

A study on visual sensor network cross-layer resource allocation using quality-based criteria and metaheuristic optimization algorithms



K. Pandremmenou, L.P. Kondi, K.E. Parsopoulos*

University of Ioannina, Department of Computer Science & Engineering, GR-45110 Ioannina, Greece

ARTICLE INFO

Article history:

Received 11 October 2013

Received in revised form 11 July 2014

Accepted 16 September 2014

Available online 6 October 2014

Keywords:

Hybrid algorithms

Metaheuristic optimization

Particle swarm optimization

Power control

Resource allocation

Visual sensor network

ABSTRACT

Visual sensor networks (VSNs) consist of spatially distributed video cameras that are capable of compressing and transmitting the video sequences they acquire. We consider a direct-sequence code division multiple access (DS-CDMA) VSN, where each node has its individual requirements in compression bit rate and energy consumption, depending on the corresponding application and the characteristics of the monitored scene. We study two optimization criteria for the optimal allocation of the source and channel coding rates, which assume discrete values, as well as for the power levels of all nodes, which are continuous, under transmission bit rate constraints. The first criterion minimizes the average distortion of the video received by all nodes, while the second one minimizes the maximum video distortion among all nodes. The resulting mixed integer optimization problems are tackled with a modern optimization algorithm, namely particle swarm optimization (PSO), as well as a hybrid scheme that combines PSO with the deterministic Active-Set optimization method. Extensive experimentation on interference-limited as well as noisy environments offers significant intuition regarding the effectiveness of the considered optimization schemes, indicating the impact of the video sequence characteristics on the joint determination of the transmission parameters of the VSN.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Wireless sensor networks have constituted a very active research topic in computer science and telecommunications over the past few years. Initially, such networks were mainly concerned with the transmission of unidimensional signals (e.g., temperature, sound etc). Nowadays, their applications have been expanded to the transmission of visual data, such as images or videos. This type of wireless sensor network that conveys visual data is the well-known *visual sensor network* (VSN).

In VSNs, each node is equipped with a camera for imaging different fields of view and detecting events of interest. VSNs support a plethora of applications, ranging from security and teleconference systems to environmental monitoring [1]. Security monitoring includes the surveillance of large areas such as motorways, airports, banks, and other public or private places, mainly for the prevention of unpleasant events or illegal activities. Environmental monitoring is concerned with the monitoring of natural environment in remote locations where human presence is often impossible. These

monitoring systems deter illegal construction, logging, hunting etc. Concerning teleconference systems, they enable users to participate remotely in events of interest, such as attending a meeting or a lecture that is taking place in a different location, or even virtually visit an exhibition or a museum.

In the present work, we consider a *direct-sequence code division multiple access* (DS-CDMA) VSN, where the sensor nodes record scenes with varying motion levels, which is a common approach in real-time VSN applications. All the nodes of the network communicate directly with a central processing server, called the *centralized control unit* (CCU), and they exchange information in order to achieve the ideal tradeoff between the transmitted video quality and energy consumption. This policy is based on a cross-layer design that differs from usual layered network architectures like the *open systems interconnection* (OSI) [2], and the *transmission control program/internet protocol* (TCP/IP) models. In a layered network design, each layer is responsible for executing specific tasks, sequentially transferring information from layer to layer according to their hierarchical ordering. On the contrary, the cross-layer design employed in our study is more flexible since it allows for different network layers to interact with each other in order to exchange information, even they are not adjacent. Hence, it relaxes the austere limits of a layered network infrastructure, ameliorating the overall system performance.

* Corresponding author. Tel.: +30 2651008839; fax: +30 2651008890.

E-mail addresses: apandrem@cs.uoi.gr (K. Pandremmenou), lkon@cs.uoi.gr (L.P. Kondi), kostasp@cs.uoi.gr (K.E. Parsopoulos).

Various cross-layer designs have been proposed in the literature, based on application-specific requirements [3–8]. Our study is based on such a network, where the physical layer, data link layer, network layer, and application layer are allowed to cooperate with each other with the goal of overcoming the delay latency difficulties and improving the *quality of service* (QoS) of real-time applications. Indeed, the majority of VSNs applications require real-time data from sensor nodes [1], i.e., data shall be transmitted from nodes to the end-user within an extremely limited time frame, such that the total transmission delay is imperceptible. Yet, there are some negative factors that affect real-time system performance. Data imaging, processing, and transmission, in conjunction with the constraints that govern wireless channels (e.g., available bandwidth or bit rate, modulation scheme, video coding standard, and wireless access method) can dramatically slow down real-time system response, thereby deteriorating the overall efficiency of the VSN.

Wireless channels are generally unreliable due to frequent errors that occur during wireless data transmissions. Thus, if our primary concern is to maintain a good level of video quality, we should aim at maintaining a low transmission *bit error rate* (BER). To this end, suitable dynamic adjustment of sensor nodes' transmission parameters is required to maximize the VSN's performance. Specifically, each sensor node has a bit rate that can be used for both source coding and channel coding, while it also has an amount of power necessary for sensing, processing, and transmission of the captured data. Hence, the source coding rate, channel coding rate, and power constitute the transmission parameters of each node. Naturally, each node compresses the captured data at a different source coding rate according to the detected amount of motion in each scene. Thus, channel coding rate shall be different for each node. Under a total bit rate constraint, a higher source coding rate results in a lower channel coding rate, and vice versa. Consequently, higher levels of power are required for data transmission due to the lower protection from channel errors.

Data transmission is scourged mainly due to multi-path fading, shadowing at the physical layer, and co-channel interference at the medium access control layer [9]. Hence, channel coding is used to increase the reliability of transmissions. *Automatic repeat request* (ARQ) and *forward error correction* (FEC) are two widely used schemes, suitable for error-correction over wireless channel transmissions [10]. ARQ is based on the retransmission of missing data packets. Its weak points are the need for a feedback channel and the time required for recovering missing packets, which impose significant limitations in real-time applications. Instead, FEC schemes can tolerate some amount of losses, allowing data transmissions under lower power consumption. However, in the special cases where there are too many losses, they can be handled by usual ARQ techniques.

In addition, given that sensor nodes are battery-operated systems, energy control determines the lifetime of their battery. The high complexity of data processing and analysis that accompanies large amounts of video information, along with the real-time application requirements for data transmissions, result in rapid battery drain of the sensor. Energy conservation can be achieved through proper power control in order to maximize the sensors' energy-efficiency [11–13]. Besides that, the multimedia content transmissions that require high bandwidth demands shall be taken into consideration. In [14] a bandwidth management framework is proposed to coordinate multiple video flows in order to overcome wireless channel resource limitations. Further VSN challenges are highlighted in [15].

In the present study, a cross-layer multi-node optimization design accounts for the overall system performance through all network layers. Particularly, it is responsible for the optimal determination of the source coding rates, channel coding rates, and power levels at the application layer, data link layer, and

physical layer, respectively, while the CCU lies at the network layer. The source coding rate determines the compression rate of a video sequence, while the channel coding rate defines the relative protection of the transmitted video sequence. Regarding the transmission power, on one hand, it should be adequately high to permit data transmission and maintain the quality of the video reception. On the other hand, it needs to be adequately low in order to prolong battery lifetime, keep interference at low levels among nodes, and efficiently exploit channel capacity, resulting in high system QoS.

In order to optimally and jointly allocate system resources to all nodes, we considered two quality-based optimization criteria aiming at video distortion minimization. The first one, called the *minimized average distortion* (MAD), minimizes the overall average video distortion of the network, neglecting fairness among the nodes. The second criterion, called the *minimized maximum distortion* (MMD), minimizes the maximum distortion among all nodes of the network, promoting a rather unbiased treatment of the nodes. For both criteria, we require that the total bit rate is identical for all nodes. These criteria have been previously used in similar resource allocation problems [16–18]. However, in these works, they were applied to purely discrete optimization problems, since the source coding rates, channel coding rates, and power levels assumed discrete values solely. Contrary to this, in the present work we considered continuous (and bounded) power levels, while source coding rates retain discrete values, since channel coding rates can take values only within a finite discrete set [19]. Promising preliminary results have offered strong motivation for the further investigation of the specific problems [20].

The solution methodology for our problem was a significant issue for both our optimization criteria. Deterministic mixed integer programming methods can be used to tackle such problems. Indeed, branch and bound approaches combined with outer approximation algorithms have been proposed in the literature [21]. Despite the robustness that usually accompanies deterministic algorithms, various issues may arise. For instance, they may require significant implementation effort and their required memory and running-time can be exponentially increased with the number of integer variables. In addition, sensitivity to initial conditions shall be taken into consideration.

On the other hand, established population-based optimization algorithms can offer satisfactory solutions at the cost of reasonable computational requirements and minor implementation effort. Also, they concurrently evolve a population of candidate solutions that may constitute useful suboptimal alternatives with slightly different characteristics than the optimal one. Applications of population-based metaheuristics such as evolutionary algorithms, ant colony, greedy randomized adaptive search procedure, and particle swarm optimization in wireless sensor networks can be found in [22–25]. The highly appreciable properties of metaheuristic algorithms triggered our interest in using such methods to the resource allocation problems under investigation.

In particular, we employed the *particle swarm optimization* (PSO) algorithm, which is a computational intelligence approach that draws inspiration from social dynamics. PSO has proved to be very efficient in a plethora of complex engineering problems [26]. This fact, along with its stochastic nature, relieves the user from the burden of presenting an appropriate initialization to the algorithm. The promising preliminary experimental results in [20] suggested the use of PSO as the optimizer for solving our resource allocation problems.

For comparison reasons, the performance of PSO was compared to that of the deterministic *active-set* (AS) algorithm [27–29]. In addition, motivated by the promising performance of hybrid algorithms that combine population-based approaches with deterministic schemes (often called *memetic algorithms*) [30], we also considered a hybrid algorithm that combines PSO with AS. The new

algorithm, henceforth denoted as HPSOAS (Hybrid PSO-AS), aims at exploiting the benefits of both PSO and AS, thereby increasing efficiency.

The rest of the paper is organized as follows: in Section 2, we present the transmission parameters of the network under consideration along with procedures for their determination, the wireless channel access method, as well as the universal rate distortion characteristics model. The two optimization criteria that minimize either the average or the maximum distortion of the network are analyzed in Section 3. The employed optimization algorithms are described in Section 4. In Section 5, the system and algorithms' parameters are defined, followed by experimental results meticulously presented and discussed. Finally, Section 6 summarizes the key-concepts and conclusions of our study.

2. Considered wireless VSN

In this section, we discuss the considered source and channel coding techniques for encoding video sequences, as well as the basic characteristics of the considered wireless DS-CDMA VSN. Moreover, we present the estimation procedures for the expected video distortion.

2.1. Video compression

Video compression is essential in communications due to limitations in the bandwidth of the communication channel. Different video sequences have different bit rate requirements for their compression. Generally, these requirements depend on the amount of motion in the video sequence. Clearly, video sequences with less motion can be source encoded at a lower bit rate while still maintaining good perceptual quality. On the other hand, video sequences that contain intense motion activity shall be compressed at a higher bit rate in order to avoid significant degradation of the video quality.

Thus, assuming that the total bit rate is fixed, if a node needs a higher source coding rate, a lower percentage of the total bit rate is assigned to channel coding for error correction. Hence, in order to keep the BER at acceptable levels, the transmission power must be increased. In our study, we assumed that each node has the power required for video transmission over the VSN. Inevitably, the energy consumed for data transmission leads to shortening of the battery life and, due to the nature of DS-CDMA, to increased interference imposed to the other nodes of the network.

The CCU at the network layer plays the role of the coordinator of the overall process of resource allocation. Specifically, it communicates directly with each node, performing source and channel decoding so as to receive the transmitted video sequences. Depending on the amount of motion detected in each video sequence, the central server can request from the nodes to properly adjust their transmission parameters, namely the source coding rate, channel coding rate, and power levels. For instance, if the CCU considers that a node is imaging scenes of great interest, it tries to maximize the picture quality of the specific video by appropriately adjusting its transmission parameters.

We employed the H.264/advanced video codec (AVC) video coding standard to compress the video sequences imaged by the nodes, using the High profile for 4:2:0 color format video. This standard is targeted at many applications such as video telephony, storage, broadcast, and streaming [31]. The coded video data is organized in *network abstraction layer* (NAL) units, which are packets containing an integer number of bytes. These units are grouped into *video coding layer* (VCL) NAL units and non-VCL NAL units. The VCL NAL units contain the data that represents the values of the pixels in the video pictures, while non-VCL NAL units contain parameter sets and supplemental enhancement information [32]. Concerning the

H.264/AVC High profile for the aforementioned color format, it has proved to be extremely efficient in coding, taking into consideration the available coding tools for the encoder [31].

2.2. Channel coding

Channel coding is used to increase the communication channel reliability by increasing resistance to channel errors. Specifically, it adds redundancy in the video bitstream, unlike source coding which intends to represent data with the smallest possible number of bits. In our paper, an adaptive FEC scheme using *rate compatible punctured convolutional* (RCPC) codes is utilized for channel coding [19]. These are families of codes with different rates, which can be decoded by the same Viterbi decoder. However, other channel coding schemes can also be used.

The use of RCPC codes allows the use of Viterbi's upper bounds on the bit error probability. Thus, for the bit error probability P_b , it holds that [19],

$$P_b \leq \frac{1}{P} \sum_{d=d_{\text{free}}}^{\infty} c_d P_d, \quad (1)$$

where P is the period of the code; d_{free} is the free distance of the code; c_d is the information error weight; and P_d is the probability that the wrong path at distance d is selected.

Let us assume that information is sent over a channel subjected to *additive white Gaussian noise* (AWGN). Also, let *binary phase shift keying* (BPSK) be the employed modulation scheme. Then, the probability P_d becomes [19],

$$P_d = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{dR_c E_k}{N_0}} \right), \quad (2)$$

where,

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} \exp(-t^2) dt, \quad (3)$$

is the *complementary error function*; R_c is the channel coding rate; and E_k/N_0 is the energy per bit to *multiple access interference* (MAI) ratio. The index k denotes the corresponding node of the network.

2.3. Direct-sequence code division multiple access

DS-CDMA is the wireless VSN access method that we adopted in the current study. This method allows all nodes to transmit over the same channel, sharing the same bandwidth. Furthermore, since all nodes transmit over the same channel, transmissions are affected by generated interference from the other nodes, mainly due to non-orthogonal spreading codes, possible asynchronous transmissions, and multi-path fading. The target is to limit the interference as much as possible in order to ameliorate the video quality, retain low power consumption, and achieve the effective exploitation of the system's capacity without affecting the integrity of the data transmission procedure.

After source and channel coding, each data signal is assigned a spreading code, usually orthogonal or pseudo-random to the codes assigned to the other signals, such that the interference between two signals is minimized. In order to transmit a single bit, a node actually transmits L chips, where L is the spreading code length. Usually, the *chip rate* (number of transmitted chips per second) is identical for all nodes. We assume that the spreading code length is identical for all nodes. Thus, a constraint on the chip rate corresponds to a constraint on the bit rate. DS-CDMA systems are usually interference-limited systems and therefore, it is common for the

thermal noise and background noise to be neglected. The power level for each node $k = 1, 2, \dots, K$, is given by,

$$S_k = E_k R_k, \quad (4)$$

and it is measured in Watts (W), where E_k is the energy per bit, and R_k is the total bit rate used for both source and channel coding.

In fact, S_k refers to the power received by the CCU from node k . Therefore, for given power levels, the required transmission powers for the nodes can be determined through a propagation model. Assuming the *two-ray ground reflection model* as the propagation model, the transmission power for node k is given by [33],

$$S_{k\text{trans}} = \frac{S_k d_{tr}^4}{G_t G_r h_t^2 h_r^2}, \quad (5)$$

where d_{tr} is the distance between the transmitter (node) and the receiver (CCU); G_t is the transmitter antenna gain; G_r is the receiver antenna gain; h_t is the height of the transmitter; and h_r is the height of the receiver.

Concerning the total bit rate R_k , it is defined as,

$$R_k = \frac{R_{s,k}}{R_{c,k}}, \quad (6)$$

and it is measured in bits per second (bps). The quantity $R_{s,k}$ represents the source coding rate, also measured in bps, while $R_{c,k}$ is the channel coding rate of k -th node. Obviously, since $R_{c,k}$ is the ratio of the number of information bits over the total number of bits, it is a scalar within the range (0, 1) [3].

In our investigation, we followed the assumption that the interference can be approximated by AWGN [34,35]. Thus, the energy per bit to MAI ratio is given by,

$$\frac{E_k}{I_0} = \frac{S_k/R_k}{\sum_{j \neq k}^K (S_j/W_t)}, \quad k = 1, 2, \dots, K, \quad (7)$$

where $I_0/2$ is the two-sided noise power spectral density due to MAI, measured in Watts/Hertz (W/Hz), and W_t is the total available bandwidth, measured in Hertz (Hz). Again, k refers to the corresponding node, while j refers to each interfering node.

In Eq. (7), the following fundamental assumptions were made:

- The thermal and background noise were ignored.
- The spreading codes used are random and do not have any special properties.
- Interference suppression filters are not used.

Assumptions (b) and (c) suggest that no means is used to suppress or limit the co-channel interference, implying that each node admits the power of the other nodes totally as interference. If we drop assumption (a), i.e., assuming that thermal and background noise are rather significant, then, instead of E_k/I_0 , we must use the following energy per bit to multiple access interference and noise ratio,

$$\frac{E_k}{I_0 + N_0} = \frac{S_k/R_k}{\sum_{j \neq k}^K (S_j/W_t) + N_0}, \quad k = 1, 2, \dots, K, \quad (8)$$

where $N_0/2$ is the power spectral density of the AWGN.

2.4. Universal rate-distortion characteristics

Image quality is highly related with the number of occurred errors during data transmission. Therefore, the BER determines the expected video distortion, which has an immediate impact on the video quality. In this paper, in order to compute the expected video distortion, *universal rate-distortion characteristics* (URDC) are used [3]. URDC express the expected distortion as a function of the bit error probability after channel decoding. Specifically, having

compressed the video sequences with the H.264/AVC video codec at specific source coding rates, and for given BERs at each time, we can estimate the expected distortion value [18].

Similarly to relevant studies [18,20,36–38], the model we considered for the expected video distortion per node k is given by,

$$E[D_{s+c,k}] = \alpha \left[\log_{10} \left(\frac{1}{P_b} \right) \right]^{-\beta}, \quad (9)$$

where P_b corresponds to the bit error probability or, in other words, to the BER. The parameters α and β are positive, and they are determined through a mean squared error optimization procedure using a limited number of pairs $(E[D_{s+c,k}], P_b)$, which are experimentally obtained for specific BERs. The values of these parameters depend on the source coding rate and the video sequence characteristics. Specifically, the parameter α usually takes lower values for video sequences with low amount of motion and higher values for video sequences with high amount of motion [39].

Due to the fact that channel errors and packet drops occur randomly, the video distortion attributed to the lossy compression and channel errors is a random variable. To this end, the video distortion is averaged over a number of independent experiments. Another assumption made in our study was that all nodes transmit data using the same total bit rate. In fact, this constraint results from a fixed overall transmission chip rate and the same processing gain (spreading code length) for all nodes, since it holds that,

$$R_k = \frac{R_{\text{chip}}}{L}, \quad (10)$$

where R_{chip} is the chip rate, measured in chips per second, and L is the spreading code length, measured in chips.

We further assumed that the channel coding rates, $R_{c,k}$, $k = 1, 2, \dots, K$, can take only discrete values [19] from a set $\mathbf{R}_c$. Assuming that R_k is fixed, from Eq. (6) it follows that the source coding rates $R_{s,k}$, $k = 1, 2, \dots, K$, must also take discrete values from a set $\mathbf{R}_s$. Namely,

$$R_{c,k} \in \mathbf{R}_c, \quad R_{s,k} \in \mathbf{R}_s, \quad k = 1, 2, \dots, K.$$

Let the index $cb = 1, 2, \dots, CB$, denote the admissible source coding rate-channel coding rate combinations. Then, the combination $(R_{s,k}, R_{c,k})$ assumes discrete values from a set,

$$\mathbf{R}_{s+c} = \{(R_{s,k,1}, R_{c,k,1}), \dots, (R_{s,k,cb}, R_{c,k,cb}), \dots, (R_{s,k,CB}, R_{c,k,CB})\}.$$

The cardinality of $\mathbf{R}_{s+c}$ is CB . Evidently, the cardinalities of the sets $\mathbf{R}_s$, $\mathbf{R}_c$, and $\mathbf{R}_{s+c}$ shall be equal. Increasing the cardinality of these sets, results in significant augmentation of the search space with a consequent impact on the corresponding problem's complexity.

Regarding the power levels of the nodes, unlike previous work [18], in this study we assumed that they can take real values within a predetermined continuous range,

$$S_k \in \mathbf{S} = [S_{\min}, S_{\max}] \subset \mathbb{R}, \quad k = 1, 2, \dots, K.$$

Moreover, the parameters α and β of Eq. (9) are functions of the source coding rate and video content characteristics, as previously discussed. Concerning the parameters d_{free} and c_d of Eq. (1), they are functions of the channel coding rate. Thus, α , β , d_{free} , and c_d are functions of the source coding rate-channel coding rate combinations. Substituting Eq. (2) (with I_0 instead of N_0) into Eq. (1) (considering equality in order to use the BER's upper limit), and Eq. (1) into Eq. (9), the expected video distortion becomes,

$$E[D_{s+c,k}](R_{s,k}, R_{c,k}, S) = \alpha(cb) \left[\log_{10} \left(\frac{1}{P_b} \right) \right]^{-\beta(cb)}, \quad (11)$$

where,

$$P_b = \frac{1}{P} \sum_{d=d_{\text{free}}(cb)}^{\infty} \left(c_d(cb) \frac{1}{2} \operatorname{erfc} \left(\sqrt{dR_{c,k} \left(\frac{S_k/R_k}{\sum_{j \neq k}^K (S_j/W_t) + N_0} \right)} \right) \right),$$

and $k = 1, 2, \dots, K$. Obviously, the expected video distortion for node k is a function of the source coding rate, $R_{s,k}$, channel coding rate, $R_{c,k}$, as well as of the power levels, $S = (S_1, S_2, \dots, S_K)^T$, of all nodes of the network. Therefore, we eventually need to determine the source-channel coding rate combinations, and the power levels of all nodes, in order to compute the expected video distortion.

3. Optimization criteria

We assumed that the sensor nodes participating in the network image scenes that include various motion levels. This is a common feature for the majority of real-time VSN applications. The two optimization criteria that we considered to tackle the problem of optimal resource allocation among the nodes of the wireless VSN are both based on the concept of minimizing video distortion and they are analyzed in the following paragraphs.

3.1. Minimized average distortion

According to the *minimized average distortion* (MAD) criterion, we need to determine the optimal vectors of source coding rates, $R_s = (R_{s,1}, R_{s,2}, \dots, R_{s,K})^T$, channel coding rates, $R_c = (R_{c,1}, R_{c,2}, \dots, R_{c,K})^T$, and power levels, S , such that the overall *average distortion* $D_{\text{ave}}(R_s, R_c, S)$ of the network is minimized, subject to the constraint of equal target bit rate R_{target} for all nodes. This problem can be formally given as follows,

$$\min_{R_s, R_c, S} D_{\text{ave}}(R_s, R_c, S), \quad (12)$$

subject to $R_1 = R_2 = \dots = R_K = R_{\text{target}}$,

where $D_{\text{ave}}(R_s, R_c, S)$ is defined as follows:

$$D_{\text{ave}}(R_s, R_c, S) = \frac{1}{K} \sum_{k=1}^K E[D_{s+c,k}](R_{s,k}, R_{c,k}, S), \quad (13)$$

where k is the node's index and K is the total number of nodes in the VSN. Obviously, this criterion does not assert fairness among the nodes. Hence, distortion is allowed to vary significantly from node to node as far as the average distortion is kept to minimal levels.

3.2. Minimized maximum distortion

The *minimized maximum distortion* (MMD) criterion requires the determination of the optimal vectors of source coding rates, $R_s = (R_{s,1}, R_{s,2}, \dots, R_{s,K})^T$, channel coding rates, $R_c = (R_{c,1}, R_{c,2}, \dots, R_{c,K})^T$, and power levels, S , such that the *maximum distortion* $D_{\text{max}}(R_s, R_c, S)$ among all nodes is minimized subject to the constraint of equal target bit rate R_{target} for all nodes, i.e.,

$$\min_{R_s, R_c, S} D_{\text{max}}(R_s, R_c, S), \quad (14)$$

subject to $R_1 = R_2 = \dots = R_K = R_{\text{target}}$,

where $D_{\text{max}}(R_s, R_c, S)$ is defined as follows,

$$D_{\text{max}}(R_s, R_c, S) = \max_{k \in \{1, 2, \dots, K\}} E[D_{s+c,k}](R_{s,k}, R_{c,k}, S), \quad (15)$$

where k denotes the corresponding node. The MMD criterion may also exhibit deviations of the distortion from node to node, but, in

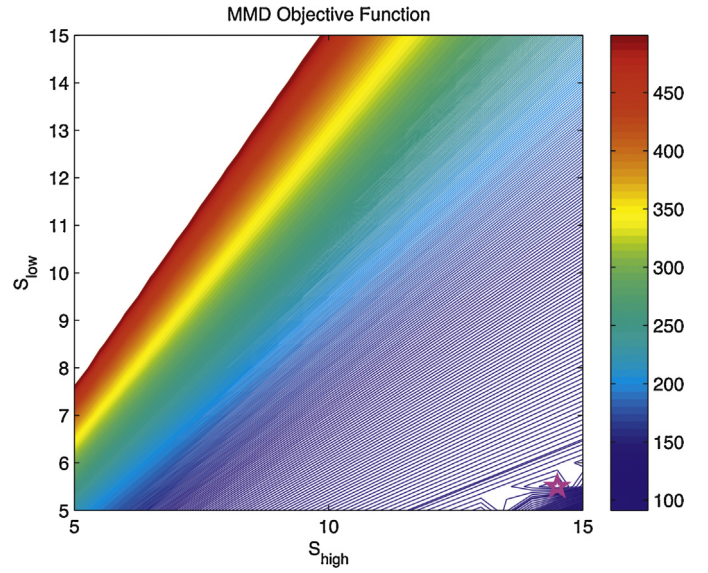


Fig. 1. Contour plot of the objective function for the MMD criterion with respect to the power level values of the two motion classes (fixed source-channel coding rate combination).

contrast to the MAD criterion, it guarantees that all distortions are kept within acceptable ranges.

3.3. Further assumptions

An additional assumption in our study is that the K nodes of the network are clustered into C motion classes, based on the amount of motion in the detected scenes. Without loss of generality, let us consider the case of $C = 2$. In this case, there are two motion classes: a high-motion class, which includes the nodes that detect high levels of motion, and a low-motion class consisting of the nodes that image relatively stationary fields. Each class has its own set of parameters α and β (see Eq. (11)), since they are affected by the amount of motion of each considered video sequence.

A reasonable question that follows the aforementioned assumption, is what happens in case of a possible change in the motion level of a scene. For example, what happens if the relatively stationary scenes of a forest-monitoring application are disturbed by an unexpected passage of an animal or, in a motorway-surveillance application, the scenes that capture intense traffic succeed scenes with infrequent vehicle passing? In such cases, a new classification of the scenes into high- and low-motion classes is required, corresponding to a new optimal resource allocation that is adjusted to the current state of the observed system.

Regarding the node clustering into two classes, the quantities that need to be determined for each class under a total transmission bit rate constraint, are the following,

$$R_{s+c, \text{high}} = (R_{s, \text{high}}, R_{c, \text{high}})^T, R_{s+c, \text{low}} = (R_{s, \text{low}}, R_{c, \text{low}})^T, \\ S = (S_{\text{high}}, S_{\text{low}})^T,$$

where $(R_{s, \text{high}}, R_{c, \text{high}})$ and S_{high} , are the source-channel coding rate combination, and the power level, respectively, for the high-motion class of nodes, while $(R_{s, \text{low}}, R_{c, \text{low}})$ and S_{low} are the corresponding quantities for the low-motion class of nodes.

Since the source-channel coding rate combinations assume discrete values, while the power levels are continuous, the resulting optimization problems for both the above criteria are of mixed integer type. Fig. 1 illustrates the contour plot of the corresponding landscape for a fixed source-channel coding rate combination

(the one that corresponds to the best solution). The two axes stand for the real-valued power levels of the two motion classes. Darker lines denote lower objective values. As we can see, the function has extremely steep regions (upper left part of the figure) as well as almost flat regions (middle to lower part). The star mark denotes the globally optimal power level vector (for the specific source-channel coding rate combination), which lies in a small region near the right lower bound of the search space. The figure refers to the MMD criterion. Similar landscapes are produced for all bit rate-bandwidth combinations, and for both the MAD and MMD criteria. Note that the optimization algorithm optimizes the source-channel coding rate combinations as well as the power levels, concurrently. It is easily conceived that the interplay of the discrete variables along with the remarkable changes in slope for the real-valued variables, as well as the discontinuities that may be produced by criteria as the MMD can impose serious difficulties for any optimization algorithm.

4. Employed optimization algorithms

In the following paragraphs we present the employed optimization algorithms, namely PSO, AS, and the hybrid HPSOAS.

4.1. Particle swarm optimization

Particle swarm optimization (PSO) was introduced in 1995 by Eberhart and Kennedy [40]. It is based on models that simulate flocking behavior and it has close ties with the concurrent concepts of *emergent* and *collective behavior* [41]. PSO is categorized as *swarm intelligence* algorithm within the wider field of intelligent optimization [26,42]. Its ongoing increasing popularity can be attributed to its efficiency in tackling a plethora of scientific and technological applications, as well as to its easy implementation, which renders it accessible to researchers from various disciplines [26].

PSO uses a population, called *swarm*, of search points, called *particles*, to probe the search space. The particles are randomly initialized (usually uniformly) in the search space. Each particle has three essential features: its *current position* in the search space, a *memory* where it retains the best position it has ever visited, and an adaptable *velocity* (position shift) that iteratively defines its new position. Also, it assumes a *neighborhood* consisting of other particles, i.e., a subset of the swarm, with which it interacts by means of information exchange. The information originating from the particle's own experience as well as the collective experience, are the main sources of influence for its move in the search space.

Let the general minimization problem,

$$\min_{x \in \mathbf{X} \subset \mathbb{R}^n} f(x),$$

with $f(x)$ being the objective function. Let the set $I = \{1, 2, \dots, N\}$ denote the indices of the N particles of the swarm. Then, the swarm can be represented as a set of search points,

$$S = \{x_1, x_2, \dots, x_N\}.$$

Each particle is an n -dimensional vector,

$$x_i = (x_{i1}, x_{i2}, \dots, x_{in})^T \in \mathbf{X}, \quad i \in I,$$

and its velocity is defined as,

$$v_i = (v_{i1}, v_{i2}, \dots, v_{in})^T, \quad i \in I.$$

Its best position is also an n -dimensional vector,

$$p_i = (p_{i1}, p_{i2}, \dots, p_{in})^T \in \mathbf{X}, \quad i \in I,$$

stored in the memory and iteratively updated as long as the particle moves in $\mathbf{X}$.

The neighborhood, $\mathcal{N}_i$, of the i -th particle can be defined in various ways. A straightforward approach considers as neighbors the closest particles in the search space. However, this approach was shown to produce clusters of particles that rapidly collapse on local minimizers, thereby reducing the (collective) exploration ability of the swarm. An alternative idea is the determination of neighborhoods in abstract spaces instead of the actual search space. An instance that has proved to be very efficient assumes that the particles are ordered on a *ring* based on their indices. In this case, the neighborhoods consist of particles with neighboring indices, having the form,

$$\mathcal{N}_i = \{i - r, \dots, i - 1, i, i + 1, \dots, i + r\} \subseteq I,$$

where $r \in \{1, 2, \dots, N/2\}$ is called the neighborhood's *radius*. The indices are assumed to recycle at the ends, i.e., index 1 follows immediately after index N . Evidently, increasing r results in neighborhoods that approximate the whole swarm. Different neighborhood topologies have been proposed in the literature [43,44]. The neighborhoods control the information flow among the particles as well as the available information that influences the particles' position shifts at each iteration. Therefore, they can have a tremendous impact on PSO's performance.

Let g_i denote the index of the best in the neighborhood of the i -th particle, i.e.,

$$g_i = \arg \min_{j \in \mathcal{N}_i} f(p_j), \quad i \in I, \quad (16)$$

and let t denote the iteration number. Then, the swarm is updated at each iteration as follows [45]:

$$v_{ij}^{(t+1)} = \chi \left[v_{ij}^{(t)} + c_1 \mathcal{R}_1 \left(p_{ij}^{(t)} - x_{ij}^{(t)} \right) + c_2 \mathcal{R}_2 \left(p_{g_{ij}}^{(t)} - x_{ij}^{(t)} \right) \right], \quad (17)$$

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + v_{ij}^{(t+1)}, \quad (18)$$

where $i \in I; j = 1, 2, \dots, n$; and χ is a parameter called the *constriction coefficient*, which can deter the *swarm explosion* effect, i.e., the rapid divergence of the particles due to excessively large velocities [45–47]. Regarding c_1 and c_2 , they are two positive acceleration parameters called the *cognitive* and *social* parameter, respectively. These parameters control the influence of the personal and collective experience (memory) on the particle's move, with equal values promoting a fair tradeoff between them. Finally, $\mathcal{R}_1$ and $\mathcal{R}_2$ are random variables uniformly distributed in the range $[0, 1]$. They introduce stochasticity in PSO and assume a different value for each i and j . Evidently, PSO's update is inherently parallel, since it is performed componentwise.

After updating and evaluating the swarm, memory update takes place in two stages. In the first stage, the personal best position of each particle is updated as follows,

$$p_i^{(t+1)} = \begin{cases} x_i^{(t+1)}, & \text{if } f(x_i^{(t+1)}) < f(p_i^{(t)}), \\ p_i^{(t)}, & \text{otherwise,} \end{cases} \quad i \in I.$$

The determination of new best positions is followed, in the second stage, by the update of all indices $g_i, i \in I$, according to Eq. (16). This completes a PSO iteration. The procedure is repeated until a stopping criterion is satisfied, such as exceeding a prespecified number of function evaluations or reaching a target function value.

Clerc and Kennedy [45] have extensively studied the stability of PSO. Their analysis offered significant mathematical evidence on its proper parameter settings. Based on their analysis, the parameter values,

$$\chi = 0.729, \quad c_1 = 2.05, \quad c_2 = 2.05,$$

have been shown to be a satisfactory starting choice, considered as the default parameter set of the constriction coefficient variant

of PSO. Further information and alternative settings can be found in [45,47].

PSO belongs among the most studied metaheuristics. In addition to [45,47], further theoretical analyses can be found in [48–50]. Its theoretical background, well-understood dynamic, as well as its frequently verified efficiency renders PSO a very appealing optimizer. Recently, it has been used with remarkable success in VNS [20,39,51–57].

4.1.1. Tackling discrete variables

Although PSO was primarily designed to handle continuous variables, it has been successfully applied also on integer optimization problems [58–62]. This can be achieved by introducing integer-arithmetic based operators in PSO. However, in most cases, the resulting PSO variants barely resemble the original PSO dynamics. Alternatively, discrete values can be tackled by solving an extended version of the problem in the continuous space and rounding the candidate solutions to the nearest integers prior to their evaluation with the objective function. The latter procedure has minor effect on the algorithm. Also, it has been shown to work efficiently in various problems, offering motivation for selecting the latter approach in our study.

In the mixed integer optimization problems of the present study, each particle should normally consist of integer and continuous components, corresponding to the discrete and continuous variables described in previous sections. Instead, we considered also the integer parameters to be continuous (retaining their bounds) and applied the presented PSO scheme. However, whenever a particle was evaluated with the objective function, its corresponding components were rounded to the nearest integers as follows:

$$x_{ij} = \lfloor x_{ij} + 0.5 \rfloor,$$

where $\lfloor \cdot \rfloor$ is the floor function. This scheme was successfully tested in previous works [20,39,51–57].

4.1.2. Estimation of maximum velocity

A feature usually neglected in PSO implementations is that of *maximum velocity*. Specifically, whenever the velocities are computed by Eq. (17), they undergo a magnitude-restriction test as follows,

$$v_{ij}^{(t+1)} = \begin{cases} v_j^{\max}, & \text{if } v_{ij}^{(t+1)} > v_j^{\max}, \\ -v_j^{\max}, & \text{if } v_{ij}^{(t+1)} < -v_j^{\max}, \\ v_{ij}^{(t+1)}, & \text{otherwise,} \end{cases} \quad \forall i, j, t,$$

where $v_j^{\max}$ is a predefined positive value, possibly different for each $j = 1, 2, \dots, n$. Obviously, this procedure restricts the velocity components within the corresponding ranges $[-v_j^{\max}, v_j^{\max}]$, preventing the particles from taking large steps that could lead to wide-range oscillations around the best positions or frequently escaping out of the search space. Naturally, this can have considerable impact on PSO's convergence speed. We can easily infer that large values of $v_j^{\max}$ are more appropriate for search spaces with wide flat or low-curvature regions, while significantly smaller values may be required in steep functions with large number of minimizers, especially when they are closely concentrated.

Typically, maximum velocity is determined as the maximum absolute distance allowed to be traveled by the particle in a single step at each component direction. For this purpose, it is usually defined as a fraction of the corresponding search space's range in the specific component direction. For example, if the search space

is defined as $\mathbf{X} = [x_1^{\min}, x_1^{\max}] \times \dots \times [x_n^{\min}, x_n^{\max}]$, then the following restriction is commonly used,

$$v_j^{\max} = \gamma_j (x_j^{\max} - x_j^{\min}), \quad \gamma_j \in (0, 1], \quad j = 1, 2, \dots, n. \quad (19)$$

Available information on the form of the objective function may dictate larger or smaller values of the parameters γ_j . For example, the Lipschitz property can provide useful insight regarding the degree of variation of the objective function in the whole search space. However, in most cases such information is either unavailable or very laborious to be computed.

In such cases, we can approximately estimate the Lipschitz constant by considering its *modulus of continuity* (MoC) $\delta > 0$, which is locally defined in a subset $\mathbf{B} \subset \mathbf{X}$ of the search space as follows,

$$|f(x) - f(y)| \leq \delta \|x - y\|, \quad \forall x, y \in \mathbf{B}.$$

Estimating the MoC around solutions obtained in preliminary experiments as well as on randomly selected points in the search space can partially reveal the local behavior of the objective function. This, in turn, can lead to more appropriate selection of the maximum velocity thresholds described above. The estimation can be easily conducted through Monte Carlo sampling within the corresponding region $\mathbf{B}$.

In our preliminary experiments, we observed that PSO performance in terms of convergence speed exhibited large deviations per optimization criterion for some cases. Thorough examination of the corresponding landscapes revealed the importance of proper velocity setting. Thus, we employed the procedure described above to obtain estimations of the maximum velocities for each optimization criterion based on Monte Carlo approximations of the MoC.

4.2. Active-set optimization method

Constrained optimization problems are usually tackled by splitting the initial problem into simpler subproblems than can be solved and used as the basis of an iterative process. The *Active-Set* (AS) is an iterative method that is used for solving a sequence of quadratic subproblems, guaranteeing the feasibility of the final solution [28,29]. The main mechanism is based on the solution of the Karush-Kuhn-Tucker (KKT) equations, which guarantee the optimality for a constrained optimization problem.

Let us assume again the minimization problem as declared in Section 4.1,

$$\min_{x \in \mathbf{X} \subset \mathbb{R}^n} f(x),$$

subject to m constraints (these constraints may be implicitly given as defining relations of the search space $\mathbf{X}$),

$$G_i(x) = 0, \quad i = 1, \dots, m_e, \quad (20)$$

$$G_i(x) \leq 0, \quad i = m_e + 1, \dots, m. \quad (21)$$

The vector function $G(x) = (G_1(x), \dots, G_m(x))^T$ returns a vector of length m that includes the equality and inequality constraint values at x . The corresponding KKT equations are given by,

$$\nabla f(x) + \sum_{i=1}^m \lambda_i \nabla G_i(x) = 0 \quad (22)$$

$$\lambda_i G_i(x) = 0, \quad i = 1, \dots, m_e, \quad (23)$$

$$\lambda_i \geq 0, \quad i = m_e + 1, \dots, m. \quad (24)$$

Eq. (22) depicts the canceling process of the gradients between the objective function $f(x)$ and the active constraints $G_i(x)$ at x , through the use of the Lagrange multipliers $\lambda_i, i = 1, \dots, m$. Lagrange multipliers are used in order to balance the deviations in magnitude of the objective function and constraint gradients. Due to the fact

that only active constraints are included in the gradients canceling, non-active constraints are assigned $\lambda_i = 0$, as it is stated implicitly by Eqs. (23) and (24).

Thus, the AS method is based on the solution of the KKT equations and attempts to compute the Lagrange multipliers directly. It searches solutions in the feasible sets and if a minimizer is found during each iteration, followed by a decrease in the value of the objective function, the algorithm terminates after a user-defined stopping criterion. Such a criterion can be the maximum iteration number, the maximum number of function evaluations, the function tolerance, the tolerance of the optimal point x etc.

In our problem, we used the robust implementation of the original Matlab® Optimization Toolbox. Further details on this implementation can be found in [63].

4.3. Hybrid PSO-AS optimization method

Motivated by the benefits of both PSO and AS optimization algorithms, we combined their features introducing a hybrid PSO-AS approach, which is denoted as HPSOAS. This hybrid approach can be categorized as a *memetic algorithm* [30] and employs AS as local optimizer for further improving the findings of PSO.

Algorithm 1. HPSOAS

```

Require: Initialize PSO algorithm.
1: loop
2:   if (not stopping) then
3:     Update swarm and best positions.
4:     if (new overall best position is found) then
5:       Apply AS on the new best position.
6:       Make AS's solution the new overall best position.
7:     end if
8:   end if
9: end loop

```

Specifically, when the overall best position of PSO changes, a local search procedure with AS is initiated from this point, in order to further improve it. The procedure is sketched in Algorithm 1. Although PSO and in many cases also AS were capable of successfully approximating the optimal solution in our experiments, the HPSOAS scheme was significantly more time-efficient, yet retaining the solutions' quality. Its success lies on the fact that AS was rapidly improving the PSO's best findings, thereby providing better attractors (best positions) for the particles, while at the same time it surmounted the sensitivity of AS on the initial conditions (starting point).

5. Experimental configuration and results

In the first part of this section, we describe the system's parameter configuration as well as the procedure for computing the expected video distortion, which is needed in Eq. (11) in order to estimate the parameters α and β . Also, the optimization algorithms' parameters are given. In the second part of the section, we assess the two optimization criteria described in Section 3, on two real video sequences. In our experiments, we assessed the source-channel coding rate combinations, and power levels, for the two considered motion classes, as well as the received video quality for each video sequence, under the same experimental settings, i.e., for the same node distributions, bit rates and bandwidths. The experiments considered both the cases of neglecting and adding background and thermal noise.

5.1. System parameters

Based on the observation that high values of the parameter α (see Eq. (11)) imply high video sequence motion and vice versa, we assumed that the K nodes of the network are clustered into $C=2$ motion classes, namely a *high-motion* and a *low-motion* class. The "Foreman" video sequence was used to represent the class of nodes that detect high motion levels, while the "Akiyo" video sequence was used to represent the class of nodes that capture more stationary fields. The resolution of both video sequences was the *quarter common intermediate format* (QCIF). Thus, one set of URDC curves was needed per video sequence.

The procedure for the computation of the expected video distortion can be concisely described as follows: for given BER, we determine the rate of the packet loss according to the *real-time transport protocol* (RTP). Then, packets are dropped from the video bit stream under investigation. We continue decoding the corrupted video sequence with the H.264/AVC video codec and, finally, the expected video distortion is obtained. Due to the existence of random channel errors in VSNs, the same procedure is repeated for 300 times and the expected video distortion is averaged over all these experiments to offer a more reliable estimation.

After the computation of the expected video distortion, the parameters α and β of Eq. (11) are determined using least squares optimization from data obtained using a few BERs. Specifically, we considered the BER values 10^{-7} , 10^{-6} , and 10^{-5} , while each of the "Foreman" and "Akiyo" video sequences is compressed at 32 kbps, 48 kbps, 64 kbps, 72 kbps, and 96 kbps. Furthermore, the characteristics for both video sequences were obtained at a rate of 15 frames per second.

The employed modulation scheme was the BPSK, while RCPC codes with mother code of rate 1/4 were used for channel coding [19]. Also, the considered target bit rate constraints were equal to 96 kbps and 144 kbps. Thus, the corresponding source-channel coding rate combinations for each motion class, resulting from the above bit rate constraints were,

(a) $R_{\text{target}} = 96$ kbps that results in,

$$(R_{s,\text{high}}, R_{c,\text{high}}), (R_{s,\text{low}}, R_{c,\text{low}}) \in \{(32, 1/3), (48, 1/2), (64, 2/3)\}.$$

(b) $R_{\text{target}} = 144$ kbps that results in,

$$(R_{s,\text{high}}, R_{c,\text{high}}), (R_{s,\text{low}}, R_{c,\text{low}}) \in \{(48, 1/3), (72, 1/2), (96, 2/3)\}.$$

For the power levels S , we assumed continuous values within the range $S = [5.0, 15.0]$ (Watts), while for the bandwidth W_t two different values were examined, namely 20 MHz and 15 MHz. The total number of nodes that composed our network was $K = 100$.

5.2. Algorithms parameters

PSO was considered with its default parameter set defined in Section 4.1. Also, a swarm of $N = 40$ particles was employed, under the ring topology of radius $r = 1$. In our problem, each particle x_i was 4-dimensional, defined as,

$$x_i = (S_{\text{high}}, S_{\text{low}}, R_{s+c,\text{high}}, R_{s+c,\text{low}})^T, \quad i = 1, 2, \dots, 40,$$

containing the power level and the source-channel coding rate combination for each motion class (denoted as "high" and "low", respectively). Furthermore, the discrete parameters, i.e., source and channel coding rate combinations, were represented in the particle with continuous values within the range $R = [0.6, 3.4]$. However, as explained in Section 4.1.1, they were rounded to the nearest integer

for the particle's evaluation. Specifically, we assumed the following correspondences,

$$R_{\text{target}} = 96 \text{ kbps} \begin{cases} 1 \rightarrow (32, 1/3) \\ 2 \rightarrow (48, 1/2) \\ 3 \rightarrow (64, 2/3) \end{cases},$$

and

$$R_{\text{target}} = 144 \text{ kbps} \begin{cases} 1 \rightarrow (48, 1/3) \\ 2 \rightarrow (72, 1/2) \\ 3 \rightarrow (96, 2/3) \end{cases}.$$

Besides that, the maximum velocities were set based on the MoC estimation procedure described in Section 4.1.2. The corresponding values of γ in Eq. (19) for the 4 component directions in our problem were determined as follows,

$$\gamma^{\text{MAD}} = (0.1, 0.1, 0.03, 0.03)^T, \quad \gamma^{\text{MMD}} = (1, 1, 1, 1)^T.$$

For each problem instance, PSO was executed 30 times for a maximum number of 1000 iterations and the best solution was recorded.

Regarding the AS method, it was also applied on each problem instance 30 times from different initial conditions. However, AS was capable of providing only the power levels, since it works only on continuous search spaces. For the discrete source-channel coding rate combinations, we used exhaustive search among all possible pair values, i.e., 3 admissible values for each of the two motion classes, resulting in $3 \times 3 = 9$ cases. Thus, a single application of AS required 9 optimization runs, one for each discrete combination, and the final solution was selected as the best one among the obtained 9 solutions.

Finally, identical experiments were conducted using the proposed hybrid algorithm HPSOAS. The parameter setting of HPSOAS was the same with that of PSO. HPSOAS exhibited similar performance with PSO in terms of solution quality for each problem instance. Note that all algorithms (PSO, AS, HPSOAS) were equipped with exactly the same total computational budget in terms of function evaluations, namely 40000 function evaluations, in order to achieve fair comparisons among them.

5.3. Presentation and analysis of results

All experiments were conducted on an Intel® Core™ 2 Quad CPU @ 2.50 GHz with 4.00 GB RAM, using the Matlab environment. For each problem instance, PSO and HPSOAS converged on the same solutions in all 30 experiments. Specifically, the obtained values of the objective functions (for the two optimization criteria) were identical up to 15–20 decimal digits, and therefore, they were considered to be essentially identical. However, this was not the case for AS, which produced inferior solutions in some cases due to its dependency on the initial conditions and the peculiarities of the objective function landscape. Nevertheless, in the rest problem instances it achieved the same solutions as the other two algorithms.

Our first group of experiments was conducted under the assumption that thermal and background noise were neglected, namely $N_0 = 0 \text{ W/Hz}$. The experiments were repeated in a second round, assuming that the aforementioned noise was significant. In this case, noise of magnitude $N_0 = 10^{-7} \text{ W/Hz}$ was added in our computations.

In the case where thermal and background noise were neglected and the AWGN was introduced entirely from the interference among the nodes, PSO detected a multitude of solutions for the power levels of both motion classes, all of which had the same ratio $S_{\text{high}}/S_{\text{low}}$, up to 4–5 decimal digits. All these solutions achieved the best objective values for both the MAD and MMD optimization

criteria. This is a consequence of the fact that the ratio E_k/I_0 in Eq. (7) is invariant under the multiplication of all power levels with the same constant. Therefore, estimation of the optimal ratio $S_{\text{high}}/S_{\text{low}}$ becomes the main goal rather than the determination of specific values for the power levels. In contrast to the power levels, the corresponding optimal source and channel coding rate combinations were unique in all solutions provided by PSO. Moreover, when thermal and background noise were added, PSO reported unique solutions for all transmission parameters.

It is worth mentioning that the video quality for each motion class $cl \in \{1, 2\}$, was estimated by using the *peak signal-to-noise ratio* (PSNR), which is measured in decibel (dB),

$$\text{PSNR}_{cl} = 10 \log_{10} \left(\frac{255^2}{E[D_{s+c,cl}]} \right), \quad (25)$$

with $E[D_{s+c,cl}]$ representing the expected video distortion due to source and channel coding for the cl -th class of nodes.

Tables 1–4 report the obtained direction components of the solution vectors (transmission parameters) that correspond to the best objective values of both MAD and MMD, by all algorithms (the inferior solutions produced by AS in some cases are omitted), for the case where thermal and background noise are considered to be negligible, namely $N_0 = 0 \text{ W/Hz}$. More specifically, the tables report the obtained source coding rates, channel coding rates, power levels and PSNR values, for both motion classes, optimization criteria and different node distributions and settings of the bit rate and bandwidth values. Since in this case we are interested in the optimal power level ratios, rather than the determination of the specific values of S_{high} and S_{low} , we cite indicative S_{high} and S_{low} values that correspond to the best objective value.

Tables 5–8 report the corresponding results under the assumption that thermal and background noise power spectral density are equal to $N_0 = 10^{-7} \text{ W/Hz}$. The corresponding tables per combination of bit rate and bandwidth are summarized below,

- (a) Bit rate 96 kbps and bandwidth 20 MHz: Tables 1 and 5.
- (b) Bit rate 96 kbps and bandwidth 15 MHz: Tables 2 and 6.
- (c) Bit rate 144 kbps and bandwidth 20 MHz: Tables 3 and 7.
- (d) Bit rate 144 kbps and bandwidth 15 MHz: Tables 4 and 8.

The $K = 100$ nodes of the network were assigned to the two motion classes in different proportions, called *node distributions*. Each line in the tables corresponds to a different node distribution, denoted as “ $N_{\text{high}} - N_{\text{low}}$ ”, where

$$N_{\text{high}}, N_{\text{low}} \in \{10, 30, 50, 70, 90\}, \quad N_{\text{high}} + N_{\text{low}} = K = 100,$$

meaning that the corresponding classes consist of a number of N_{high} nodes capturing high-motion scenes and N_{low} nodes capturing low-motion scenes, respectively. Moreover, the combination of source-channel coding rates, the power level and the PSNR of the high-motion class are represented as $(R_{s,\text{high}}, R_{c,\text{high}}), S_{\text{high}}, \text{PSNR}_{\text{high}}$, respectively, while $(R_{s,\text{low}}, R_{c,\text{low}}), S_{\text{low}}, \text{PSNR}_{\text{low}}$, are the corresponding parameters for the low-motion class.

A close inspection of the results, demonstrates that the MAD criterion works favorably for the low-motion class of nodes, equipping it with better image quality than the high-motion class. Concerning the MMD criterion, it is rather unbiased, offering identical PSNR values to both motion classes. These remarks are derived from all combinations of bit rate-bandwidth considered in our experiments.

Moreover, the MMD criterion assigns higher PSNR values to high-motion class of nodes than the MAD criterion, in the corresponding cases. Indeed, the PSNR differences between the two criteria is inversely proportional to the cardinality of the high-motion class of nodes. Thus, the MMD criterion can be considered as the most appropriate choice in cases where we are interested in

Table 1 $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 20 \text{ MHz}$, $N_0 = 0 \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(64, 2/3)	10.7080	(64, 2/3)	5.9845	32.9787	36.7642
30–70	(64, 2/3)	8.6240	(64, 2/3)	5.0000	31.3844	35.1131
50–50	(64, 2/3)	15.0000	(32, 1/3)	7.0428	30.9419	32.8537
70–30	(48, 1/2)	15.0000	(32, 1/3)	7.6070	29.2296	32.2582
90–10	(48, 1/2)	9.8451	(32, 1/3)	5.0000	28.2705	31.2943

MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(64, 2/3)	13.3488	(64, 2/3)	5.0000	35.7218	35.7218
30–70	(64, 2/3)	12.6814	(32, 1/3)	5.0000	33.4049	33.4049
50–50	(64, 2/3)	13.0847	(32, 1/3)	5.0000	31.6114	31.6114
70–30	(64, 2/3)	13.4344	(32, 1/3)	5.0000	29.7737	29.7737
90–10	(48, 1/2)	15.0000	(32, 1/3)	5.7234	28.3919	28.3919

Table 2 $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 15 \text{ MHz}$, $N_0 = 0 \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(64, 2/3)	10.5000	(32, 1/3)	5.0000	31.4521	33.3203
30–70	(48, 1/2)	9.8486	(32, 1/3)	5.0000	29.0488	32.0672
50–50	(48, 1/2)	9.8262	(32, 1/3)	5.0000	27.7763	30.7930
70–30	(32, 1/3)	11.2833	(32, 1/3)	7.4632	26.7596	31.6151
90–10	(32, 1/3)	15.0000	(32, 1/3)	10.1943	26.4203	31.1774

MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(64, 2/3)	12.7296	(32, 1/3)	5.0000	33.0047	33.0047
30–70	(64, 2/3)	13.2312	(32, 1/3)	5.0000	30.7191	30.7191
50–50	(48, 1/2)	13.1569	(32, 1/3)	5.0000	28.5991	28.5991
70–30	(32, 1/3)	12.3831	(32, 1/3)	5.0213	27.1521	27.1521
90–10	(32, 1/3)	15.0000	(32, 1/3)	6.7090	26.5321	26.5321

the amelioration of the high-motion scenes rather than improving the quality of the low-motion scenes. Surveillance applications are typical examples of such cases. On the contrary, the MAD criterion appears to be more suitable in cases where high video quality of low-motion scenes is desirable.

Considering the noisy case, in general, no significant changes in PSNR values are observed after the addition of noise, except for a marginal reduction of no more than 0.01 dB, which has

imperceptible impact to the video quality. Similarly to the noiseless case, the high-motion class of nodes requires more power than the low-motion class. More specifically, the power levels of the high-motion class of nodes are not only higher than that of the low-motion class, but also they actually take their maximum possible value.

Special attention shall be paid to the case of distributing 10 nodes in the high-motion class and 90 nodes in the low-motion

Table 3 $R_{\text{target}} = 144 \text{ kbps}$, $W_t = 20 \text{ MHz}$, $N_0 = 0 \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(72, 1/2)	15.0000	(48, 1/3)	7.7859	30.3384	32.7780
30–70	(48, 1/3)	15.0000	(48, 1/3)	9.3816	28.0307	31.8708
50–50	(48, 1/3)	7.7427	(48, 1/3)	5.0000	27.1339	30.9034
70–30	(48, 1/3)	15.0000	(48, 1/3)	9.9501	26.3745	30.0843
90–10	(48, 1/3)	15.0000	(48, 1/3)	10.1822	25.7146	29.3731

MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$PSNR_{\text{high}}$	$PSNR_{\text{low}}$
10–90	(72, 1/2)	15.0000	(48, 1/3)	6.3505	32.3213	32.3213
30–70	(48, 1/3)	11.6678	(48, 1/3)	5.0000	29.6328	29.6328
50–50	(48, 1/3)	10.5707	(48, 1/3)	5.0000	28.0449	28.0449
70–30	(48, 1/3)	9.8198	(48, 1/3)	5.0000	26.8317	26.8317
90–10	(48, 1/3)	9.2654	(48, 1/3)	5.0000	25.8461	25.8461

Table 4 $R_{\text{target}} = 144 \text{ kbps}$, $W_t = 15 \text{ MHz}$, $N_0 = 0 \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(48, 1/3)	15.0000	(48, 1/3)	10.0856	26.0091	29.6842
30–70	(48, 1/3)	15.0000	(48, 1/3)	10.4477	24.9842	28.5810
50–50	(48, 1/3)	15.0000	(48, 1/3)	10.7484	24.1375	27.6704
70–30	(48, 1/3)	15.0000	(48, 1/3)	11.0045	23.4146	26.8932
90–10	(48, 1/3)	15.0000	(48, 1/3)	11.2265	22.7829	26.2145
MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(48, 1/3)	12.3251	(48, 1/3)	5.6041	28.7266	28.7266
30–70	(48, 1/3)	9.6409	(48, 1/3)	5.0000	26.5538	26.5538
50–50	(48, 1/3)	15.0000	(48, 1/3)	8.4876	25.0406	25.0406
70–30	(48, 1/3)	15.0000	(48, 1/3)	9.0465	23.8711	23.8711
90–10	(48, 1/3)	11.0000	(48, 1/3)	6.9700	22.9148	22.9148

Table 5 $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 20 \text{ MHz}$, $N_0 = 10^{-7} \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(64, 2/3)	15.0000	(64, 2/3)	8.4148	32.9280	36.7442
30–70	(64, 2/3)	15.0000	(64, 2/3)	8.7092	31.3549	35.0942
50–50	(64, 2/3)	15.0000	(32, 1/3)	7.0445	30.9207	32.8401
70–30	(48, 1/2)	15.0000	(32, 1/3)	7.6110	29.2156	32.2485
90–10	(48, 1/2)	15.0000	(32, 1/3)	7.6210	28.2575	31.2847
MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(64, 2/3)	15.0000	(64, 2/3)	5.6298	35.6857	35.6857
30–70	(64, 2/3)	15.0000	(32, 1/3)	5.9121	33.3853	33.3853
50–50	(64, 2/3)	15.0000	(32, 1/3)	5.7300	31.5918	31.5918
70–30	(64, 2/3)	15.0000	(32, 1/3)	5.5814	29.7539	29.7539
90–10	(48, 1/2)	15.0000	(32, 1/3)	5.7257	28.3789	28.3789

class, which is reported in Table 7, for the MAD criterion. Despite the addition of noise, an increase of PSNR is observed for the low-motion class compared to the corresponding case in Table 3, for the same criterion. This is attributed to the fact that, under the influence of noise, the power level of the low-motion class is increased. Thus, the ratio E_k/I_0 also increases as follows from Eq. (7).

Additional information regarding the performance of the proposed schemes is graphically illustrated in figures. Specifically, Fig. 2(a) depicts the differences of the received PSNR between the MAD and MMD criteria for both the high- and low-motion class of nodes, for all node distributions and refers to the case of $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 20 \text{ MHz}$, and $N_0 = 0 \text{ W/Hz}$. The last column of the same figure shows the accumulated PSNR difference between

Table 6 $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 15 \text{ MHz}$, $N_0 = 10^{-7} \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(64, 2/3)	15.0000	(32, 1/3)	7.1567	31.4147	33.3082
30–70	(48, 1/2)	15.0000	(32, 1/3)	7.6223	29.0315	32.0576
50–50	(48, 1/2)	15.0000	(32, 1/3)	7.6366	27.7619	30.7834
70–30	(32, 1/3)	15.0000	(32, 1/3)	9.9299	26.7537	31.6109
90–10	(32, 1/3)	15.0000	(32, 1/3)	10.2012	26.4153	31.1736
MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(64, 2/3)	15.0000	(32, 1/3)	5.8898	32.9863	32.9863
30–70	(64, 2/3)	15.0000	(32, 1/3)	5.6668	30.7003	30.7003
50–50	(48, 1/2)	15.0000	(32, 1/3)	5.7028	28.5851	28.5851
70–30	(32, 1/3)	15.0000	(32, 1/3)	6.0882	27.1463	27.1463
90–10	(32, 1/3)	15.0000	(32, 1/3)	6.7141	26.5271	26.5271

Table 7 $R_{\text{target}} = 144 \text{ kbps}$, $W_t = 20 \text{ MHz}$, $N_0 = 10^{-7} \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(48, 1/3)	15.0000	(48, 1/3)	9.0621	29.0841	33.0421
30–70	(48, 1/3)	15.0000	(48, 1/3)	9.3985	28.0076	31.8577
50–50	(48, 1/3)	15.0000	(48, 1/3)	9.6979	27.1164	30.8916
70–30	(48, 1/3)	15.0000	(48, 1/3)	9.9590	26.3597	30.0736
90–10	(48, 1/3)	15.0000	(48, 1/3)	10.1896	25.7016	29.3630

MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(72, 1/2)	15.0000	(48, 1/3)	6.3510	32.2887	32.2887
30–70	(48, 1/3)	15.0000	(48, 1/3)	6.4367	29.6111	29.6111
50–50	(48, 1/3)	15.0000	(48, 1/3)	7.1026	28.0276	28.0276
70–30	(48, 1/3)	15.0000	(48, 1/3)	7.6444	26.8170	26.8170
90–10	(48, 1/3)	15.0000	(48, 1/3)	8.1007	25.8331	25.8331

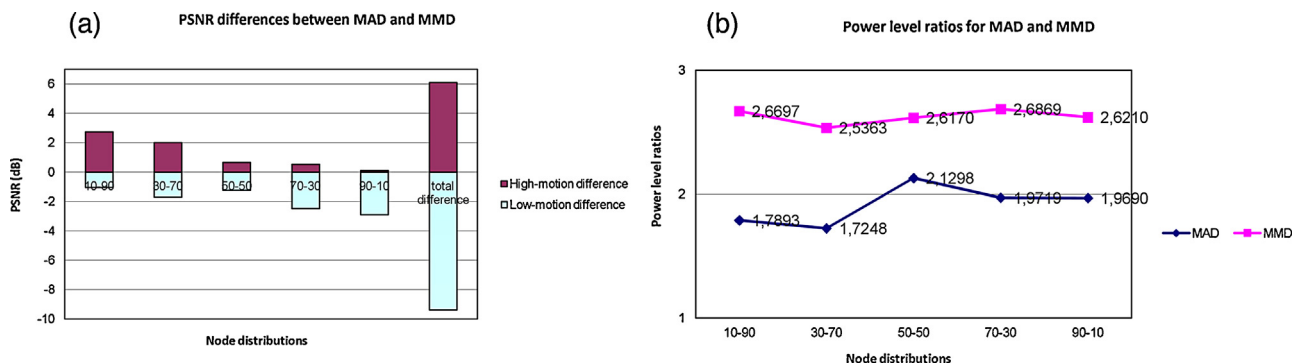
Table 8 $R_{\text{target}} = 144 \text{ kbps}$, $W_t = 15 \text{ MHz}$, $N_0 = 10^{-7} \text{ W/Hz}$.

MAD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(48, 1/3)	15.0000	(48, 1/3)	10.1179	25.9716	29.6706
30–70	(48, 1/3)	15.0000	(48, 1/3)	10.4608	24.9646	28.5689
50–50	(48, 1/3)	15.0000	(48, 1/3)	10.7574	24.1222	27.6593
70–30	(48, 1/3)	15.0000	(48, 1/3)	11.0116	23.4013	26.8828
90–10	(48, 1/3)	15.0000	(48, 1/3)	11.2325	22.7709	26.2046

MMD						
$N_{\text{high}} - N_{\text{low}}$	$(R_{s,\text{high}}, R_{c,\text{high}})$	S_{high}	$(R_{s,\text{low}}, R_{c,\text{low}})$	S_{low}	$\text{PSNR}_{\text{high}}$	PSNR_{low}
10–90	(48, 1/3)	15.0000	(48, 1/3)	6.8309	28.7016	28.7016
30–70	(48, 1/3)	15.0000	(48, 1/3)	7.7878	26.5355	26.5355
50–50	(48, 1/3)	15.0000	(48, 1/3)	8.4948	25.0255	25.0255
70–30	(48, 1/3)	15.0000	(48, 1/3)	9.0528	23.8579	23.8579
90–10	(48, 1/3)	15.0000	(48, 1/3)	9.5102	22.9029	22.9029

MAD and MMD, also for all node distributions. This figure manifests that, cumulatively for all node distributions, the decrease in PSNR achieved by the MMD criterion for the low-motion class of nodes is considerably higher than the corresponding gain for the high-motion class of nodes. For the case of a bit rate of 96 kbps and a bandwidth of 20 MHz, the MMD increases the total PSNR for all members of the high-motion class by 6.0986 dB, while the total PSNR for all members of the low-motion class decreases by 9.3798 dB. Therefore, despite the fact that the MMD offers equal PSNR values to both motion classes, it is proved to be less fair than it was initially perceived, since it disfavors the low-motion class of nodes.

An additional piece of information is provided by Fig. 2(b) that depicts the optimal ratios $S_{\text{high}}/S_{\text{low}}$ for the two optimization criteria when the number of nodes imaging high levels of motion increases, while that of the nodes imaging low levels of motion decreases. The specific figure refers to the case of $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 20 \text{ MHz}$, and $N_0 = 0 \text{ W/Hz}$ and offers strong evidence that the MAD criterion requires much less power than the MMD. Spending less power for data transmission means that a stronger channel coding is used, capable of correcting a higher number of channel errors. From Eq. (6), it follows that the channel coding rate is inversely proportional to the data compression rate, highlighting the importance of considering the special characteristics of the

**Fig. 2.** PSNR differences (a) and power level ratios (b) for the MAD and MMD criteria, for various node distributions, with $R_{\text{target}} = 96 \text{ kbps}$, $W_t = 20 \text{ MHz}$, and $N_0 = 0 \text{ W/Hz}$.

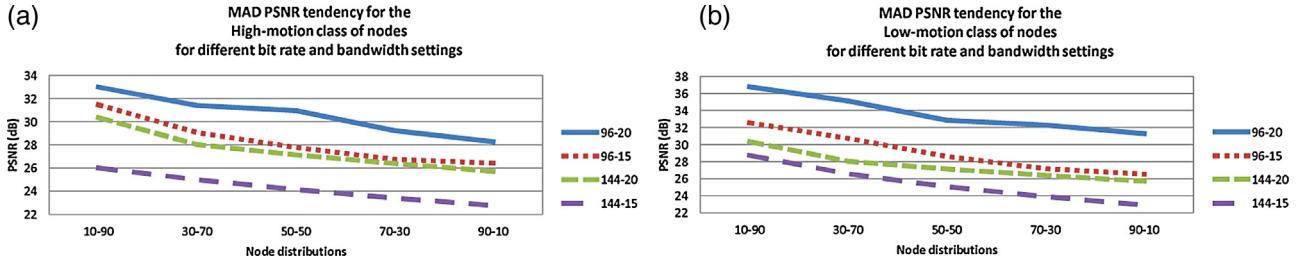


Fig. 3. PSNR tendency for the MAD criterion (a) for the high-motion, and (b) low-motion class of nodes, for various node distributions, with $R_{\text{target}} = 96$ kbps, $W_t = 20$ MHz, and $N_0 = 0$ W/Hz.

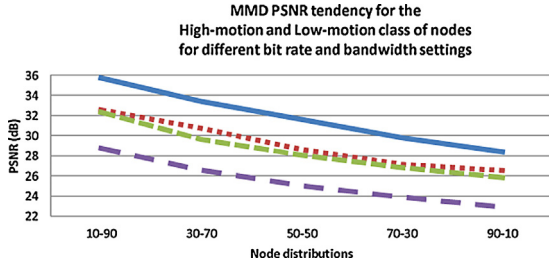


Fig. 4. PSNR tendency for the MMD criterion, for both motion classes, for various node distributions, with $R_{\text{target}} = 96$ kbps, $W_t = 20$ MHz, and $N_0 = 0$ W/Hz.

video sequence (high- against low-motion), when determining the optimal power levels.

Moreover, the power received by the CCU from the low-motion class of nodes is always less than that of the high-motion class for both optimization criteria, as it is confirmed in the tables for all the examined bit rate-bandwidth combinations. Hence, since the nodes that image high levels of motion need higher power levels than those in the low-motion class, it is reasonable that the ratio E_k/I_0 of Eq. (7) exhibits further decrease when the cardinality of the high-motion class is larger than that of the low-motion class. Naturally, this incurs a reduction to the PSNR values for both motion classes.

Actually, this fact explains the downward trend of the PSNR values illustrated in Figs. 3 and 4, for both motion classes, for all examined bit rate and bandwidth combinations. Specifically, Fig. 3(a) includes the PSNR variations for the high-motion class of nodes, while Fig. 3(b) illustrates the same results for the low-motion class of nodes for the MAD criterion. Fig. 4 offers the corresponding information for the MMD criterion. Note that for the MMD case, the PSNR values for both motion classes were identical.

Another issue that gained our interest was the efficiency of PSO against that of AS in terms of the number of iterations required for obtaining solutions of same quality with AS. Fig. 5 offers this information. In particular, it illustrates the number of iterations required by PSO, averaged over the 30 independent experiments, to detect the same solution with AS for different levels of precision, namely 3, 4, 5, and 6 decimal digits. In each subfigure, both criteria are compared for all examined bit rate and bandwidth combinations. Fig. 5(a) and (b) present the aforementioned results for the node distribution “30–70”, while Fig. 5(c) and (d) for the node distribution “70–30”. Moreover, Fig. 5(a) and (c) refer to the noiseless case ($N_0 = 0$ W/Hz), and Fig. 5(b) and (d) to the noisy case ($N_0 = 10^{-7}$ W/Hz).

From the figures, we corroborate that PSO requires fewer iterations to achieve same quality solutions with the AS algorithm for lower precisions. As expected, increasing precision is accompanied by a higher number of iterations. Furthermore, it is obvious that the MMD criterion needs more function evaluations than MAD to achieve the optimal solution, while the addition of noise incurs an increase to the number of iterations for both

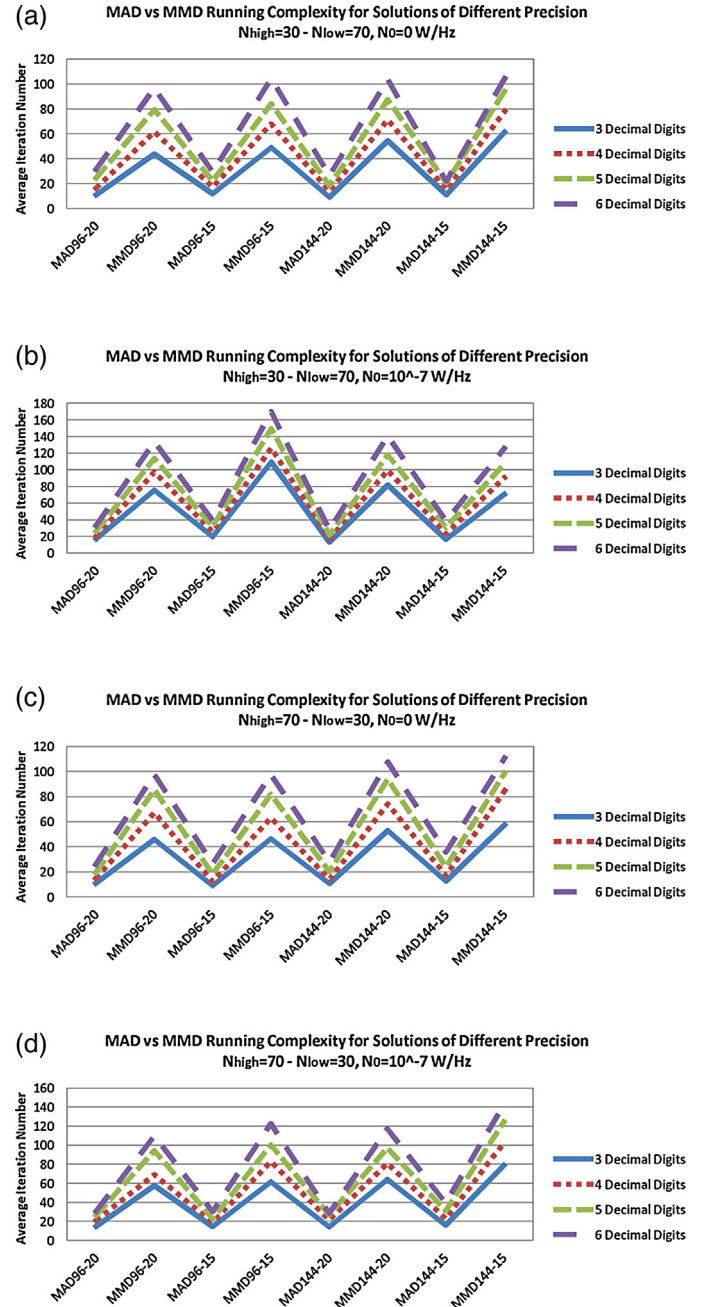


Fig. 5. Average number of iterations required by PSO to obtain the solution given by AS, for both criteria, and precision of 3, 4, 5 and 6 decimal digits, for all examined bit rate-bandwidth combinations, for two different node distributions. Fig. 5(a) and (c) correspond to the noiseless case, while Fig. 5(b) and (d) refer to the noisy case.

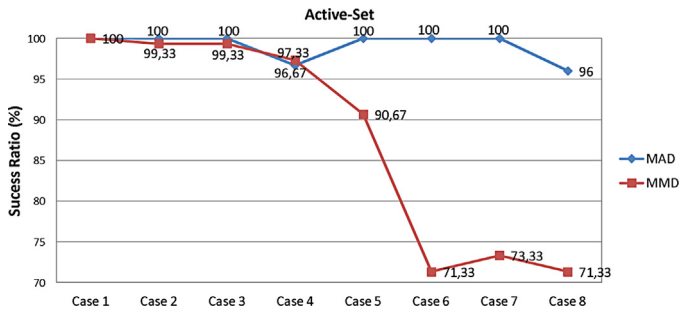


Fig. 6. Success ratio of AS in finding the optimal solution.

optimization criteria. The aforementioned confirmations derive from all considered bit rate-bandwidth combinations and node distributions.

Fig. 6 presents the success ratio of AS in achieving the optimal solution, for both MAD and MMD criteria. Particularly, each of the cases as referred to this figure corresponds to the results of each corresponding table. For example, Case 3 corresponds to the results of Table 3, when $R_{\text{target}} = 144$ kbps, $W_t = 20$ MHz, and $N_0 = 0$ W/Hz. For this case, we observe that for the MAD it has a success ratio of 100% and 99.33% for MMD. For the MAD criterion, this means that each of the node distributions manages to achieve the same best solution in all 30 experiments. In contrast, the success ratio for MMD implies that one experiment out of 30, for a specific node distribution, failed to reach the best solution.

Interpreting the AS efficiency for MAD and MMD, we infer that for the MAD criterion it is capable of detecting the best solution in most of the cases and exhibits better performance than for MMD. The superiority for MAD is clearer in noisy cases (Cases 5–8) where for MMD it presents success ratios of nearly 70%. Although it may seem high, we indicatively refer that for the case of $N_{\text{high}} = 90 - N_{\text{low}} = 10$ of Table 8, only 7/30 experiments reach the optimal solution. Therefore, this figure sheds light on the weakness of AS in detecting always the best solution, while it also demonstrates that its efficiency depends both on the initially supplied starting point as well as on the objective function to be optimized.

Lastly, Table 9 offers an intuition regarding the execution time required by each optimization algorithm averaged over 30 experiments per problem instance. Also, the total running times needed for the execution of all cases of node distributions for each specific bit rate and bandwidth combination, are presented. The same table includes results for both criteria and all cases of considered bit rate and bandwidth for both the noiseless and noisy cases. We shall note that AS was set to execute 40,000 function evaluations (as PSO and HPSOAS), with the function tolerance set to 10^{-15} and the tolerance of the candidate optimal solution to 10^{-15} . We set the function and optimal point tolerances to such levels since we confirmed that the PSNR results provided by the AS were different from those of PSO even to the first decimal digit for some cases. At this point, we should recall that the AS method was run sequentially 9 times for each tested case in order to evaluate all possible combinations of bit rate and bandwidth. The execution times of this method are averaged over the number of successful experiments per case.

An overall inspection of the results denotes that despite PSO is able to reach the optimal solutions efficiently in all 30 experiments of each considered case, as opposed to AS, it still needs more time to be executed compared to both AS and HPSOAS. Particularly, more time is needed as more nodes are assigned to the low-motion class of nodes. Moreover, it follows that MAD takes less time to reach the optimal point compared to MMD, while in the noisy case the MMD

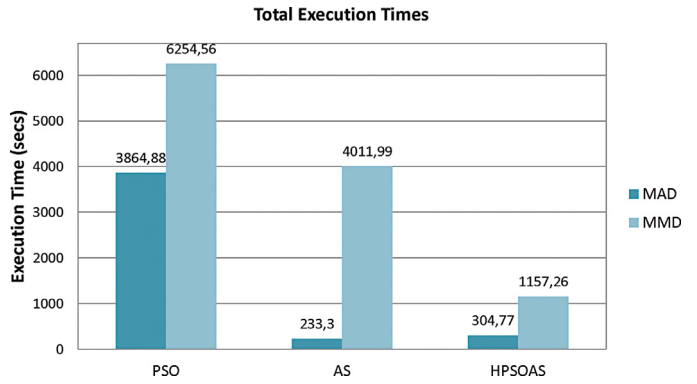


Fig. 7. Total execution times of all node distributions and bit rate-bandwidth combinations of both noiseless and noisy cases, for MAD and MMD, using all optimization methods.

needs the double time to achieve the optimal solution compared to the noiseless case, using the PSO.

Commenting on the AS execution times, it is clear from Table 9 that it requires less time compared to PSO, with two exceptions for the noiseless case of MMD criterion when the bit rate is set to 144 kbps. This probably means that the randomly supplied starting points are far away from the optimal solution and that the shape of the objective function complicates further the finding of the solution. Additionally, we may think that the AS method follows the brute force approach testing all possible cases of source and channel pair values, increasing the overall problem's complexity. Therefore, this method is impractical to applications where problems with higher dimensions exist. For example, if we had made the assumption for a node clustering into 10 motion classes, the AS should be executed 3^{10} times in order to determine the optimal source and channel coding rate combinations for each motion class, instead of the 3^2 times of our problem.

Therefore, as it was also mentioned in Section 4.3 in order to surmount all weaknesses of both PSO and AS keeping their benefits at the same time, we resorted to the development of the HPSOAS optimization method. Table 9 demonstrates that the optimization of MMD criterion using the HPSOAS method requires significantly less time compared to both PSO and AS. Similarly, the MAD optimized under the use of the HPSOAS saves much execution time compared to PSO and the same holds also in the half cases compared to AS.

Interesting enough are also the conclusions drawn by Fig. 7. The goal of the specific illustration is to give us an insight regarding the overall execution times for each of the MAD and MMD criteria, when we aggregate the total times for each considered bit rate and bandwidth combination, for both the noiseless and noisy cases. Evidently, it follows that the MMD takes longer to converge compared to the MAD, for all considered optimization methods. Moreover, for the MMD it is clear that HPSOAS needs about 1/6th and 1/4th of the corresponding time of PSO and AS, respectively, in order to converge to the optimal point. Further is the gain in time by adopting the HPSOAS to the MAD criterion instead of PSO. In this case, HPSOAS reaches the optimal solution in about 1/12th of PSO's time. Comparable execution times are required accumulatively by both AS and HPSOAS for the MAD criterion.

Summarizing, HPSOAS keeps the following strong points that renders its use very appealing: (i) it is not sensitive to initial conditions, (ii) there is no need to execute an exhaustive search in order to determine the discrete parameters of this work, (iii) it can be applied to problems of higher dimensions without a tremendous effect on problem's complexity, and (iv) it takes much less time for its execution compared to other stochastic methods.

Table 9

Average execution times (in seconds).

$N_{\text{high}} - N_{\text{low}}$	$R_{\text{target}} = 96 \text{ kbps}, W_t = 20 \text{ MHz}, N_0 = 0 \text{ W/Hz}$						$R_{\text{target}} = 96 \text{ kbps}, W_t = 20 \text{ MHz}, N_0 = 10^{-7} \text{ W/Hz}$					
	MAD			MMD			MAD			MMD		
	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS
10–90	148.78	4.67	5.43	154.34	7.36	9.71	154.83	4.93	10.70	319.55	7.08	13.24
30–70	117.27	4.91	3.74	125.29	97.94	16.41	124.11	5.20	11.90	245.21	51.37	41.91
50–50	81.74	6.43	3.59	94.55	53.01	32.22	87.08	5.61	0.85	185.45	47.82	48.75
70–30	51.77	6.27	0.79	53.56	43.78	8.78	56.75	5.95	1.23	122.51	53.13	23.14
90–10	25.44	6.47	0.61	29.01	125.44	13.29	28.15	8.88	4.04	58.87	70.79	50.42
Total time	425.00	28.75	14.16	456.75	327.53	80.41	450.92	30.57	28.72	931.59	230.19	177.46

$N_{\text{high}} - N_{\text{low}}$	$R_{\text{target}} = 96 \text{ kbps}, W_t = 15 \text{ MHz}, N_0 = 0 \text{ W/Hz}$						$R_{\text{target}} = 96 \text{ kbps}, W_t = 15 \text{ MHz}, N_0 = 10^{-7} \text{ W/Hz}$					
	MAD			MMD			MAD			MMD		
	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS
10–90	182.11	3.32	15.89	183.22	12.38	37.24	169.20	4.64	2.85	352.83	28.09	27.30
30–70	151.86	6.32	6.32	129.84	20.86	21.51	141.15	5.76	13.33	295.61	51.79	13.05
50–50	121.44	6.17	18.32	99.39	131.27	28.98	103.18	5.58	10.96	243.32	92.02	27.58
70–30	68.89	6.63	5.52	63.67	165.39	40.93	61.64	6.00	0.61	133.04	86.56	17.84
90–10	29.69	9.53	1.18	26.83	114.67	17.44	29.10	8.97	2.65	52.54	90.14	31.35
Total time	553.99	25.45	47.23	502.95	444.57	146.10	504.27	30.95	30.40	1077.34	348.60	117.12

$N_{\text{high}} - N_{\text{low}}$	$R_{\text{target}} = 144 \text{ kbps}, W_t = 20 \text{ MHz}, N_0 = 0 \text{ W/Hz}$						$R_{\text{target}} = 144 \text{ kbps}, W_t = 20 \text{ MHz}, N_0 = 10^{-7} \text{ W/Hz}$					
	MAD			MMD			MAD			MMD		
	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS
10–90	183.50	4.51	19.78	176.71	26.23	13.75	148.45	5.50	7.07	359.43	48.07	85.15
30–70	137.70	5.60	4.72	147.39	106.24	12.33	120.21	5.16	10.33	284.55	89.94	47.53
50–50	107.84	6.17	12.90	119.47	116.33	22.95	88.51	4.46	1.24	217.15	95.29	26.18
70–30	69.08	6.38	22.96	86.76	198.52	59.87	60.34	5.18	21.44	166.85	109.61	24.80
90–10	26.34	7.57	4.26	29.97	201.45	24.68	25.05	6.65	2.35	90.57	105.50	51.44
Total time	524.46	30.23	64.62	560.30	648.77	133.58	442.56	26.95	42.43	1118.55	448.41	235.10

$N_{\text{high}} - N_{\text{low}}$	$R_{\text{target}} = 144 \text{ kbps}, W_t = 15 \text{ MHz}, N_0 = 0 \text{ W/Hz}$						$R_{\text{target}} = 144 \text{ kbps}, W_t = 15 \text{ MHz}, N_0 = 10^{-7} \text{ W/Hz}$					
	MAD			MMD			MAD			MMD		
	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS	PSO	AS	HPSOAS
10–90	152.84	3.65	2.44	155.06	176.00	13.98	166.51	4.60	1.67	343.07	77.41	25.07
30–70	122.65	6.08	0.54	127.42	101.08	8.32	142.24	4.92	16.71	299.31	75.09	17.36
50–50	93.27	7.39	10.83	100.01	178.71	50.09	106.31	5.22	22.97	243.20	98.60	23.78
70–30	57.95	6.85	0.95	59.56	333.43	17.66	61.56	5.54	12.71	172.14	102.95	53.07
90–10	32.31	9.38	4.95	31.83	313.93	27.19	28.04	6.77	3.44	75.48	106.72	30.97
Total time	459.02	33.35	19.71	473.88	1103.15	117.24	504.66	27.05	57.5	1133.20	460.77	150.25

6. Conclusions

We studied the problem of optimal resource allocation among the nodes of a wireless DS-CDMA VSN. More specifically, we optimally allocated the source coding rates, channel coding rates, and power levels to all nodes, based on the detected amount of motion per node. For the source and channel coding rates, discrete values were considered, while for the power levels, we assumed continuous values within a predetermined range.

The MAD criterion that minimizes the average network distortion and the MMD criterion that minimizes the maximum distortion among all nodes of the network, both aiming at achieving the highest possible video quality, were employed in the present study. In order to solve the underlying mixed integer optimization problems, the established PSO algorithm was employed, motivated by preliminary results in previous works. The performance of this algorithm was assessed in comparison with the performance of the AS method, justifying its capability in tackling such problems. Additionally, we developed a hybrid optimization method that is based on PSO, using the AS as a local optimizer. The experimental

results using all optimization methods highlighted the superiority of HPSOAS over both AS and PSO, under various aspects.

Having clustered the nodes into two motion classes based on the amount of the detected motion per node, extensive experimentation showed that the MAD criterion works favorably for the nodes that image low motion, since they are offered considerably higher PSNR values. On the other hand, the MMD assigns equal PSNR values to both motion classes. Nevertheless, it is not sufficiently clear how fair the MMD can be for the nodes that record low motion, taking into consideration the significant PSNR reduction observed in this case, for this class of nodes. Furthermore, our results confirmed that the CCU receives less power with the MAD than with the MMD, implying that MAD requires less power for data transmission.

Experiments conducted under the presence of thermal and background noise, verified the conclusions derived for the noiseless case. The main impact of noise was a marginal reduction of the PSNR of both motion classes and optimization criteria, with only a minor exception. Also, the nodes that detected high levels of motion required considerably higher power levels than the nodes that detected low levels of motion, to accomplish data transmission.

Summarizing, our investigation aligns with the general feeling that the most appropriate approach for tackling resource allocation problems in wireless VSNs, is rather problem-dependent. The application's special characteristics and requirements shall dictate the methodology to use, inhibiting the possibility of a panacea that would simultaneously favor all nodes.

Acknowledgements

The authors would like to thank the editor and the anonymous reviewers for their constructive comments. Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-12-1-0001. The US Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright notation thereon.

References

- [1] S. Soro, W. Heinzelman, A survey of visual sensor networks, *Adv. Multimed.* 2009 (2009) 1–22.
- [2] J. Day, H. Zimmermann, The OSI reference model, *Proc. IEEE* 71 (1983) 1334–1340.
- [3] L.P. Kondi, F. Ishtiaq, A.K. Katsaggelos, Joint source-channel coding for motion-compensated DCT-based SNR scalable video, *IEEE Trans. Image Process.* 11 (2002) 1043–1052.
- [4] A.K. Katsaggelos, Y. Eisenberg, F. Zhai, R. Berry, T.N. Pappas, Advances in efficient resource allocation for packet-based real-time video transmission, *Proc. IEEE* 93 (2005) 135–147.
- [5] H. Jiang, W. Zhuang, Resource allocation with service differentiation for wireless video transmission, *IEEE Trans. Wirel. Commun.* 5 (2006) 1456–1468.
- [6] D. Triantafyllou, N. Passas, E. Zervas, L. Merakos, Optimized cross-layer adaptation for improved resource management in wireless networks, in: *Proceedings of the WiOpt'10: Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, Avignon, France, 2010, pp. 380–386.
- [7] J. Tang, X. Zhang, Cross-layer resource allocation over wireless relay networks for Quality of Service provisioning, *IEEE J. Sel. Areas Commun.* 25 (2007) 645–656.
- [8] T. Ozcelebi, M.O. Sunay, A.M. Tekalp, M.R. Civanlar, Cross-layer optimized rate adaptation and scheduling for multiple-user wireless video streaming, *IEEE J. Sel. Areas Commun.* 25 (2007) 760–769.
- [9] I.F. Akyildiz, T. Melodia, K.R. Chowdhury, A survey on wireless multimedia sensor networks, *Comput. Netw. ISDN Syst.* 51 (2007) 921–960.
- [10] M. Wu, C.W. Chen, Multiple bitstream image transmission over wireless sensor networks, in: *Proceedings of the IEEE Sensors*, vol. 2, 2003, pp. 727–731.
- [11] H. Kwon, T.H. Kim, S. Choi, B.G. Lee, A cross-layer strategy for energy-efficient reliable delivery in wireless sensor networks, *IEEE Trans. Wirel. Commun.* 5 (2006) 3689–3699.
- [12] R. Madan, S. Cui, S. Lal, A. Goldsmith, Cross-layer design for lifetime maximization in interference-limited wireless sensor networks, *IEEE Trans. Wirel. Commun.* 5 (2006) 3142–3152.
- [13] W.J. Huang, Y.W.P. Hong, C.C.J. Kuo, Lifetime maximization for amplify-and-forward cooperative networks, *IEEE Trans. Wirel. Commun.* 7 (2008) 1800–1805.
- [14] Z. Yang, K. Nahrstedt, A bandwidth management framework for wireless camera array, in: *Proceedings of the International Workshop on network and operating systems support for digital audio and video*, ACM, 2005, pp. 147–152.
- [15] T. Zahariadis, S. Voliotis, Open issues in wireless visual sensor networking, in: *Systems, Signals and Image Processing, 2007 and 6th EURASIP Conference Focused on Speech and Image Processing*, 2007, pp. 335–338.
- [16] E.S. Pynadath, L.P. Kondi, Cross-layer optimization with power control in DS-CDMA visual sensor networks, in: *Proceedings of the IEEE International Conference on Image Processing*, 2006, pp. 25–28.
- [17] E.S. Bentley, J.D. Matyjas, M.J. Medley, L.P. Kondi, Wireless visual sensor network resource allocation using cross-layer optimization, in: *Visual Communications and Image Processing*, vol. 7257, SPIE, 2009.
- [18] E.S. Bentley, L.P. Kondi, J. Matyjas, M. Medley, B. Suter, Spread spectrum visual sensor network resource management using an end-to-end cross-layer design, *IEEE Trans. Multimed.* 13 (2011) 125–131.
- [19] J. Hagenauer, Rate-compatible punctured convolutional codes (RCP codes) and their applications, *IEEE Trans. Commun.* 36 (1988) 389–400.
- [20] K. Pandremmenou, L.P. Kondi, K.E. Parsopoulos, Optimal power allocation and joint source-channel coding for wireless DS-CDMA visual sensor networks, in: *Visual Information Processing and Communication II*, vol. 7882, SPIE, 2011.
- [21] C.A. Floudas, *Nonlinear and Mixed-Integer Optimization*, Oxford University Press, New York, 1995.
- [22] B. Dieber, C. Micheloni, B. Rinner, Resource-aware coverage and task assignment in visual sensor networks, *IEEE Trans. Circuits Syst. Video Technol.* 21 (2011) 1424–1437.
- [23] M. Shindarov, S. Fidanova, P. Marinov, Wireless sensor positioning algorithm, in: *6th IEEE International Conference on Intelligent Systems (IS)*, 2012, pp. 419–424.
- [24] A. Raiconi, M. Gentili, Exact and metaheuristic approaches to extend lifetime and maintain connectivity in wireless sensors networks, *INOC Lecture Notes in Computer Science*, 6701, Springer, 2011, pp. 607–619.
- [25] R. Kulkarni, G. Venayagamoorthy, Particle swarm optimization in wireless-sensor networks: a brief survey, *IEEE Trans. Syst. Man Cybern. C: Appl. Rev.* 41 (2011) 262–267.
- [26] K.E. Parsopoulos, M.N. Vrahatis, *Particle Swarm Optimization and Intelligence: Advances and Applications*, Information Science Publishing (IGI Global), 2010.
- [27] P.E. Gill, W. Murray, M.H. Wright, *Practical optimization*, Academic Press, 1981.
- [28] D.G. Luenberger, Y. Ye, *Linear and Nonlinear Programming*, Springer, New York, 2008.
- [29] K.G. Murty, *Linear Complementarity, Linear and Nonlinear Programming, Sigma Series in Applied Mathematics 3*, Heldermann Verlag, Berlin, 1988.
- [30] F. Neri, C. Cotta, P. Moscato, *Handbook of Memetic Algorithms*, Springer-Verlag, Berlin/Heidelberg, 2012.
- [31] H. Wang, L.P. Kondi, A. Luthra, S. Ci, *4G Wireless Video Communications*, Wiley Publishing, 2009.
- [32] T. Wiegand, G. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC video coding standard, *IEEE Trans. Circuits Syst. Video Technol.* 13 (2003) 560–576.
- [33] T. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed., Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.
- [34] K.S. Gilhousen, I.M. Jacobs, R. Padovani, A.J. Viterbi, L.A. Weaver, C.E. Wheatley, On the capacity of a cellular CDMA system, *IEEE Trans. Veh. Technol.* 40 (1991) 303–312.
- [35] Y.S. Chan, J. Modestino, A joint source coding-power control approach for video transmission over CDMA networks, *IEEE J. Sel. Areas Commun.* 21 (2003) 1516–1525.
- [36] L.P. Kondi, E.S. Bentley, Game-theory-based cross-layer optimization for wireless DS-CDMA visual sensor networks, in: *Proceedings of the 17th IEEE International Conference on Image Processing (ICIP)*, 2010, pp. 4485–4488.
- [37] L.P. Kondi, A.K. Katsaggelos, Joint source-channel coding for scalable video using models of rate-distortion functions, in: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, pp. 1377–1380.
- [38] M. Bystrom, T. Stockhammer, Modeling of operational distortion-rate characteristics for joint source-channel coding of video, in: *Proceedings of the International Conference on Image Processing*, vol. 1, 2000, pp. 359–362.
- [39] A.V. Katsenou, L.P. Kondi, K.E. Parsopoulos, Motion-related resource allocation in dynamic wireless visual sensor network environments, *IEEE Trans. Image Process.* 23 (2014) 56–68.
- [40] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: *Proceedings of the Sixth Symposium on Micro Machine and Human Science*, IEEE Service Center, Piscataway, NJ, 1995, pp. 39–43.
- [41] G. Minati, E. Pessa, *Collective Beings*, Springer, New York, 2010.
- [42] A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, Wiley, 2006.
- [43] J. Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, in: *Proceedings of the IEEE Congress on Evolutionary Computation*, IEEE Press, Washington, DC, USA, 1999, pp. 1931–1938.
- [44] P.N. Suganthan, Particle swarm optimizer with neighborhood operator, in: *Proceedings of the IEEE Congress on Evolutionary Computation*, Washington, DC, USA, 1999, pp. 1958–1961.
- [45] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. Evol. Comput.* 6 (2002) 58–73.
- [46] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: *Proceedings of the IEEE Congress on Evolutionary Computation*, Piscataway, NJ, 2000, pp. 84–88.
- [47] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, *Inf. Process. Lett.* 85 (2003) 317–325.
- [48] F. Van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, *Inf. Sci.* 176 (2006) 937–971.
- [49] M. Jiang, Y.P. Luo, S.Y. Yang, Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm, *Inf. Sci.* 102 (2007) 8–16.
- [50] M. Smitt, R. Wanka, Particle swarm optimization almost surely finds local optima, *Theor. Comput. Sci.* (2014), <http://dx.doi.org/10.1016/j.tcs.2014.05.017>.
- [51] K. Pandremmenou, L.P. Kondi, K.E. Parsopoulos, E.S. Bentley, Game-theoretic solutions through intelligent optimization for efficient resource management in wireless visual sensor networks, *Signal Proces. Image Commun.* 29 (2014) 472–493.
- [52] K. Pandremmenou, L.P. Kondi, K.E. Parsopoulos, Optimal power allocation and joint source-channel coding for wireless DS-CDMA visual sensor networks using the Nash bargaining solution, in: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 2340–2343.
- [53] A.V. Katsenou, L.P. Kondi, K.E. Parsopoulos, Resource management for wireless visual sensor networks based on individual video characteristics, in: *18th IEEE International Conference on Image Processing (ICIP)*, 2011, pp. 149–152.
- [54] A.V. Katsenou, L.P. Kondi, K.E. Parsopoulos, E.S. Bentley, Quality-driven power control and resource allocation in wireless multi-rate visual sensor networks, in: *19th IEEE International Conference on Image Processing (ICIP)*, 2012, pp. 1117–1120.

- [55] E.G. Datsika, A.V. Katsenou, L.P. Kondi, E. Papapetrou, K.E. Parsopoulos, Priority-based cross-layer optimization for multihop DS-CDMA visual sensor networks, in: 19th IEEE International Conference on Image Processing (ICIP), 2012, pp. 1101–1104.
- [56] A.V. Katsenou, E.G. Datsika, L.P. Kondi, E. Papapetrou, K.E. Parsopoulos, Power-aware QoS enhancement in multihop DS-CDMA visual sensor networks, in: 18th International Conference on Digital Signal Processing (DSP), 2013, pp. 1–6.
- [57] A.V. Katsenou, L.P. Kondi, K.E. Parsopoulos, On the use of clustering for resource allocation in wireless visual sensor networks, in: Visual Information Processing and Communication III, vol. 8305, SPIE, 2012, p. 830505.
- [58] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of the Conference on Systems, Man and Cybernetics, 1997, pp. 4104–4109.
- [59] A. Farmahini-Farahani, S. Vakili, S.M. Fakhraie, S. Safari, C. Lucas, Parallel scalable hardware implementation of asynchronous discrete particle swarm optimization, *Eng. Appl. Artif. Intell.* 23 (2010) 177–187.
- [60] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization, *Nat. Comput.* 1 (2002) 235–306.
- [61] K.E. Parsopoulos, M.N. Vrahatis, Studying the performance of unified particle swarm optimization on the single machine total weighted tardiness problem, in: A. Sattar, B.H. Kang (Eds.), *Lecture Notes in Artificial Intelligence (LNAI)*, volume 4304, Springer, 2006, pp. 760–769.
- [62] E.C. Laskari, K.E. Parsopoulos, M.N. Vrahatis, Particle swarm optimization for integer programming, in: Proceedings of the IEEE Congress on Evolutionary Computation, Hawaii (HI), USA, IEEE Press, 2002, pp. 1582–1587.
- [63] MathWorks: constrained non-linear optimization algorithms: R2012a documentation, (2014). <http://www.mathworks.de/de/help/optim/ug/constrained-nonlinear-optimization-algorithms.html> (accessed 02.07.14).