Journal of Network and Computer Applications 66 (2016) 236–249

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Big data analytic architecture for intruder detection in heterogeneous wireless sensor networks

Suvendu Kumar Mohapatra [a], Prasan Kumar Sahoo [b], Shih-Lin Wu [b,c,*]

[a] Department of Electrical Engineering, Division of Computer Science and Information Engineering, Chang Gung University, Kwei-Shan 333, Taiwan, Republic of China
[b] Department of Computer Science and Information Engineering, Chang Gung University, Kwei-Shan 333, Taiwan, Republic of China
[c] Center for Biomedical Engineering, Chang Gung University, Kwei-Shan 333, Taiwan, Repubic of China

ARTICLE INFO

Article history:
Received 28 August 2015
Received in revised form
14 December 2015
Accepted 7 March 2016
Available online 10 March 2016

Keywords:
WSN
Barrier coverage
Intruder detection
Big data
Spark

ABSTRACT

Barrier coverage in Wireless Sensor Networks (WSNs) is an important research issue as intruder detection is the main purpose of deploying wireless sensors over a specified monitoring region. In WSNs, excessive volume and variety of sensor data are generated, which need to be analyzed for accurate measurement of the image in terms of width and resolution. In this paper, a three layered big data analytic architecture is designed to analyze the data generated during the construction of the barrier and detection of the intruder using camera sensors. Besides, a cloud layer is designed for storing the analyzed data to study the behavior of the intruder. In order to minimize the number of camera sensors for constructing the barrier, algorithms are designed to construct the single barrier with limited node mobility and the barrier path Quality of Sensing (QoS) is maintained with a minimum number of camera sensors. Simulation results show that our algorithms can construct 100% of the barrier with fewer number of camera sensors and average data processing time can be reduced by using parallel servers even if for larger size of data.

1. Introduction

Advanced development and improvement in Micro-Electro-Mechanical Systems (MEMS) technology, mixed with low power, small in size, minimum cost sensors can be equipped in Wireless Sensor Network (WSN). Now-a-days various types of sensors such as microwave sensors, thermal sensors, laser sensors and camera sensors are available according to the applications and working environments. Those sensors are static or having limited mobility (Dantu et al., 2005; El-Moukaddem et al., 2013; Janansefat et al., 2013; Chellappan et al., 2007) to gather and process environmental information. Camera sensors are different from the general sensors in terms of image capture capability and are used for number of applications such as border surveillance (Tao et al., 2012; Cheng and Tsai, 2012), and intruder detection (Keung et al., 2012; Sahoo et al., 2013). Area, point and barrier coverage are critical coverage issues in WSN, and are the parameters to appraise the quality of surveillance. In this paper, we are interested to focus on the barrier coverage problem. Barrier coverage (Shih et al., 2010) is the line coverage to cover all possible crossing paths of the intruder within the monitoring region. In directional camera sensor (Tao et al., 2012; Cheng and Tsai, 2012; Wang and Cao, 2012), the sensing range is an arc having a field of view with a finite angle and the intruder is picked up within the arc range. However, previous barrier construction protocols do not consider the mobility of the sensors, camera rotation and Quality of Measurement altogether. Hence, we design here an efficient barrier construction algorithm by considering all the above parameters.

In our barrier construction algorithm, we use microwave sensors (Dual Technology Motion Sensor) and directional camera sensors. Microwave sensors detect the movement of the intruders whereas camera sensors are used to identify the image of an intruder. Intruder detection (Keung et al., 2012; Sahoo et al., 2013) is a part of border surveillance in which the intruder is captured by the sensors which are present along the barrier line. Once the barrier network is established, colossal amount of streaming data are generated by the camera sensors, which is difficult to handle and analyze using the traditional data processing platforms. Therefore, big data analytic platform is used here to process the gigantic image data.

In order to analyze the data generated by the camera sensors, big data is the best solution to manage those unstructured streaming data with a cost effective manner, which is very decisive in terms of volume, velocity, variety and value. Volume handles

* Corresponding author.
E-mail addresses: d0121007@stmail.cgu.edu.tw (S.K. Mohapatra), pksahoo@mail.cgu.edu.tw (P.K. Sahoo), slwu@mail.cgu.edu.tw (S.-L. Wu).

http://dx.doi.org/10.1016/j.jnca.2016.03.004
1084-8045/

enormous amount of data generated continuously by many different camera sensors. Velocity focuses on the tremendous speed at which the camera sensor data (bytes) are coming for processing. Variety defines that the diversified data format arrives from various sources. Value represents the meaningful information by converting the data insights. In the industry level data scale and cutting-edge network technology, new challenges force the researchers as well as developers to improve in solutions for data collection, transmission, processing and storage. Virtualization technology acts as a backbone of various big data analysis tools such as Hadoop, where chunks of data are processed in parallel. To support such batch processing parallel execution, Hadoop MapReduce framework (Dean and Ghemawat, 2008; Yang and Chen, 2015) is used. However, the Spark platform (Zaharia et al., 2012) is employed for the realtime coordinated streaming data processing for the intruder detection. A cluster of slave nodes are used in the data analysis with complex work flows, which are controlled by the Spark master nodes. The analyzed data are stored in Cassandra (Lakshman and Malik, 2010) distributed database in the data centers.

1.1. Motivations

Barrier construction and intruder detection using wireless camera sensor network is highly essential and is convenient in surveillance system. However, to the best of our knowledge no barrier construction protocol proposed so far considers all the three functionalities such as node mobility, rotation of the camera sensors and Quality of Measurement of WSN to detect the intruder efficiently. Moreover, camera sensors are normally expensive and efficient detection of an intruder with a minimum number of camera sensors is a challenging research issue. Hence, the main motivation of our proposed work is to reduce the number of camera sensors for constructing the barrier to detect the intruder by combining all those three parameters. Besides, high volume of streaming data are generated from the camera sensors once the barrier is constructed, which need to be analyzed for detecting the intruder properly. However, it is very difficult to analyze those low-latency and high volume of unstructured streaming data manually or in any batch processing big data framework such as Hadoop. In Hadoop, the complete batch data must be loaded before the processing is done, which encounters the startup delay with intermediate data shuffling overhead during computation. Therefore, we propose an in-memory data processing Spark platform to handle such real-time data sets.

1.2. Contributions

The main contributions of our work can summarized as follows.

- A barrier construction mechanism in heterogeneous WSN is designed with a minimum number of camera sensors to collect the huge amount of real time image data for the analysis.
- Quality of Sensing (QoS) is maintained throughout the barrier path with a minimum number of camera sensors.
- A big data analytic architecture is designed to analyze and store the low-latency big data generated from the wireless camera sensors.
- A Graphics Processing Unit (GPU) enabled Spark cluster is proposed for the in-memory data processing and frame-by-frame analysis of the realtime visual streaming data.
- Based on the analysis of the big data in our proposed Spark platform, intruder detection mechanism is also designed.
- Our proposed algorithms can provide 100% barrier coverage with a minimum number of camera sensors.

Remainder of this paper is organized as follows. Section 2 describes the related works on barrier coverage and intruder detection using heterogeneous sensors. In Section 3, the big data analytic architecture is described. The wireless sensor layer is described in Section 4, which includes the barrier construction algorithms. Big data analytic with cloud based storage layer is described in Section 5. Simulation results are given in Section 6, and Section 7 concludes the work.

2. Related work

Comprehensive studies have been carried out on barrier coverage issues in WSN. Directional sensor networks (Tao et al., 2012) use directional sensors to construct the strong barrier. The objective was to diminish the total number of sensors and save the energy by minimizing the maximum rotation angle. However, most of the existing solutions are centralized and take longer time to detect the intruder. In distributed barrier coverage with β-QoM (Cheng and Tsai, 2012), wireless visual sensors construct the barrier by maintaining β-breadth to increase the quality of monitoring (QoM). Authors have proposed two β-breadth belt-barrier construction algorithms without rotation of the sensors, in which barrier is constructed with β-breadth. Distributed β-breadth belt-barrier construction algorithm with rotation is proposed, in which barrier is constructed by camera sensors with rotation capability. Their main contribution is to minimize the number of visual sensors by maintaining the quality, the number of sensors can still be minimized by adding limited mobility to the sensors. Also resolution factor can be calculated by maintaining the distance from the location of the intruder.

It is to be noted that mobility in camera sensor networks (Dantu et al., 2005; El-Moukaddem et al., 2013) has heightened the monitoring quality. Sensors with controlled mobility (Vecchio and Lopez-Valcarce, 2015) can enhance the deployment strategy, adaptive sampling, hole detection and repair capability and event detection mechanism can even become better. MICAbot (Janansefat et al., 2013) is inexpensive, adaptable and modular mobile robots, which are used in large scale distributed sensor networks. By using those mobile sensors, we can build the barrier network in an efficient way. Now-a-days, many camera sensors (Mehta et al., 2009) are available for constructing the wireless sensor networks to detect the intruders. In Hoseini et al. (2012), the coverage problem of three dimensional objects by enabling the tilt, zoom and pan functionalities of the camera sensors is investigated. In their proposed solution, a circular target model is used to determine the full coverage. In Chen et al. (2010a), the object coverage problem with rotating capabilities of camera sensors is explored. In order to reduce the redundant image data, they map the proposed problem to the set coverage problem.

Authors in Chow et al. (2007) have analyzed the angle coverage problem in visual sensors and propose an algorithm to achieve full view of the target. By preserving 360° angle, they propose an energy efficient algorithm, which tries to minimize the transmission cost over the network. In Zanella et al. (2014), authors have focused on a smart city vision as an application of Internet of Things (IoT). The main goal is to collect environmental data and monitor the public street light. However, the massive IoT data storage is not considered. Authors in Jiang et al. (2014) have proposed a cloud based data storage framework for both structured and unstructured data. The data are collected by sensors and RFID readers and the main advantage of this framework is to combine and extend multiple databases with Hadoop to store. But no analysis is done on this huge stored data.

Recently, many computation intensive (Bhattacharya et al., 2014) and data intensive applications like border surveillance,

Table 1
Comparison and contributions.

Related works	Barrier	QoS	Mobility	Intruder Det.	Big Data	Cloud
Robomote (Dantu et al., 2005), MRCD (El-Moukaddem et al., 2013), IACW (Vecchio and Lopez-Valcarce, 2015)			✓			
S_{BP} (Tao et al., 2012), LBCP (Chen et al., 2010b)	✓			✓		
D-Trib (Cheng and Tsai, 2012), B^3CA (Guo et al., 2014)	✓	✓		✓		
MR (Dean and Ghemawat, 2008), DATA (Yang and Chen, 2015), EO (Bhattacharya et al., 2014)					✓	✓
Our algorithm	✓	✓	✓	✓	✓	✓

digitized medical records, scientific data reports, semantic web and bioinformatics have generated a substantial amount of data, which need to be processed continuously and systematically. An effective data management and analysis technique is required for large-scale data, which is quite interesting but challenging too. Therefore, big data has drawn attention from industry, academia, scientist and government as well. However, inadequate research is not only on the quality measurement of the image in terms of width and resolution but also on the limited mobility on camera sensors. To get rid of this problem, we propose an energy efficient barrier construction algorithm where all camera sensors are having limited mobility. Also we provide a better solution for intruder detection with the help of this barrier line. Eventually, the camera sensor data are processed by the Spark big data analytic platform. Comparison of our proposed protocol with the existing works are listed in Table 1.

3. Big data analytic architecture

Recently, inadequate research have accomplished on the architectural design of big data and its analytical aspects referring to the border surveillance applications. Big data is an emerging solution for the sensor data analysis as it deals with very large sets of complex data originating from camera and microwave sensors, which become very difficult to process using traditional database management tools. We have pointed out some of the key requirements of a new approach to design the big data platform in an innovative way such that it would address the major challenges related to the big data applications in security domain with existing technologies. We are going to present a logical big data architecture in border surveillance applications that will be supported by extended cloud computing platform. In this section, we are inclined to propose a cloud based big data architecture for analyzing the camera sensor's data, which is used in the purpose of border surveillance.

A layered-wise big data analytic architecture is presented for the intruder detection and analysis as shown in Fig. 1. In order to get efficient results in terms of analysis, storage and performance in border security applications, we develop the big data architecture to integrate the wireless network of camera sensors, big data and cloud platform altogether. Accordingly, as shown in Fig. 1, the whole architecture is divided into Wireless sensor layer, Big data layer and Cloud layer. Each layer not only addresses the data flow in big data but also emphasizes on the efficient way of data collection, optimum storage and effective way of data analysis.

Wireless sensor layer is the 1st layer, where sensors are deployed to form the barrier and to collect the data time to time. This layer is further divided into wireless microwave sensor and wireless directional camera sensor sub-layers. In microwave sensor sub-layer, the barrier is constructed with the help of microwave sensors to sense the presence of any intruder. Once the microwave sensors sense the presence of any intruder, the signal is passed to the camera sensor sub-layer to activate them. Upon receiving the signal from the microwave sensors, the camera sensors are activated and are enabled to take the image of the intruder. The barrier is constructed with the help of camera sensors to capture the image of the intruders. The image captured by the camera sensors are sent to the sink for analysis. Basically, the

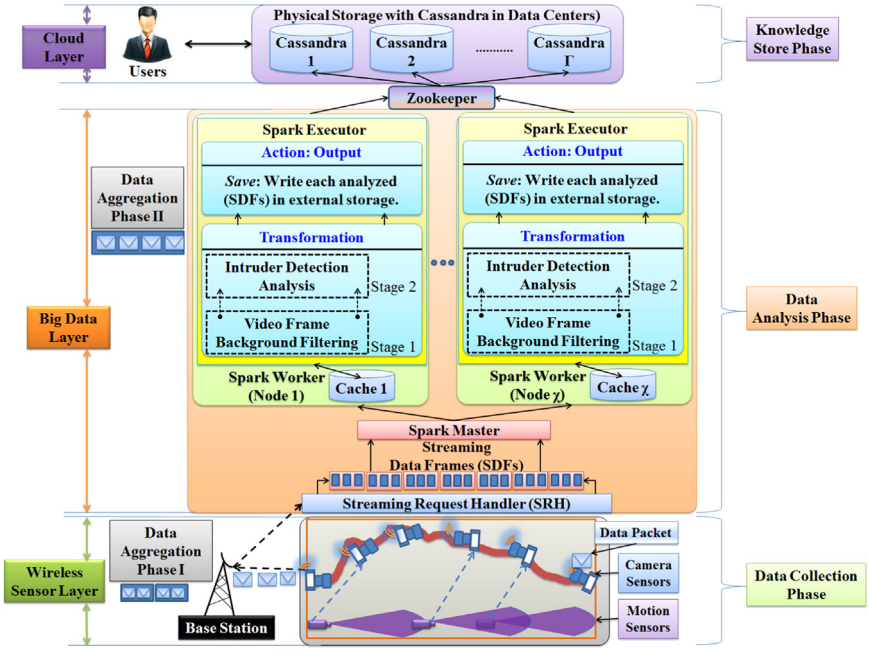

Fig. 1. Big Data analytic architecture for intruder detection.

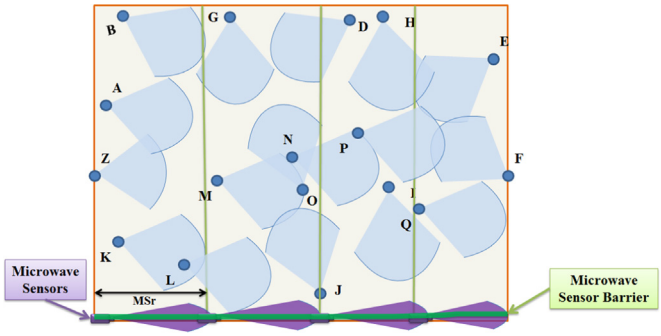

Fig. 2. Deployment of microwave motion sensors and camera sensors.

wireless sensor layer is responsible for the collection of data through the camera sensors.

Big data layer is the 2nd sub-layer of the architecture, which is responsible for the streaming data processing, analysis and identifying the intruders. In this layer, the data collected by the camera sensors are managed, analyzed and integrated so that the functionalities of the big data analysis is performed on the collected data. A Spark platform is used to process the streaming camera data, where a Streaming Request Handler (*SRH*) is used as an interface between the wireless sensor layer and big data layer to manage the flow of large volume of streaming data. The continuous streaming video frames are divided into sub-frames based on the data processing response time, which could be less than one second. In Spark, the sliced Streaming Data Frames (*SDFs*) known as Resilient Distributed Data sets (*RDDs*) are transferred to the local cache present in the Spark workers through Spark manager. Without any ambiguity, *SDFs* and *RDDs* are used interchangeably in this paper.

Let χ number of Spark workers be present in our architectural model, which are controlled by the master node. A Spark Executor is present inside the Spark worker for the task execution. *Transformation* and *Output* are two sub-modules that exist inside the Spark Executor, where *Transformation* is responsible to analyze the data sets and *Output* is used to save the processed data. *Transformation* is executed through two stages, i.e. Video Frame Background Filtering in Stage 1 and Intruder Detection Analysis in Stage 2. The filter and join operations are facilitated by Spark on the window frames present in the local memory, which is utilized for the background elimination (Lima et al., 2014) to prune large volume of unwanted data and aggregation, respectively in the stage 1. Hence, the images of the intruders are only present as the output of the stage 1 and proceed to detection phase present in stage 2 of *Transformation* module. The data saving operation is carried out on each *SDFs* for external storage purpose. However, the fault tolerant, storage and backups are controlled by the zookeeper as a coordinator present in-between the big data and cloud layer.

Finally, the 3rd layer known as the Cloud layer in the architecture is responsible for storing the analyzed data for visualization. This upper layer is extended to the physical storage layer and Cassandra distributed database, which are used for storing all types of analyzed data in the data centers (DC). However, Cassandra is also used to store the intermediate data in the local cache, which is synchronized by the zookeeper. In this cloud layer, the useful information like intruder image, detection time and location are stored in image and text formats for the future use or decision making.

Let us consider a scenario where an intruder enters into the surveillance area. At the entrance of the surveillance area, first the intruder is detected by the microwave motion sensors and the sensed signal is transmitted to make the camera sensors active. Then the detected video frames or images by the camera sensors are sent from the sensor layer to the Spark platform for processing. Hence, the intruder image analysis jobs are assigned to the respective Workers by the Spark Master present in the big data logical layer, where all the data analyses are accomplished. The analyzed data are sent to the Cassandra distributed databases present in the cloud's logical layer via Zookeeper for storage and future usage. In this fashion, the wireless sensor layer, big data layer and cloud layer collaborate with each other for data collection, analysis and storage, respectively.

4. Wireless sensor layer

The sensor layer comprises the construction of double barrier with help of camera and microwave sensors. Let us consider a Heterogeneous Wireless Sensor Network (HWSN) in which wireless camera sensors are deployed randomly over a rectangular monitoring region and microwave sensors are deployed deterministically along the border of the monitoring region as shown in Fig. 2. It is assumed that sensors are self-organized after the deployment and can be connected with each other. Each camera sensor has a finite field of view, which is different from the conventional sensors. The complete deployment method with construction of the barrier is described in the following sections.

4.1. Problem formulation

Let R be the rectangular monitoring region that comprises m number of microwave sensors $(M)=\{M_1, M_2,\ldots,M_m\}$ and n number of camera sensors $(C)=\{C_1, C_2,\ldots,C_n\}$. The microwave sensors are static and are deployed uniformly along the border of the monitoring region such that entry of any target can be first detected by them. The camera sensors have limited mobility and are deployed randomly with the Poisson distribution. Let $\lambda > 0$ be the density of the nodes and $n(R)$ be the number of camera sensors deployed over the region R.

Definition 1 (*Crossing path*). A crossing path is defined as the breadth to breadth movement of an intruder from one side to opposite side across the monitoring region. Else, it is referred to as a non-crossing path.

Definition 2 (*Length of an image* (L_i)). Length of an image i is defined as the vertical contribution of an object within the sensing range of a camera sensor.

Definition 3 (*Width of an image* (W_i)). Width of an image i is defined as the horizontal contribution of an object within the sensing range of a camera sensor.

Definition 4 (*Quality of Sensing (QoS)*). The lowest width and length of an image i that are maintained by a sensor within its sensing range is called Quality of Sensing. As shown in Fig. 3, W_i

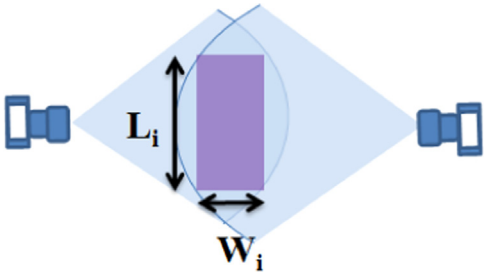

Fig. 3. Length and width of an image for QoS.

and L_i are taken as the image width and length respectively, hence $QoS = W_i \times L_i$.

Definition 5 (*Residual Energy (RE)*). Residual energy is defined as the remaining energy state of a sensor node at any point of time.

Definition 6 (*Threshold Energy (TH_E)*). Threshold energy is the minimum energy level maintained by the sensor node to perform any task.

Let radius of sensing range, radius communication range and location of ith directional camera sensor C_i be R_s, R_c and $Loc(x_i, y_i)$, respectively. The sensing region of a camera sensor is a sector in the two dimensional plane, which refers to the directional sensing model. These camera sensors have a finite field of view of angular range ($0 \le \Theta \le \frac{\Pi}{2}$), where Θ represents the offset angle of a camera sensor. The communication region of a camera sensor is a circle in two dimensional plane. It is assumed that communication range of the camera sensor (R_c) is twice of its sensing range (R_s). If sensors C_i, and C_j are located at $Loc(x_i, y_i)$, and $Loc(x_j, y_j)$, respectively, they can collaborate to detect any event within their sensing range if the following two conditions are satisfied.

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \le 2R_s \quad (1)$$

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \le R_c \quad (2)$$

Taking S_i and S_j as the sensing range of the camera sensors C_i and C_j, respectively, a barrier is constructed if the sensing range of those two overlap with each other, i.e. if $S_i \bigcap S_j \ne \Phi$, where $i \ne j$. Considering this condition, the next goal is to construct the barrier with a minimum number of camera sensors and to maintain the Quality of Sensing (QoS).

4.2. Barrier construction algorithm

In this section, three phases of the barrier construction algorithm are explained in detail. The three phases describe about the barrier construction with microwave sensors, selection of boundary camera sensors and selection of non-boundary camera sensors. As shown in Fig. 4, let B_E, B_W, B_N, and B_S be the borderlines of the monitoring region (R) along east, west, north and south, respectively. ML_E, and ML_W are the middle points of the border lines B_E, and B_W, respectively and the central line (CN_L) is obtained by connecting these mid points ML_E, and ML_W. Taking this physical scenario, three phases of the barrier construction algorithm can be designed as follows.

4.2.1. Barrier construction with microwave sensors

The microwave sensors are deployed deterministically along a straight line on the south side (B_S) of the monitoring region, which is the entrance side as shown in Fig. 2. Let m number of microwave sensors $(M) = \{M_1, M_2, \ldots, M_m\}$ be present at their locations

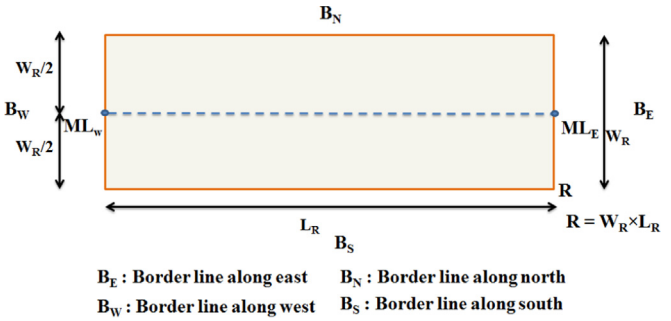

Fig. 4. Monitoring region with border and central lines.

$Loc_x = \{x_1, x_2, \ldots x_m\}$, where $x_i \le x_j$. MS_r is the sensing radius and M_c is the communication radius of each M. If L_R is the length of the monitoring region R, the required number of microwave sensors M to be deployed is $\frac{L_R}{MS_r}$.

Any movement of an intruder within sensing range of a microwave sensor can be captured. Two microwave sensors M_i and M_j can cooperate with each other to detect any event within their sensing range if $|Loc_x^i - Loc_x^j| \le 2MS_r$. It implies that the sensing range of ${MS_r}^i$ and ${MS_r}^j$ are overlapping with each other, i.e. $MS_r^i \bigcap MS_r^j \ne \Phi$ for $i \ne j$. By drawing a line from the west (B_W) to the east borderline (B_E) of the monitoring region through the sensing range of all microwave sensors M, the *microwave sensor barrier* is constructed as shown in Fig. 2. Any intruder that crosses this line can be detected by a microwave sensor M_i.

4.2.2. Selection of boundary camera sensors

In this algorithm, one camera sensor located on the western (B_W) borderline of the monitoring region is selected as the initial node and another one located on the eastern (B_E) borderline of the monitoring region is selected as the terminal node. These two nodes are termed as the boundary nodes of the monitoring region. In order to select the starting node, the boundary camera sensor selection algorithm as given in Table 2 is executed by a camera sensor located at the western side (B_W) border of the monitoring region. Initially, location of the initiator is assumed to be at (0,0)

Table 2
Algorithm 1: Boundary node selection algorithm.

1. *Notation* : CN_L: Central line;
2. B_W : Borderline at West side;
3. H: Horizontal contribution;
4. S_k : Sink node;
5. C_i : ith camera sensor;
6. N_i : One hop neighbor of node i;
7. B_i : Set of Boundary nodes i, for all $i = 1,2,\ldots,n$;
8. BN : Boundary node;
9. BH_i : Horizontal contribution of a boundary node i;
10. Θ_i : Rotation angle of ith camera sensor;
11. BN_{msg} : Message that contains location, radius of sensing range, residual energy and ID of a boundary node;
12. BN_{Rply} : Reply message from the node;
13. R_c : Radius of communication range of a camera sensor;
14. R_s : Radius of sensing range of a camera sensor;
15. RE_i : Residual energy of ith camera sensor;
16. TH_E : Threshold Energy;
17. $d(B_i, C_L)$: Euclidian distance between ith BN and central line (CN_L);
18. $d(B_i, B_W)$: Euclidian distance between ith BN and west borderline (B_W);
19. *BoundaryNodeSelection*()
20. $S_k = (0, 0)$; #Starting location
21. $BN_1 = S_k$;
22. H=horizontal contribution of S_k;
23. do{
24. S_k broadcast BN_{msg} to its one hop neighbor;
25. N_i unicast BN_{Rply} message to sender S_k;
26. if($RE_i \ge TH_E$){
27. BN verifies whether it has horizontal contribution or not;
28. if($BH_i > \mathrm{BH}_{i-1}$)
29. $H = BH_i$;
30. else
31. $H = \mathrm{BH}_{i-1}$;
32. if((d(B_i, B_W) > R_s) && ($B_W \bigcap B_i = \phi$))
33. Rotate an angle Θ_i to intersect with B_W;
34. else if(d(B_i, C_L) < d(B_{i-1}, C_L) && ($BH_i > \mathrm{BH}_{i-1}$))
35. $BN = B_i$;
36. else $BN = B_{i-1}$;
37. }
38. else{
39. Discard B_i
40. }
41. i++;
42. }*While*: C_n is not visited();
43. All other boundary nodes go to sleep mode except BN;

and is considered as the sink node (S_k) among all camera sensors C_i. S_k is treated as the first boundary node BN and S_k sends the BN_{msg} to its one hop neighbors N_i and each N_i sends the BN_{Rply} message to S_k, which contains the identification of the node, sensing contribution toward the border line along with the monitoring region and location information. Residual energy (RE_i) of ith sensor is compared with Threshold energy (TH_E) and continue for all n number of boundary nodes. If the residual energy of a node is below the threshold energy, then simply discard it from the boundary node selection set. The horizontal contribution between any two neighboring camera sensors is compared and the camera sensor having highest value of horizontal contribution is stored in set H. Then, the distance between the nodes stored in set H is calculated from the central line. However, if sensing range of a neighboring node does not intersect the borderline, whereas its horizontal contribution is more than its one-hop neighbors after rotating through an angle Θ_i, then that node is selected as the initial node.Finally, the camera sensor having the highest value of the horizontal contribution and more closer to the central line is selected as the initial node.

For example, as shown in Fig. 5, initially, sensing range of camera sensor BH_i was not intersecting the borderline, whereas its horizontal contribution is more than its one-hop neighbor BH_j after rotating through an angle Θ_i and is closest to the central line. Hence, BH_i becomes the initial node after this rotation. This process continues for all the n number of nodes. Once the boundary node is selected, all other $n-1$ number of boundary nodes go to the sleep mode. Execution of this algorithm is initiated by a camera sensor located along the west side (B_W) border of the monitoring region and is continued until the terminal camera sensor is selected along the east side (B_E) border of the monitoring region. Selection of intermediate non-boundary camera sensors is done based on the algorithm described in Section 4.2.3. Once the boundary camera sensors (initial and terminal nodes) are selected, rest of the boundary nodes in the network can go to the power saving mode.

4.2.3. Selection of non-boundary camera sensors

The non-boundary camera sensor selection algorithm is designed to select the elite camera sensors to construct the barrier. Each node is assigned a weight based on its mobility distance from the central line of the monitoring region, residual energy level, and angle of rotation. The entire monitoring region is divided into subregions based on the sensing range (MS_r) of the microwave sensor (M). Though we present here the construction of barrier in one subregion, practically barrier construction phase is executed in parallel in each subregion. The whole procedure of barrier construction is executed in two different phases as follows. It is to be noted that each camera sensor knows its location information and equation of the central line CN_L. After deploying the camera sensors randomly, each node calculates its distance from the central line, which is considered as the magnitude of the mobility distance. As shown in Fig. 6(a), each node knows its required mobility

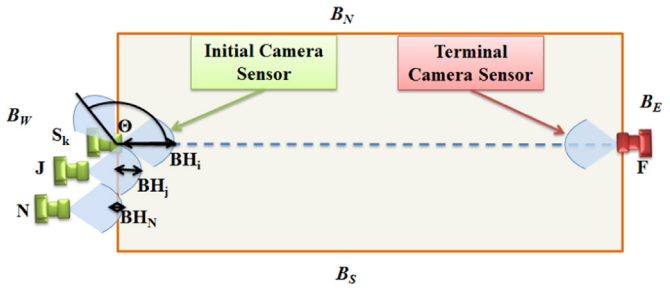

Fig. 5. Example of boundary camera sensor selection.

distance (d_i) from the central line CN_L, for $i=1,2,\ldots,n$. Let RE_i and TH_E be the current residual energy and threshold energy of a node, respectively.

It is to be noted that threshold energy TH_E of a node is the required energy to perform the sensing operation and to monitor the network after construction of the barrier and is a constant for all nodes of the network. Each node exchanges the value of RE_i and required mobility distance d_i with its one-hop neighbors as shown in Fig. 6(b). Upon receiving these values, each node compares the value of RE_i with the value of TH_E and ignores the neighbors whose $RE_i < TH_E$ as shown in Fig. 6(c). This procedure is continued for all nodes of the network and let k nodes out of n be the remaining nodes in the monitoring area after discarding the nodes whose $RE_i < TH_E$.

Let d_i be the distance of a camera sensor i from central line, γ be the amount of energy consumed by moving it for unit distance, and Θ_i be the angle of rotation of the camera sensor i such that its sensing range can intersect with the sensing range of its one-hop neighbors. Taking E_i as the initial energy of camera sensor i, energy consumption of ith camera sensor E_i^c due to its mobility and then rotation can be calculated as given in Eq. (3). The residual energy RE_i can be calculated as given in Eq. (4). If $RE_i > TH_E$, the ith camera sensor can move toward the central line based on the algorithm given in Table 3.

$$E_i^c = (d_i * \gamma) + \Theta_i \tag{3}$$

$$RE_i = E_i - E_i^c \tag{4}$$

4.2.4. Selection of best non-boundary camera sensors

After selecting the non-boundary camera sensors based on their mobility distance (d), the angle of rotation (Θ) is used to select the best candidate for constructing the barrier with quality of sensing. Here, the angle of rotation is used among the non-boundary sensors to select a node that can have limited mobility distance with least angle of rotation as shown in Fig. 7.

Prior to this selection, boundary node selection and limited mobility non-boundary camera sensor selection algorithm are executed and the values are stored in set BN and V, respectively. Based on the communication range of a node, location of its one-hop neighbors is stored in set N_i. For each node i, the weight Ψ_w^i is calculated from the mobility distance and angle of rotation of a camera sensor. The user defined threshold of image quality (α) is the percentage of an image that is required for monitoring an intruder calculated in advance and compared with the quality of sensing (QoS_i) of ith camera sensor. The weight (Ψ_w^i) and quality of sensing (QoS_i) of a camera sensor are compared with corresponding values of its one-hop neighbors. A node having more weight with better QoS is selected as the best non-boundary camera sensor to form the barrier.

After several iterations of selection procedure of the non-boundary camera sensors based on the angle of rotation, the threshold of image quality α is compared with QoS to check whether the barrier is constructed or not. As shown in Fig. 8, if the sensing range of the boundary node BN and intermediate node C_i intersects with each other, there exists a barrier between them by maintaining the quality α. This procedure is continued until the sensing range of intermediate sensors intersects with the terminal sensor BN_E present on east side of the monitoring region. The algorithm of selecting the best non-boundary camera sensor to form the barrier with the help of boundary camera sensors is given in Table 4.

As described in Table 4, steps 1 through 13 are used to initialize the data sets. By using step 15, S_k is selected as the boundary sensor node as shown in Fig. 8, where $S_k \in BN$. Two sensor nodes, A and B are chosen as the neighboring camera sensors of S_k.

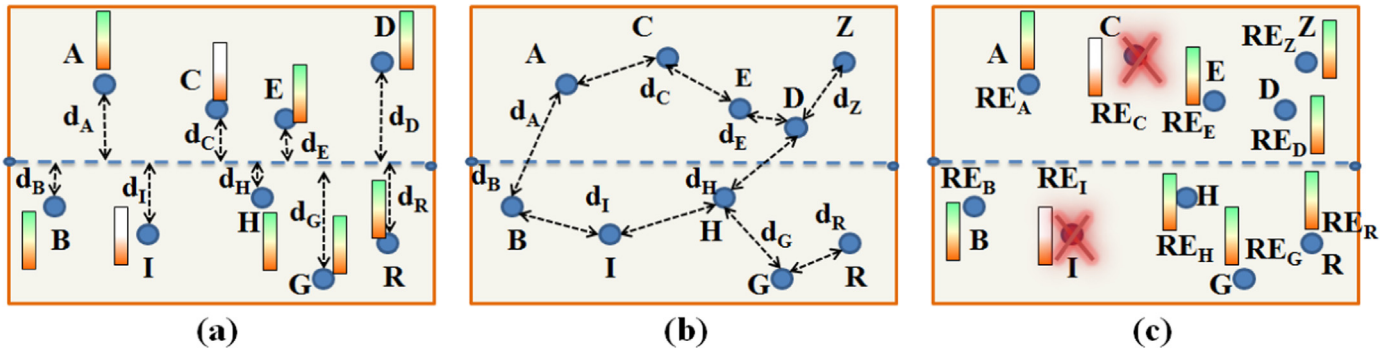

Fig. 6. Example of selecting non-boundary camera sensors.

Table 3
Algorithm 2: Selection of non-boundary camera sensors.

1. *Notation* : CN_L: Centerline;
2. C_i : ith camera sensor node;
3. Ns_i : Non-boundary camera sensors, where Ns_i=Ns_1,Ns_2,…,Ns_n;
4. U : Intermediate remaining sensor nodes;
5. V : Total remaining sensor nodes;
6. W_i : Calculated weight for each camera sensor;
7. RE_i : Residual energy of ith camera sensor;
8. TH_E : Threshold energy;
9. *Potentialnon-boundarycamerasensors*()
10. W_i=0;
11. RE_i=0;
12. for($i=0$; $i<n$; $i++$)
13. {
14. Calculate $d(C_i, CN_L)$;
15. Calculate RE_i;
16. if($RE_i < TH_E$)
17. {
18. Discard NS_i, ∀i=1,2,…,n;
19. $U = |C_i - NS_i|$;
20. }
21. V=U;
22. Return V;
23. }

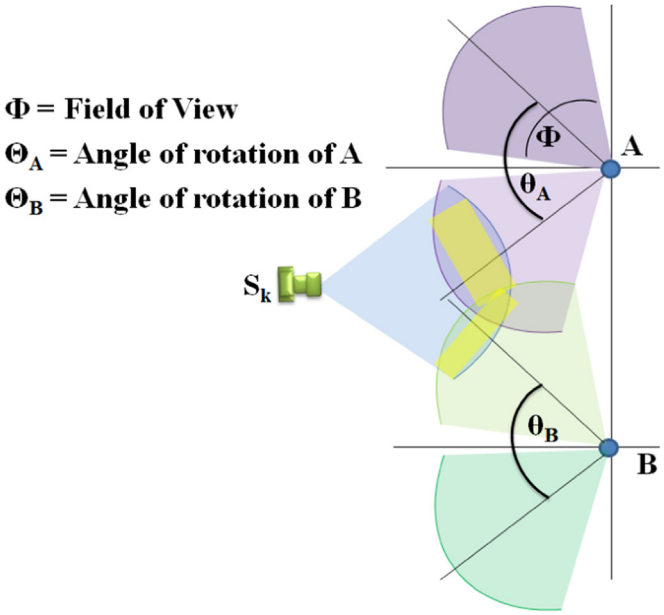

Fig. 7. Example of selecting non-boundary camera sensors based on angle of rotation.

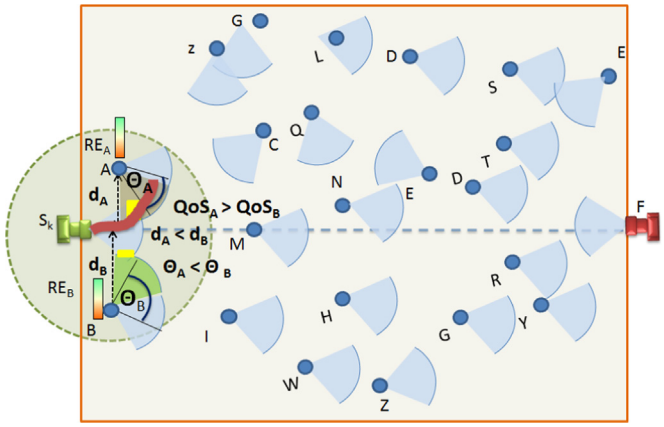

Fig. 8. Barrier construction procedure.

The weight factor Ψ_w^i is calculated for each node as given in steps 18 and 20 by considering the angle of rotation Θ_A and Θ_B as given in step 19. The QoS parameter for both nodes A and B is checked in step 21 and QoS_A and QoS_B are compared with the predefined quality (α) in steps 22 through 30. As given in steps 31 through 35, the barrier is constructed between nodes A and B by comparing the sensing range of camera sensors A and B. By repeating the steps from 18 through 35, the barrier construction procedure is continued until the complete barrier is formed by taking the non-boundary and boundary camera sensors. The example of such barrier construction is also shown in Fig. 8. Here minimum numbers of sensors are selected to construct the barrier as a result of which data redundancy and length of the routing path is reduced. Once the barrier is constructed, the data collection phase is carried out and the data packets are sent frame by frame to the base station with a minimum routing cost.

4.3. Energy consumption analysis

In this section, we analyze the energy consumption of a sensor including the motion and camera sensors by constructing the barrier, detecting the intruder and transmitting the image data to the sink for the ultimate processing by the big data analytic centers. According to Sinha and Chandrakasan (2001), Chen et al. (2010c) and Halgamuge et al. (2009), energy consumption depends on the sensing energy (E_{SE}), rotational energy (E_{Rot}), processing energy, transmission and receiving energy and state transition energy.

Taking P_{Wrk}, S_{Vol}, S_{Cur} and S_{Dur} as the working power of a sensor, supply voltage, sensing current, and sensing duration, respectively, the following equations are derived.

$$P_{Wrk} = S_{Cur} * S_{Vol} \tag{5}$$

Table 4
Algorithm 3: Selection of non-boundary camera sensors using angle of rotation.

1. *Notation* : CN_L: Central line;
2. C_i : ith camera sensor;
3. N_i : One-hop neighbors of camera sensor i;
4. N_s^i : Non-boundary camera sensors, where $N_s^i = N_s^1, N_s^2, \ldots, N_s^n$;
5. U: Intermediate remaining camera sensors;
6. V : Potential non-boundary camera sensors;
7. Ψ_w^i : Calculated weight for each camera sensor i;
8. Θ_i : Angle of rotation;
9. RE_i : Residual energy of ith camera sensor;
10. TH_E : Threshold energy;
11. BN_s=Sensing range of boundary sensor;
12. BN_E=Boundary sensor along east side;
13. BN_W=Boundary sensor along west side;
14. *SNCSPAR*(){:
15. BN=BoundarynodeSelection();
16. **for**($i=0$; $i<j$; $i++$)
17. {
18. Calculate $\Psi_w^i = E_i - (d_i * \gamma)$, where γ=unit of energy consumed to move a unit distance;
19. Check for Θ_i;
20. Update $\Psi_w^i = \Psi_w^i$ - $(\Theta_i * \delta)$, where δ=unit of energy consumed to rotate a unit angle;
21. Calculate $QoS_i = W_i$ x L_i;
22. if(QoS_i=QoS$_{i-1}$=α)
23. if($\Psi_w^i > \Psi_w^{i-1}$)
24. Select C_i;
25. else Select C_{i-1};
26. if($QoS_i > QoS_{i-1}$)
27. if($QoS_i > \alpha$)
28. Select C_i;
29. else if(QoS$_{i-1} > \alpha$)
30. Select C_{i-1};
31. if(R_s&&$BN_s \neq \phi$)
32. Draw horizontal line (barrier) from BN_W to R_s;
33. else if(R_s&&$N_s^i \neq \phi$)
34. Draw horizontal line (barrier) from R_s to N_s^i;
35. else Draw horizontal line (barrier) from R_s of $(i-1)$ th sensor to N_s^{i-1};
36. }
37. **do**{
38. Repeat Barrier Construction steps from 31 to 35 to draw the barrier line in the network;
39. }*while* (R_s&&$BN_E \neq \phi$)

For transmitting k bytes of data, energy consumed by a sensor can be calculated as follows.

$$E_{SE} = k * S_{Cur} * S_{Vol} * S_{Dur} \tag{6}$$

The energy consumption for reading/writing k bytes of data can be calculated as follows.

$$E_{RW} = k * S_{Vol} * (C_{Wr} * T_{Wrt} + C_{Rd} * T_{Rd}) \tag{7}$$

where E_{Wrt}=energy consumption for writing data, E_{Rd}=energy consumption for reading k bytes of data, T_{Wrt}=time required to write, T_{Rd}=time required to read, C_{Wr}=current for writing 1 byte of data, C_{Rd}=current for reading 1 byte of data. Finally, the total energy consumption (E_{Tot}) is given in Eq. (8).

$$\left.\begin{aligned} E_{Tot} = (S_{Cur} * S_{Vol} * S_{Dur}) + E_{Rot} \\ + S_{Vol} * (C_{Wr} * T_{Wrt} + C_{Rd} * T_{Rd}) + E_{Pro} + E_{Trn} + E_{Rcv} + E_{St} \end{aligned}\right\} \tag{8}$$

5. Big data analytic and storage layer

5.1. Big data analytic layer

The big data analytic layer is responsible for processing, analyzing and identifying the intruders based on the data collected by the camera sensors. In this layer, the detail process of trespasser detection is explained. Data are the substantial element of big data analysis and in case of border surveillance, the camera sensor data are even more essential for security purpose. However those streaming video data are huge in size with high arrival rate. Though several big data processing models such as Hadoop, Storm, and Spark exist to deal with these situations, we select the Spark streaming model as it provides the most appropriate data engineering technique for real-time intruder detection. Prior to data analysis, it is assumed that huge amount of data is collected by those camera sensors to obtain the quality image without losing any necessary information. This colossal intruder data are processed in three different phases, i.e. intruder detection and data acquisition phase, intruder data management phase, and intruder data analysis phase, which are described in the subsequent subsections.

5.1.1. Intruder detection and data acquisition phase

Intruder detection is a major activity under border security monitoring along with habitat monitoring. In Keung et al. (2012), authors propose a k-barrier coverage model, which uses mobile sensors to detect the moving intruder. Prior to this work all others have used the static sensor for target detection. Authors in Keung et al. (2012) mapped this problem to the classical kinematic theory to establish a relationship between the moving intruder and moving sensors as gas molecules. In this formulation, detection depends on sensor's moving speed (V_m), sensing range (R_s) and time duration (τ). Both sensors and intruders move with a constant velocity where intruders path intersect with the sensor's traveling path. In this process, an intruder is detected, if it intersects with the sensor's path along with the sensing range of the sensor. However, it could be possible that an intruder is undetected, if it is not within the sensing range of a sensor and also even if the intruder is detected, the sensor cannot communicate instantly with other sensors to pass the information. In Sahoo et al. (2013), authors have proposed the centralized, distributed, sequential boundary node selection algorithms with target tracking algorithm. However, the quality of sensing in terms of image resolution is not discussed in this work. In our proposed work, we not only focus on the detection of an intruder but also maintain the quality of detection.

In this subsection, the intruder detection with guaranteed quality of sensing and data transmission mechanism are illustrated. We use dual technology to save energy and enhance the lifetime of the network. It is assumed that the intruder will maintain a constant velocity while crossing the monitoring region. The main purpose of this analysis is to detect an intruder with minimum energy consumption that trespasses across the monitoring region. Dual technology microwave sensors (Dual Technology Motion Sensor) have a field of view consisting of a Doppler microwave detector with two receiving channels and a dual element infrared detector, which are used for primary detection of the intruders. Detection of an intruder occurs only when the microwave and passive infrared protection patterns overlap. In order to detect the intruders using microwave motion sensors, the whole monitoring region of area $W_R \times L_R$ is divided into grids. Thus, the total number of horizontal partitions is $W_R/R_s = P$ and the total number of vertical partitions is $L_R/MS_r = Q$. After partitioning the monitoring region into grids, Red and Yellow zones are demarcated as shown in Fig. 9. The *Red zone* (Z^R) is the most probable zone to identify the intruder, whose area can be demarcated as $L_R * (W_R/2R_s)$. *Yellow zone* (Z^Y) is the alert zone to wake up the camera sensors whose area is demarcated as $L_R * (W_R/R_s)$.

When an intruder enters into the monitoring region, M_m can detect the intruder within its sensing range. M_m will wait for Δ_{ty} units of time in the yellow zone and then communicate to the camera sensor present within its communication range. After Δ_t

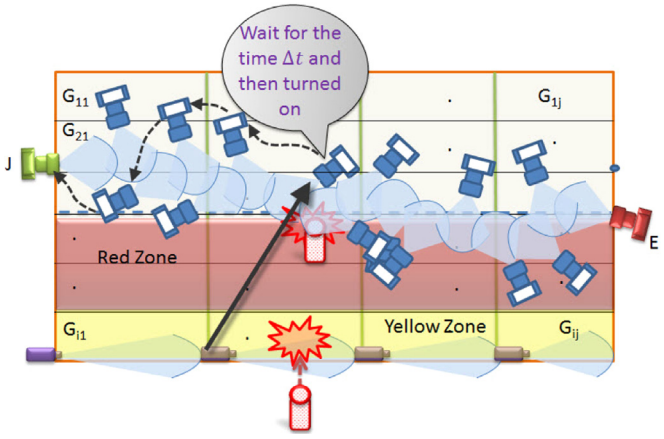

Fig. 9. Detection of intruder and data transmission. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

time, camera sensors are activated as shown in Fig. 9. If a camera sensor does not detect any intruder within Δ_{tr} time, it informs to neighboring camera sensor present in the next grid to active. For quality monitoring, the resolution of the intruder's image needs to be checked and maintained during detection. Taking Z_W^Y, Z_W^R and V_I as the width of the Yellow zone, width of the Red zone and velocity of the intruder, respectively. Waiting duration of the microwave sensor (Δ_{ty}) and wireless camera sensors (Δ_{tr}) can be derived as given in Eqs. (9) and (10), respectively.

$$\Delta_{ty} = Z_W^Y / V_I, \tag{9}$$

$$\Delta_{tr} = Z_W^R / V_I, \tag{10}$$

where Δ_{ty} and Δ_{tr} are the time to detect the intruder in yellow and red zone, respectively. Hence, the image frames are captured within the red zone only and are transmitted to the next hop camera sensors with a beacon message including the destination sink node's id represented as the dotted arrows in Fig. 9 or we can use any existing collaborative multi-hop routing algorithm (Jiang et al., 2015). The data packets are guaranteed due to short range, energy saving, low power transmission mode by which the quality of the data packets are maintained. The first label of the data aggregation is accomplished by combining the images within each active grid. Subsequently, the accumulated data are dispatched to big data processing system for analysis.

5.1.2. Intruder data management phase

It is to be noted that the image of the intruders captured by the barrier camera sensors are of real time streaming data. It is very difficult to handle those gigantic real-time video frames for processing. A Spark streaming model is used to execute the surge camera sensor data. A Streaming Request Handler (SRH) interface is used in between the wireless sensor layer and big data layer to control the stream of data sets. In Fig. 10, the input video frames ($Iv(x,y,t)$) at time t are divided into multiple small video segments ($Vs(x,y,t)$) by the SRH, where $Vs(x,y,t) = \{Vs_1(x,y,t), Vs_2(x,y,t), Vs_3(x,y,t), \ldots, Vs_m(x,y,t)\}$. The response time of each sub-frame is less than one second, which enhances the execution speed. The sliced Streaming Data Frames (SDFs) known as Resilient Distributed Datasets (RDDs) in Spark are transferred to local caches by calling *persist* method present in the Spark. In this case, $\{Vs_1(x,y,t), Vs_4(x,y,t), Vs_k(x,y,t)\}$ SDFs are stored in the local cache χ as shown in Fig. 10. The persistent data are stored in local memory for future use. However, if the data size exceeds the cache limit, then the selected data can move to the disk based on the priority. Hence, the data are selected either from the input stream directly or from the local cache for execution by the Spark Executor for intruder detection.

5.1.3. Intruder data analysis phase

The intruder data are processed and analyzed in the Spark Executor present in the core module of the Spark as shown in Fig. 11. In the processing phase, we also intend to eliminate the background image (Bouwmans, 2014) of the captured border images by the camera sensors. The streaming data processing is accomplished in two stages, i.e. *Transformation* and *Action*. The background of the video frames are eliminated in the stage 1 of transformation phase to identify the intruder and reduce the data volume as the interest is on object rather than the background scene. The *filter* method is used during this background subtraction process. In Fig. 11, the data stream ($Vs_1(x,y,t)$) has three different frames, i.e. $Vs_1^1(x,y,t)$, $Vs_1^2(x,y,t)$, $Vs_1^3(x,y,t)$ in which the partial image of the intruder is captured with the background ($B(x,y,t)$). Therefore, the filter method is used on each frame to generate the intruder image only. Taking n as the number of frames per second in the video, the average ($B_{Avg}(x,y,t)$) can be calculated as follows.

$$B_{Avg}(x,y,t) = \frac{1}{n}\sum_{i=1}^{n}(1-\omega)Vs_i(x,y,t-1) + (\omega)Vs_i(x,y,t_i) \tag{11}$$

where $\omega = \frac{1}{t}$ is the learning parameter. Hence, the incoming frame $Vs_i(x,y,t)$ at time t is compared with the previously calculated average background $B_{Avg}(x,y,t)$. Let us consider an example where 5 number of frame images ($n=5$) are coming at time instances $t = \{1,2,3,4,5\}$. The learning parameter ω is assumed to be 1. According to Eq. (11), the background is calculated by taking the average of 5 frames instead of a single frame, which is more realistic. Hence, $B_{Avg}(x,y,t)$ can be represented as follows.

$$\left.\begin{aligned} B_{Avg}(x,y,5) &= \frac{1}{5}\sum_{i=1}^{5}(1-1)Vs_i(x,y,t-1) + (1)Vs_i(x,y,t_i) \\ B_{Avg}(x,y,5) &= \frac{1}{5}\sum_{i=1}^{5}Vs_i(x,y,t_i) \end{aligned}\right\} \tag{12}$$

Once the background ($B_{Avg}(x,y,t)$) is calculated, a subtraction or *filtering* process is carried out by comparing each upcoming frame with the background frame to find the intruder image as shown in Eq. (13).

$$\left.\begin{aligned} Int(x,y,t) &= Vs_i(x,y,t) - B_{Avg}(x,y,t) \\ s.t.\ Int(x,y,t) &\geq I_{Th} \end{aligned}\right\} \tag{13}$$

where, $Int(x,y,t)$ is the intruder image at time t and I_{Th} is the quality threshold parameter. However, in one frame the complete image of the intruder may not be visible. To get rid of this problem, the partial image of the trespasser is combined together by using the *join* function to get the complete picture. Hence, only the intruder image is sent to the stage 2 for intruder analysis. The

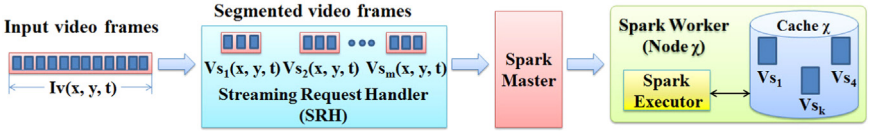

Fig. 10. Intruder streaming data processing.

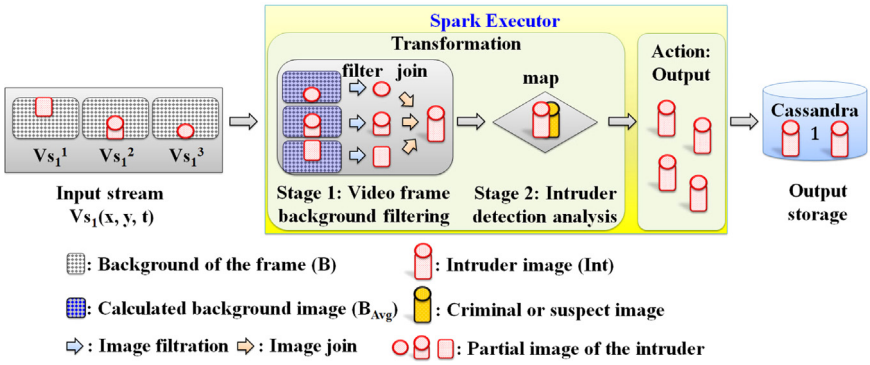

Fig. 11. Intruder streaming data analysis.

intermediate data are stored in the local cache to achieve the data locality in the next stage. This in-memory execution reduces the processing time of large streaming data. In stage 2, the intruder image is compared with the suspected person by using the *map* as a transformation. If we want to compare the image with multiple targets, then *flatmap* is used. We can also use any existing face detection algorithms (O'Toole et al., 2007) as our map or flatmap function. Eventually the analyzed data are transferred to the output phase for storage purpose. All analyzed data are stored using *save* method present in the Spark output.

5.2. Cloud layer

In order to analyze and detect the intruders, an organization has to setup a large scale barrier coverage system with the help of camera sensors and the collected data need to be stored and analyzed time to time in a distributed service architecture. For example, in military application, the data collected by the camera sensors need to be stored in the systems, which should be fault tolerant, highly available, and highly durable. It can be achieved only if the data are stored in multiple facilities with error checking and self-healing processes and should be accessible at any location. By storing the larger size of image data in a highly virtualized, distributed Cloud, features such as scalability, elasticity, fault-tolerance, self-manageability, and ability to run on commodity hardware can be achieved.

Moreover, by storing data in on-premise data centers within an organization's local network may have single point of failure and is not cost effective as building an infrastructure from the ground up will have an organization's own maintenance and administration. Beside, a traditional data center can have limited capacity of storage and we cannot be able to change the amount of storage and workload without purchasing and installing more equipment once it is built. However, a cloud platform for the big data analysis can act as an off-premise computing environment to store the data on the Internet and can be available for analysis irrespective of any locality. Hence, we propose here a Cloud layer for the physical storage of big data through the Zookeeper.

In our proposed architecture, Cloud layer is the physical storage layer, which is solely used for storing the analyzed data across different data centers (DCs) in a distributed fashion. All DCs are networked and distributed geographically. In this layer, it is proposed that Zookeeper is used as a coordinator between the Spark Executor and cloud repository for data storage purpose. Cassandra distributed databases are used in the cloud and local cache as it can store the real-time data sets in an effective manner. It can support the fault-tolerant by clustering multiple database nodes and backup nodes, which are also synchronized by the zookeeper. Any data visualization technique can be applied on the stored data to present the results in graphical or chart formats for the military and research purpose by extracting the useful information. By using this kind of architecture, the camera sensor streaming data are efficiently handled, analyzed and stored for the intruder detection application.

6. Simulation results

In this section, the performance of our proposed algorithms are evaluated using NS-3 and CloudSim (http://www.cloudbus.org/cloudsim/) simulators on Ubuntu platform. Directional camera sensors are deployed randomly over a rectangular monitoring region of size 1000 m × 100 m. All simulation parameters are setup according to the IEEE 802.15.4 MAC mechanism with AODV routing protocol. The number of deployed sensors varies from 100 to 1000 in the simulation and the sensing range is set to be 5 m, 10 m and 20 m. The communication range is twice of the sensing range and accordingly it is set to be 10 m, 20 m, 40 m. Throughout the simulation, we have fixed the field of view (FOV) as 30°, 45°, 60°, 90°. Quality of monitoring is defined as the percentage of the area of the image detected. In the simulation, the mobility distance of each camera sensor is set to be 1 m, 2 m and 5 m. The size of the control packet is 128 kb and the simulation is run for 25 rounds to get average of the each simulated data. Performance evaluation is done taking different number of directional sensors, sensing range, field of view and quality of sensing in different scenarios and detailed list of parameters used in the simulation is shown in Table 5.

As shown in Fig. 12, it is observed that the percentage of barrier construction rate increases monotonically with the increase in the number of camera sensors.However, a less number of camera sensors are required to achieve the same coverage with an increase in the sensing range. Here, FoV and mobility distance are kept constant. The reason is that when the radius of sensing range increases, it enhances the barrier coverage probability.

In this process, we should decrease the number of sensors to achieve 100% of barrier coverage. In Fig. 13, it is noticed that the

Table 5
List of simulation parameters.

Number of nodes	100–1000
Monitoring area	1000 m × 100 m
Sensing range (R_s)	5–20 m
Communication range (R_c)	10–40 m
Field of View (FoV)	30–90°
Quality of monitoring (α)	5 m × 10 m
Mobility distance	1–5 m
Size of control packet	128 kb
Initial residual energy	100 J

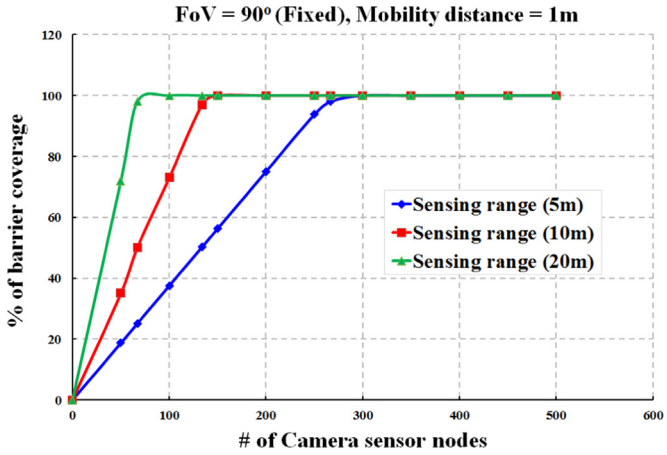

Fig. 12. % of barrier construction with different sensing range.

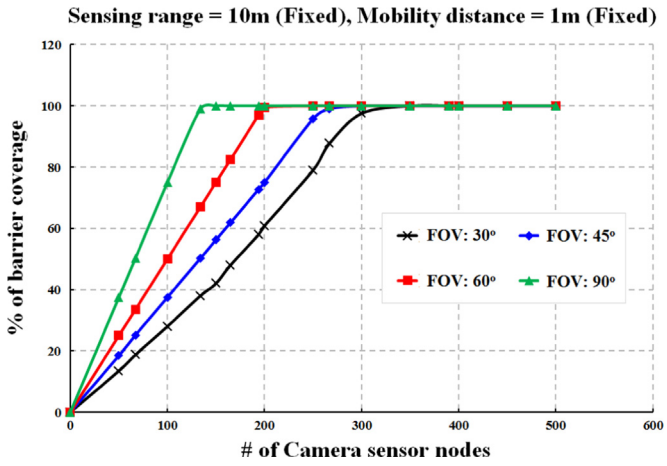

Fig. 13. % of barrier construction with FOV.

barrier coverage percentage grows when the number of sensors increases with different FOVs.

The increment in barrier construction depends on the angle of FOVs with constant sensing range and mobility distance. It is analyzed that the barrier construction rate is also increased with a small number of camera sensors, if the FOV increases.

The barrier construction time is simulated as shown in Fig. 14. The time taken for barrier construction is reduced with an increase in the sensing range where FOV and mobility distance are steady.

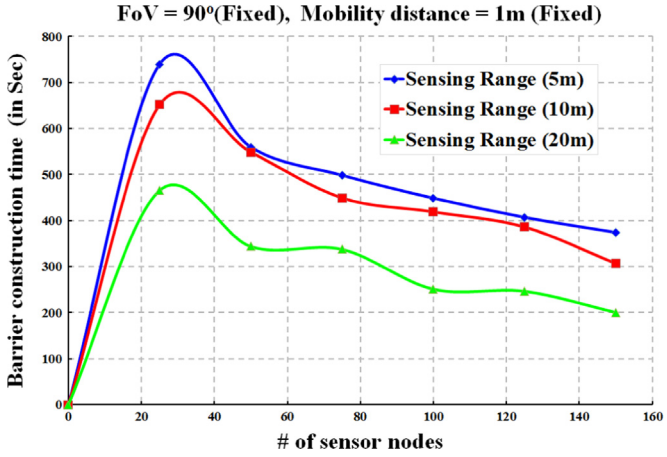

Fig. 14. Barrier construction time.

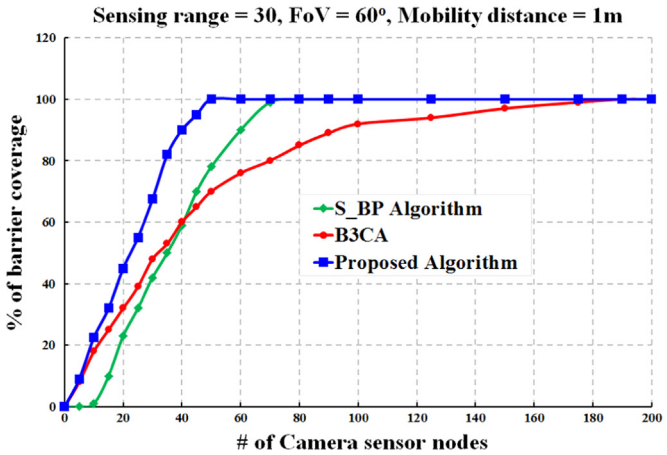

Fig. 15. Performance comparison based upon # of nodes.

In Fig. 15, barrier construction performance comparison is done based on the number of camera sensors.

In this figure, it is observed that a less number of camera sensors are used to construct the barrier in our proposed algorithm as compared to the S_{BP} and B^3CA (Guo et al., 2014) algorithm.

For 100% barrier construction, 50 number of camera sensors are required by our protocol, whereas 80 and 190 number of camera sensors are needed by S_{BP} and B^3CA, respectively. In Fig. 16, successful barrier construction rate is compared with respect to the Quality of Sensing (QoS). It is observed that our algorithm outperforms as compared to the *D-TriB* and B^3CA with different widths. In this experiment, the length of the image is fixed to be 1 unit as it is not considered by other protocols. Mobility of the sensors helps us to minimize the number of camera sensors during the barrier construction, by maintaining the QoS. It is clearly observed that the sensors having smaller FOV consume more energy for rotation to maintain 100% barrier coverage as shown in Fig. 17, where sensing range and mobility distance are unchanged.

The reason is that to achieve the QoS for a fixed percentage, we need to intersect the sensing range of two neighboring camera sensors. Hence, camera sensor with smaller FOV needs more rotation to continue the barrier along with the QoS.

In Fig. 18, the total energy consumption is calculated taking camera sensors as well as the static microwave sensors to maintain 100% coverage with fixed sensing range of the camera sensors. It is found that the sensors having more mobility distance and less FOV consume more power. Therefore, it is advisable to select a less number of mobile sensors with larger FOVs to exhaust less energy. In Fig. 19, the network lifetime is simulated and is confirmed that the lifetime can be enhanced by choosing a less number of mobile

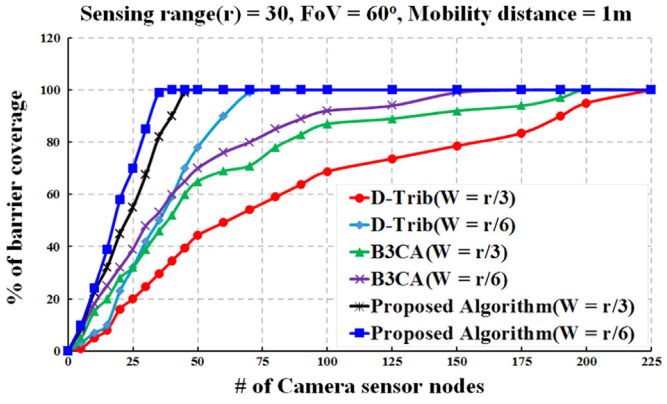

Fig. 16. Performance comparison based upon QoS.

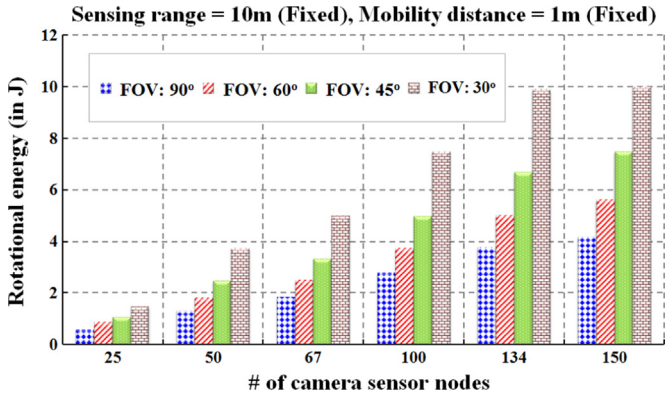

Fig. 17. Energy consumption due to rotation.

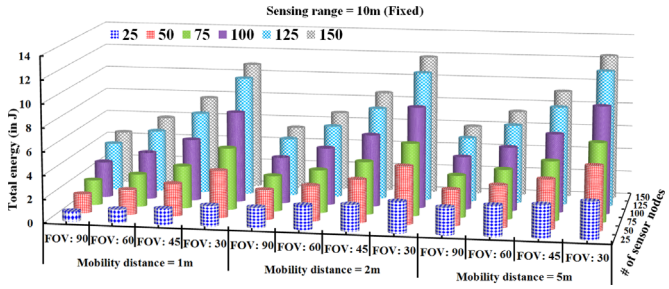

Fig. 18. Total energy consumption with mobility distance.

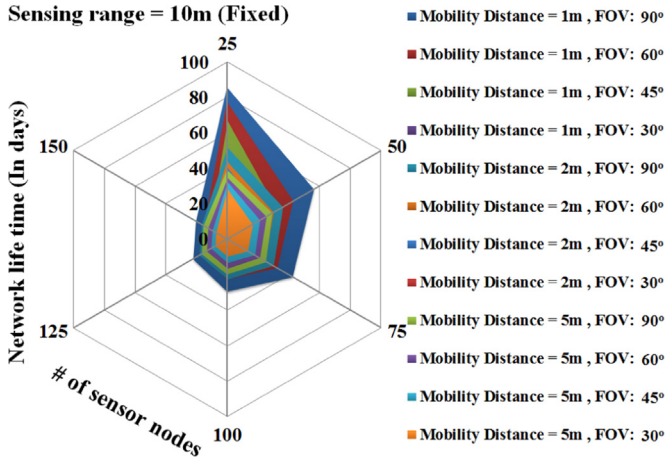

Fig. 19. Network lifetime for different number of camera sensors.

sensors with larger FOVs and medium sensing range. However, in this figure, the sensing range is fixed as 10 m. Number of nodes, FOVs and sensing range greatly influence the network lifetime to improve it.

We have also simulated the proposed algorithm using in CloudSim simulator to study the processing time for different number of servers. An efficient way of streaming data processing is the main focus with processing and storage cost optimization. For this reason, we have considered a scenario that consists of five data centers with many servers in each data center. It is assumed that the data centers are networked and are geographically distributed. The processing time is defined as the summation of both execution time and data transfer time from different locations. Our goal is to utilize the optimal number of data centers to study the processing cost incurred by those data centers.

From Fig. 20, it is observed that the parallel servers can be beneficial only when a large number of streaming data need to be processed for background elimination. Initially, parallel servers

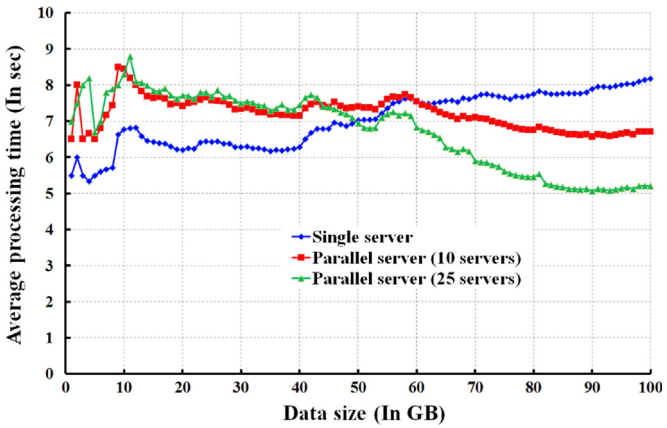

Fig. 20. Average processing time with different number of servers.

took more time for a less number of data size as compared to the single server, since the data are distributed over different locations. However, when the camera sensor data size increases significantly, a single server may not be able to accommodate all the data. Hence, it takes much time as compared to the parallel servers. However, always increasing the number of parallel servers does not give better result. In our simulation, it is observed that 25 parallel servers give higher performance in terms of processing time when the data size is up to 85 GB, but the processing time is almost constant until the data size is 100 GB as the data are highly distributed.

In Fig. 21, the image analysis is performed by using the Spark with Graphics Processing Unit (GPU). In the process of execution, NumbaPro (http://docs.continuum.io/anaconda-cluster/examples/spark-numbapro), NVIDIA and Compute Unified Device Architecture (CUDA) platform are used to support the image analysis in Spark. From the simulation figure, it is clearly noticed that the Spark-GPU works efficiently for the image analysis. The processing time of Spark-GPU is 2 × time (on average) faster than the Spark-CPU. The processing time is higher in Spark-CPU with the increase in the number of images (900–1000 images) as more disk I/O operations are needed to achieve the data locality. By taking advantage of GPU, if we scale up the number of CPU and GPU, the execution time can be 18.6 × faster (Li et al., 2015) as compared to the only Spark platform.

Fig. 22 shows the utilization of data centers while processing a huge number of records. In our simulation, we have set the input data size in gigabytes ranging from 5 GB to 50 GB. Beside, we have taken heterogeneous servers for each data center with an unequal number of servers ranging from 30 to 50. From Fig. 22, we can conclude that the CPU utilization of servers increases with an

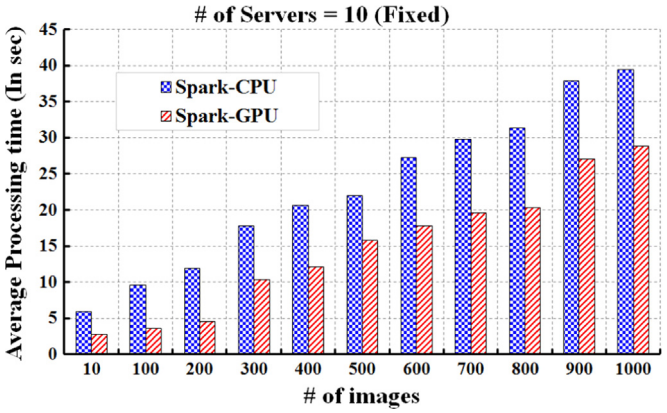

Fig. 21. Spark-CPU Vs Spark-GPU.

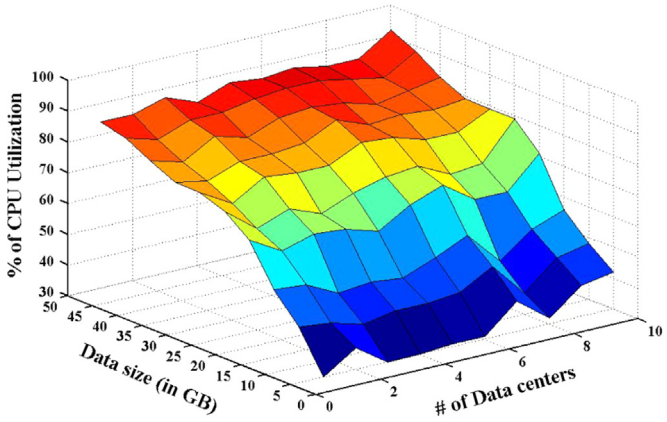

Fig. 22. % of CPU utilization.

increase in amount of data size coming to different data centers. However, when the amount of data exceeds the capacity of a single data center, a new data center needs to be deployed to balance the utilization. Our goal is to maximize the resource utilization without compromising the processing deadline. However, after attaining an optimum CPU utilization of data centers with fixed size of data, if we increase the number of data centers, the utilization percentage will decrease due to over distribution.

In Fig. 23, the system throughput is shown over a period of time in the presence of multiple servers. The processing throughput is defined as the amount of data segments executed among different servers within the cloud network. Initially, the system throughput is high with 5# of servers up to 12 s as the data volume is low. Later on, the system throughput is increased with the increase in # of servers over the time due to the increase in the data volume. However, an increasing trend of throughput is noticed up to 55 s, which remains steady after that time.

In Fig. 24, the average link load of the cloud network over time is valuated. The ratio between the used bandwidth over the capacity is defined as the link load between the data centers. In this case, the average link load is observed during the peak time by taking different size of the data packets. During the initial phase of the graph, the load is steady up to 20 s due to data locality and less workload. However, the load is increased between 20 and 40 s due to high data volume and the intermediate data transfer between the data centers for next level of processing. It is observed that the packet sizes of the data are also responsible to increase the load on the link of the data center network.

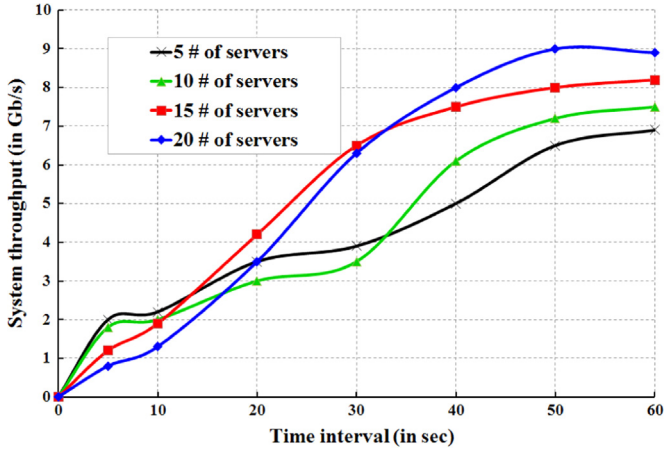

Fig. 23. System throughput.

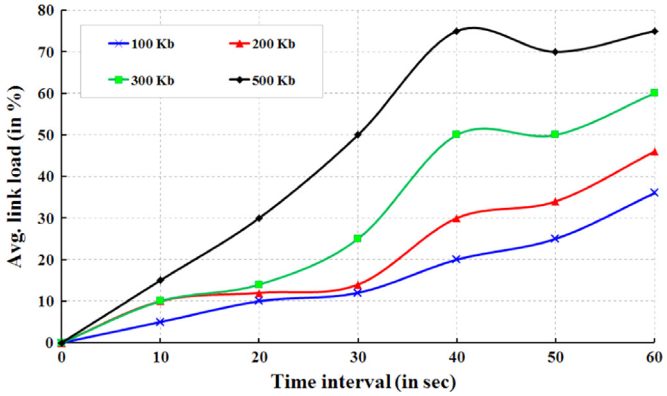

Fig. 24. Average link load in the cloud network.

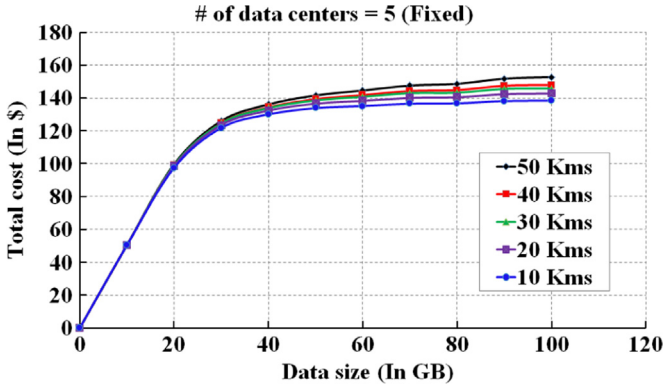

Fig. 25. Cost incurred for processing data in distributed data centers.

As shown in Fig. 25, the incurred cost associated with different number of distributed data centers across different geographical locations is simulated. In the simulation, bandwidth cost, storage, computation cost and data migration cost are taken into account. For cost calculation, we have taken the pricing model of Amazon Web Server (AWS) (http://www.aws.amazon.com/cloudcomputing) for reference and found that higher cost is incurred if the geographical distance is increased.However, this trend is not always exactly the same for all the scenarios as many different parameters are interrelated with each other.

7. Conclusion

In this paper, a big data analytic architecture is proposed to process and analyze the data generating from the camera sensors to form the barrier and to detect the intruders. In addition, a barrier construction algorithm is designed to construct the barrier with the help of a minimum number of camera and microwave sensors. Beside, intruder detection technique is also introduced to detect and identify the intruder within the specified region with a threshold resolution. The QoS is maintained by taking size of the intruder across the network. The intruder detection mechanism is also introduced with cloud layer to store the data about the intruders. The data size of the captured image is reduced by eliminating the background images. A Spark streaming framework is introduced to handle and process the huge volume of camera sensor data in parallel. Taking different number of data centers distributed geographically, the throughput, link load processing cost are simulated and CPU utilization is evaluated with different number of servers in the cloud environment. It is observed that our proposed algorithms can be used for the border surveillance applications to monitor the border round the clock using camera

sensors and our big data architecture with cloud layer can give the complete solution of analyzing the large volume of captured data.

Acknowledgments

This work is co-sponsored by the Ministry of Science and Technology (MOST), Taiwan, under Grants 103-2221-E-182-029, 104-2221-E-182-004, 101-2923-E-182-001-MY3, 104-2221-E-182-032 and is partly supported by Chang Gung University, Taiwan under Grant UERPD2D0061.

References

⟨http://www.aws.amazon.com/cloudcomputing⟩.
Bhattacharya M, Islam R, Abawajy J. Evolutionary optimization: a big data perspective. J. Netw. Comput. Appl. 2014.
Bouwmans T. Traditional and recent approaches in background modeling for foreground detection: An overview. Comput. Sci. Rev. 2014;11(2):31–66.
Chellappan S, Gu W, Bai X, Xuan D, Ma B, Zhang K. Deploying wireless sensor networks under limited mobility constraints. IEEE Trans. Mob. Comput. 2007;6(10):1142–57.
Chen, Tzung-Shi, Tsai, Hua-Wen, Chen, Chih-Ping, Peng, Jiun-Jie, 2010a. Object coverage with camera rotation in visual sensor networks. In: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference (IWCMC '10), ACM, pp.79–83.
Chen A, Kumar S, Lai TH. Local barrier coverage in wireless sensor networks. IEEE Trans. Mob. Comput. 2010b;9(4):491–504.
Chen, J., Salim, M.B., Matsumoto, M., 2010c. Modeling the energy performance of object tracking in wireless sensor network using dual-sink. In: 16th Asia–Pacific Conference on Communications (APCC), pp. 204–209.
Cheng Chien-Fu, Tsai Kuo-Tang. Distributed barrier coverage in wireless visual sensor networks with β-QoM. IEEE Sens. J. 2012;12(6):1726–35.
Chow, Kit-Yee, Lui, King-Shan, Lam, E.Y. 2007. Achieving 360° angle coverage with minimum transmission cost in visual sensor Networks. In: IEEE Wireless Communications and Networking Conference, pp.4112-4116.
⟨http://www.cloudbus.org/cloudsim/⟩.
Dantu, K., Rahimi, M., Shah, H., Babel, S., Dhariwal, A., Sukhatme, G., 2005. Robomote: enabling mobility in sensor networks. In: Fourth International Symposium on Information Processing in Sensor Networks, pp. 404–409.
Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Commun. ACM 2008:107–13.
⟨http://docs.continuum.io/anaconda-cluster/examples/spark-numbapro⟩.
El-Moukaddem F, Torng E, Guoliang X, Torng E, Xing G. Mobile relay configuration in data-intensive wireless sensor networks. IEEE Trans. Mob. Comput. 2013;12(2):261–73.
Guo, L., Kim, D., Li, D., Chen, W., Tokuta, A.O., 2014. Constructing belt-barrier providing β-quality of monitoring with minimum camera sensors. In: 23rd International Conference on Computer Communication and Networks, pp. 1–8.
Halgamuge MN, Zukerman M, Ramamohanarao K, Vu HL. An estimation of sensor energy consumption. Prog. Electromagn. Res. B 2009;12:259–95.
Hoseini, S.M., Dehghan, M., Pedram, H. 2012. Full angle coverage in visual sensor networks. In: 2nd International eConference on Computer and Knowledge Engineering (ICCKE), pp. 260–265.
Janansefat, S., Akkaya, K., Senturk, I.F., Gloff, M., 2013. Rethinking connectivity restoration in WSNs using feedback from a low-cost mobile sensor network testbed. In: IEEE 38th Conference on Local Computer Networks Workshops (LCN Workshops), pp. 108–115.
Jiang L, Xu LD, Cai H, Jiang Z, Bu F, Xu B. An IoT-oriented data storage framework in cloud computing platform. IEEE Trans. Ind. Inform. 2014;10(2):1443–51.
Jiang D, Xu Z, Wang W, Wang Y, Han Y. A collaborative multi-hop routing algorithm for maximum achievable rate. J. Netw. Comput. Appl. 2015.
Keung GY, Li B, Zhang Q. The intrusion detection in mobile sensor network. IEEE/ACM Trans. Netw. 2012;20(4):1152–61.
Lakshman A, Malik P. Cassandra: a decentralized structured storage system. SIGOPS Oper. Syst. Rev. 2010;44(2):35–40.
Li, P., Luo, Y., Zhang, N., Cao, Y. 2015. HeteroSpark: s heterogeneous CPU/GPU Spark platform for machine learning algorithms. In: IEEE International Conference on Networking, Architecture and Storage (NAS), pp. 347–348.
Lima DHS, Aquino ALL, Ramos HS, Almeida ES, Rodrigues JJPC. OASys: an opportunistic and agile system to detect free on-street parking using intelligent boards embedded in surveillance cameras. J. Netw. Comput. Appl. 2014;46:241–9.
Mehta, V., Sheng, W., Chen, T., Shi, Q. 2009. Development and calibration of a low cost wireless camera sensor network. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, pp. 110–115.
O'Toole AJ, Phillips PJ, Jiang Fang, Ayyad J, Penard N, Abdi H. Face recognition algorithms surpass humans matching faces over changes in illumination. IEEE Trans. Pattern Anal. Mach. Intell. 2007;29(9):1642–6.
Sahoo PK, Sheu JP, Hsieh KY. Target tracking and boundary node selection algorithms of wireless sensor networks for internet services. Inf. Sci. 2013;230:21–38.
Shih, Kuei-Ping, Chou, Chien-Min, Liu, I-Hsin, Li, Chun-Chih, 2010. On barrier coverage in wireless camera sensor networks. In: 24th IEEE International Conference on Advanced Information Networking and Applications (AINA '10), pp. 873–879.
Sinha A, Chandrakasan A. Dynamic power management in wireless sensor networks. IEEE Des. Test Comput. 2001;18(2):62–74.
Tao Dan, Tang Shaojie, Zhang Haitao, Mao Xufei, Ma Huadong. Strong barrier coverage in directional sensor networks. Comput. Commun. 2012;35(8):895–905.
Vecchio M, Lopez-Valcarce R. Improving area coverage of wireless sensor networks via controllable mobile nodes: a greedy approach. J. Netw. Comput. Appl. 2015;48:1–13.
Wang, Yi, Cao, Guohong, 2012. Barrier coverage in camera sensor networks. In: Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '11), ACM, New York, NY, USA, p. 12.
Yang SJ, Chen YR. Design adaptive task allocation scheduler to improve MapReduce performance in heterogeneous clouds. J. Netw. Comput. Appl. 2015;57:61–70.
Zaharia, M., Das, T., Li, H., Shenker, S., Stoica,I. 2012. Discretized streams: an efficient and fault-tolerant model for stream processing on large clusters. In: Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing, Berkeley, CA, USA, p. 10.
Zanella A, Bui N, Castellani A, Vangelista L, Zorzi M. Internet of things for smart cities. IEEE Internet Things J. 2014;1(1):22–32.