IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 1, 2018, accepted February 27, 2018, date of publication March 15, 2018, date of current version April 18, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2816162

Analysis of Binary Image Coding Methods for Outdoor Applications of Wireless Vision Sensor Networks

KHURSHEED AURANGZEB[1,2], MUSAED ALHUSSEIN[1], AND MATTIAS O'NILS[3]

[1]Computer Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
[2]Department of Electrical Engineering, COMSATS Institute of Information Technology, Attock 43600, Pakistan
[3]Electronics Design Division, Mid Sweden University, 851 70 Sundsvall, Sweden

Corresponding author: Khursheed Aurangzeb (kaurangzeb@ksu.edu.sa)

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through the research group NO (RG-1438-034).

ABSTRACT The processing of images at the vision sensor nodes (VSN) requires a high computation power and their transmission requires a large communication bandwidth. The energy budget is limited in outdoor applications of wireless vision sensor networks (WVSN). This means that both the processing of images at the VSN and the communication to server must be energy efficient. The wireless communication of uncompressed data consumes huge amounts of energy. Data compression methods are efficient in reducing data in images and can be used for the reduction in transmission energy. We have evaluated seven binary image coding techniques. Our evaluation is based on the processing complexity and energy consumption of the compression methods on the embedded platforms. The focus is to come up with a binary image coding method, which has good compression efficiency and short processing time. An image coding method with such attributes will result in reduced total energy requirement of the node. We have used both statistically generated images and real captured images, in our experiments. Based on our results, we conclude that International Telegraph and Telephone Consultative Committee Group 4, gzip_pack and JPEG-LS are suitable coding methods for the outdoor applications of WVSNs.

INDEX TERMS Embedded systems, energy consumption, image compression, wireless vision sensor network.

I. INTRODUCTION

There are a lot of applications of Wireless Vision Sensor Network (WVSN) but our discussion is limited to those which are based on binary images. The cyclist and human detection, for ensuring their safety, based on segmented bi level images has been presented in [1]. The sky surveillance in relation to the detection of birds which are flying towards the wind mills and wind turbines, thus providing the ability to divert them and avoid possible collisions with the wind turbines was investigated in [2].

Some other examples which are based on binary images include the automatic monitoring of meter readings [3], the detection of magnetic particles in hydraulics systems [4] for failure prediction/avoidance, human detection [5] and robot localization [6]. In all these applications, bi level image processing algorithms have been applied.

This miscellaneous set of applications may necessitate a significantly huge amount of Vision Sensor Nodes (VSNs) in relation to uninterrupted monitoring [1]. Connecting the VSN through wires for communicating with each other's and with server for such outdoor applications is too expensive and is impractical [2].

Hence for such outdoor applications, the deployment of battery operated VSN is essential. Typically, every node consists of a camera, processor, storage and a radio link. The camera based sensor networks are appropriate for their installation in faraway areas. The camera sensors provide wider coverage and richer information which enable WVSNs for monitoring events in wide areas [1]–[6].

The image processing flow from image capturing up to feature extraction includes many complex image processing algorithms such as filtering, frame-subtraction, pixels based operations (segmentation, morphology), labeling, object dimension extraction etc.

The VSN must be able to perform these complex image processing tasks using onboard tiny/weak processor and

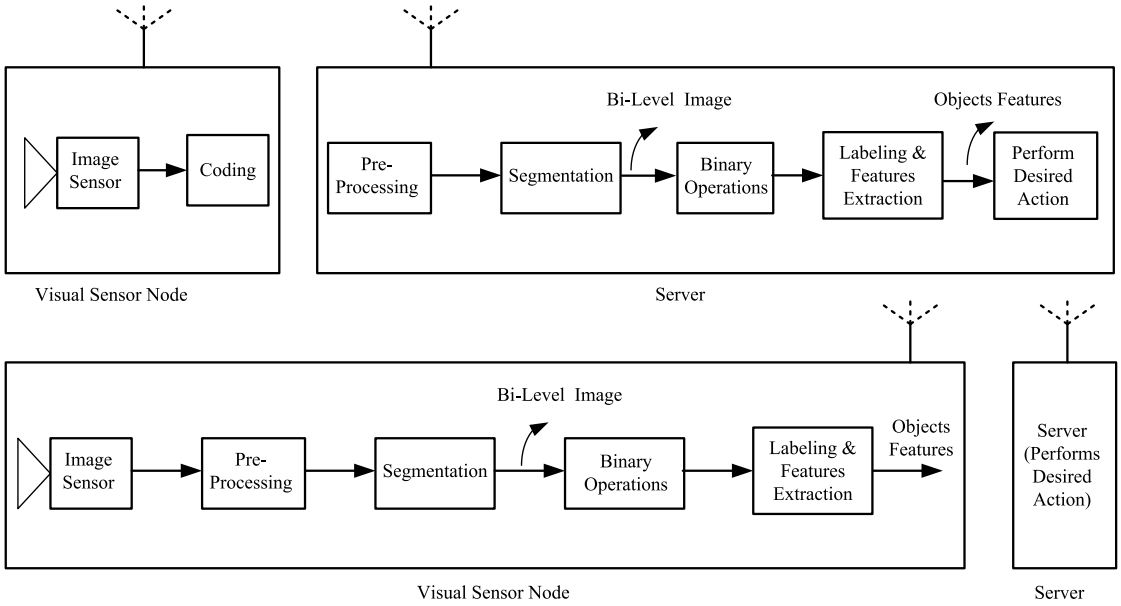

FIGURE 1. The two image processing extremes in WVSN.

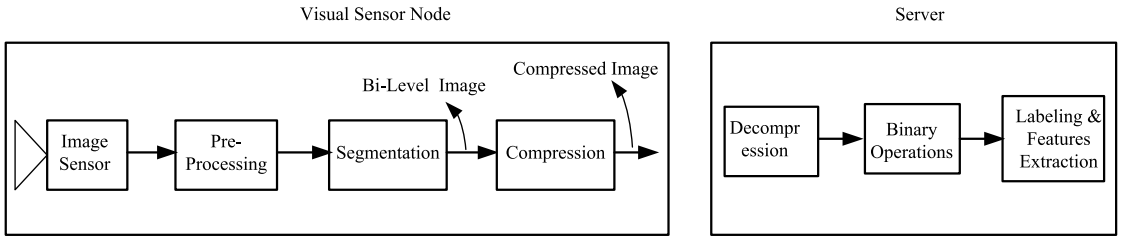

FIGURE 2. Proposed architecture for the implementation of the VSN.

communicate the results wirelessly. Typically, the VSN is composed of the hardware components with low power consumption characteristics. The communication bandwidth and the energy budget are the major constraints in outdoor applications of WVSN.

Tasks execution at VSN and wireless transmission consume significant power. Transmitting the raw data results in reduced processing time but its consequence is increased transmission energy. At the other extreme, implementing all processing tasks locally at VSN and communicating the processed images, reduces transmission energy, but, the disadvantage is the increased onboard execution time. These two extremes of processing are shown in Fig. 1.

We have concluded in our previous work on intelligence partitioning (IP) in [7] and [8], that selecting a suitable IP method results in reduced energy consumption. However, wireless communication of the uncompressed data will quickly drain total energy of VSN. Transmission energy is mainly reliant on data which is transmitted to server.

In our previous work in [9], we proposed an architecture for some applications of WVSN. In this architecture, we proposed to compress the binary image after pixels based operations at node and transmit the results to server. The general architecture from [9] is shown in Fig. 2 which shows that the remainder of the tasks are shifted to the server. The size of the compressed image in Fig. 2 is reliant on the applied compression technique. Additionally, the execution time of the used compression technique has an effect on the node's overall energy consumption. The algorithmic steps for our proposed framework is shown in Fig. 3.

In machine vision, images contain few randomly placed objects. There is a possibility for there to be significant variations in the shape, the sizes and positions of objects in the set of captured frames. So, for analyzing the compression standards, a big volume of statistically generated and real taken images are needed.

In this work, our aim is to explore various well known binary image coding methods for their usage in outdoor applications of WVSNs. The energy budget is limited in outdoor applications of WVSN and hence the selected method must be efficient in terms of compression efficiency, compression time and total energy consumption on the embedded platform. We have evaluated seven well known binary image coding methods based on their compression efficiency, compression time and total energy consumption on the embedded platforms. Reduced total energy consumption will lead to

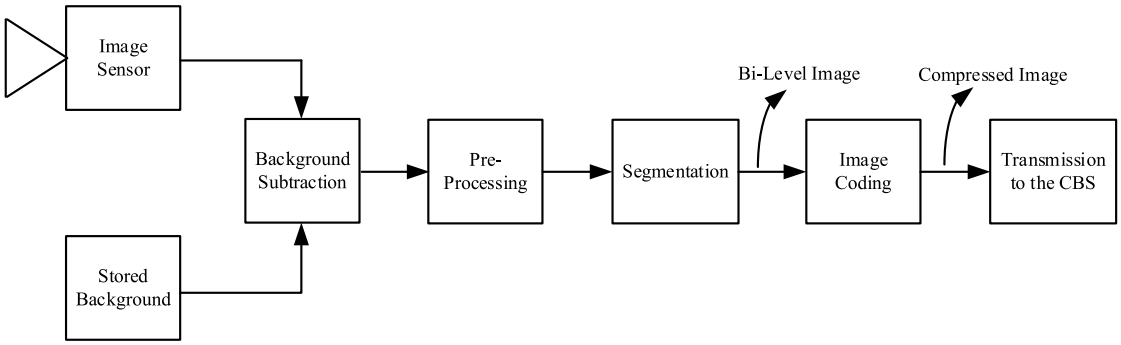

FIGURE 3. Algorithm flow for the proposed framework.

longer lifetime of the VSN which is desirable in outdoor applications of WVSNs.

Our goal is to come up with a coding technique which is robust to the varying characteristic of objects in both statistically generated as well as real captured images. This method must be efficient in terms of compression ratio, processing time and energy consumption and hence will be appropriate for outdoor applications of WVSNs.

The remainder of the paper is planned as follows. Section II provides the related work while Section III elaborates the evaluation criterion in relation to analyzing the processing complexity of the bi-level image coding technique. The performance evaluation of the compression methods is discussed in section IV. Section V provides the verification of the results based on real captured images. Finally, the conclusion is provided in section VI.

II. RELATED WORK

Typical examples of WVSNs were studied in [10]–[13]. Downes *et al.* [10], developed a mote for image based sensor network. They analyzed the processing and memory limitations in current mote designs and have developed a powerful platform. Their mote is based on ARM7 micro-controller operating at 48 MHz and 64 KB RAM.

Rowe *et al.* [11] presented a CMUcam3, which is an open source platform that can perform image processing. The platform is composed of a camera, an ARM7TDMI ucontroller and memory. The impact of Relaying Packets in a network with different node densities, on the development/Implementation concerns of the camera based mote, was explored in [12]. Chen *et al.* [13], proposed and demonstrated CITRIC, which is a camera based wireless network. The CITRIC is composed of a camera, a processing unit and memory and is capable of executing in-network image processing tasks. In our current analysis, three computing architectures have been used, based on AVR32, ARM and Intel, which are NGW100 mkII, BeagleBoard-xM and a laptop machine. The two embedded computing platforms are selected for our analysis because they are comparable with those typically used for the implementation of a VSN.

The NGW100 mkII kit uses the AT32AP7000 which has a 32-bit digital signal processor. The kit has 256 MB Random Access Memory (RAM) and 256 MB NAND flash. The AT32AP7000 operates at 150 MHz clock. The other computing platform is BeagleBoard-xM which has an ARM ® Cortex TM-A8 running at 1 GHz and 512MB RAM. The third computing platform is a laptop with an Intel ® processor (Core^{TM} 2 Duo, 1.86 Giga Hrtz) and 3 GB RAM.

Based on the enormous amount of information in the images and consequently the high communication energy consumption of VSN, image compression has received significant attention from researchers [14]–[20].

Nasri *et al.* [14] performed image coding based on discrete wavelet transform and embedded block coding with optimized truncation, in a distributed way. They used energy consumption and image quality for assessing the performance of their technique.

Another study of image compression has been conducted in [15], in which the JPEG standard was implemented in reconfigurable hardware. They applied a modular implementation of the JPEG algorithm for increasing the lifetime of the node. The selection of binary image compression methods based on compression efficiency has been explored in [16]–[19]. But in these explorations, some of the main parameters such as processing complexity and the energy consumption of the compression methods on the physical embedded platform, were not considered.

A distributed image compression in image based sensor networks has been explored in [20]. The authors presented a novel technique for collaborative image communication in WVSN for avoiding the extra energy consumption during unnecessary data transmission.

The issues associated with the majority of the work relating to image compression in WVSN, is in relation to the high computational complexity of the compression process. For some applications of WVSN such as those in [1]–[6], the captured frames could be converted into binary frames. The binary image compression standards are computationally faster than those in [14], [15], and [20].

However, many binary image coding methods based on different algorithms exist, but, to the best of our knowledge, there is no analysis of which method have low total energy consumption characteristic for the energy constrained outdoor applications of WVSNs.

Few researchers, including those in [21]–[24] have compared the image compression standards, however, the problem associated with all of these comparisons is that their focus is only on the investigation of the compression ratio and the speed on the personal computer. Analysis based on the energy consumption and computational complexity of the compression standards on the embedded computing platform has not been explored.

Veeraputhiran and Sankararajan [25] explored a security enhanced video codec for bandwidth reduction in sensor network using compressive sensing. Hamzaa *et al.* [26] explored a framework for summarization of video which is highly secure. They suggested to extract key-frames by applying a video summarization scheme which is very light-weight.

Hamza *et al.* [27], studied confidentiality of key-frames. They applied video summarization for extracting key-frames from diagnostic hysteroscopy data. They explored a color image coding method for increasing the security of extracted key-frames.

The exploration of DISCOVER [28], H.264 Intra [29] and DCVS [30] for WVSNs has been recently presented in [31]. They evaluated these methods based on total energy consumption, computation complexity, lifetime, decoding complexity and quality of reconstructed images. The processing time of these codecs is high, which result in excessive energy burden.

Another most recent research work on data reduction in WVSN is presented by Girod *et al.* [32]. They investigated Distributed Video Coding (DVC) which employs encoders with reduced complexity at node, by transferring most of the computationally extensive operations to the central base station. In other words, in DVC, the computationally extensive and energy draining tasks are shifted to the server, which is anticipated to be resourceful.

A widespread review of the codecs which are based on DVC and their target was WVSN applications, was recently reported by Imran *et al.* [33]. Their widespread review consists of a reasonable debate of various video codecs. The DVC uses discrete cosine transformation (DCT) for compression. DCT requires extensive computation and its computational complexity is very high. Performance of change coding for data reduction in WVSN has recently been explored in [41].

Hence, a study is required in relation to the selection of the most suitable bi-level image coding method for energy constrained outdoor embedded applications of WVSN. In current work, we explore binary image coding methods in terms of processing complexity and energy consumption for outdoor applications of WVSN. We have studied seven binary image coding techniques and evaluated the computational complexity and energy consumption associated for each of these methods. The considered methods include CCITT Group 4 & CCITT Group 3 [34], JBIG2 [35] which is based on Arithmetic Coding in [36], Gzip based on [37] (and Gzip_pack for packed images), JPEG-LS [38] which is based on LOCO-I [39], and the rectangular compression method [40].

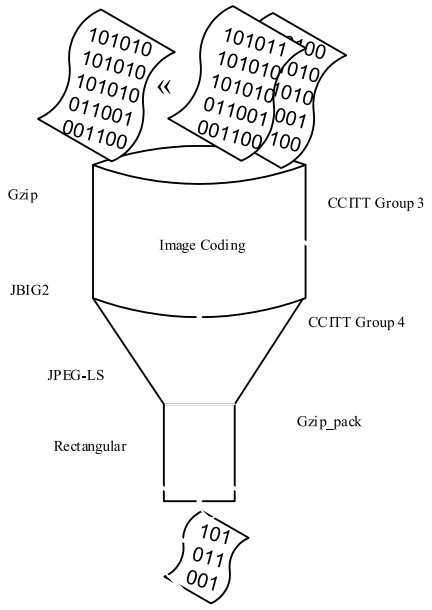

FIGURE 4. The pool of the image coding methods.

III. EVALUATED COMPRESSION METHODS

The focus of our work is on the analysis of various compression methods for their usage in a set of applications of WVSN. Fig. 4 shows the set of image compression methods which we have considered in our evaluation. In order to analyze the pool of image compression methods for such applications, a comprehensive set of statistically generated images with various features of the objects is required. These computer-created images should have objects with different desired features such as various sizes, locations, shapes and numbers. We have developed a statistical model which generates images with all such features.

We want to use these computer-generated images to explore the trend of seven binary image coding techniques in various ways. Precisely, our aim is to explore the impact of an increasing number, increasing size and different geometrical shapes (of objects), on compression ratio and execution time of the considered methods. We want to find a coding standard which is robust to the mentioned attributes of the objects in the Matlab-generated set of statistical images. Additional, we have used the real captured images to verify the results obtained using statistical analysis.

The well-known binary image compression methods are briefly described below and appropriate references are provided for interested readers.

A. CCITT GROUP 3 (2D) AND GROUP 4

The CCITT has established some communication protocols for coding binary images. Two of their established protocols are commonly known as Group 3 and Group 4. These coding methods uses Huffman coding for compressing images. For each pixel of the image, in Group 3 and Group 4 algorithms, there is a comparison operation in order to detect a changing picture element (transition from white to black & vice versa).

When a changing picture element is detected, subtraction is performed in order to calculate the distance from the previous changing picture element.

If the distance is greater than 64 then it is repeatedly divided by 64 until the remainder is less than 64. Based on the distance and remainder, there could be two to four memory accesses for accessing Huffman codes which are accessed and placed in the output buffer. This procedure is repeated for the whole image. Finally, the output buffer, together with the header information, is written into the output image. Interested readers are referred to a detailed explanation of these standards in [34].

B. THE JBIG2

JBIG2 [35] is an image coding method for compressing binary images with best compression ratio. The biggest problem with IBIG2 is that it uses arithmetic coding [36] for compressing images. The computational complexity of JBIG2 is high, which results in longer execution time on the embedded platform. If it is provided with correct probabilities, then it produces highly compressed images. In arithmetic encoding the frequently occurring characters are represented by smaller amount of bits and vice versa. This procedure results in fewer bits for the overall representation of any string of characters. Arithmetic coding produces a single number for the entire message.

C. THE Gzip

The Gzip is based on the algorithm in [37]. In our experiments, Gzip is used for compressing binary images in two ways: standard Gzip and Gzip_pack. The same implementation is involved in both cases, but, for standard Gzip, normal binary images are compressed which are one byte per pixel. In relation to the Gzip_pack, eight pixels of the binary image are filled to compose a byte. Then this composed byte is coded with standard GNU zip.

In Gzip coding, each new pixel of the image is compared to a pixel in the sliding window until a match is found. In the worst case scenario, it is possible that there could be N comparison for each new byte (pixel). With this method, the compression is achieved by replacing long sequences of pixels with two binary numbers. The interested readers are referred to the details of Gzip in [37].

D. JPEG-LS

The JPEG-LS in [38] compression standard uses LOCO-I [39] for compressing images. The JPEG-LS achieved compression ratios which are similar to those which obtained best compression results and its compression time is comparatively low. The complete description of JPEG-LS and LOCO-I can be found in [38] and [39] respectively.

E. RECTANGULAR COMPRESSION METHOD

Mohamed and Fahmy [40] explored a binary image coding method which is based on extracting rectangular regions from the image. According to this method, non-overlapping rectangular regions of black color are determined in image and then their two corners (vertices) are saved. These vertices can be used at the receiver side in order to fully reconstruct the bi-level image and, by this means, compression is achieved. This method is efficient in relation to the compression of bi-level images which only involve rectangle shaped objects. The compression efficiency of this method is poor for images with irregular shaped objects such as curves which is discussed with details in [18].

IV. EVALUATION CRITERION

An important performance feature, which must be considered in analyzing compression methods, is the processing complexity (execution time) for both the compression and decompression processes. A compression algorithm is deemed to be of little use if its execution time introduces an insufferable delay in the image processing application.

An increased execution time results in greater processing energy consumption. Since, in remote applications of WVSNs, the energy consumption is the main concern because of the limited energy resources. Thus, the processing complexity and energy consumption must be evaluated for the considered compression standards for deciding which ones provide best features.

The execution time and compression ratio of binary coding methods depends on the switching of the pixel's values from one to zero and vice versa in the image. For our evaluating the compression methods, we produced images (640 columns and 400 rows) with randomly placed objects in the frames.

The number of objects were increased in the first set of generated images while the sizes of the objects were increased in our second set of generated images. The objects in both the sets have different shapes such as quarters of ellipses, semi-ellipses, ellipses, quarters of circles, semi-circles, circles, curves and rectangles. Our aim is to find the trend in relation to the processing time on the embedded platform for the compression methods, relating to a variety of changes in the generated sets of images.

The compression ratio of a compression method does not depend on the software/hardware design of the system. However, processing complexity of a compression method is dependent on both the hardware and software design of the system.

The aim of our current study is not to find the absolute computational complexity of a compression method on a specific machine, but is, instead, to analyze the trend of the different compression methods on the two embedded computing platforms.

Generally speaking, the finding of current research work will assist the readers in deciding which compression method is computationally fast. In other words, it will enable the readers to decide which coding methods are suitable for outdoor applications of WVSNs.

A. EFFECT OF INCREASING NUMBER OF OBJECTS

For the different object' shapes, such as circles, rectangles, ellipses, curves, we generated 10 images

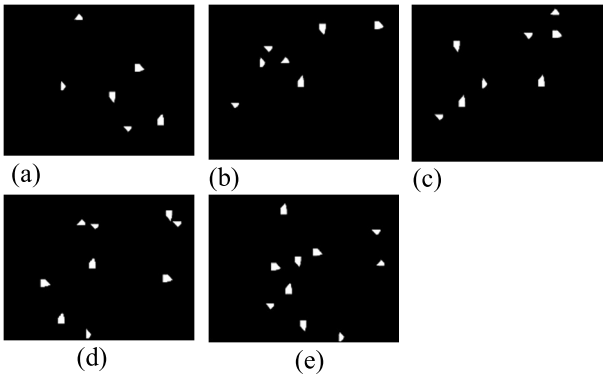

FIGURE 5. Five frames with increasing no. of randomly placed objects.

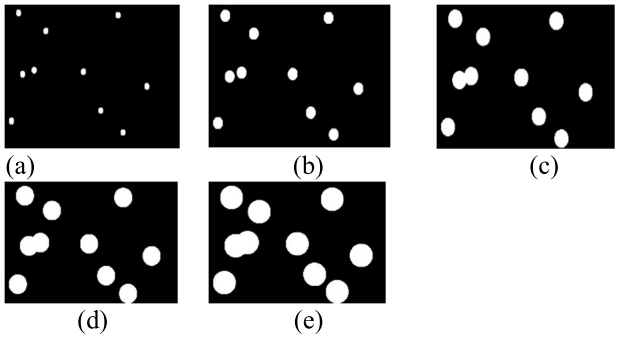

FIGURE 6. Five frames showing increase in size of randomly placed objects.

TABLE 1. Analysis for increasing no. of objects.

Method	Mean Processing Time(t in ms)	Standard Deviation (σt ms)
Group 4	140	7
Group 3	140	2
JBIG2	348	5
Rectangular	554	111
Gzip	614	162
Gzip_pack	138	120
JPEG-LS	50	2

(using Matlab). From Image1 to Image10, the number of objects were increased by 10% in the consecutive sequence of the frames. In total, 80 images were analyzed for eight shapes of objects.

The Fig. 5 shows 5 images with a quarter of an ellipse. The gradual increase as well as the arbitrariness in the occurrence of the objects in the images is clear in Fig. 5. We presented the compression ratio for these compression methods based on the sets of the generated images in [18].

In the current work, all the frames with various shapes of objects are executed on the three computing platforms and the processing complexity of each of the compression methods is determined. Some of the methods proved to have a higher standard deviation in the processing complexity on the embedded platform. The mean value and the standard deviation in processing complexity based on based NGW100 mkII, for the compression methods are shown in Table 1 in section IV. The analysis of processing complexity and the energy consumption based on two embedded computing platforms is provided in section V.

B. EFFECT OF INCREASING SIZE OF THE OBJECT

For different object' shapes including ellipses, rectangles, ellipses and circles, we generated ten frames using our developed code (Matlab code) with 10% increase in the objects' sizes in the consecutive from Image1 to Image10.

In total, we analyzed 80 images having different shapes of objects (8 different shapes). The 5 images with increasing size of the randomly placed circles are shown in Fig 6. We presented the compression ratio for these compression methods based on the sets of the generated images in [18].

In the current work, all these sets of images are executed on the target embedded platforms. The processing complexity and the associated standard deviations are determined. For some of the methods, the standard deviation in relation to the processing complexity on the embedded platform is high, while for others it is low.

The mean value and standard deviation of processing complexity are determined for NGW100 mkII and are shown in Table 2. The analysis for processing complexity and the energy consumption for the two embedded computing platforms is provided in section V.

V. PERFORMANCE EVALUATION

This section consists of the results of the processing complexity of the compression methods for two sets of images, having objects with a variety of features including increasing numbers, sizes and shapes. The execution file and respective libraries of all the compression standards are downloaded to the target embedded platforms and are used to compress all the images. In order to be able to provide a high accuracy, the average execution time in relation to compressing each image ten times, is determined.

The mean processing time and its standard deviation are shown in Table 1 for the sets of images with increasing no. of objects. The larger values for the standard deviation in processing time in Table 1, show that the Gzip and Rectangular compression are highly dependent on the contents of the input image (i.e. number of objects).

On the other hand, the JPEG-LS, CCITT Group 3, 4 and JBIG2 are resilient, in terms of processing complexity, to the contents of the input image. Table 1, also shows that Group 3, 4, JBIG2 and JPEG-LS are the least sensitive (low standard deviation) for an increase in the no. of objects in the frames. Additionally, average processing time for the JPEG-LS, CCITT Group 3, 4 and gzip_pack on the target embedded platform are lesser compared to the other coding methods.

Table 2 shows the standard deviation and the mean execution time on the embedded computing systems based on sets of generated images having objects with gradual increase in their sizes. It is clear from Table 2, that the processing

TABLE 2. Analysis for increasing size of objects.

Method	Mean Processing time (t in ms)	Standard Deviation (σt ms)
Group 4	140	3
Group 3	140	2
JBIG2	351	8
Rectangular	600	215
Gzip	616	136
Gzip_pack	184	178
JPEG-LS	51	3

TABLE 3. Analysis of the compression ratio.

Method	Original File Size (OFS) Bits	Original File Size (OFS) Bytes	Compressed File Size (CFS) Bytes	Compression Ratio = OFS/CFS
Group 4	256000	32000	488	66
Group 3	256000	32000	2198	15
JBIG2	256000	32000	315	102
Rectangular	256000	32000	777	42
Gzip	256000	32000	2152	15
Gzip_pack	256000	32000	619	52
JPEG-LS	256000	32000	746	43

time for the Gzip, Gzip_pack and Rectangular compression techniques have high sensitivity for the increasing size of objects.

On the other hand, in terms of processing time, JPEG-LS, Group 3, 4 and JBIG2 are less sensitive (low standard deviation) to the gradual growth of the objects' size. Additionally, the mean processing time for the JPEG-LS, Group 3, 4 and gzip_pack on the embedded platform are quite low.

Based on the observation in Table 1, 2, it is possible to generalize that the CCITT Group 3, 4, JBIG2 and JPEG-LS are resilient to the variations in terms of number of object and the size of the objects in the images.

Table 3 shows the compression ratio for the considered compression standards where the compressed file size (column 4) is from our previous work [18]. It must be observed that the compression ratio of JBIG2 is the highest and that for Group 3 and Gzip are the lowest.

Fig. 7 shows the processing complexity (processing time) vs. mean compressed file size for the studied coding methods. The average (mean) compressed file size is determined using all the images in the two sets of generated images (including both the sets with increasing number and sizes of the objects). The circles in Fig. 7, show the mean processing complexity on NGW100 mkII at the y-axis while the x-axis shows the average compressed file size.

The vertical lines in Fig. 7 show the standard deviation in the mean execution time (processing complexity). Similarly, the standard deviation in the mean compressed file size is shown using the horizontal lines in the same figure.

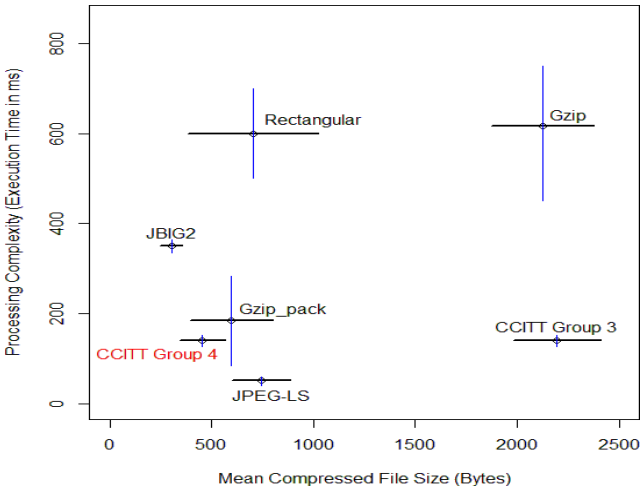

FIGURE 7. Processing complexity vs. average files size.

The Fig. 7 shows that Gzip and CCITT Group 3 have higher mean compressed file sizes. We must also observe that processing complexities are high for Gzip, Rectangular and JBIG2 coding methods. So, we may say that JPEG-LS, Gzip_pack and CCITT Group 4 are suitable for energy limited outdoor applications of WVSNs.

However, the trends of the compression methods must be verified on another computing platform and for the real captured images (which is done in Fig. 10). In addition, the energy consumption of all the compression methods needs to be evaluated for providing an accurate conclusion.

VI. VERIFICATION OF THE PERFORMANCE METRICS

The processing time of any compression method needs to be analyzed on various embedded platforms. However, the compression ratio is independent of the speed of the machine. Due to this reason, we have analyzed processing time of the considered compression methods on three different machines with different processing capabilities.

The compression ratio of the considered compression methods is analyzed and shown in Fig. 8. The analysis of the processing time for the considered compression methods on NGW100 mk II is shown in Fig. 9. Furthermore, the comparative analysis of the processing time for the considered compression methods on three machines is shown in Fig. 10.

The analysis based on simulation results are far from the physically measured energy consumption on the actual embedded platform. So, we have evaluated the considered compression methods based on both real and statistically generated images. In addition to simulation, we have measured the actual energy consumption on two embedded platforms i.e. NGW100 mkII and BeagleBoard-xM and have shown the results in Fig. 11. Our aim is to come up with an image coding method which has low total energy consumption and hence is suitable for outdoor applications of WVSNs.

In order to verify the results of section 5, 50 frames with varying number of objects of various shapes were captured and are used in our analysis. The average compressed file size for both the statistically generated images and the captured

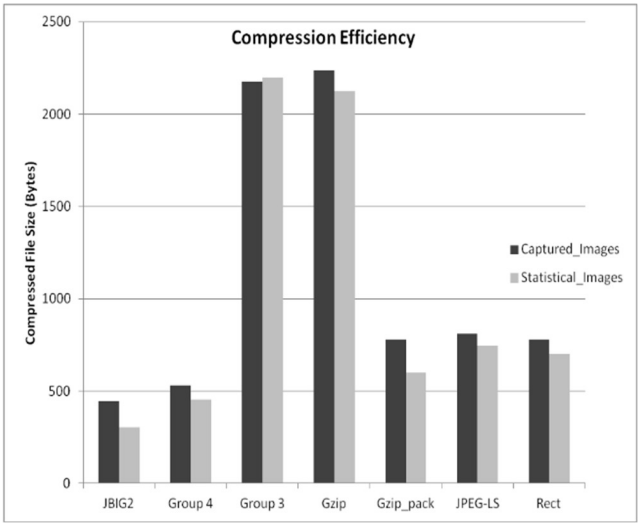

FIGURE 8. The mean compressed files size for captured and generated images.

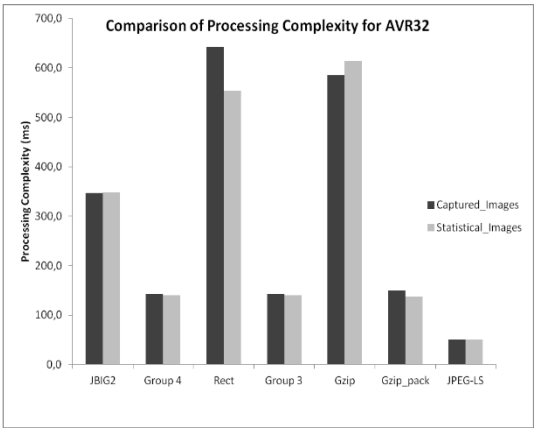

FIGURE 9. Encoding time for captured and statistical images.

images is shown in Fig. 8. The significance of the results in Fig. 8 is that the compressed file sizes for the statistical and captured images are almost identical. This verifies one aspect of the developed statistical model, i.e. that the average compressed file size is almost identical for both the statistically generated and the real captured images.

Fig. 9 shows the comparison of the encoding time for the captured and statistically generated images. It shows that for the same computing architecture (AVR32), the encoding times of all the compression methods for both the statistical and captured images are almost identical with the exception of Gzip and Rectangular. This verifies the high standard deviation of these methods, which was observed in the results of Section IV.

The encoding times (processing complexity) of all the seven compression methods for the three computing platform based on the real captured images are shown in Fig. 10. The significant observation in Fig. 10 is that the encoding time is lowest on the Intel machine and highest on the AVR32 embedded platform (The faster the computing system the lower the execution time).

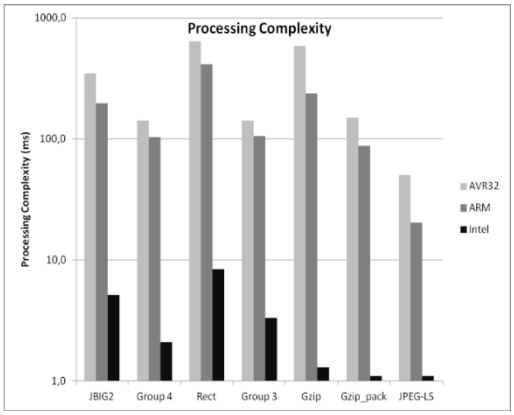

FIGURE 10. The encoding time of computing platforms for the captured images.

Another trend that must be observed in Fig. 10 is that the encoding times of Gzip, Rectangular and JBIG2 compression methods are higher as compared to the other compression methods. The most important result in Fig. 10 is that the trend in the encoding time of the compression methods is almost identical in relation to the different computing architectures.

Table 4 shows the measured energy consumption of the seven compression methods for both the embedded platforms i.e. NGW100 mkII and BeagleBoard-xM.

In Table 4, the processing energy consumption is calculated by multiplying the applied voltage, the measured current and the measured mean processing time on the embedded platforms.

We have used Equation (1) for determining the energy consumption in Table 4. In Equation (1), Imeasured is the measured current, Vapplied is the applied voltage and Tmeasured is the measured execution time of the compression methods. We measured these values while the embedded computing platforms were compressing the images using the compression methods.

$$\text{Energy\_consumed} = I_{measured} * V_{applied} * T_{measured} \quad (1)$$

It can be observed from Table 4 that, because of higher clock frequency, the measured current consumption of the BeagleBoard-xM is higher than that of NGW100 mkII. For the same reason, the measured execution time on the BeagleBoard-xM is lower than that for the NGW100 mkII. This resulted in low energy consumption of the BeagleBoard-xM compared to NGW100.

The energy consumption of the various coding methods is portrayed in Fig. 11. The energy consumption is the combined processing and communication energy. The most significant result is that the total energy consumptions for the JPEG-LS, CCITT Group 4 and Gzip_pack are lower in comparison to the remaining coding techniques (Fig. 11). Total energy consumptions of the Gzip, JBIG2 and Rectangular coding methods are the highest in relation to the other compression methods. Based on these explanations, we can say that CCITT Group 4, JPEG-LS and Gzip_pack is appropriate for energy constrained outdoor applications of WVSNs.

TABLE 4. Processing energy consumption of the coding method.

Measured Parameters for NGW100 mkII				
	Voltage (V)	Current (mA)	Time (ms)	Energy (mJ)
Group 4	9	120	142	154
Group 3	9	130	142	192
JBIG2	9	140	347	406
Rectangular	9	135	642	751
Gzip	9	110	585	579
Gzip_pack	9	110	149	148
JPEG-LS	9	115	50	52
Measured Parameters for BeagleBoard-xM				
	Voltage (V)	Current (mA)	Measured Time (ms)	Measured Energy(mJ)
Group 4	5	300	103	154
Group 3	5	310	105	165
JBIG2	5	335	197	325
Rectangular	5	330	414	683
Gzip	5	270	238	333
Gzip_pack	5	270	88	123
JPEG-LS	5	280	20	26

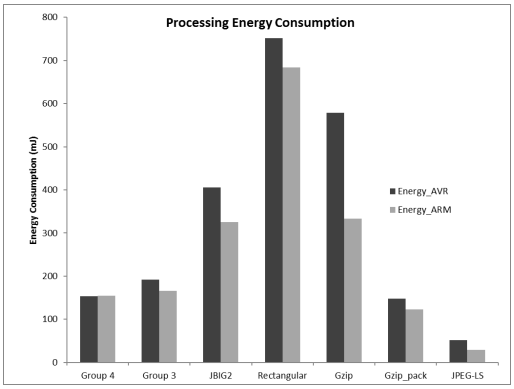

FIGURE 11. The total energy consumption of the compression methods.

It's true that there may be fluctuations in the current during the compression process, but we have determined the average value and still the variation in the current is very small. Hence the average value of the current is a good measure. The voltage will also vary with the passage of time, but as long as the VSN remains functional, the difference in the voltage will be very small and hence our measured energy consumption will provide a good analysis.

VII. CONCLUSION

We have evaluated the impact of various features of objects in the sets of images including different shapes, the growing sizes and increasing number, on the encoding time and measured energy consumption of the seven binary image coding methods. The encoding times of the JPEG-LS and Group 4 are highly resilient for both the increasing number and size of dissimilar shaped objects within the input bi-level images. The Rectangular, Gzip and Gzip_pack compression methods showed higher standard deviations in their encoding times for both an increasing size and number of dissimilar objects in the binary images. The JBIG2 is greatly robust to the variations of objects in the images and its compression efficiency is the highest compared to other compression methods. However, because of its bigger encoding time, its total energy consumption is large. The consequence of increased energy consumption is a short lifetime of the VSN, which is a characteristic that is not desirable in outdoor applications of Wireless Vision Sensor Networks (WVSNs). The hardware implementation of JBIG2 may be faster and this may result in lower total energy consumption but the design and implementation time will be high. The measured energy consumptions on the embedded platform for JPEG-LS, CCITT Group 4 and gzip_pack are lower compared to other coding methods and hence these are the suitable coding methods for the energy constrained outdoor applications of WVSNs.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through the research group NO (RG-1438-034).

REFERENCES

[1] M. Imran, B. Rinner, S. Z. Zand, and M. O'Nils, "Exploration of preprocessing architectures for field-programmable gate array-based thermal-visual smart camera," *J. Electron. Imag.*, vol. 25, no. 4, p. 041006, 2016, doi: 10.1117/1.JEI.25.4.041006.

[2] N. Ahmad, M. Imran, K. Khursheed, N. Lawal, M. O'Nils, and B. Oelmann, "Model, placement optimisation and verification of a sky surveillance visual sensor network," *Int. J. Space-Based Situated Comput.*, vol. 3, no. 3, pp. 125–135, 2013.

[3] L. Ferrigno, S. Marano, V. Paciello, and A. Pietrosanto, "Balancing computational and transmission power consumption in wireless image sensor networks," in *Proc. IEEE Int. Conf. Virtual Environ., Human-Comput. Interfaces, Meas. Syst.*, Messina, Italy, Jul. 2005, p. 6.

[4] M. Imran, K. Khursheed, N. Lawal, M. O'Nils, and N. Ahmad, "Implementation of wireless vision sensor node for characterization of particles in fluids," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 11, pp. 1634–1643, Nov. 2012.

[5] J. McHugh, J. Konrad, V. Saligrama, and P. M. Jodoin, "Foreground-adaptive background subtraction," *IEEE Signal Process. Lett.*, vol. 16, no. 5, pp. 390–393, May 2009.

[6] H. Lee, H. Dong, and H. Aghajan, "Robot-assisted localization techniques for wireless image sensor networks," in *Proc. 3rd Annu. IEEE Commun. Soc. Sensor Ad Hoc Commun. Netw.*, vol. 1. Sep. 2006, pp. 383–392.

[7] K. Khursheed, M. Imran, M. O'Nils, and N. Lawal, "Exploration of local and central processing for a wireless camera based sensor node," in *Proc. IEEE Int. Conf. Signal Elect. Syst.*, Gliwice, Poland, Sep. 2010, pp. 147–150.

[8] K. Khursheed, M. Imran, A. W. Malik, M. O'Nils, N. Lawal, and T. Benny, "Exploration of tasks partitioning between hardware software and locality for a wireless camera based vision sensor node," in *Proc. Int. Symp. Parallel Comput. Elect. Eng. (PARELEC)*, Apr. 2011, pp. 127–132.

[9] M. Imran, K. Khursheed, M. O'Nils, and N. Lawal, "Exploration of target architecture for a wireless camera based sensor node," in *Proc. 28th Norchip Conf.*, Tampere, Finland, Nov. 2010, pp. 1–4.

[10] I. Downes, L. B. Rad, and H. Aghajan, "Development of a mote for wireless image sensor networks," in *Proc. Cognit. Syst. Interact. Sensors (COGIS)*, Paris, France, Mar. 2006, pp. 1–8.

[11] A. Rowe, A. Goode, D. Goel, and I. Nourbakhsh, "CMUcam3: An open programmable embedded vision sensor," School Comput. Sci., Carnegie Mellon Robot. Inst., Pittsburgh, PA, USA, Tech. Rep. RI-TR-07-13, May 2007.

[12] K. Khursheed, N. Ahmad, M. Imran, and M. O'Nils, "The effect of packets relaying on the implementation issues of the visual sensor node," *Elektron. Elektrotechn.*, vol. 19, no. 10, pp. 155–161, 2013.

[13] P. Chen *et al.*, "CITRIC: A low-bandwidth wireless camera network platform," in *Proc. ACM/IEEE Int. Conf. Distrib. Smart Cameras*, Sep. 2008, pp. 1–10.

[14] M. Nasri, A. Helali, H. Sghaier, and H. Maaref, "Adaptive image transfer for wireless sensor networks (WSNs)," in *Proc. Int. Conf. Design Technol. Integr. Syst., Nanoscale Era*, Mar. 2010, pp. 1–7.

[15] S. R. Mallireddy and S. Commuri, "Run time compression of image data in wireless sensor networks," in *Proc. IEEE Int. Conf. Netw., Sens. Control (ICNSC)*, Apr. 2010, pp. 512–517.

[16] K. Aurangzeb, M. Alhussein, and S. I. Haider, "Impact of complexity and compression ratio of compression method on lifetime of vision sensor node," *Elektron. Elektrotechn.*, vol. 23, no. 3, pp. 64–67, 2017.

[17] K. Khursheed, M. Imran, N. Ahmad, and M. O'Nils, "Efficient data reduction techniques for remote applications of a wireless visual sensor network," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 5, p. 240, 2013, doi: 10.5772/55996.

[18] K. Khursheed, M. Imran, N. Ahmad, and M. O'Nils, "Selection of bi-level image compression method for reduction of communication energy in wireless visual sensor networks," *Proc. SPIE*, vol. 8437, p. 84370M, May 2012.

[19] K. Khursheed, M. Imran, N. Ahmad, and M. O'Nils, "Bi-level video codec for machine vision embedded applications," *Elektron. Elektrotechn.*, vol. 19, no. 8, pp. 93–96, 2013. [Online]. Available: http://dx.doi.org/10.5755/j01.eee.19.8.5401

[20] M. I. Razzak, S. A. Hussain, A. A. Minhas, and M. Sher, "Collaborative image compression in wireless sensor networks," *Int. J. Comput. Cognit.*, vol. 8, no. 1, pp. 24–29, Mar. 2010.

[21] R. B. Arps and T. K. Truong, "Comparison of international standards for lossless still image compression," *Proc. IEEE*, vol. 82, no. 6, pp. 889–899, Jun. 1994.

[22] S. R. Kodituwakku and U. S. Amarasinghe, "Comparison of lossless data compression algorithms for text data," *Indian J. Comput. Sci. Eng.*, vol. 1, no. 4, pp. 416–425, 2010.

[23] J. Kivijärvi, T. Ojala, T. Kaukoranta, A. Kuba, L. Nyúl, and O. Nevalainen, "A comparison of lossless compression methods for medical images," *Comput. Med. Imag. Graph.*, vol. 22, no. 4, pp. 323–339, 1998.

[24] R. Starosolski, "Performance evaluation of lossless medical and natural continuous tone image compression algorithms," *Proc. SPIE*, vol. 5959, p. 59590L, Sep. 2005.

[25] A. Veeraputhiran and R. Sankararajan, "Quantization and security enabled compressive video CODEC for WSN," in *Multimedia Tools and Applications*. Amsterdam, The Netherlands: Springer, Sep. 2017. [Online]. Available: https://doi.org/10.1007/s11042-017-5140-9

[26] R. Hamzaa, K. Muhammad, Z. Lv, and F. Titounaa, "Secure video summarization framework for personalized wireless capsule endoscopy," *Pervasive Mobile Comput.*, vol. 41, pp. 436–450, Oct. 2017. [Online]. Available: https://doi.org/10.1016/j.pmcj.2017.03.011.

[27] R. Hamza, K. Muhammad, A. Nachiappan, and G. R. González, "Hash based encryption for keyframes of diagnostic hysteroscopy," *IEEE Access*, to be published, doi: 10.1109/ACCESS.2017.2762405.

[28] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The DISCOVER codec: Architecture, techniques and evaluation," in *Proc. Picture Coding Symp.*, vol. 17. 2007, pp. 1103–1120.

[29] H. Kalva, "The H.264 video coding standard," *IEEE MultiMedia*, vol. 13, no. 4, pp. 86–90, Oct./Dec. 2006.

[30] K. Li-Wei and L. Chun-Shein, "Distributed compressive video sensing," in *Proc. IEEE Int. Conf. Acoust., Speech (ICASSP)*, Taipei, Taiwan, Apr. 2009, pp. 1169–1172.

[31] N. Imran, B.-C. Seet, and A. C. M. Fong, "A comparative analysis of video codecs for multihop wireless video sensor networks," *Multimedia Syst.*, vol. 18, no. 5, pp. 373–389, 2012, doi: 10.1007/s00530-012-0258-0.

[32] B. Girod, A. M. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proc. IEEE*, vol. 93, no. 1, pp. 71–83, Jan. 2005.

[33] N. Imran, B.-C. Seet, and A. C. M. Fong, "Distributed video coding for wireless video sensor networks: A review of the state-of-the-art architectures," *SpringerPlus*, vol. 4, no. 1, p. 513, 2015.

[34] *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*, Standard ITU-T Rec. T.6, Volume VII, Fascicle VII.3, Terminal Equipment and Protocols for Telematic Services, International Telegraph and Telephone Consultative Committee (CCITT), Geneva, Switzerland, 1988. [Online]. Available: http://www.itu.int/rec/T-REC-T.6-198811-I/en

[35] F. Ono, W. Rucklidge, R. Arps, and C. Constantinescu, "JBIG2—The ultimate bi-level image coding standard," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Vancouver, BC, Canada, Sep. 2000, pp. 140–143.

[36] J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM J. Res. Develop.*, vol. 23, no. 2, pp. 149–162, Mar. 1979.

[37] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory*, vol. 23, no. 3, pp. 337–343, May 1977.

[38] M. E. Papadonikolakis and A. P. Kakarountas, "Efficient high-performance implementation of JPEG-LS encoder," *J. Real Time Image Process.*, vol. 3, pp. 303–310, 2008, doi: 10.1007/s11554-008-0088-7.

[39] M. J. Weinberger, G. Seroussi, and G. Sapiro, "LOCO-I: A low complexity, context-based, lossless image compression algorithm," in *Proc. Data Compress. Conf.*, Snowbird, UT, USA, Mar. 1996, pp. 140–149.

[40] S. A. Mohamed and M. M. Fahmy, "Binary image compression using efficient partitioning into rectangular regions," *IEEE Trans. Commun.*, vol. 43, no. 5, pp. 1888–1893, May 1995.

[41] K. Aurangzeb, M. Alhussein, and M. O'Nils, "Data reduction using change coding for remote applications of wireless visual sensor networks," *IEEE Access*, vol. 6, no. 1, pp. 1–11, Apr. 2018, doi: 10.1109/ACCESS.2018.2799958.

KHURSHEED AURANGZEB received the B.S. degree in computer engineering from the COMSATS Institute of Information Technology, Abbottabad, Pakistan, in 2006, the M.S. degree in electrical engineering (system on chip) from Linkoping University, Sweden, in 2009, and the Ph.D. degree from Mid Sweden University, Sundsvall, Sweden, in 2013. From 2013 to 2016, he was an Assistant Professor/Head of the Electrical Engineering Department, Abasyn University, Peshawar, Pakistan. He is currently an Assistant Professor with the College of Computer and Information Science, King Saud University, Riyadh, Saudi Arabia. His research interests include wireless visual sensor networks, design methods and implementation of embedded systems, applied image/signal processing, image compression, mobile cloud computing, Internet of Things, smart grids, smart buildings, and traffic monitoring in smart cities.

MUSAED ALHUSSEIN was born in Riyadh, Saudi Arabia. He received the B.S. degree in computer engineering from King Saud University, Riyadh, in 1988, and the M.S. and Ph.D. degrees in computer science and engineering from the University of South Florida, Tampa, FL, USA, in 1992 and 1997, respectively. Since 1997, he has been on the Faculty of the Computer Engineering Department, College of Computer and Information Science, King Saud University. He is currently the Founder and the Director of Embedded Computing and Signal Processing Research Laboratory. His research activity is focused on typical topics of computer architecture and signal processing and with an emphasis on VLSI testing and verification, embedded and pervasive computing, cyber-physical systems, mobile cloud computing, big data, eHealthcare, and body area networks.

MATTIAS O'NILS received the B.S. degree in electrical engineering from Mid Sweden University, Sundsvall, Sweden, in 1993, and the Licentiate and Ph.D. degrees in electronics system design from the Royal Institute of Technology, Stockholm, Sweden, in 1996 and 1999, respectively. He is currently a Professor and the Dean with the Department of Electrical Engineering and leads a research group in embedded systems design at Mid Sweden University. His current research interests include design methods and implementation of embedded systems, with a specific interest in implementation of real-time video processing systems.