer objects which represent the FG.

The BG ESR for the *MF* and the DOF methods are compared in "Fig. 2(b)" with the *MF* method having a higher value than DOF. This indicates that more BG objects are encrypted in the videos, while the lower value for the DOF method demonstrates that DOF detects relatively limited BG information in the videos.

From the FG and BG ESR/PSR analysis, it can be deduced that the DOF method has a larger False Positive (FP) detection rate for FG objects and a greater True Negative (TN) detection rate for BG objects in the tested videos when compared with the results from using *MF*.

4) Video Quality Metrics: Structural Similarity Index (SSIM) compares the similarity between the detected and encrypted test videos (output) with the original test videos (taken as input). SSIM index ranges from 0-1, so if it is closer to 1, then the structure of selectively detected/encrypted parts in the frames and the original frames are closely resemble with each other, with the inverse applying if a SSIM index is closer

TABLE V
VISUAL REPRESENTATION COMPARING THE EXISTING DOF AND THE PROPOSED MF ON DYNAMIC BACKGROUND VIDEOS

Original Frame	Detected Frame	FG Encrypted Frame	BG Encrypted Frame	Decrypted Frame
	 $a(ii)$ *DOF*	 $a(iv)$ *DOF*	 $a(vi)$ *DOF*	
$a(i)$ Horse moving Frame_65	 $a(iii)$ *MF*	 $a(v)$ *MF*	 $a(vii)$ *MF*	$a(viii)$ *Decrypted*
	 $b(ii)$ *DOF*	 $b(iv)$ *DOF*	 $b(vi)$ *DOF*	
$b(i)$ MOT16-12 Frame_45	 $b(iii)$ *MF*	 $b(v)$ *MF*	 $b(vii)$ *MF*	$b(viii)$ *Decrypted*
	 $c(ii)$ *DOF*	 $c(iv)$ *DOF*	 $c(vi)$ *DOF*	
$c(i)$ Car moving Frame_120	$c(iii)$ *MF*	 $c(v)$ *MF*	 $c(vii)$ *MF*	$c(viii)$ *Decrypted*
	 $d(ii)$ *DOF*	 $d(iv)$ *DOF*	 $d(vi)$ *DOF*	
$d(i)$ Mountain Hiking Frame_115	 $d(iii)$ *MF*	 $d(v)$ *MF*	 $d(vii)$ *MF*	$d(viii)$ *Decrypted*

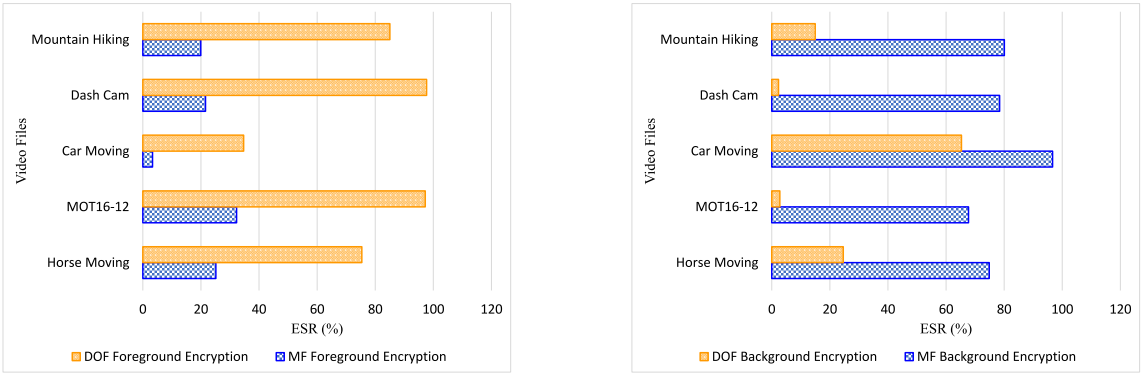
to 0. SSIM is calculated in "(6)" as:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c1)(2\sigma_{xy} + c2)}{(\mu_x^2\mu_y^2 + c1)(\sigma_x^2 + \sigma_y^2 + c2)}, \quad (6)$$

where x = original tested videos, y = encrypted tested videos, μ_x = average of x, μ_y = average of y, σ_x^2 = variance of x, σ_y^2 = variance of y, σ_{xy} = covariance of x and y, c1 = $(K_1L)^2$ c2 = $(K_2L)^2$, L = dynamic range, $(K_1) = 0.01$, $(K_2) = 0.03$

SSIM was calculated after applying the *MF* and DOF algorithms and after selective encryption of either the detected FG or BG ROIs using Chacha20 (see "Table V" (columns 3 and 4)). The results drawn from these operations are shown in "Table VII".

MF for the FG has a SSIM value that ranges between 0.4684 to 0.7255 giving an average of 0.5553. In contrast, for DOF the FG values range from 0.0766 to 0.3391. These latter values indicate that virtually the whole of the frames are identified as FG by the DOF algorithm, indicating that virtually whole frames are encrypted. Consequently, the metric's values indicate limited resemblance between the original video frames and the video frames when applying selectively-detected/encrypted FGs. In other words, the DOF algorithm has erroneously detected and consequently encrypted more objects as FG elements of frames, leading to an enlarged FG ROI.



(a) ESR calculated on detected FG

(b) ESR calculated on detected BG

Fig. 2. Encryption space ratio (ESR) comparison of the *SecureCam* scheme using MF and DOF detection algorithms on tested videos.

TABLE VI
COMPARATIVE ACCURACY AND PRECISION FOR MF AND DOF ALGORITHMS

File	MF		DOF	
	Accuracy	Precision	Accuracy	Precision
Horse Moving (FG)	85.71%	85.19%	55.56%	69.70%
Horse Moving (BG)	85.71%	87.50%	55.56%	16.67%
MOT16-12 (FG)	68.75%	87.50%	21.88%	21.88%
MOT16-12 (BG)	68.75%	62.50%	46.88%	46.88%
Car Moving (FG)	81.82%	75.00%	40.00%	37.50%
Car Moving (BG)	81.82%	85.71%	40.00%	42.86%
Mountain Hiking (FG)	71.43%	66.67%	37.50%	25.00%
Mountain Hiking (BG)	71.43%	75.00%	30.77%	28.57%

TABLE VII
COMPARATIVE SSIM INDEXES FOR MF AND DOF ALGORITHMS

File	Foreground		Background	
	MF	DOF	MF	DOF
Horse Moving	0.4684	0.0966	0.1361	0.5077
MOT16-12	0.5374	0.0766	0.3062	0.8663
Car Moving	0.5472	0.3391	0.1494	0.3609
Dash Cam	0.4982	0.0814	0.2210	0.9540
Mountain Hiking	0.7255	0.1362	0.2248	0.8256

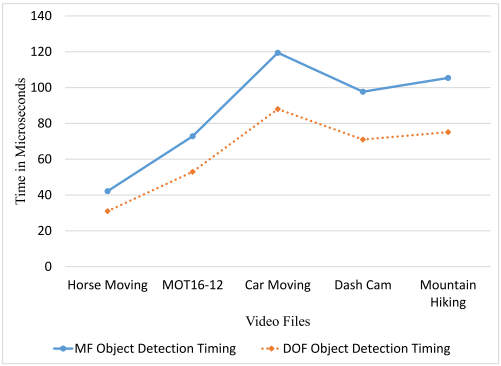
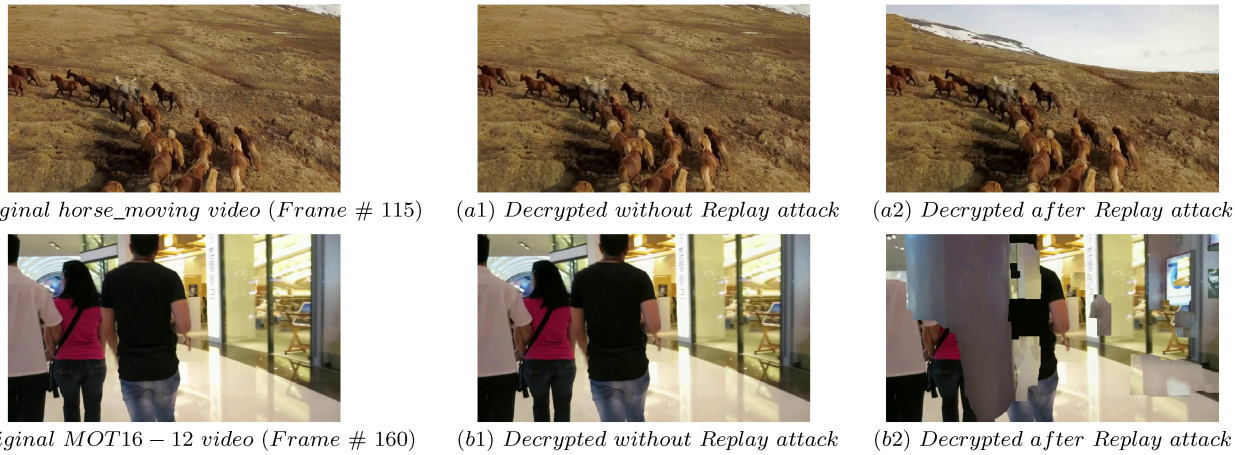


Fig. 3. Object Detection Timing Comparison.

On the other hand, selectively encrypting the BGs resulting from the *MF* algorithm results in SSIM values between 0.1361 to 0.3062 with an average of 0.2075 for the test videos. This should not be surprising as the frames with the BGs encrypted can be viewed as the obverses of the frames with the FGs encrypted, as discussed in the previous paragraph. Thus, for *MF*, according to the SSIM values, the BGs are a much greater proportion of their frames and consequently the SSIM results are relatively low (indicating a minor resemblance between the frames with selectively encrypted FGs and the original frames).

Conversely, the BG SSIM values after application of DOF and SE are closer to 1 (ranges from 0.3609 to 0.9540 across the videos). Thus, the DOF algorithm identifies a smaller sized ROI as part of the static background, which again indicates that the DOF algorithm is more likely to identify static objects as part of a larger moving foreground, which visual inspection shows is the case. The average SSIM value for the encrypted FG and BG with *MF* is 0.381.

5) Computational Cost Analysis for Selective Detection: The time taken for object detection of FG objects using either *MF* or DOF algorithms was also calculated and compared in this paper. It is visible from "Fig. 3" that *MF* took a longer time in comparison with the DOF. For example, the variation between the *MF* and DOF for object detection in the MOT16-12 video file was 19920103 (μs) while object detection in the Mountain Hiking video file was 30262966 (μs). On average (arithmetic mean) *MF* has a time difference of 23894911.4 (μs) across the five tested videos during object detection. However, such a time difference is relatively minimal, especially when set against the accurate detection exhibited by *MF*.



(a) original horse_moving video (Frame # 115) (a1) Decrypted without Replay attack (a2) Decrypted after Replay attack

(b) original MOT16 − 12 video (Frame # 160) (b1) Decrypted without Replay attack (b2) Decrypted after Replay attack

Fig. 4. Visual representation of the Replay attack performed on test videos (a,b) original video frames, (a1,b1) Decrypted video frame without Replay attack, (a2, b2) Decrypted video frame with Replay attack applied.

B. Security Analysis of SecureCam

The Security paradigm of *SecureCam* was considered on two implementation points, i.e., a Replay attack and an extensive key guessing attack. The analysis of these attacks against *SecureCam* are discussed below.

1) Replay Attack: Replay attacks occur when valid data transmission is fraudulently or maliciously repeated or delayed. However, the *SecureCam* scheme renders Replay attack infeasible.

To verify the strength of *SecureCam* against Replay attacks, the attack was simulated against tested videos and it could be deduced from "Fig. 4" that the output video frames after decryption were looking tampered. The attacker could not guess the accurate position to inject the attack since the segmented FG and the BG were separately stored in our scheme. The visual results of the Replay attack for two different frames from two test videos are shown in "Fig. 4". "Fig. 4 (a) and (b)" are the original video frame of horse_moving video (frame number 115) and MOT16-12 video (frame number 160) respectively. "Fig. 4 (a1) and (b1)" are the decrypted frames without the application of Replay attack while "Fig. 4 (a2) and (b2)" are the decrypted frames after launching a Replay attack against the frame. The differences in the original and the attacked video frames are obvious in "Fig. 4", proving the effectiveness of *SecureCam* segmentation scheme against the prevention of Replay attacks.

To further elaborate the verification of tampered frames by executing Replay attacks, we also performed pixel correlation testing. Pixels in a frame are constituted of two properties, position and colour. Verifying the originality of videos is dependent on the number of pixels within a frame/image relative to these two properties. For testing, we have calculated the number of pixels within the original frames, and decrypted frames with and without Replay attacks ("Fig. 4"). "Table VIII" verifies that the total pixel counts of the attacked frames after decryption (column 4) are increased. This also reduces the pixel correlation within the attacked frames in contrast to the original. Testing pixels count/correlation within frames is an easy way to detect tampering within frames, as they always change even with subtle alterations in a video frame.

TABLE VIII
PIXELS COUNT FOR DECRYPTED FRAMES WITH AND WITHOUT ATTACK

Videos	Frame no.	Orig/Decrypted frame without Replay attack	Decrypted frame after Replay attack
Horse Moving	115	107895722	143581526
MOT16-12	160	181505504	184302004
Car Moving	155	282418036	275280320
Dash Cam	140	353730839	355250205
Mountain Hiking	140	408553321	359447070

2) Extensive Key Guessing Attack: Extensive key guessing is an approach of finding the correct key by continuously trying every possible key by guessing until the correct key is discovered.

Implementation of the *SecureCam* application with Chacha20 uses 256-bit keys and it is not feasible to find a 256-bit key by extensive key guessing techniques. However, quantifying the security effectiveness of the *SecureCam*, the number of generated attacks on data and keys could be related with the Poisson probability distribution, given by $P(\mu; n) = \frac{e^{-\mu}\mu^n}{n!}$, where e is constant at approximately 2.71828, μ refers to the number of attacks and n represents how many attacks occur within a fixed period of time.

Based on a given number of events (attacks), P is the probability of attacks occurring within a set time interval. According to Cisco security statistics [64] in 2020, there is an attack on a host machine every 5 minutes, which is approximately 300 attacks per day.

Despite the inherent vulnerabilities of the host machine, the encryption process of *SecureCam* is robust enough to meet the security requirements since it does not only use the keys (256-bit), but it also uses a nonce of 64-bit for random generation of a strong cipher output. In other words,

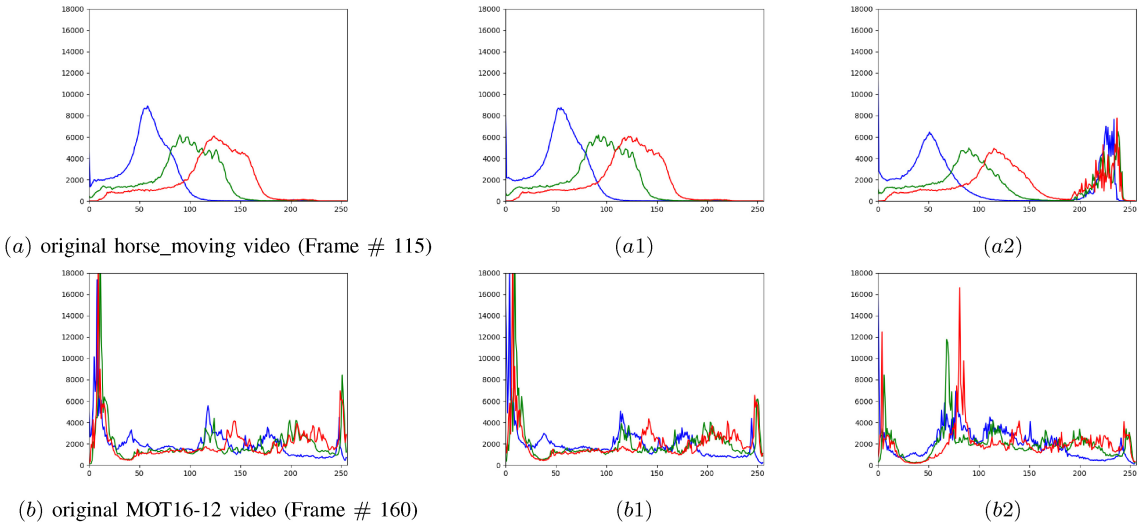


Fig. 5. Histogram analysis of the Replay attack performed on test videos (a, b) original video frames, (a1, b1) Decrypted video frames without Replay attack, (a2, b2) Decrypted video frame with Replay attack.

the previously rendered successful attack wouldn't work with the *SecureCam* scheme.

C. Statistical Analysis of SecureCam

The statistical analysis of the *SecureCam* was evaluated with the histogram of the pixel colors and the SSIM values of the decrypted test video with and without simulated attack.

1) Histogram Analysis: The histogram analysis, is a plot of the frequency distribution of the pixel values based on the color components RGB (red, green, and blue) of the original vs. decrypted video frames with and without simulated Replay attack. Additionally, the histogram determines the correlation between these frames as shown in "Fig. 5". The lower correlation of the plot shows the greater variance, and vice versa.

"Fig. 5 (a) and (b)" shows the histogram plot of the color values (R, G, B) for the original video frame of horse_moving video (frame number 115) and MOT16-12 video (frame number 160) respectively. "Fig. 5 (a1) and (b1)" are the decrypted histogram color values (R, G, B) without the application of Replay attack while "Fig. 5 (a2) and (b2)" are the decrypted histogram color values (R, G, B) after launching a Replay attack against the frame.

Comparing the histogram results of the original with the decrypted frames without Replay attack signifies a high correlation with each other, which means there is little or no variance with both frames. But when comparing the original frame with decrypted frame "Fig. 5 (a2) and (b2)" after Replay attack reveals a low correlation with both frames resulting in high variance. This proves that an attempted Replay attack against a frame protected with the *SecureCam* scheme can be easily detected.

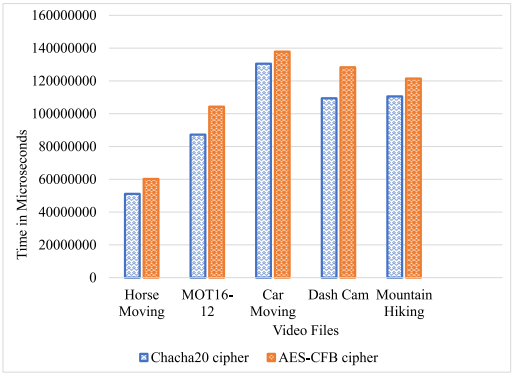
2) Structural Similarity Analysis: As discussed earlier (Section IV-A4) SSIM evaluates the structural distortions by comparing the luminance values of two different images as a proxy for similarity ranging from 0 to 1. A Replay attack was performed on all test videos and the results of the SSIM of the decrypted frames with and without attack are compared with the original frames. From "Table IX" the decrypted frames without Replay attack (column 3) have higher SSIM values closer to 1 which means the decrypted frames without Replay attack are closer to the original frame. However, the decrypted frames after attack (column 4) have a lower value, hence tampering can be easily detected.

TABLE IX
SSIM FOR DECRYPTED FRAMES WITH AND WITHOUT ATTACK

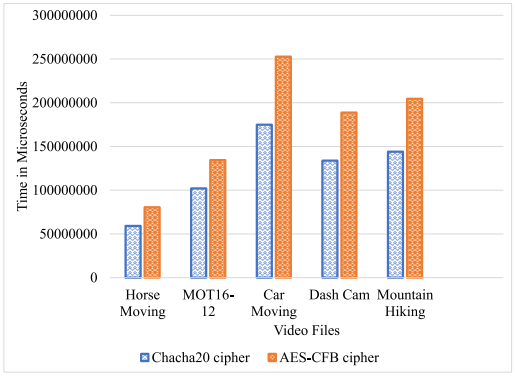
Videos	Frame no.	Decrypted frames without Replay attack	Decrypted frames after Replay attack
Horse Moving	115	0.943818	0.725737
MOT16-12	160	0.956050	0.666668
Car Moving	155	0.961324	0.611006
Dash Cam	140	0.964251	0.762203
Mountain Hiking	140	0.966834	0.764785

D. Comparative Computational Cost Analysis of Used Cipher

The computational complexity of the proposed *SecureCam* is 3n (3 × Frame_Size) (Section III-E) which is dependent on the number of iterations (video length or frames). For evaluating the execution complexity of proposed *SecureCam* scheme (stage 4) with other SOTA cipher, we also performed their comparative results in this section. The computation timing for using Chacha20 as the encryption algorithm in



(a) Execution timing on detected FG objects



(b) Execution timing on detected BG objects

Fig. 6. Comparative computational cost analysis of the *SecureCam* scheme (stage 4) using Chacha20 and AES-CFB ciphers on tested videos.

TABLE X
COMPARATIVE ANALYSIS OF *SecureCam* APPLICATION WITH THE EXISTING APPROACHES

Existing schemes	Proposed Model	Method	Detection	Machine Learning	Encryption	Other Metrics: Accuracy	Precision	SSIM Encrypted	SSIM Decrypted	ESR for FG detection
Maier et al., (2013) [49]	Anticipated geometry pixel deviation	DOF and fundamental matrix	YES	Unsupervised	X	65.03	X	X	X	X
Qu et al., (2016) [47]	Spatial Gabor filter texture	DOF and texture feature	YES	Unsupervised	X	X	X	X	X	X
Wu et al., (2017) [39]	Detection of objects in 3-D scene	Decomposition, trajectories, background subtraction	YES	Unsupervised	X	X	decrease from 92 to 72	X	X	X
Xiang et al., (2019) [46]	Optical-to-SAR Flow (OS-Flow)	DOF and Sparse Optical Flow	YES	Unsupervised	X	X	X	X	X	X
Kushwaha et al., (2020) [48]	Dense Optical Flow based background subtraction technique	Homography matrix, single Gaussian and DOF	YES	Unsupervised	X	52.49	X	X	X	X
Jang et al., (2020) [66]	Partial image encryption using format-preserving encryption in image processing systems for Internet of things environment	FF1 and FF3-1	X	X	$FF1\ and\ FF3-1\ (SE)$	X	X	X	X	X
Liu et al., (2020) [67]	A Data Encryption and Fast Transmission Algorithm Based on Surveillance Video	frame differencing, and video compression	YES	Unsupervised	$image\ or\ text\ information\ (SE)$	X	X	X	X	60%
Kang et al., (2021) [68]	ROI Image Encryption using YOLO and Chaotic Systems	YOLO, Chen system	YES	Supervised	$chaotic-based\ (SE)$	X	X	X	X	X
Tian et al., (2021) [69]	Robust Privacy-Preserving Motion Detection and Object Tracking in Encrypted Streaming Video	Density of Non-zero Residual Coefficient (DNRC) (Kalman filter, H.264/AVC)	YES	Unsupervised	$H.264/AVC\ (SE)$	X	0.6434	0.163	X	X
SecureCam (2022)	Selective Privacy protection with *MF* algorithm	*SecureCam* (*MF*, thresholding, Chacha20)	YES	Unsupervised	$Chacha20\ (SE)$	80	78	0.381	0.96	21%

SecureCam was tested against advanced encryption standard (AES) cipher feedback (CFB) mode [65]. The AES algorithm is an industry cipher and has proven resistance against attacks, while the CFB is the only self-synchronizing mode used by AES operating as a stream cipher. The measured execution time when Chacha20 was applied for SE in *SecureCam* at stage 4 was less than the time taken by AES-CFB as shown in "Fig. 6 (a) and (b)" respectively. The FG and the BG encryption and storage with the Chacha20 cipher was faster in comparison with AES-CFB, which proves the efficiency of *SecureCam* application by utilizing the Chacha20 cipher for IoMT environment.

E. Comparative Analysis of SecureCam With Existing Approaches

The proposed *SecureCam* application for selective detection and encryption of objects in the dynamic camera videos is compared with other approaches in "Table X". Even though other approaches employed the use of object detection to identify the objects in the videos, none of them applied the attack prevention strategies using video segmentation and encryption altogether. Furthermore, the newly proposed object detection algorithm (*MF*) also proved to be more accurate and efficient in the FG and BG detection of the test videos than the SOTA DOF algorithm.

F. Limitations and Future work

This paper proposes a novel *SecureCam* application for protecting mobile camera videos against Replay and Man-in-the-Middle attacks using video frame segmentation and encryption techniques. Video segmentation into FG and BG was performed through a newly developed *MF* algorithm and encryption on detected parts was applied with the Chacha20 cipher. Despite its significance, this application has some limitations.

There is always an additional cost that should be paid to achieve Security as a service (SECaaS). There is an obvious trade-off among security, computational cost, and storage in all applications. Same is the case with *SecureCam* application, as the selective detection and encryption on segmented FG and BG result in the increase of video size by adding additional bits to these objects in order to change their appearance. As a result of storing them separately (FG and BG), the amount of storage space will automatically double. For static camera videos (CCTV), the storage issue can be handled through video summarisation methods (by not storing all detected static BG parts). However, for moving camera videos, the remedy should be further explored as a future research challenge.

Although this paper provides a detailed security and statistical analysis against attacks on *SecureCam*, in future we also intend to develop a threat model that will provide effective countermeasures to other types of stealthy threats and attacks on ROI based encrypted surveillance videos captured by dynamic camera devices.

V. Conclusion

The contribution of this paper is two-fold; firstly we develop an unsupervised learning algorithm *MF* for precisely detecting foreground and background ROIs in mobile/moving camera videos, secondly this paper proposes a novel application *(SecureCam)* for these videos. The *SecureCam* encompasses five implementation stages ("Fig. 1") with the linear time complexity of 3n (3 × Frame_Size). The application and the *MF* algorithm were extensively tested and compared with existing SOTA approaches in this study. The performance analysis of *MF* algorithm with SOTA DOF was compared using different parameters such as Accuracy, Precision, ESR, SSIM metrics, and detection timings. The results of these comparisons prove the efficiency and accuracy of object detection with *MF* for moving camera videos. The SE was applied using Chacha20 cipher in the *SecureCam*, and the computational cost analysis in terms of their execution timing (for encryption and decryption) was also compared to the SOTA AES cipher. In contrast to the industry cipher(AES), ChaCha20 is found to be efficient, making it an ideal choice for real-time video encryption on low-powered camera devices. Overall, the comparative performance, security, and statistical analysis of *SecureCam* demonstrate its effectiveness, efficiency, and robustness against Replay and/or Man-in-the-Middle attacks for moving camera surveillance videos in the IoMT infrastructure.

References

[1] S. Wang, L. Yu, and S. Xiang, "A low complexity compressed sensing-based codec for consumer depth video sensors," *IEEE Trans. Consum. Electron.*, vol. 65, no. 4, pp. 434–443, Nov. 2019.

[2] C. S. Lai, K. F. Tsang, and Y. Wang, "Electrification of smart cities," *Electronics*, vol. 11, no. 8, p. 114, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/8/1235

[3] E. Rubio-Drosdov, D. Díaz-Sánchez, F. Almenárez, P. Arias-Cabarcos, and A. Marín, "Seamless human-device interaction in the Internet of Things," *IEEE Trans. Consum. Electron.*, vol. 63, no. 4, pp. 490–498, Nov. 2017.

[4] G. Quiroz and S. Kim, "A confetti drone: Exploring drone entertainment," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, 2017, pp. 378–381.

[5] P. S. Chatterjee, N. K. Ray, and S. P. Mohanty, "LiveCare: An IoT-based healthcare framework for livestock in smart agriculture," *IEEE Trans. Consum. Electron.*, vol. 67, no. 4, pp. 257–265, Nov. 2021.

[6] J. Guo, M. Amayri, N. Bouguila, and W. Fan, "A hybrid of interactive learning and predictive modeling for occupancy estimation in smart buildings," *IEEE Trans. Consum. Electron.*, vol. 67, no. 4, pp. 285–293, Nov. 2021.

[7] M. R. Manesh and N. Kaabouch, "Cyber-attacks on unmanned aerial system networks: Detection, countermeasure, and future research directions," *Comput. Security*, vol. 85, pp. 386–401, Aug. 2019. [Online]. Available: https://doi.org/10.1016/j.cose.2019.05.003

[8] "Man arrested after hijacking car and threatening driver with knife." 2021. [Online]. Available: https://www.thejournal.ie/man-arrested-car-hijacked-limerick-portlaoise-5402052-Apr2021/

[9] B. Krishnamurthy, K. Naryshkin, and C. Wills, "Privacy leakage vs. protection measures: The growing disconnect," *Proc. Web*, vol. 2, pp. 1–10, May 2011.

[10] "What is GDPR, the EU's new data protection law?" GDPR.EU. 2022. [Online]. Available: https://gdpr.eu/what-is-gdpr/

[11] A. Boulmerka and M. S. Allili, "Foreground segmentation in videos combining general Gaussian mixture modeling and spatial information," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1330–1345, Jun. 2018.

[12] S. Li, J. Wu, C. Long, and Y.-B. Lin, "A full-process optimization-based background subtraction for moving object detection on general-purpose embedded devices," *IEEE Trans. Consum. Electron.*, vol. 67, no. 2, pp. 129–140, May 2021.

[13] Y. Yang, J. Ruan, Y. Zhang, X. Cheng, Z. Zhang, and G. Xie, "STPNet: A spatial-temporal propagation network for background subtraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2145–2157, Apr. 2022.

[14] A. Mohanty and S. Sanjivani, "A survey on moving object detection using background subtraction methods in video," in *Proc. Nat. Conf. Knowl. Innov. Technol. Eng. (NCKITE)*, Raipur, India, 2015, pp. 5–10.

[15] R. Firdousi and S. Parveen, "Local thresholding techniques in image binarization," *Int. J. Eng. Comput. Sci.*, vol. 3, no. 3, pp. 4062–4065, 2014.

[16] N. Senthilkumaran and S. Vaithegi, "Image segmentation by using thresholding techniques for medical images," *Comput. Sci. Eng. Int. J.*, vol. 6, no. 1, pp. 1–13, 2016.

[17] M. N. Asghar, N. Kanwal, B. Lee, M. Fleury, M. Herbst, and Y. Qiao, "Visual surveillance within the EU general data protection regulation: A technology perspective," *IEEE Access*, vol. 7, pp. 111709–111726, 2019.

[18] Y. Negi, "A survey on video encryption techniques," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 3, pp. 234–237, Apr. 2013. [Online]. Available: www.ijetae.com

[19] J. Shah and V. Saxena, "Video encryption: A survey," *Int. J. Sci.*, vol. 8, no. 2, pp. 525–534, 2011.

[20] F. Liu and H. Koenig, "A survey of video encryption algorithms," *Comput. Security*, vol. 29, no. 1, pp. 3–15, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2009.06.004

[21] M. Abomhara, O. Zakaria, and O. O. Khalifa, "An overview of video encryption techniques," *Int. J. Comput. Theory Eng.*, vol. 2, no. 1, pp. 103–110, 2009.

[22] P. K. Shukla, A. Khare, M. A. Rizvi, S. Stalin, and S. Kumar, "Applied cryptography using chaos function for fast digital logic-based systems in ubiquitous computing," *Entropy*, vol. 17, no. 3, pp. 1387–1410, 2015.

[23] "ChaCha20 and XChaCha20—PyCryptodome 3.9.9 documentation." Accessed: Mar. 20, 2022. [Online]. Available: https://pycryptodome.readthedocs.io/en/latest/src/cipher/chacha20.html

[24] R. Alyassi, M. Khonji, A. Karapetyan, S. C.-K. Chau, K. Elbassioni, and C.-M. Tseng, "Autonomous recharging and flight mission planning for battery-operated autonomous drones," 2017, *arXiv:1703.10049*.

[25] *Basic Safety*, IEC Standard 61010-031, 1998.

[26] M. Piccardi, "Background subtraction techniques: A review," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 4, 2004, pp. 3099–3104.

[27] P. Shah and H. Modi, "Comprehensive study and comparative analysis of different types of background subtraction algorithms," *Int. J. Image, Graph. Signal Process.*, vol. 6, no. 8, pp. 47–52, 2014.

[28] A. S. Murugan, K. S. Devi, A. Sivaranjani, and P. Srinivasan, "A study on various methods used for video summarization and moving object detection for video surveillance applications," *Multimedia Tools Appl.*, vol. 77, no. 18, pp. 23273–23290, 2018.

[29] S. Javed, P. Narayanamurthy, T. Bouwmans, and N. Vaswani, "Robust PCA and robust subspace tracking: A comparative evaluation," in *Proc. SSP*, Jun. 2018, pp. 836–840.

[30] P. Narayanamurthy and N. Vaswani, "A fast and memory-efficient algorithm for robust PCA (MEROP)," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 4684–4688.

[31] J. H. Giraldo and T. Bouwmans, "Semi-supervised background subtraction of unseen videos: Minimization of the total variation of graph signals," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2020, pp. 3224–3228.

[32] J. H. Giraldo, S. Javed, and T. Bouwmans, "Graph moving object segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 5, pp. 2485–2503, May 2022.

[33] M. Mandal and S. K. Vipparthi, "An empirical review of deep learning frameworks for change detection: Model design, experimental frameworks, challenges and research needs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6101–6122, Jul. 2022.

[34] T. Minematsu, A. Shimada, and R.-I. Taniguchi, "Rethinking background and foreground in deep neural network-based background subtraction," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 3229–3233.

[35] H.-G. Kim and G. Y. Kim, "Deep neural network-based indoor emergency awareness using contextual information from sound, human activity, and indoor position on mobile device," *IEEE Trans. Consum. Electron.*, vol. 66, no. 4, pp. 271–278, Nov. 2020.

[36] J. Bai, S. Lian, Z. Liu, K. Wang, and D. Liu, "Deep learning based robot for automatically picking up garbage on the grass," *IEEE Trans. Consum. Electron.*, vol. 64, no. 3, pp. 382–389, Aug. 2018.

[37] J.-H. Kim, S.-J. Choi, and J.-W. Jeong, "Watch & do: A smart IoT interaction system with object detection and gaze estimation," *IEEE Trans. Consum. Electron.*, vol. 65, no. 2, pp. 195–204, May 2019.

[38] Y. Yang, C. Liu, F. Chang, Y. Lu, and H. Liu, "Driver gaze zone estimation via head pose fusion assisted supervision and eye region weighted encoding," *IEEE Trans. Consum. Electron.*, vol. 67, no. 4, pp. 275–284, Nov. 2021.

[39] Y. Wu, X. He, and T. Q. Nguyen, "Moving object detection with a freely moving camera via background motion subtraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 2, pp. 236–248, Feb. 2017.

[40] C. H. Yeh, C. Y. Lin, K. Muchtar, H. E. Lai, and M. T. Sun, "Three-pronged compensation and hysteresis thresholding for moving object detection in real-time video surveillance," *IEEE Trans. Ind. Electron.*, vol. 64, no. 6, pp. 4945–4955, Jun. 2017.

[41] Y. Yu, L. Kurnianggoro, and K. H. Jo, "Moving object detection for a moving camera based on global motion compensation and adaptive background model," *Int. J. Control, Autom. Syst.*, vol. 17, no. 7, pp. 1866–1874, 2019.

[42] M. N. Chapel and T. Bouwmans, "Moving objects detection with a moving camera: A comprehensive review," *Comput. Sci. Rev.*, vol. 38, Nov. 2020, Art. no. 100310. [Online]. Available: https://doi.org/10.1016/j.cosrev.2020.100310

[43] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transp. Res. C, Emerg. Technol.*, vol. 6, no. 4, pp. 271–288, 1998.

[44] B. Leibe, K. Schindler, and L. Van Gool, "Coupled detection and trajectory estimation for multi-object tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[45] G. Farneb, "Two-frame motion estimation based on polynomial expansion," in *Proc. 13th Scandinavian Conf. Image Anal. (SCIA)*, Halmstad, Sweden, Jun./Jul. 2003, pp. 363–370.

[46] Y. Xiang, F. Wang, L. Wan, N. Jiao, and H. You, "OS-flow: A robust algorithm for dense optical and SAR image registration," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6335–6354, Sep. 2019.

[47] Z. Qu, L. Liang, and J. Lu, "Dense optical flow computation using textural features," in *Proc. 8th Int. Conf. Wireless Commun. Signal Process.*, 2016, pp. 1–5.

[48] A. Kushwaha, A. Khare, O. Prakash, and M. Khare, "Dense optical flow based background subtraction technique for object segmentation in moving camera environment," *IET Image Process.*, vol. 14, no. 14, pp. 3393–3404, Aug. 2020. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-ipr.2019.0960

[49] J. Maier and M. Humenberger, "Movement detection based on dense optical flow for unmanned aerial vehicles," *Int. J. Adv. Robotic Syst.*, vol. 10, pp. 1–11, Jan. 2013.

[50] M. Yazdi and T. Bouwmans, "New trends on moving object detection in video images captured by a moving camera: A survey," *Comput. Sci. Rev.*, vol. 28, pp. 157–177, May 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013716301794

[51] A. M. Khan, "Image segmentation methods: A comparative study," *Int. J. Soft Comput. Eng.*, vol. 3, no. 4, pp. 84–92, 2013.

[52] D. Bradley and G. Roth, "Adaptive thresholding using the integral image," *J. Graph. Tools*, vol. 12, no. 2, pp. 13–21, 2007.

[53] A. K. Chaubey, "Comparison of the local and global thresholding methods in image segmentation," *World J. Res. Rev.*, vol. 2, no. 1, pp. 1–4, 2016. [Online]. Available: https://www.wjrr.org/download_data/WJRR0201009.pdf

[54] S. Ferrari, *Image Segmentation (Image Processing I)*, Università Degli Studi di Milano, Milan, Italy, 2012.

[55] O. A. Khashan, A. M. Zin, and E. A. Sundararajan, "Performance study of selective encryption in comparison to full encryption for still visual images," *J. Zhejiang Univ. Sci. C*, vol. 15, no. 6, pp. 435–444, 2014.

[56] P. Yadav, I. Gupta, and S. K. Murthy, "Study and analysis of eSTREAM cipher salsa and ChaCha," in *Proc. 2nd IEEE Int. Conf. Eng. Technol.*, 2016, pp. 90–94.

[57] K. Deshpande and P. Singh, "Performance evaluation of cryptographic ciphers on IoT devices," 2018, *arxiv:1812.02220*.

[58] S. Ghatak, S. Rup, B. Majhi, and M. N. S. Swamy, "HSAJAYA: An improved optimization scheme for consumer surveillance video synopsis generation," *IEEE Trans. Consum. Electron.*, vol. 66, no. 2, pp. 144–152, May 2020.

[59] E. Bursztein, "Google online security blog: Speeding up and strengthening HTTPS connections for chrome on Android." Apr. 2014. [Online]. Available: https://security.googleblog.com/2014/04/speeding-up-and-strengthening-https.html

[60] Y. Nir and A. Langley, "ChaCha20 and Poly1305 for IETF protocols," IETF, RFC 7539, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7539

[61] "Pixabay Webpage." Accessed: Jan. 30, 2022. [Online]. Available: https://pixabay.com/videos/

[62] "Motchallenge Webpage." Accessed: Feb. 20, 2022. [Online]. Available: https://motchallenge.net/data/MOT16/

[63] "Pexels Webpage." Accessed: Jan. 30, 2022. [Online]. Available: https://www.pexels.com/

[64] "Cisco Stealthwatch security events and alarm categories 7.2.1." 2020. [Online]. Available: https://www.cisco.com/c/dam/en/us/td/docs/security/stealthwatch/management_console/securit_events_alarm_categories/SW_7_2_1_Security_Events_and_Alarm_Categories_DV_1_0.pdf

[65] M. N. Asghar and M. Ghanbari, "An efficient security system for CABAC bin-strings of H.264/SVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 425–437, Mar. 2013.

[66] W. Jang and S.-Y. Lee, "Partial image encryption using format-preserving encryption in image processing systems for Internet of Things environment," *Int. J. Distrib. Sens. Netw.*, vol. 16, no. 3, 2020, Art. no. 1550147720914779. [Online]. Available: https://doi.org/10.1177/1550147720914779

[67] X. Liu, S. Qiu, Y. Cui, and X. Meng, "A data encryption and fast transmission algorithm based on surveillance video," *Wireless Commun. Mobile Comput.*, vol. 2020, p. 12, Aug. 2020. [Online]. Available: https://doi.org/10.1155/2020/8842412

[68] S. W. Kang and U.-S. Choi, "ROI image encryption using YOLO and chaotic systems," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 7, pp. 1–9, Jan. 2021.

[69] X. Tian, P. Zheng, and J. Huang, "Robust privacy-preserving motion detection and object tracking in encrypted streaming video," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 5381–5396, 2021.

Ifeoluwapo Aribilola received the B.Tech. degree from the Department of Computer Science, School of Science, Federal University of Technology Akure, Nigeria, in 2012 and the M.Sc. degree from the Department of Computer and Software Engineering, Faculty of Engineering and Informatics, Athlone Institute of Technology, Athlone, Ireland, in 2019. She is currently pursuing the Ph.D. degree with the Software Research Institute, Technological University of the Shannon: Midlands Midwest (Athlone Campus), Athlone. She has over seven years of working experience as a Software Engineer in different IT companies. Her research interests include, but are not limited to, deep learning, computer vision, visual privacy, threat modeling, cryptology, and cybersecurity.

Mamoona Naveed Asghar (Senior Member, IEEE) received the Ph.D. degree from the School of Computer Science and Electronic Engineering, University of Essex, Colchester, U.K., in 2013. She is a former Marie Sklodowska-Curie Career-Fit Postdoctoral Research Fellow with AIT, Ireland, from 2018 to 2021. She is currently working as a Lecturer/Assistant Professor in the Cybersecurity Discipline with the School of Computer Science, COSE, University of Galway, Ireland. She has published several ISI-indexed journal articles along with numerous international conference papers. Her research interests include security aspects of multimedia, video compression, visual privacy, encryption, steganography, secure transmission in future networks, video quality metrics, key management schemes, computer vision algorithms, malware detection/classification deep learning models, and General Data Protection Regulation. She is actively involved in reviewing research articles for renowned journals/conferences and has also participated as the session chair in different conferences.

Nadia Kanwal (Senior Member, IEEE) received the masters and Ph.D. degrees in computer sciences from the University of Essex, U.K., in 2009 and 2013, respectively. She has also received a prestigious Marie Skłłodowska-Curie Research Fellowship (for three years) in 2019 to do an industry led project related to secure and privacy protected CCTV video storage and retrieval as a Principal Investigator. Her research interests include computer vision and machine learning. She is actively working on applying machine learning to develop cutting-edge solutions for health, security, and vision applications. Her publications on multimedia data security, visual privacy, low-level image features, virtual reality, EEG/ECG signal analysis, and pupillometry have been very well supported by the research community. She is an active reviewer of good-standing journals and conferences.

Martin Fleury (Member, IEEE) received the degree in modern history from Oxford University, U.K., the degree in mathematics/physics from The Open University, Milton Keynes, U.K., the first M.Sc. degree in astrophysics from QMW College, University of London, U.K., in 1990, the second M.Sc. degree in parallel computing systems from the University of South West England, Bristol, in 1991, and the Ph.D. degree in parallel image-processing systems from the University of Essex, Colchester, U.K., where he became a Senior Lecturer and after a Visiting Fellow. He is currently associated with the University of Suffolk, Ipswich, U.K. He is also a Free-Lance Consultant. He has authored or coauthored around 300 research articles and book chapters on topics, such as document and image compression algorithms, performance prediction of parallel systems, software engineering, recongurable hardware, and video systems. He has additionally published or edited books on high-performance computing for image processing and peer-to-peer streaming. His current research interests include video communication over wireless networks and multimedia network security.

Brian Lee received the Ph.D. degree from Trinity College Dublin, Dublin, Ireland, in the application of programmable networking for network management. He is the Director of the Software Research Institute, Athlone Institute of Technology, Athlone, Ireland. He has over 25 years research and development experience in telecommunications network monitoring, their systems and software design and development for large telecommunications products with very high impact research publications. Formerly, he was the Director of research for LM Ericsson, Ireland, with responsibility for overseeing all research activities, including external collaborations and relationship management. He was an Engineering Manager with Duolog Ltd., where he was responsible for strategic and operational management of all research and development activities.