Kinematics Based Approach for Data Reduction in Wireless Video Sensor Networks

Christian Salim[a,b], Abdallah Makhoul[a], Rony Darazi[b] and Raphaël Couturier[a]
[a]FEMTO-ST Institute, Univ. Bourgogne Franche-Comté, CNRS, France, Email: firstname.lastname@univ-fcomte.fr
[b]TICKET Lab, Antonine University, Lebanon, Email: firstname.lastname@ua.edu.lb

***Abstract*—Recently, Wireless Video Sensor Networks (WVSNs) have been one of the most used technologies for surveillance, event tracking, nature catastrophe and other sudden events. Those networks are composed of small embedded camera motes which help to extract the needed information for the monitored zone of interest. A WVSN is divided into 3 different layers: the video sensor-node layer, the coordinator layer and the sink. Every video sensor-node is in charge of capturing the raw data of images and videos and sending it to the coordinator for further analysis before sending the analyzed data to the sink. In a normal scenario, the load of collected images and videos from different sensor nodes on the same network is huge. Sending all the images from all the sensor nodes to the coordinator consumes a lot of energy on every sensor, and may cause a bottleneck. In this paper, some processing and analysis are added based on the similarity between frames on the sensor-node level to send only the important frames to the coordinator. Kinematic functions are defined to predict the next step of the intrusion and to schedule the monitoring system accordingly. Compared to a fully scheduling approach based on predictions, this approach minimizes the transmission on the network. Thus, it reduces the energy consumption and the possibility of any bottleneck while guaranteeing the detection of all the critical events at the sensor-node level as shown in the experiments.**

***Index Terms*—wireless video sensor networks; shot similarity; video aggregation; frames similarity; event detection.**

I. Introduction

In wireless video sensor networks, WVSN, the event driven and periodic approaches are combined. The wireless video sensor networks are 3 layers network: The Wireless Sensor Node level, The Coordinator level and the Sink as shown in Figure 1. The wireless video sensor nodes have very limited energy resources. Those nodes are responsible of monitoring a well determined area of interest. Thus, to monitor an area, they film it according to their field of views (FOVs) and send those videos to the coordinator. Filming a video and capturing its frames consume a lot of energy especially since each sensor is sending a big number of frames to the coordinator. To reduce the energy consumption on the sensor node level, the main target is to reduce the energy consumption related to the sensing process, and to the transmission phase.

Normally if there is no critical event in the area of interest, the WVSN operates periodically [1]. To reduce the energy consumption related to the sensing process on the sensor level, a simple probabilistic method has been proposed to adapt the frame rate of every sensor node, depending on the level of criticality, the position and the trajectory direction of the intrusion in the scene (the movement's vector) based on a probability based prediction [2].

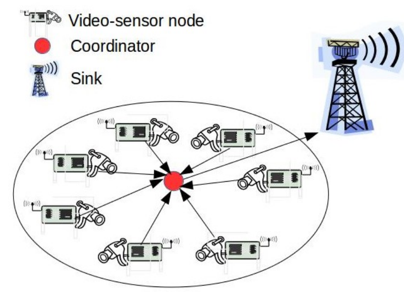

Fig. 1. Architecture of WVSN

After adapting the number of frames sensed by the node in every period, the main goal is to reduce the number of frames sent from the sensor node to the coordinator. To reduce this number, each sensed frame is compared with the last frame sent. This comparison is an edge based comparison: according to [3], we are interested in the simple image processing algorithms. Local (on-board) processing of the image data reduces the total amount of data that needs to be communicated through the network. Local processing can involve simple image processing algorithms (such as background substraction for motion/object detection, and edge detection). According to a predefined threshold of similarity, the node decides whether to send this frame to the coordinator or not [1].

The coordinator of an area of interest serves as a cluster head leading a defined number of nodes. To sum up, we try to detect critical events and track intrusions through the area of interest via a wireless video sensor network, while using an energy efficient method.

The reminder of this paper is divided as follows: section II introduces the state of the art, section III explains in brief the Movement State Vector for later use in the kinematic based approach. For Data Reduction on the sensing phase, the kinematic based and position based approaches used for predictions, as well as the frame rate adaptation technique are explained in section IV. Data Reduction technique on the transmission phase is discussed in section V. In section VI, some experimental results validate the approach. At the end, section VII concludes the paper.

II. Related work

In this section, some previous works regarding this topic in the literature are explored. Several works dealt with the location of the intrusion and the target in the zone of interest [4], [5], [6], [7], [8], [9]. In [4], the authors implement an evolution of the time difference of arrival (TDOA) using 4 stationnary sensor nodes as reference nodes in the area of interest. The intersection of several hyberboles are studied to perform the localization process which needs 4 equations from the 4 reference nodes. This new method outperforms the traditional TDOA in terms of error rate. Machine learning has been used for target localization in WSN in [5]. In [6], [7], [8], [9], the Intrusion Detection System (IDS) has been used. This system divides the network into several areas and each area into sub-areas. The location of the intrusion is equal to the group position. In [10], the authors develop a small algorithm that detects and tracks a moving target in WSN. The sensor that detects the intrusion sends an alert according to the projected path of the target. They use the triangulation method to locate the target.

Several works have studied the path of an intrusion in the area of interest in the literature. In [11], the authors develop a method that exploits the traces of target presence. The sensor node plays a role in decreasing a trace intensity with time and propagates them to the network. A tracking agent is used to follow the traces. This agent can be a pure software originated by a mobile sink, a human being with a device to communicate with the network or a mobile robot. The objective is to follow the traces from the first alert message detecting a target, the agent can immediately start with the first trace by studying the intensity of the trace, and then by grouping 2 or more traces, the path can be built. In [12] the target is considered as an unkown sensor with RFF. Measurement selection in this work are based on fuzzy modeling, the position estimation is aggregated through neighborhood functions. An optimization of this work is done by the Generalized Kalman Filter method. The authors in [13] compare between several target tracking approaches in WSN. They show the different aspects of tracking: security, energy efficiency, network structure, accuracy, mobility of the target, fault tolerance... Several metrics in [13] have been taken into consideration:

1) The network structure : Tree Structure, Face structure or Cluster structure

2) Prediction-based Tracking: to predict the next position of the target several approaches use the kinematics functions, Kalman filter, extended Kalman filter and particle filter.

3) The number of targets: it differs if there is 1 target or several targets which can be more consuming with high complexity.

4) Type of the Object: Discrete (people, animals, vehicles) or Continuous (forest fires, oil spills, ...).

Several reasons may cause the loss of the target in [13] such as communication failures, abrupt changes in the speed and direction of the target, the energy-hole problem, the inaccuracy and the delay caused by the computation of the algorithms.

Authors in [14], [15] adopted spanning tree methods to locate and track the intrusions in the area of interest in WSN.

A lot of works focus on cluster-based approaches to track the targets in the network [16], [17], [18], [19], [20], [21]. In [19], they use a hybrid cluster-based target tracking. This method has static clusters, each node cooperates with its cluster head to track the mobile target. Once a target approaches the boundaries, sensors from different clusters can cooperate forming a dynamic cluster on the borders. When the target moves away, the dynamic cluster is dismissed. It is energy consuming and uses a lot of overhead to form and dismiss a cluster.

Some papers concentrate on the contour tracking using minimal contour tracking algorithms to improve energy-efficency such as [22] where all sensors in the contour are on and all the others are off depending on a sleep schedule. In [23], [24], the authors transform the whole tracking area to voronoi polygons while having 3 kinds of sensors: workers, border and computational.

Other approaches are based on kinematics rules and the probability theory [2], kinematics rules are used to detect the path of the target in the network, and the probability theory can help to schedule the sleep mechanism of the sensors according to the trajectory of the target and its future predicted positions, acceleration and speed.

In this paper, 2 sensor nodes are neighbors as shown in Fig 2 if there is a geometrical correlation between their FOVs. A node to node connection is established. If an intrusion is detected in sensor node S_1, this sensor studies the direction vector of the trajectory D of this intrusion, and send this vector to all its neighbors. In this figure S_2 is the neighboor that receives the vector. In other terms, this vector also represents an alert that an intrusion is taking place in the monitored area.

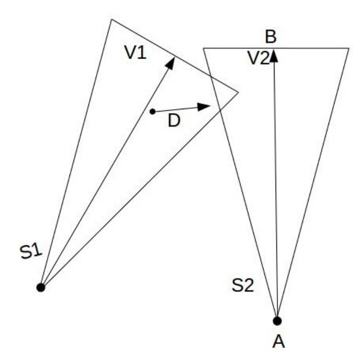

Fig. 2. Direction Vector

In the kinematic based approach (KBA) proposed in this paper, every sensor node starts the sensing process by taking as a frame rate the minimum frame rate possible [1]. When this sensor is triggered by another node (after the probability study of the trajectory and the new location of the intrusion) that an intrusion may pass by its FOV, it adapts the frame rate according to the level of probability sent in the alarm message from the sending sensor. In this paper, we consider that a sensor node can determine the position of the intrusion at detection [2].

III. Sensor node level: Sensing Phase

In the sensing phase section, the probability based prediction method is introduced to determine the trajectory of the intrusion. This process leads to adapt the frame rate of the triggered sensors in the region of the predicted path. This adaptation of the frame rate differs in the same region between sensor nodes according to the probability. This probability defines the percentage of the intrusion to pass by this sensor's field of view (FOV). It all depends on the trajectory vector generated from the first sensor that detects the intrusion.

When a node detects a target, it sends an alarm to actively raise the frame rate of all its neighbor sensor nodes. In this case all these nodes will be ready to detect the approaching target. But the target may move to a side that is disproportional to some sensor's field of views, in this case the variation of the frame rate varies between sensor nodes depending on the direction of the target. A node by which the intrusion may not pass by is called a low-probability sensor node, this node raises its frame rate to a certain extent in case any abrupt change of direction or speed of the intrusion happens. However, a node by which the intrusion has a high probability to pass by, raises its frame rate to the maximum frame rate.

The target prediction technique is based on two big studies: first, the kinematics rules to calculate the expected displacement of the intrusion (position after a certain time t and moving direction). Secondly, the probability theory to adapt the frame rate of each sensor node. Based on these predictions, the decision of raising the frame rate in each sensor node is made according to the probabilistic study that also schedules the time to raise the frame rate when the probability is close to one. This approach reduces the energy consumption of the whole network by selecting the sensors that need to adapt their frame rate not only in the whole network but also in a very specific region.

To summarize, 3 steps are needed to apply the KBA (kinematics based approach) approach on the sensor node level:
1) The target prediction.
2) The reduction of the number of triggered node to adapt their frame rate.
3) The adaptation time control: based on the probability, the approach schedules the node to adapt its frame rate when the probability of detecting the target in its FOV is equal or close to 1.

A. Movement State Vector

Let us start by defining the Movement Vector. The movement state vector is a combination of 5 parameters related to the movement. $\overrightarrow{MS}(n)$={t_n, x_n, y_n, $\overrightarrow{v}(n)$, $\overrightarrow{a}(n)$}, where: t_n is the actual time, x_n and y_n define the position of the target at t_n, $\overrightarrow{v}(n)$ is the average velocity vector defined by its scalar speed v_n and its moving direction θ_n and $\overrightarrow{a}(n)$ refer to the acceleration during [t_{n-1},t_n].

B. Current State Movement Calculation

First to calculate the first movement state vector of the intrusion, the first two detected positions of the intrusion are needed. To be able to calculate the movement state vector at time t_n, all we need to have are $\overrightarrow{MS}(n-1)$ and the position x_n, y_n at t_n so the functions can calculate the velocity $\overrightarrow{v}(n)$(v_n,θ_n) and acceleration $\overrightarrow{a}(n)$ vectors to compose the new movement state vector $\overrightarrow{MS}(n)$ at t_n as shown in Fig 3 and in the equations below:

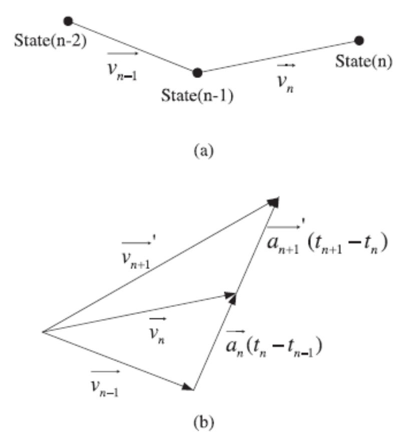

Fig. 3. Movement State Vector and kinematics rules

$$v_n = \frac{\sqrt{(y_n - y_{n-1})^2 + (x_n - x_{n-1})^2}}{t_n - t_{n-1}} \tag{1}$$

$$\theta_n = \arctan \frac{y_n - y_{n-1}}{x_n - x_{n-1}}, x_n \neq x_{n-1} \tag{2}$$

$$\overrightarrow{a_n} = \frac{\overrightarrow{v_n} - \overrightarrow{v_{n-1}}}{t_n - t_{n-1}} \tag{3}$$

In this approach, the sensor nodes are time synchronized locally in a small range, so that the received t_{n-1} may be used in the calculation. Local time synchronization may be easily achieved with a protocol such as RBS [26], or simply with HELLO message exchange [27].

IV. Predictions

In this section, the kinematics-Based prediction technique is discussed.

In this type of prediction, we suppose that $\tau = t_{n+1} - t_n$ as the lapse of time between two states, and all the prediction vectors are labeled with an apostrophe such as $\overrightarrow{MS_{n+1}}'$.

A. Kinematics-Based

The kinematics rules are used to generate a prediction about $\overrightarrow{v_{n+1}}'$ and $\overrightarrow{MS_{n+1}}'$. To do so, we consider the acceleration $\overrightarrow{a_n}$ as constant during a defined time of $t_{n+1} - t_n$ indeed, according to the displacements taylor polynomial, its rate of change can be ignored because it is the third derivative of displacement. In this case, the predicted velocity, acceleration, trajectory and position of any intrusion can be computed based on the kinematics rules and equations as follows:

$$\overrightarrow{a_{n+1}}' = \overrightarrow{a_n} \tag{4}$$

$$\overrightarrow{v_{n+1}}' = \overrightarrow{v_n} + \overrightarrow{a_{n+1}}' \times \tau \tag{5}$$

$$\overrightarrow{MS_{n+1}}' = \overrightarrow{v_{n+1}}' \times \tau + \overrightarrow{a_{n+1}}' \times \tau^2 \tag{6}$$

B. Position-Based

To be able to compute the following predicted positions, at least the first two positions of the intrusion must be detected before any prediction.

$$\overrightarrow{(x_{n+1}, y_{n+1})}' = \overrightarrow{(x_n, y_n)} + \overrightarrow{MS_{n+1}}' \tag{7}$$

The fact of considering $\overrightarrow{a_{n+1}}' = \overrightarrow{a_n}$ should to be respected over a time $\tau = t_{n+1} - t_n$ for the estimation. But once the sensor senses the new position of the intrusion, the new $\overrightarrow{a_{n+1}}$ is computed and taken into account in the rest of the processing. The distance between position n and $n + 1$ is calculated as follows:

$$d'_{n+1} = v'_{n+1} \times \tau \tag{8}$$

$$d'_{n+1} = \sqrt{(x_n - x_{n+1})^2 + (y_n - y_{n+1})^2} \tag{9}$$

The main target in this work is to get the estimation of the new position of the intrusion (x_{n+1}, y_{n+1}). A straight line can be drawn for y displacement as a function of x such as:

$$y'_{n+1} = \frac{y_n - y_{n-1}}{x_n - x_{n-1}} \times x'_{n+1} + b \tag{10}$$

This above equation is also valid for x_n and y_n already sensed and stored, their values are used to get b. In the next step, a system of two equations for x and y are used to get x'_{n+1} and y'_{n+1} as follows:

$$\begin{aligned} x'_{n+1} &= \frac{1}{2} a'_{n+1}\tau^2 + v'_{n+1}\tau + x_n \\ y'_{n+1} &= \frac{1}{2} a'_{n+1}\tau^2 + v'_{n+1}\tau + y_n \end{aligned} \tag{11}$$

C. Adaptive frame rate function

In this section the Adaptive Frame Rate function is discussed. This function is dedicated to changing the frame rate of each sensor node according to the direction conditions, the generated probabilty and the maximum Frame Rate of the sensor node.

Each sensor that detects the intrusion sends an alarm message to its neighbor nodes. This message contains the ID and the position of the alarm node, as well as the movement state vector $\overrightarrow{MS_n}$ and the prediction results ($\overrightarrow{MS_{n+1}}'$,$x'$,$y'$).

Every sensor that receives the message checks the received values and predictions with its own FOV. If the intrusion is set to pass by this sensor the probability $prob$ is different than 0. In this case, this sensor node increases its frame rate FR according to the predefined maximum frame rate FR_{max} as follows:

$$FR = FR_{max} \times prob \tag{12}$$

As shown in Figure 4, if the probability $prob$ is close to 1 then the frame rate of this sensor node increases and approximately reaches its maximum frame rate available. The probability is computed based on the number of periods nb_p needed by the intrusion to reach the sensor-node in question as the equation below shows:

$$prob = \frac{1}{nb_p} \tag{13}$$

The number of periods needed is computed based on the distance between the intrusion and the sensor node and the speed of the intrusion.

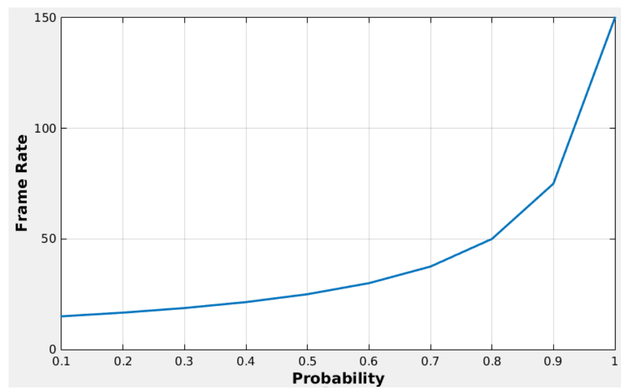

Fig. 4. Adaptative Frame Rate Function

V. Sensor node level: Transmission Phase

After adapting the number of frames sensed by the node in every period, the aim is to reduce the number of frames sent from the sensor node to the coordinator. To reduce this number, each sensed frame is compared with the last sent frame to the coordinator. This comparison is an edge based comparison: according to the study in [3], we are interested in the simple image processing algorithms. Local (on-board) processing of the image data reduces the total amount of data that needs to be communicated through the network. Local processing can involve simple image processing algorithms (such as background substraction for motion/object detection, and edge detection). According to a predefined threshold of similarity, the node decides whether to send this frame to the coordinator or not [1].

VI. Experimental Results

In this section, several experiments have been conducted using MATLAB simulator, connected to multiple Microsoft Vx-800 cameras. A defined scenario has been taken into consideration as shown in Figure 5 and Table I. This table represents the sensor nodes in their active mode, when the intrusion is in their FOV, and in their passive mode when no intrusion is detected in their area of interest.

TABLE I
Scenario Time Table

Sensor	Active mode	Passive mode
S1	64s	656s
S2	64s	656s
S3	304s	416s
S4	244s	476s
S5	64s	656s
S6	0s	720s

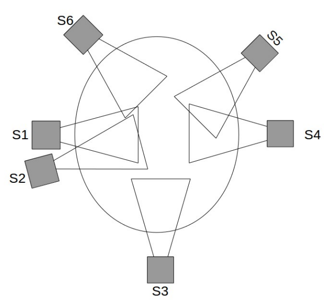

Fig. 5. Network Setup

A stable frame rate of 30 fames per second (Fps) is set to be the frame rate of the cameras. The maximum frame rate in those experiments is decreased to 15 Fps. This approach is compared to the PPSS algorithm proposed in [2]. The main purpose in this work is to send to the coordinator the frames that represent the critical situations. The coordinator reacts accordingly. We have used 6 Microsoft LifeCam VX-800 cameras to film a short video of 750 seconds, each camera is connected to a laptop to do the processing via a Matlab simulator. In this study, an intrusion has been detected in the sensor-nodes at the following time-intervals:
S1: 60 seconds from 30 to 90.
S2: 60 seconds from 70 to 130.
S3: 300 seconds from 120 to 420
S4: 240 seconds from 400 to 640
S5: 60 seconds from 630 to 690
S6: 0 seconds. The process has been run for 720 periods, each period consists of 1 second, with a frame rate equal to 30 frames per second. The frame rate in each sensor node changes independantly according to the theory explained above. In each period, every sensor node senses a certain number of frames according to the assigned frame rate. The minimum frame rate is set to $FR = 1$ frame per period. The initial and maximum frame rate is considered as $FR = 15$ frames per period. In this case the sensor node senses 15 frames from the 30 ones in the period.

Then, the PPSS approach has been implemented like in [2] for the same video sequence. This algorithm adopts the normal law of probability and the kinematics rules. Its role is to schedule the monitoring time of the sensor-node depending on the trajectory of the intrusion and the time needed to reach its FOV and the sensor-node sends all the sensed frames to the coordinator while the intrusion is in its FOV, and then it goes back to the sleep mode. But after several experiments, this approach tends to lose information up to 15% due to probability errors. This loss of data in PPSS is shown in Figures 6 and 7 for sensor S1 in our scenario.

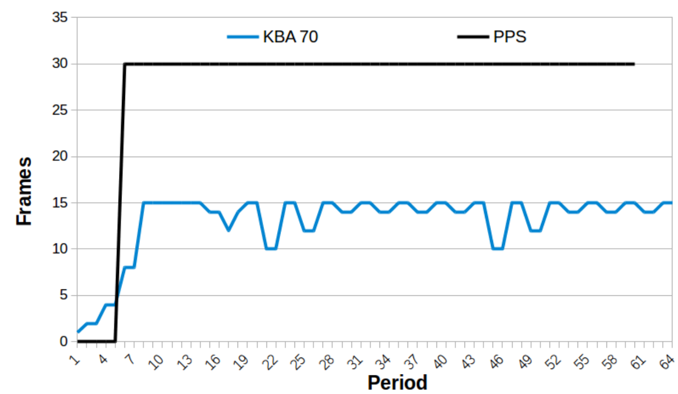

Fig. 6. Difference between KBA and PPSS on the sensing phase

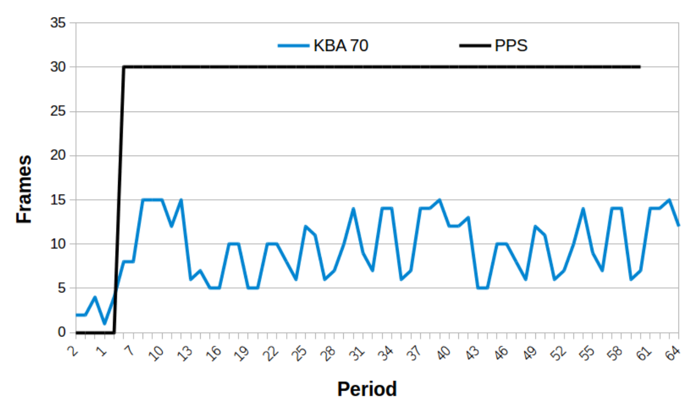

Fig. 7. Difference between KBA and PPSS on the transmission phase

A. Data Reduction

In this section, the biggest challenge in WSN is exploited, which is the energy consumption problem due to the limited resources of the sensor nodes and to the big number of data (frames) transmitted all over the network. On the sensor node level, the energy consumption problem is the main issue as well as the bandwidth limitations. In the proposed scenario, when no adapted frame rate is implemented on each sensor node, the amount of sensed frames remains at 30 for each period. In terms of energy consumption and bandwidth usage, sending all the frames is costly while a lot of frames are identical and do not represent any criticality. Sending frames with a time difference inferior to 0.03 seconds in a video surveillance does not represent any additional information. For this reason, we set the initial and maximal frame rate to $FR = 15$ frames sensed per period. The proposed method is implemented on every video-sensor node to reduce the number of frames sensed and sent to the coordinator. For every sensor node, the frame rate is adapted after two periods where $P = 1$ second. Every sensor node sends the first frame of each period. For sensor node $S1$, as seen in Figure 7, this technique only sends the critical frames to the coordinator according to a predefined threshold of similarity as explained in the upper sections, this threshold is set to 70% [1]. The number of frames sent in each period is the parameter that influences the frame rate. The frame rate variation seen in Figure 6 validates the frame rate adaptation method in the active mode of sensor S1, when an intrusion is detected.

Figure 7 reveals the number of critical frames sent to the coordinator via $S1$, this variation in the number of critical frames per period is proportional to the adaptation of the frame rate. As seen in Tables II and III for S1 and in Tables IV and V for S3, adapting the frame rate reduces the sent data by more than 90%. Then, applying the similarity function proposed in this paper causes the degradation of the number of sent frames by 92% from $129,600$ frames to $11,041$ frames. Tables VI,VIII show the data reduction caused by KBA on the sensing and transmission levels.

TABLE II
DATA REDUCTION FOR S1 OVER 64S

Nb of Periods	All Frames	Sampled Frames	Critical Frames
64	1920	730	540

TABLE III
DATA REDUCTION FOR S1 OVER 720S

Nb of Periods	All Frames	Sampled Frames	Critical Frames
720	21600	1386	1196

TABLE IV
DATA REDUCTION FOR S3 OVER 300S

Nb of Periods	All Frames	Sampled Frames	Critical Frames
304	9120	4407	3100

By comparing these numbers to the number of frames in Tables IX, X, XI, XII, while applying PPSS algorithm, the efficiency of the KBA approach for the sensing and transmission process surpasses the PPSS algorithm. This gain grows

TABLE V
DATA REDUCTION FOR S3 OVER 490S

Nb of Periods	All Frames	Sampled Frames	Critical Frames
720	21600	4437	3530

TABLE VI
NUMBER OF FRAMES IN ACTIVE AND PASSIVE MODES FOR KBA APPROACH

Sensor	P. Mode Sensed	P. Mode Transmitted	A. Mode Sensed	A. Mode Transmitted
S1 (64s)	656	656	730	540
S2 (64s)	658	658	788	585
S3 (304s)	430	430	4407	3100
S4 (244s)	482	482	3523	2620
S5 (64s)	650	650	760	600
S6 (0s)	720	720	0	0
Total	3596	3596	10190	7445

TABLE VII
NUMBER OF FRAMES IN ACTIVE AND PASSIVE MODES FOR PPSS METHOD

Sensor	P. Mode Sensed	P. Mode Transmitted	A. Mode Sensed	A. Mode Transmitted
S1 (60s)	0	0	1618	1618
S2 (60s)	0	0	1454	1454
S3 (300s)	0	0	6778	6778
S4 (240s)	0	0	5439	5439
S5 (60s)	0	0	1405	1405
S6 (0s)	0	0	0	0
Total	0	0	17694	17694

TABLE VIII
DATA REDUCTION ON THE OVERALL NETWORK

	All-Frames	KBA Sensed	KBA Transmitted	PPSS Sensed	PPSS Transmitted
Total	129600	13786	11041	17694	17694

furthermore when the time interval of the active mode of the sensor grows, as shown for sensor-node S3 when comparing the values in Table V to the values in Table XII. For probability reasons, the first sequence of frames for every sensor is lost in PPSS, once the intrusion opts in the FOV of the sensor node. Tables VI, VII and VIII show the efficiency of KBA approach sensor by sensor and on the overall network regarding the number of sensed and transmitted frames.

TABLE IX
DATA REDUCTION FOR S1 OVER 60S PPSS

Nb of Periods	All Frames	Sampled Frames	Critical Frames
60	1800	1618	1618

TABLE X
DATA REDUCTION FOR S1 OVER 700S PPSS

Nb of Periods	All Frames	Sampled Frames	Critical Frames
700	21000	1618	1618

TABLE XI
DATA REDUCTION FOR S3 OVER 300S PPSS

Nb of Periods	All Frames	Sampled Frames	Critical Frames
300	9000	7778	7778

TABLE XII
DATA REDUCTION FOR S3 OVER 700S PPSS

Nb of Periods	All Frames	Sampled Frames	Critical Frames
700	21000	7778	7778

As for the bottleneck issue, the bandwidth capacity is the main concern and the KBA approach decreases the use of this capacity by reducing the number of frames transmitted all over the network as shown in Table XIII. Table XIII shows the upper hand of KBA over PPSS in the bandwidth consumption reduction.

TABLE XIII
THE ULTIMATE BANDWIDTH TOTAL REDUCTION KBA AND PPSS

Approach	Nb of Periods	All Frames	Sampled	Critical
KBA	720	2640 MB	275 MB	220 MB
PPSS	700	2520 MB	354 MB	354 MB

B. Energy Consumption Study

In this section, an energy model is adopted from [27] where the radio energy for the transmission of the data on the radio and the computational energy for the in-node processing are the core of the energy consumption of every sensor node as shown in the equation below:

$$E = E_{radio} + E_{comp} \tag{14}$$

Table XIV shows the different parameters to compute the energy consumption while considering:
I_{TX} and I_{RX} the electric power needed to respectively send and receive by the radio while T_{TX} and T_{RX} the respective corresponding operating times over 1 byte, and V is the constant voltage supply throughout the transmission.

$$E_{radio}(k) = k.I_{TX}.V.T_{TX} + k.I_{RX}.V.T_{RX} \tag{15}$$

Taking into account that k is the number of bytes sent from a specific sender to a specific receiver. For the computational energy consumption:
ϵ_{add},ϵ_{mul},ϵ_{cmp},ϵ_{sht} are the basic operations (shift,addition,comparison,multiplication, etc...), Table XIV shows the required energy for each operation. To compute this energy consumption, the number of each basic operation in the algorithm must be counted:

$$E_{comp} = N_{add}\times\epsilon_{add}+N_{sht}\times\epsilon_{sht}+N_{mul}\times\epsilon_{mul}+N_{cmp}\times\epsilon_{cmp} \tag{16}$$

TABLE XIV
PARAMETERS OF THE ENERGY MODEL

Parameter	Value
I_{TX}	17.4 mA
I_{RX}	19.7 mA
T_{TX}	3.2×10^{-5} s
T_{RX}	3.2×10^{-5} s
V	3.3 V
I_{cpu}	31 mA
f_{cpu}	48 MHz
ϵ_{add}	2.13 nJ
ϵ_{mul}	6.39 nJ
ϵ_{cmp}	2.13 nJ
ϵ_{sht}	4.26 nJ

In order to compare both approaches, the two components of the energy consumption have been computed using a CC2420 radio transceiver and an ARM7TDMI microprocessor. Table XIV displays the parameters that are used in the calculations and which are found in the data sheets of the node's components [27].

C. Sensor Node Level

In the experiments, when running the KBA technique, 9262 frames were sensed and compared using the similaritiy function. For a 640×480 frame size, $307,200$ pixels exist in each frame. Each similarity takes into account all the pixels in every frame. The KBA approach consists of 1 comparison. The maximum computational energy for E_{comp} for $9,262$ similarities is computed as follows:

$$E_{comp} = 13,768 \times 640 \times 480 \times \epsilon_{cmp} \tag{17}$$

For KBA, $E_{comp,KBA}$=9 J.
For PPSS, $E_{comp,PPSS}$=0.1 J.
To move on to the transmission phase, using KBA, where the network transmits $11,041$ frames = 220 MB, comparing to the $17,694$ frames = 354 MB for PPSS.

$$E_{radio,KBA} = 423J \tag{18}$$

$$E_{radio,PPSS} = 682J \tag{19}$$

The total energy consumption is computed as follows:

$$E_{KBA} = E_{comp,KBA} + E_{radio,KBA} = 432J \tag{20}$$

$$E_{PPSS} = E_{comp,ppss} + E_{radio,ppss} = 682J \tag{21}$$

Based on Figure 8, KBA algorithm consumes more energy on the computational level, but reduces much more energy on the transmission level. Figure 8 compares both approaches in terms of energy consumption on the overall network while considering a starting energy of $1,000$ J for the network.

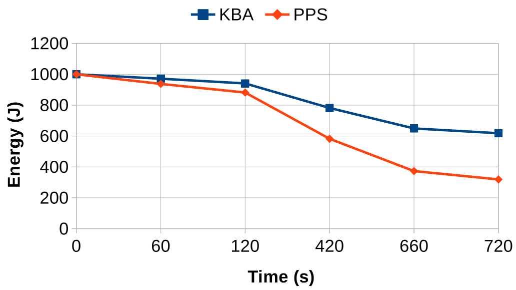

Fig. 8. Energy consumption comparison for KBA and PPSS

VII. Conclusion

In this paper, a kinematics based approach for an adaptive frame rate with a similarity detection function for wireless video sensor nodes has been introduced. The conducted experiments show that the proposed algorithm did not miss any event in the recorded video sequence. Thus, the algorithm sends the minimum required frames to the coordinator node by using a similarity detection function at the sensor node level. The selected frames are transmitted by the sensor nodes to the coordinator without missing any required information. The results show that the size of the transmitted data in each period is reduced and the energy consumption is decreased, thus, preventing any bottleneck problem regarding the bandwidth limitation issue.

Comparing KBA approach with [26] in terms of data reduction and energy consumption, helps us to find out that KBA approach outperforms PPSS, and reduces the number of data for more than 40% than PPSS. Thus, PPSS consumes 2 times more energy than KBA. As future works, some real experimentations are needed on real sensor-nodes.

Acknowledgement

This project has been performed in cooperation with the Labex ACTION program (contract ANR-11-LABX-0001-01) and partially funded with support from the National Council for Scientific Research in Lebanon CNRS-L, the Hubert Curien CEDRE programme n40283YK, and the Agence Universitaire de la Francophonie AUF-PCSI programme.

References

[1] C. Salim, A. Makhoul, R. Darazi, and R. Couturier. Combining frame rate adaptation and similarity detection for video sensor nodes in wireless multimedia sensor networks. *IWCMC*, pages 327 – 332, 2016.

[2] B. Jiang, B. Ravindran, and H. Cho. Probability-based prediction and sleep scheduling for energy-efficient target tracking in sensor networks. *IEEE TRANSACTIONS ON MOBILE COMPUTING*, 12(4), 2013.

[3] Stanislava Soro and Wendi Heinzelman. A survey of visual sensor networks. *Advances in Multimedia*, 2009:21, 2009.

[4] Alireza Ghelichi, Kumar Yelamarthi, and Ahmed Abdelgawad. Target localization in wireless sensor network based on time difference of arrival. *IEEE 56th International MWSCAS 2013*, 2013.

[5] Jaehyun Yoo and H. Jin Kim. Target localization in wireless sensor networks using online semi-supervised support vector regression. *Sensors 2015*, 15:12539–12559, 2015.

[6] Amrita Ghosal and Subir Halder. A survey on energy efficient intrusion detection in wireless sensor networks. *Journal of Ambient Intelligence and Smart Environments*, 9:239261, 2017.

[7] I. Krontiris, T. Dimitriou, and F.C. Freiling. Towards intrusion detection in wireless sensor networks. *13th European Wireless Conference*, page 17, 2007.

[8] G. Li, J. He, and Y. Fu. Group-based intrusion detection system in wireless sensor networks. *Computer Communications*, 31(18):43244332, 2010.

[9] S. Misra, P.V. Krishna, and K.I. Abraham. A simple learning automata-based solution for intrusion detection in wireless sensor networks. *Wireless Communications and Mobile Computing, Special Issue: Architectures and Protocols for Wireless Mesh, Ad Hoc, and Sensor Networks*, 11(3):426441, 2011.

[10] R. Gupta and S.R. Das. Tracking moving targets in a smart sensor network. *in proceedings of the 58th IEEE Vehicular Technology Conference*, 2004.

[11] A. Marculescu, S. Nikoletseas, O. Powell, and J. Rolim. Efficient tracking of moving targets by passsively handling traces in sensor networks. *IEEE GLOBECOM 2008*, 2008.

[12] Yingyou Wen, Rui Gao, and Hong Zhao. Energy efficient moving target tracking in wsn. *SENSORS*, 16(29), 2016.

[13] Asmaa Ez-Zaidi and Said Rakrak. A comparative study of target tracking approaches in wireless sensor networks. *Journal of Sensors*, 2016(1), 2016.

[14] A. Alaybeyoglu, A. Kantarci, and K. Erciyes. A dynamic lookahead tree based tracking algorithm for wireless sensor networks using particle filtering technique. *Computers and Electrical Engineering*, 40(2):374383, 2014.

[15] M.-X. Chen, C.-C. Hu, and W.-Y. Weng. Dynamic object tracking tree in wireless sensor network. *EURASIP Journal on Wireless Communications and Networking*, 2010, 2010.

[16] K. A. Darabkh, S. S. Ismail, M. Al-Shurman, I. F. Jafar, E. Alkhader, and M. F. Al-Mistarihi. Performance evaluation of selective and adaptive heads clustering algorithms over wireless sensor networks. *Journal of Network and Computer Applications*, 35(6):20682080, 2012.

[17] L. Cheng, C. Wu, Y. Zhang, H. Wu, M. Li, and C. Maple. A survey of localization in wireless sensor network. *International Journal of Distributed Sensor Networks*, 2012:12, 2012.

[18] A. Boukerche, H. A. B. F. Oliveira, E. F. Nakamura, and A. A. F. Loureiro. Localization systems for wireless sensor networks. *IEEE Wireless Communications*, 14(6):612, 2007.

[19] Z. Wang, Z. Wang W. Lou, J. Ma, and H. Chen. A hybrid cluster-based target tracking protocol for wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2013(1):16, 2013.

[20] J.-I. Kong, J.-W. Kim, and D.-S. Eom. Energy-aware distributed clustering algorithm for improving network performance in wsns. *Journal of Sensors*, 2014(1):10, 2014.

[21] F. Deldar and M. H. Yaghmaee. Designing a predictionbased clustering algorithm for target tracking in wireless sensor networks. *IEEE International Symposium on Computer Networks and Distributed Systems CNDS 11*, page 199203, 2011.

[22] J. Jeong, T. Hwang, T. He, and D. Du. Mcta: target tracking algorithm based on minimal contour in wireless sensor networks. *in proceedings og the 26th IEEE International Conference on Computer Communications*, page 23712375, 2007.

[23] A. A. U. Rahman, M. Naznin, and M. A. I. Mollah. Energy efficient multiple targets tracking using target kinematics in wireless sensor networks. *in proceedings The 4th International Conference on Sensor Technologies and Applications*, page 275280, 2010.

[24] W. Alsalih, K. Islam, Y. Rodriguez, and H. Xiao. Distributed voronoi diagram computation in wireless sensor networks. *in proceedings of the 20th Annual Symposium on Parallelism in Algorithms and Architectures*, 2008.

[25] Rui Dai and Ian F. Akyildiz. A spatial correlation model for visual information in wireless multimedia sensor networks. *IEEE TRANSACTIONS ON MULTIMEDIA*, 11(6):11481159, 2009.

[26] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Operating System Rev.*, pages 147–163, 2002.

[27] B. Jiang, B. Ravindran, and H. Cho. Cflood: A constrained flooding protocol for real-time data delivery in wireless sensor networks. *Proc. 11th Intl Symp. Stabilization, Safety, and Security of Distributed Systems*, pages 413–427, 2009.