J Supercomput (2017) 73:973–992
DOI 10.1007/s11227-016-1857-x

Internet of surveillance: a cloud supported large-scale wireless surveillance system

Mohammad A. Alsmirat[1] · **Yaser Jararweh**[1] · **Islam Obaidat**[1] · **Brij B. Gupta**[2]

Published online: 1 September 2016

Abstract Large-scale video surveillance systems are among the necessities for securing our life these days. The high bandwidth demand and the large storage requirements are the main challenges in such systems. To face these challenges, the system can be deployed as a multi-tier framework that utilizes different technologies. In such a framework, technologies proposed under the umbrella of the Internet of Things (IoT) can play a significant rule in facing the challenges. In video surveillance, the cameras can be considered as "the things" that are streaming videos to a central processing and storage server (the cloud) through the Internet. Wireless technologies can be used to connect wireless cameras to the surveillance system more conveniently than wired cameras. Unfortunately, wireless communication in general tend to have limited bandwidth that needs careful management to achieve scalability. In this paper, we design and evaluate a reliable IoT-based wireless video surveillance system that provides an optimal bandwidth distribution and allocation to minimize the overall surveillance video distortion. We evaluate our system using NS-3 simulation. The results show that the proposed framework fully utilizes the available cloud bandwidth budget and achieves high scalability.

Keywords Internet of Things · Wireless surveillance system · Cloud computing · Mobile edge computing

✉ Mohammad A. Alsmirat
masmirat@just.edu.jo

1 Department of Computer Science, Jordan University of Science and Technology, Irbid, Jordan

2 National Institute of Technology Kurukshetra, Kurukshetra, India

1 Introduction

Large-scale surveillance systems are becoming essential part of today civil defense systems. Such systems are widely deployed in our cities and urban communities. Currently, wireless-based video surveillance approach is commonly used to insure a real-time objects monitoring. Using wireless-based surveillance system is showing a momentum for its low overhead and affordable deployment cost [5,26,33–35,48]. Wireless-based large-scale video surveillance systems require the deployment of huge number of cameras in geographically distributed areas to achieve its objectives. This system of connected cameras is usually connected through the Internet to one or more video data aggregation centers.

Wireless-based video surveillance systems characteristics are similar to what it is known today as the Internet of Things (IoT), where the things in the video surveillance system are the cameras. The video surveillance system can exploit the existing Internet infrastructure which offers a low cost and ubiquitous connectivity. The video data collected by the set of deployed cameras are transferred to a centralized point or set of points that will complete the surveillance process by performing an automatic video analysis in real time.

The main challenge of such system is related to the amount of video data generated by the things (i.e., the cameras) which requires a very high channel bandwidth at the wireless link side (i.e., the Internet) and a huge computing and storage capacity at the processing side (i.e., the centralized data aggregation point).

In this paper, we utilize two new technologies to handle the aforementioned challenges, namely, the cloud and the Mobile Edge Computing (MEC) [21,22]. The MEC system will help in managing data collection and bandwidth management to insure efficient data transfer to the processing side of the system, while the cloud system will offer virtually unlimited resources capacity to perform video storage and analysis in real time. Specifically, we propose an IoT-based framework that provides an automated video surveillance solution. The framework supports a set of camera sensor groups. Each group is connected to an MEC server that is colocated with the base station that this group of cameras is connected to. This group of cameras can use the communication technology supported by the base station. A set of these groups are connected through the Internet to a central cloud server. The cameras capture and stream videos to the MEC servers. Each MEC server stores the video locally and then forwards it to the central cloud server. The MEC server may keep the video locally until it guarantees the reception of the video by the cloud and it then can delete it.

The framework provides a reliable surveillance video storage, on the central cloud server, with an optimal quality that fully utilizes the total central cloud provided bandwidth. Bandwidth management and allocation are done in two parts. Within the cell, and among the cameras in the group connected to the same base station, the bandwidth management and allocation are done by the MEC server colocated with that base station. Globally, among the MEC servers, the cloud server distributes the available bandwidth among the cells.

We evaluate our framework using NS-3 simulation. We stream real MJPEG video from the cameras over a simulated network and evaluate the system by measuring the video quality at the cloud server. The results show that the framework is highly scalable and fully utilizes the cloud available bandwidth.

2 Background and related work

2.1 IoT

Internet of Things (IoT) is a network that connects objects of any types. Objects can be any electronic or mechanical devises, persons, animals, or any thing else with a unique identifier. The network can be of any type as well. The development of IPv6 make the implementation of IoT more possible as it can identify very large number of object uniquely. IoT has many applications in the fields of agriculture [16,37], building management [4], health [18–20,25,46,47], energy saving [15,29,32], security [27,38,43] and surveillance [3,10,28,41], transportation [8], and many many more.

The most related work to the framework proposed in this paper is the work done in [3]. The authors proposes the skeleton of a face recognition-based video surveillance system. In the framework, they did not provide much details other then that they say that integrating a cloud server with a set of camera sensors to perform the required video processing solves all the problems of the system which are the high bandwidth and processing demands.

2.2 MEC

The number of mobile users and mobile applications is increasing dramatically. In traditional mobile systems that are integrated with cloud computing systems, this increase in users and applications produces high load on the core cloud servers as well. And because these cloud dependent applications rely heavily on services hosted by or performed by the cloud servers, a huge amount of data are uploaded from the mobile devices and downloaded from the core cloud servers. Consequently, network bandwidth consumption in these systems is continuously increasing. Adding to that the bandwidth demands introduced by the introduction of the Internet of Things (IoT). To deal with the growing bandwidth demands, network resources utilization needs to be improved. Moreover, enhancing network resources is needed. Furthermore, new advanced technologies should be adopted to enhance the QoS provided to end users. Long-term evolution (LET) one of the technologies that are proposed to increase the network bandwidth. However, employing new technologies requires further investments which causes higher operational cost. Recently, Mobile Edge Computing (MEC), described for the first time by Akamai et al. [13] when they described their content delivery network topology, has been proposed to overcome this issue in certain scenarios.

MEC has been proposed to ease the application development and data communications in mobile networks. The main services offered by MEC are data caching and/or providing the required processing power for the relevant data at the edge of the mobile network. Despite the fact that MEC is not yet utilized in real systems, many researchers studied and analyzed many MEC technical details and concepts. In MEC, a distributed services platform is employed instead of using a centralized platform to serve all mobile users and applications. This platform can be built by providing cloud servers on the edge of the mobile network. Initially, MEC was used to cache some of the frequently requested contents at the network edge [13]. Recent studies [2,7,14,44] propose the use of MEC with cloud computing to provide more complex services at the edge of the network. Traditionally, devices at the mobile network edge are only mobile access points, also known as base stations (BS). The BS rule is to forward traffic coming from mobile devices to a connected network and in the opposite direction. BS does not provide any processing or storage capabilities. MEC introduces new network elements at the network edge that provides computational power and storage capabilities. Therefore, these new devices can service user request locally instead of just forward these requests. In the rest of this paper, these devices are referred to as MEC servers.

2.2.1 MEC characterization

Typically, MEC-based systems can be characterized by the following:

– Self-owned: since the edge is local, this means that it can run in isolation from the rest of the overall network, while having the ability to access local resources. This is important for machine-to-machine scenarios. For example, when users need to deal with high level of security or safety systems.
– Nearer service: as the users are close to the information source, edge computing is specifically useful to gather important information for analytics and big data, to be able to provide more enhanced services to end users. Edge servers may also have the ability to directly access connected devices, which can easily be leveraged by business specific applications. since edge services run as near as possible to end users, it significantly reduces latency. And can be utilized to respond faster, to enhance QoE, or to reduce congestion in the rest of the network.
– Network environment information: real-time network data can be used by applications to offer context-related services that can differentiate the experience of mobile broadband and be monetized. Several new applications can be implemented (which will benefit from this real-time network data) to link mobile users with local points-of-interest, events and businesses.

2.2.2 MEC deployment scenarios

According to the Industry Specification Group (ISG),[1] MEC servers are usually deployed at the base station site of a cell. The cell internal network can be 3G, LTE, or WIFI. The cell site can be indoor within an enterprise (e.g., hospital, university campus, or any large building), or outdoor for a special public coverage scenario such as sport stadiums and shopping malls.

[1] https://portal.etsi.org/Portals/0/TBpages/MEC/Docs/Mobile-edge_Computing_-_Introductory_ _Technical_White_Paper_V1%2018-09-14.pdf.

2.2.3 MEC similar platforms

Mobile Edge Computing (MEC), Fog Computing (FC), and Mist Computing (MC) are all terms that have been introduced in the field of mobile cloud computing (MCC). They all refer to a platform, in which a set of devices with limited capabilities can use some nearby resources within nearby devices to accomplish their work.

MCC [12,24,45] refers to a platform that provides computation, storage, and applications, as services to mobile users. These services are usually provided by a set of centralized servers that are managed and controlled by the centralized service provider. MCC represents the incorporation between the cloud computing and the mobile devices. MCC provides many benefits, (1) it overcomes the limited mobile devices capabilities, such as the limited battery life, small memory, and the limited bandwidth, (2) it handles some environment difficulties such as heterogeneity and availability, and (3) it improves mobile security, privacy, and reliability. FC is considered as a middle solution between MCC and MEC and is defined as a virtualized platform which is not exclusively placed at the network edge and it can provide cloud services between end users and cloud computing servers [9]. MEC and FC can help reduce service latency and increase execution speed. Even that both MEC and FC seem very similar to grid computing, they defer in that the work distribution logic is implicitly handled by an underlying infrastructure layer, and not explicitly assigned into the applications. Mist Computing is another similar concept to MEC. Mis Computing is something that mix FC and MEC. It extends the traditional client–server approach to a more peer-to-peer based approach. In the rest of this paper, we will use MEC to refer to all of the above concepts.

2.2.4 MEC and Fog computing in the literature

In [2], the authors analyze a system called volley. It addresses the challenges arise from the rapidly growing cloud services across distributed datacenters. These challenges include (1) the need for automated mechanisms for placing application data among different datacenters, taking into account, the WAN bandwidth cost as well as the datacenters capacity limitations, and (2) reducing end user latency considering, shared data, data inter-dependencies, and application changes. Volley is used to automate the process of data placement among geographically distributed datacenters. The authors in [2] claimed that Volley reduces the oblique in datacenters load by more than two times. In addition, it reduces inter-datacenters traffic by more than 1.8 times. Moreover, it reduces latency.

Dsouza et al. [14] proposed a policy-based management framework for fog computing resources, and extended the current fog computing system to make it possible for the users to have a secure communication and resource sharing across distributed environment when requesting resources. The proposed policy includes five modules: (1) Policy Decision Engine which is implemented to make decisions upon data received from all connected components. Based on the requested services and the end user. It also analyzes the rules specified in the Policy Repository and makes a decision which is then enforced, (2) Application Administrator which is responsible for defining policies that provide a specific application to a particular user and ensure secure communication among multiple fog nodes, (3) Policy Resolver which follow attribute-based security procedure in which users are authenticated and identified by a set of attributes which they present, and then these attributes are analyzed by attribute finder scheme and forwarded to the policy resolver for further accumulation, after that the user access is identified based on a set of attributes defined and stored in an attribute repository, (4) Policy Repository which contains a set of rules needed by the Policy Decision Engine when making policy decisions, and (5) Policy Enforcer which resides either within a virtual instance such as fog node, fog instance or cloud datacenter, or within physical devices such as connected vehicle or mobile device.

In [1], the authors propose a communication architecture called smart gateway and its integration with fog computing. The smart gateway refers to the scheme which is responsible for preprocessing and aggregating data before it can be transferred to the cloud to reduce the load on the core cloud. The authors then test their architecture in terms of different delay aspects, including upload and bulk-data upload, synchronization and bulk-data synchronization, and jitter delays. They showed that using smart gateway-based communication with fog computing helps cloud to provide improved services and make it easier and faster to deliver services to applications that require low latency.

In [40], the authors discuss the main advantages that can be achieved when deploying fog computing platform. They also provide an analysis of some application scenarios that can benefit from deploying such a platform. These scenarios include smart grid, smart traffic lights in vehicular networks, and software-defined networks. Then, the authors discuss some security issues within fog computing and provide a case study to illustrate these issues. They studied the man-in-the-middle attack and how it can affects the performance of fog devices such as CPU utilization and memory consumption. Experiments show that man-in-the-middle attack can be very stealthy within fog computing, since it has just a little increase in the CPU utilization and the memory consumption.

3 Proposed framework

In this paper, we propose an IoT-based framework that provides an automated video surveillance solution. As shown in Fig. 1, the framework supports a set of camera sensor groups. Each group is connected to an MEC server that is colocated with the base station that this group of cameras are connected to. This group of cameras can use the communication technology supported by the base station. A set of these groups are connected through the Internet to a central cloud server. The cameras capture and stream videos to the MEC servers. Each MEC server stores the video locally and then forwards it to the central cloud server. The MEC server may keep the video locally until it guarantees the reception of the video by the cloud and it then can delete it.

The framework provides a reliable surveillance video storage, on the central cloud server, with an optimal quality that fully utilizes the total central cloud provided bandwidth. Bandwidth management and allocation is done in two parts. Within the cell, and among the cameras in the group connected to the same base station, the bandwidth management and allocation are done by the MEC server colocated with

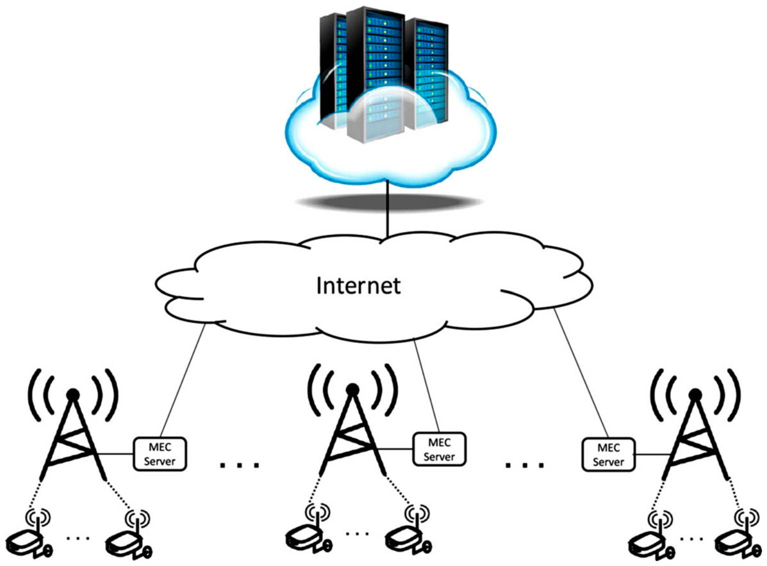


Fig. 1 The proposed framework

that base station. Globally, among the MEC servers, the cloud server distributes the available bandwidth among the cells.

3.1 Global bandwidth distribution

The global bandwidth distribution is done periodically by the cloud. Assume that B represents the total bandwidth budget, provided by a cloud service provider, to be used by a surveillance system. Assume also that M_i represents the ith MEC connected to the cloud, where $i = 1, 2, \ldots, N$ and N is the total number of MECs connected to the cloud. Moreover, assume that c_i is the number of cameras connected to the MEC M_i.

The simplest way to calculate the bandwidth for the ith MEC, B_i, is to divide the total bandwidth provided by cloud (B) by the number of connected MECs to the cloud (N) as shown in:

$$B_i = \frac{B}{N}. \tag{1}$$

This distribution achieves fairness among the areas monitored by different cells. We call this distribution, simple global distribution (SD). As we can see, the only information needed for this distribution is N which is the number of connected MECs. This calculation can be done by the cloud server and a message is then sent broadcasted to all the MEC servers with the total bandwidth that they should use locally.

Another option for bandwidth distribution is to divide the total bandwidth among the MECs, taking into consideration the number of cameras connected to each MEC. Equation 2 shows this calculation:

$$B_i = \frac{B}{\sum_i^N c_i} \times c_i. \qquad (2)$$

We call this the weighted global distribution (WD). The term $\frac{B}{\sum_i^N c_i}$ calculates the bandwidth share of each camera in the system. This term can be calculated by the cloud server after receiving the number of connected cameras c_i form each M_i. The value is then broadcasted to all participating MECs. Each MEC then can calculate its share form the total bandwidth by multiplying the received term value by the number of cameras connected to the it.

3.2 Local bandwidth distribution

Once each MEC identifies its bandwidth share B_i, it distributes B_i among the connected cameras in the MEC cell. One can say that the distribution here is very simple. It can be just dividing the MEC bandwidth by the number of cameras connected to the MEC. The problem here is much more complicated than that because of many issues that should be taken into consideration. These issues are as follows:

- The cell physical bit rate. This is the maximum bit rate (bandwidth) supported by the cell. This bit-rate depends on the communication technology used in the cell.
- The effective bandwidth in the cell. It is the bandwidth that can be used to deliver the surveillance video streams to the MEC server. In the network, a fraction of the physical bandwidth is consumed by communication protocol overhead. Also another fraction is consumed by communication errors. Moreover, another fraction is consumed by cross traffic in the network. Obviously, the effective bandwidth is much smaller than the physical bandwidth in any network. The effective bandwidth should be always larger then the bandwidth share of the MEC, so that the MEC can utilize all its bandwidth share.
- Number of cameras connected. This number may change over time because some cameras may become defective or out of battery life.
- Type of video compression used by the camera. Most surveillance cameras support three compression algorithms: MJPEG, MPEG4, and H.264.[2] MJPEG takes a quality factor as an input and usually produces better quality video steam with higher rate than the other two compression. The higher the quality factor, the higher the video quality and bit rate. MPEG4 and H.264 takes a bit-rate as an input and produces a video stream with that bit-rate. H.264 provide more compression rate than that of MPEG4.
- How to force the bandwidth distribution at the cameras.

The work in [5] proposes an optimal solution that can be used to distribute the available bandwidth in a wifi-based cell over a set of cameras to minimize the received video distortion at a central station. In this paper, we consider this as a case study. The central station can be replaced by the MEC server. The solution proposed in [5] is summarized in the following section.

[2] http://www.cctvsound.com/news/8/MJPEG-MPEG4-or-H-264-which-one-is-better.

3.2.1 Case study: wifi-based local bandwidth distribution

The study in [5] introduces enhanced distortion optimization (EDO) that manages multiple video sources that stream live videos to a central station. EDO provides a dynamic cross-layer optimization solution that distribute and allocate the network bandwidth among the multiple video sources to minimize the overall distortion of the video streams received by the central station. EDO achieves the bandwidth allocation by dynamically controls the application rate and the link layer parameters in the streaming sources. In building EDO, the bandwidth distribution problem is formulated as a cross-layer optimization problem based the sum of the distortion of all video streams received by the proxy station. In this formulation, the solution is the set of the video sources airtime fractions. The mathematical formulation of the problem was as shown in the following equations:

$$F^* = \arg\min \sum_{s=1}^{S} \text{Distortion}(r_s) \tag{3a}$$

such that:

$$\sum_{s=1}^{S} f_s = A_{\text{eff}} \tag{3b}$$

$$r_s = f_s \times y_s \tag{3c}$$

$$0 \leq f_s \leq 1 \tag{3d}$$

$$s = 1, 2, 3, \ldots, S, \tag{3e}$$

where F^* is the set of optimal airtime fractions (f_s) of all video sources, A_{eff} is the total effective airtime which is an estimation of the effective bandwidth in the network, and r_s and y_s are the optimal application rate and the physical rate of each source (s), respectively.

To find a solution for the previous formulation, a model is needed for the relationship between the video distortion and the video bit rate. In addition, an estimation is needed for the effective airtime in the network. The work in [5] characterizes the relationship between the distortion and the video bit rate as shown in:

$$\text{Distortion (RMSE)} = a \times (Z)^b + c, \tag{4}$$

where Z is the average video frame size and $a, b,$ and c are constants. Since the framework uses MJPEG video streams, Z is calculated as $Z = \frac{R}{\tau}$, where R is the video playback rate and τ is the video frame rate.

The work in [5] also uses an online estimation algorithm to estimate the network effective airtime, as shown in Algorithm 1. The algorithm is run on the central proxy station and measure the sum of the ratios of the data dropping to the physical rate of each node. All information needed by the central proxy station about the video sources are sent periodically by the sources to the central proxy station. This value,

A_δ, gives an indication on the performance of the network, and the algorithm adapts the estimated effective airtime A_{eff} to get A_δ very close to pre-specified threshold A_{thresh}. The algorithm is stopped when the difference between A_δ and A_{thresh} becomes very small.

Algorithm 1 Simplified dynamic effective airtime estimation algorithm

```
if this is the first time to run the algorithm
    $A_{eff} = \sum_{s=1}^{S} \frac{t_s}{y_s}$ ;
At the end of each estimation period{
    $A_\Delta = \sum_{s=1}^{S} \frac{d_s}{y_s}$ ;
    if ($A_\Delta < A_{thresh}$){
        if (last operation was decrement){
            $I = 0.5 * lastDecrement$;
            $A_{eff} = A_{eff} + I$;}
        else if (last operation was increment){
            $I = 0.5 * lastIncrement$;
            $A_{eff} = A_{eff} + I$;}
        else //no decrements happened before
            $A_{eff} = A_{eff} + 0.5$; //keep increasing $A_{eff}$ to cause the first decrement
        if ($lastIncrement < I_{thresh}$)
            Stop the estimation algorithm;
    }
    else if ($A_\Delta \geq A_{thresh}$){
        $A_{eff} = A_{eff} - (A_\Delta - A_{thresh})$;
        $lastIncrement = A_\Delta - A_{thresh}$;}
}
```

The problem formulated in (3) was solved using the Lagrangian relaxation technique [31]. The Lagrangian-relaxed formulation for this problem was formulated as follows:

$$L(F^*, \lambda) = \sum_{s=1}^{S} (a_s (f_s y_s/\tau)_s^b + c_s) + \lambda \left(\sum_{s=1}^{S} f_s - A_{\text{eff}} \right). \tag{5}$$

Then, the Lagrangian conditions were formulated and solved to obtain the equations for both λ and f_s as shown in the following equations:

$$\lambda = \left(\frac{A_{\text{eff}}}{\sum_{s=1}^{S} (\frac{-\tau_s}{a_s b_s y_s (y_s/\tau_s)^{(b_s-1)}})^{(1/(b_s-1))}} \right)^{(b_s-1)} \tag{6}$$

$$f_s = \left(\frac{-\lambda \times \tau_s}{a_s b_s y_s (y_s/\tau_s)^(b_s - 1)} \right)^{(1/(b_s-1))}. \tag{7}$$

A_{eff} in the formulation in Eq. 3, and that is estimated by Algorithm 1, represents the effective bandwidth fraction of the physical bit rate in the cell network. It follows that to calculate the effective bandwidth in the cell network, we can use:

$$B_{\text{eff}} = Y \times A_{\text{eff}}, \tag{8}$$

where Y is the physical bit rate in the cell network. B_{eff} should be compared to the bandwidth assigned to the MEC, B_i. If B_{eff} is larger than B_i, we should force the cell network to use only B_i. This can be achieved by applying Eq. 9 after finishing the estimation in Algorithm 1:

$$A_{\text{eff}} = \left\{ \begin{array}{ll} \frac{B_i}{Y} & \text{if } B_i < B_{\text{eff}} \\ A_{\text{eff}} & \text{if } B_i \geq B_{\text{eff}} \end{array} \right\}. \tag{9}$$

λ is then calculated at the MEC server using Eq. 6 and broadcasted to all video sources. All information needed to calculate λ are sent periodically by the sources to the MEC server. Once a video source receives λ, it calculates its f_s using Eq. 7 and then it uses Eq. 3c to calculate its application rate. Moreover, to ensure that each video source does not internally drop packets, the link layer parameter is also managed dynamically in such a way that the data received from the application layer is transmitted accordingly, once a channel access is gained.

3.3 Possible enhancements

As we can see in Eq. 9 that if B_i is greater than B_{eff}, nothing is being done and the bandwidth difference between B_i and B_{eff} is a cloud bandwidth is waisted without being utilized. The cloud bandwidth can also be waisted if the cloud-MEC link bandwidth is smaller than B_i. In the later case, many data would be lost or a huge delay can happen in communicated the video stream from the MEC to the cloud.

The first problem can be solved by making the MEC that has a smaller B_{eff} to inform the cloud with the maximum bandwidth that it can support, so that the cloud will not assign more bandwidth to it.

The second problem can be solved by making the cloud estimate the cloud-MEC link bandwidth during a period of time. The cloud should use this estimation into consideration when it distributes the available bandwidth over the participating MECs for the next period. This assures that the bandwidth is always fully utilized. The estimation should be done while the video stream is being sent from the MEC to the cloud without sending any extra data.

4 Evaluation

Several experiments were conducted using NS-3 [42] simulator version 3.22. We have extended the simulation framework that have been built in [6] and [30] to simulate the proposed framework.

We construct and stream real MJPEG videos from the video sources to the MEC and then to the cloud over the simulated network. We use the approach used in [5] to achieve that. The MJPEG video stream are sent as Real-time Transport Protocol (RTP) Packets. The MJPEG streams are constructed using the CMU/MIT [11] and the Georgia Tech [17] image databases. The frames in each video stream are randomly selected from those image databases and packetized as RTP packets and are sent over cell network to the MEC server. The MEC server then forwards those packets to the cloud server. The application layer at the cloud server, upon receiving all the RTP packets of a video frame, reassemble the RTP packets a to form the video frame. An error concealment algorithm [39] is used to reconstruct the video frames with lost RTP packets.

The network topologies used in our experiments are constructed using 5, 10, 15, and 20 cells. Each cell uses 802.11g wifi standard. Each cell has an MEC server that manages several video sources.

Two scenarios regarding the cloud-MEC link bandwidth are considered in this paper. We call the first one *The Relaxed Scenario*, and the second one *The Limited Scenario*. In *The Relaxed Scenario*, the number of video sources in each cell is chosen randomly, between 5 and 25 video sources, and the bandwidth of the links that exist between the MECs and the cloud, is randomly selected between 4 and 100 Mbps. *The Limited Scenario* is used to represent a more congested topology in which the bandwidth of the Cloud-MEC link is limited. In this scenario, the number of video sources in each cell is chosen randomly between 5 and 30 video sources and Cloud-MEC link bandwidth is randomly generated between 4 and 24 Mbps.

The random selection of Cloud-MEC link bandwidth is to represent different link types.

In our experiments, we tested four variations of our framework for both scenarios (Relaxed and Limited).

- Local management (LM). In this variation, the framework only performs bandwidth management within the cell. Each MEC server send as much data as it wishes to the cloud. The cloud will only receives data according to its budget which causes many data to be lost. This variation shows the importance of the global bandwidth management.
- Local management with global simple distribution (LM-SD). In this variation, we use local management with the global simple bandwidth distribution (SD) described in Sect. 3.1.
- Local management with global weighted distribution (LM-WD). In this variation, we use local management with the global weighted bandwidth distribution (WD) described in Sect. 3.1.
- Local management with an enhanced global weighted Distribution (LM-WD-Enhanced). In this variation, we use local management with the global weighted bandwidth distribution (WD) described in Sect. 3.1. We further enhanced this variation by incorporating the enhancements descried in Sect. 3.3. To achieve the proposed enhancements, the MECs send a control message to the cloud informing it about its B_{eff}. Also, the cloud estimates the link bandwidth between the MEC and the cloud. The control messages and the bandwidth estimation should be executed periodically. We choose period length to be 5 min.

Table 1 summarizes the main simulation parameters.

4.1 Performance metrics

In this paper, we use the following performance metrics:

Table 1 Main simulation parameters used in the AVS simulation framework

Parameter	Model/value(s)
Number of streaming sources	Random (5–25)
Number of cells	5, 10, 15, 20
Source start time	Random (1–5) s
Simulation time	10 min
Video frame rate	20 fps
Application packet size	1024 bytes
Application rate	Optimized, default = network physical rate/s
Physical characteristics	IEEE 802.11g
Physical data rate	54 Mbps
Buffer size	256 packets
Beacon interval	0.02 s
State report interval	2 s
Long retry limit	4
Short retry limit	7
CWmin, CWmax, AIFS	Default, described in the standard
Video TXOP limit	Optimized, Default = 3008 µs

- **Peak signal-to-noise ratio (PSNR)** is one of the most used metrics to assess the video transmission QoS at the application level. PSNR is described by the International Telecommunication Union [36] as:

$$\mathrm{PSNR}(n)_{\mathrm{db}} = 20\ \log\left[\frac{V_{\mathrm{peak}}}{\sqrt{\mathrm{MSE}(n)}}\right], \tag{10}$$

  where $V_{\mathrm{peak}} = 2^k - 1$, k represent the number of bits used to represent the pixel. Mean square error (MSE) is the error variance estimation, and MSE value is calculated as:

$$\mathrm{MSE}(n) = \frac{\sum_{i=1}^{N_{\mathrm{col}}} \sum_{j=1}^{N_{\mathrm{row}}} [Y_{\mathrm{S}}(n,i,j) - Y_{\mathrm{D}}(n,i,j)]^2}{N_{\mathrm{col}} N_{\mathrm{row}}}, \tag{11}$$

  where N_{row} and N_{col} are the number of rows and columns in the image, i and j are the current position of column and row, n is the number of the current frame, Y_{S} is the luminous component of the source image, and Y_{D} is the luminous component of the destination image as defined in [23].
- **The overall received data rate** it is the total amount of data received at the cloud server during a specific period time from all cameras averaged over time. It is calculated as the total size of the packets received from all cameras divided by the time in which the data is collected. In our experiments, the overall received data rate is calculated for each second. Eventually, these values are averaged to calculate the overall average.

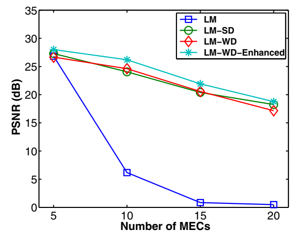
- **Packet delay** the amount of time needed for a packet to be transmitted along its entire path in the network. Packet delay is caused by (a) processing delay, which is the time required to analyze, process, and decide where the packet will be sent, (b) buffer delay, which is the time that a specific packet stay in the queue until it is dequeued to be transmitted, (c) transmission delay, which is the total time in which all the packet bits are pushed to the desired transmission medium, and (d) propagation delay, which is the time duration for the bit to propagate across the network until it reaches the end of its physical trajectory. In calculating the average delay, in each experiment, we use the time difference between the time at which the packet is received at the cloud server application layer and the time when the packet was sent from the camera application layer. At the end of each second, the average delay of all packets received within that second is calculated and then the average of these averages is calculated.
- **Overall network load** it is the total traffic (data) rate generated by all video sources in the network. The sending rate of each video source is calculated in the application layer as the total size of data packets sent to the AP divided by the time in which the data is collected. In our experiments, the sending rate is calculated each second and then it is stored. Then, the average sending rate is calculated for each video source as the average of the stored sending rate values. Eventually, the overall network load was calculated as sum of the average sending rates of all video sources participating in the experiment.
- **Overall retransmission dropping rate** it is the total rate of data dropped in the MAC layer due to exceeding the maximum retransmission tries. Each packet maintain a number of sending tries counter. Whenever an acknowledgment of a data packet is not received within a specific time period, the packet is considered to be lost and should be retransmitted, and the sending tries counter is increased by one. If this counter, however, exceeds a specific retries number, the packet is not retransmitted and considered as a dropped packet. The retransmission dropping rate is calculated in each video source, as the total size of data in the packets that are dropped because it exceeded retransmission tries, divided by the time in which the data is collected. In our experiments, the retransmission dropping rate is calculated each second and then it is stored. Then, the average retransmission dropping rate is calculated as the average of the stored values. Eventually, the overall retransmission dropping rate is calculated as sum of the average retransmission dropping Rate of all video sources participating in the experiment.
- **Overall buffer dropping rate** it is the total rate of the data dropped in the MAC layer due to overflow in the MAC queues. Each MAC queue has a limit on the size of data that it can store. Whenever the MAC queue is full, any data that arrives from the upper layers is dropped immediately. The buffer dropping rate is calculated in each video source, as the total size of data in the packets that were dropped (because the buffer was full) divided by the time in which the data are collected. In our experiments, the buffer dropping rate is calculated each second and then it is stored. Then, the average buffer dropping rate is calculated as the average of the stored values. Eventually, the overall buffer dropping rate is calculated as the sum of the average buffer dropping rate of all video sources in the network.

5 Results

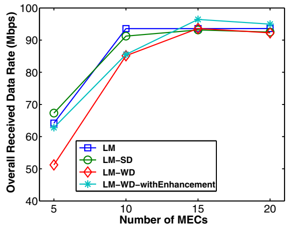
In this section, we compare the results of simulating all the variations of the proposed IoT-based video surveillance system in NS-3 for both the relaxed and the limited scenarios. The results have been compared using the performance metrics described in Sect. 4.1. The used metrics are perceptual video quality, the overall received data rate at the cloud application layer from the all of the video sources, average packet delay of the video, complete received video frames percentage, incomplete received video frames percentage, missed video frames percentage, overall network load, total buffer dropping, and total retransmission dropping rate.

Figure 2 shows the results obtained by running the simulation of the relaxed scenario with streaming videos that are constructed using the G-T image database.

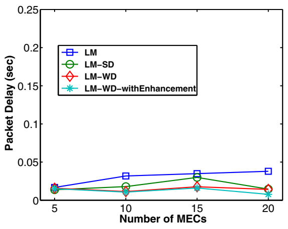
Figures 3 and 4 show the results obtained by running the simulation in the Relaxed and the Limited scenarios, respectively, with streaming videos that are constructed using the CMU–MIT image database.



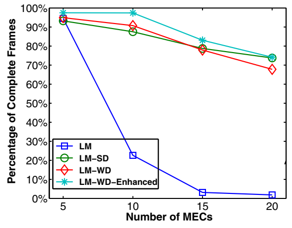
(a) Perceptual Video Quality



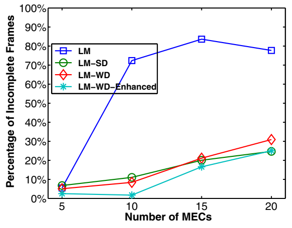
(b) Overall Received Data Rate



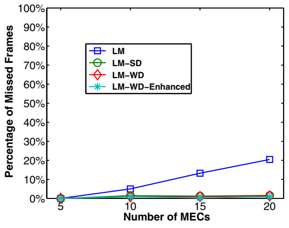
(c) Packet Delay



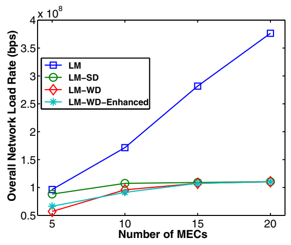
(d) Complete Video Frames



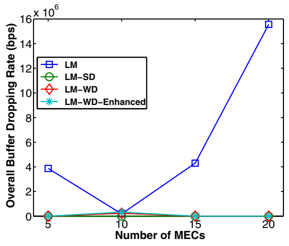
(e) Incomplete Video Frames



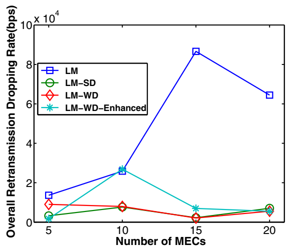
(f) Missed Video Frames



(g) Overall Network Load

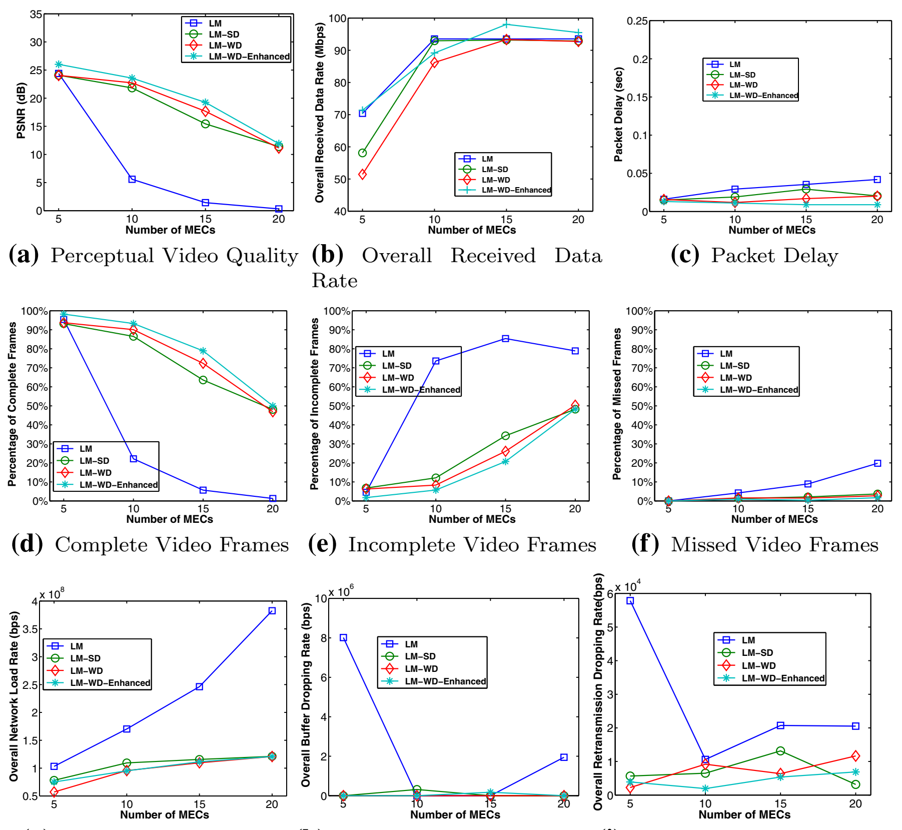


(h) Overall Buffer Dropping Rate



(i) Overall Retransmission Dropping Rate

Fig. 2 Comparison of the variations of the proposed IoT-based framework (G-T image set, relaxed scenario)



(a) Perceptual Video Quality **(b)** Overall Received Data Rate **(c)** Packet Delay

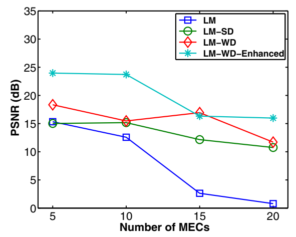
(d) Complete Video Frames **(e)** Incomplete Video Frames **(f)** Missed Video Frames

(g) Overall Network Load **(h)** Overall Buffer Dropping Rate **(i)** Overall Retransmission Dropping Rate

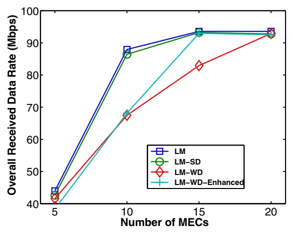
Fig. 3 Comparison of the variations of the proposed IoT-based framework (CMU-MIT image set, relaxed scenario)

These results show that the LM-SD outperforms the LM variation in terms of the important video performance metrics which are PSNR, Packet Delay, and the percentages of complete, incomplete, and missed video frames while, at the same time, requiring a lower amount of data received at the cloud. These results proofs that global management is very important to the performance of the system. In addition, the results show that the LM-WD approach provide slightly better performance when compared to the LM-SD variation. This means that taking the number of cameras into consideration in the global bandwidth solution is important to enhance the performance of the system. Furthermore, the results show that LM-WD-Enhanced outperforms all of the previous approaches since it utilizes the cooperation of the cloud and MEC server. Those results have the same trend in both the Relaxed and the Limited Scenarios. The results of the limited scenario with the G-T image set follow the same trnd and thus not shown.

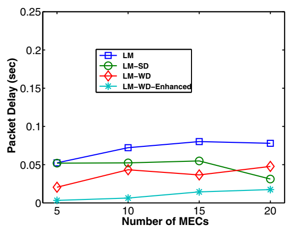
Those results show the effectiveness of the proposed IoT-based surveillance framework in its capabilities to include a large number of MECs with while maintaining high quality and lower bandwidth/storage cost at the clouds side.



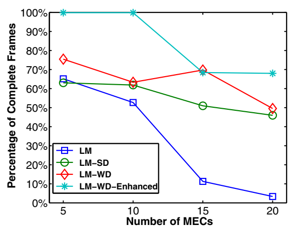
(a) Perceptual Video Quality



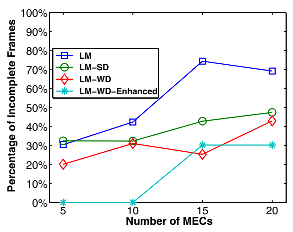
(b) Overall Received Data Rate



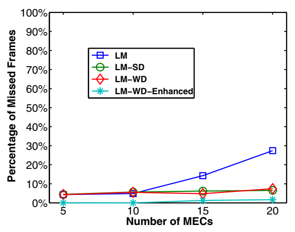
(c) Packet Delay



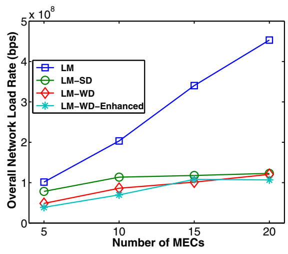
(d) Complete Video Frames



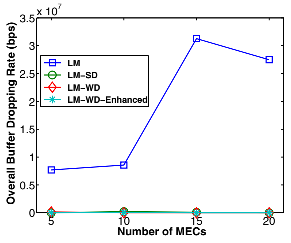
(e) Incomplete Video Frames



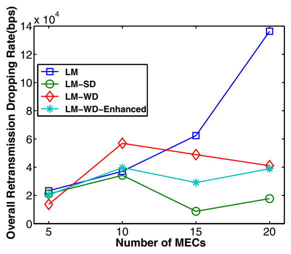
(f) Missed Video Frames



(g) Overall Network Load



(h) Overall Buffer Dropping Rate



(i) Overall Retransmission Dropping Rate

Fig. 4 Comparison of the variations of the proposed IoT-based framework (CMU-MIT image set, limited scenario)

6 Conclusion

In this paper, we presented a reliable IoT-based wireless video surveillance system. The system provides an optimal bandwidth distribution and allocation to minimize the overall surveillance video distortion. We exploited two new emerging technologies to provide a reliable data transfer medium using mobile edge technology and sufficient capacity for data processing and storage using cloud. The proposed surveillance system is evaluated using NS-3 simulation. The results show that the proposed framework fully utilizes the available cloud bandwidth budget and achieves high scalability. Results can be summarized as follows.

- Global management that is conducted by the cloud is very important to the performance of the system.
- Taking the number of cameras, connected to each MEC, into consideration in the global bandwidth management is important to enhance the performance of the system.

- The cooperation of the MEC and the cloud server can enhance the system performance further.

As a future work, we plan to study the system using different video streaming technology. We also plan to propose and study an extension of our framework to secure the video stream communication.

Acknowledgements This work is funded in part by the Jordan University of Science and Technology Deanship of Research Grant Number 20150310.

References

1. Aazam M, Huh EN (2014) Fog computing and smart gateway based communication for cloud of things. In: 2014 International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, pp 464–470
2. Agarwal S, Dunagan J, Jain N, Saroiu S, Wolman A, Bhogan H (2010) Volley: automated data placement for geo-distributed cloud services. In: NSDI, pp 17–32
3. Ali AMM, Ahmad NM, Amin AHM (2014) Cloudlet-based cyber foraging framework for distributed video surveillance provisioning. In: 2014 Fourth World Congress on Information and Communication Technologies (WICT), pp 199–204. doi:10.1109/WICT.2014.7076905
4. Alletto S, Cucchiara R, Fiore GD, Mainetti L, Mighali V, Patrono L, Serra G (2016) An indoor location-aware system for an iot-based smart museum. IEEE Internet Things J 3(2):244–253. doi:10.1109/JIOT.2015.2506258
5. Alsmirat M, Sarhan N (2012) Cross-layer optimization and effective airtime estimation for wireless video streaming. In: 2012 21st International Conference on Computer Communications and Networks (ICCCN), pp 1–7. doi:10.1109/ICCCN.2012.6289275
6. Alsmirat MA, Jararweh Y, Obaidat I, Gupta BB (2016) Automated wireless video surveillance: an evaluation framework. J Real-Time Image Process, pp 1–20. doi:10.1007/s11554-016-0631-x
7. Beck MT, Werner M, Feld S, Schimper S (2014) Mobile edge computing: a taxonomy
8. Bellavista P, Cardone G, Corradi A, Foschini L (2013) Convergence of manet and wsn in iot urban scenarios. IEEE Sensors J 13(10):3558–3567. doi:10.1109/JSEN.2013.2272099
9. Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. ACM, pp 13–16
10. Chae H, Park J, Song H, Kim Y, Jeong H (2015) The iot based automate landing system of a drone for the round-the-clock surveillance solution. In: 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp 1575–1580. doi:10.1109/AIM.2015.7222767
11. Cmu/mit image set. http://vasc.ri.cmu.edu/idb/html/face/frontal_images/ (online; accessed Nov 2015)
12. Cuervo E, Balasubramanian A, Cho DK, Wolman A, Saroiu S, Chandra R, Bahl P (2010) Maui: making smartphones last longer with code offload. In: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services. ACM, pp 49–62
13. Davis A, Parikh J, Weihl WE (2004) Edgecomputing: extending enterprise applications to the edge of the internet. In: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers and Posters. ACM, pp 180–187
14. Dsouza C, Ahn GJ, Taguinod M (2014) Policy-driven security management for fog computing: preliminary framework and a case study. In: 2014 IEEE 15th International Conference on Information Reuse and Integration (IRI). IEEE, pp 16–23
15. Faruque MAA, Vatanparvar K (2016) Energy management-as-a-service over fog computing platform. IEEE Internet Things J 3(2):161–169. doi:10.1109/JIOT.2015.2471260
16. Fleming K, Waweru P, Wambua M, Ondula E, Samuel L (2016) Toward quantified small-scale farms in africa. IEEE Internet Comput 20(3):63–67. doi:10.1109/MIC.2016.58
17. Georgia tech face database. http://www.anefian.com/research/face_reco.htm (online; accessed Nov 2015)
18. Gope P, Hwang T (2016) Bsn-care: a secure iot-based modern healthcare system using body sensor network. IEEE Sensors J 16(5):1368–1376. doi:10.1109/JSEN.2015.2502401

19. Hassanalieragh M, Page A, Soyata T, Sharma G, Aktas M, Mateos G, Kantarci B, Andreescu S (2015) Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: opportunities and challenges. In: 2015 IEEE International Conference on Services Computing (SCC), pp 285–292. doi:10.1109/SCC.2015.47
20. Islam SMR, Kwak D, Kabir MH, Hossain M, Kwak KS (2015) The internet of things for health care: a comprehensive survey. IEEE Access 3:678–708. doi:10.1109/ACCESS.2015.2437951
21. Jararweh Y, Doulat A, AlQudah O, Ahmed E, Al-Ayyoub M, Benkhelifa E (2016) The future of mobile cloud computing: integrating cloudlets and mobile edge computing. In: 2016 23rd International Conference on Telecommunications (ICT), pp 1–5. doi:10.1109/ICT.2016.7500486
22. Jararweh Y, Doulat A, Darabseh A, Alsmirat M, Al-Ayyoub M, Benkhelifa E (2016) Sdmec: software defined system for mobile edge computing. In: 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), pp 88–93. doi:10.1109/IC2EW.2016.45
23. Ke CH, Shieh CK, Hwang WS, Ziviani A et al (2008) An evaluation framework for more realistic simulations of mpeg video transmission. J Inf Sci Eng 24(2):425–440
24. Kemp R, Palmer N, Kielmann T, Bal H (2012) Cuckoo: a computation offloading framework for smartphones. In: Mobile computing, applications, and services. Springer, Berlin, pp 59–79
25. Khoi NM, Saguna S, Mitra K, Ahlund C (2015) Irehmo: an efficient iot-based remote health monitoring system for smart regions. In: 2015 17th International Conference on e-health Networking, Application Services (HealthCom), pp 563–568. doi:10.1109/HealthCom.2015.7454565
26. Kokkonis G, Psannis KE, Roumeliotis M, Schonfeld D (2016) Real-time wireless multisensory smart surveillance with 3d-hevc streams for internet-of-things (iot). J Supercomput, pp 1–19. doi:10.1007/s11227-016-1769-9
27. Kumarage H, Khalil I, Alabdulatif A, Tari Z, Yi X (2016) Secure data analytics for cloud-integrated internet of things applications. IEEE Cloud Comput 3(2):46–56. doi:10.1109/MCC.2016.30
28. Liu Z, Yan T (2013) Study on multi-view video based on iot and its application in intelligent security system. In: Proceedings of 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), pp 1437–1440. doi:10.1109/MEC.2013.6885292
29. Moness M, Moustafa AM (2016) A survey of cyber-physical advances and challenges of wind energy conversion systems: prospects for internet of energy. IEEE Internet Things J 3(2):134–145. doi:10.1109/JIOT.2015.2478381
30. Obaidat I, Alsmirat MA, Jararweh Y (2016) Completing ieee 802.11e implementation in ns-3. In: 2016 7th International Conference on Information and Communication Systems (ICICS), pp 190–195. doi:10.1109/IACS.2016.7476109
31. Ortega A, Ramchandran K (1998) Rate-distortion methods for image and video compression. IEEE Signal Process Mag 15(6):23–50
32. Pan J, Jain R, Paul S, Vu T, Saifullah A, Sha M (2015) An internet of things framework for smart energy in buildings: designs, prototype, and experiments. IEEE Internet Things J 2(6):527–537. doi:10.1109/JIOT.2015.2413397
33. Psannis K (2009) Efficient redundant frames encoding algorithm for streaming video over error prone wireless channels. IEICE Electron Express 6(21):1497–1502. doi:10.1587/elex.6.1497, http://ci.nii.ac.jp/naid/130000121968/en/
34. Psannis K, Ishibashi Y (2008) Enhanced h. 264/avc stream switching over varying bandwidth networks. IEICE Electron Express 5(19), 827–832
35. Psannis KE, Ishibashi Y (2006) Impact of video coding on delay and jitter in 3g wireless video multicast services. EURASIP J Wirel Commun Netw 2006(2):51–51. doi:10.1155/WCN/2006/24616
36. Recommendation 500-10: Methodology for the subjective assessment of the quality of television pictures. ITU-R Recommendation BT.500-10 (2000)
37. Ryu M, Yun J, Miao T, Ahn IY, Choi SC, Kim J (2015) Design and implementation of a connected farm for smart farming system. In: SENSORS, 2015 IEEE, pp 1–4 (2015). doi:10.1109/ICSENS.2015.7370624
38. Sajid A, Abbas H, Saleem K (2016) Cloud-assisted iot-based scada systems security: a review of the state of the art and future challenges. IEEE Access 4:1375–1384. doi:10.1109/ACCESS.2016.2549047
39. Shirani S, Kossentini F, Kallel S, Ward R (1997) Reconstruction of jpeg coded images in lossy packet networks. IEEE Trans Commun **(submitted)**
40. Stojmenovic I, Wen S (2014) The fog computing paradigm: scenarios and security issues. In: 2014 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE, pp 1–8

41. Taneja M (2015) A framework to support real-time applications over ieee802.15.4 dsme. In: 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pp 1–6. doi:10.1109/ISSNIP.2015.7106918
42. The network simulator ns-3. http://www.nsnam.org/
43. Trappe W, Howard R, Moore RS (2015) Low-energy security: limits and opportunities in the internet of things. IEEE Security Privacy 13(1):14–21. doi:10.1109/MSP.2015.7
44. Wang WQ, Zhang X, Zhang J, Lim HB (2012) Smart traffic cloud: an infrastructure for traffic applications. In: 2012 IEEE 18th International Conference on Parallel and Distributed Systems (ICPADS). IEEE, pp 822–827
45. Wang Y, Chen R, Wang DC (2015) A survey of mobile cloud computing applications: perspectives and challenges. Wirel Pers Commun 80(4):1607–1623
46. Yang G, Xie L, Mntysalo M, Zhou X, Pang Z, Xu LD, Kao-Walter S, Chen Q, Zheng LR (2014) A health-iot platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box. IEEE Trans Ind Inf 10(4):2180–2191. doi:10.1109/TII.2014.2307795
47. Yeh LY, Chiang PY, Tsai YL, Huang JL (2015) Cloud-based fine-grained health information access control framework for lightweight iot devices with dynamic auditing and attribute revocation. IEEE Trans Cloud Comput PP(99), 1. doi:10.1109/TCC.2015.2485199
48. Zhang T, Chowdhery A, Bahl PV, Jamieson K, Banerjee S (2015) The design and implementation of a wireless video surveillance system. In: Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, MobiCom ’15. ACM, New York, pp 426–438. doi:10.1145/2789168.2790123