Enabling Green Video Streaming over Internet of Things

(5^{th} Quarter Deliverable)

Dr. Ghalib Asadullah Shah (PI)

Next-generation Wireless Networking (NWN) Lab,
Al-Khawarizmi Institute of Computer Science,
University of Engineering and Technology, Lahore

11-3-2015

Contents

List of Figures

List of Tables

About this Document

This document reports the activities performed in the 5th quarter of our project 'Enabling Green Video Streaming over Internet of Things' and the corresponding deliverable to be submitted to ICT R & D Fund.

In the 1st deliverable we conducted a detailed literature survey of IoT communication protocols and identified the issues challenges for IoM. One of these challenges was to enable green communication for IoM. Since, multimedia traffic is bulky in size requiring frequent data transmissions, which results in high carbon footprints emission. Therefore, enabling green communication for IoM is a critical issue. In the 3rd deliverable, we identified RPL as a potential routing protocol for IoM and on the basis of a simulation study in Cooja simulator of Contiki-OS. In addition, we implemented an energy metric based objective function in Contiki-OS and compared its performance with existing ones. The outcomes of this analysis suggested that RPL is highly flexible and dynamic routing protocol, however its existing implementations do not consider green communication. Therefore, in the 4th deliverable we designed a green routing protocol based on RPL for IoM. In the proposed routing protocol referred as Green-RPL, a node chooses a preferred parent by considering a set of network metrics such as the delay constraint, battery consumption of potential parent nodes, type of energy sources along the route towards the root node, etc. In this way, the proposed Green-RPL routing protocol minimizes network carbon footprint emissions and energy consumption, while assuring application specific Quality of Service (QoS).

In this (5th) deliverable, we provide the implementation specifications for the proposed Green-RPL routing protocol. We implemented the Green-RPL in the Contiki - operating system. The implementation required some modification and addition in the already existing RPL routing protocol implementation in Contiki - operating system. In addition, to the implementation of the Green-RPL we also run several network scenarios in order to compare its performance with existing RPL implementation. The performance analysis done on the these implementations suggest that the proposed Green-RPL out performs the prior implementation of the RPL routing protocol.

In the previous deliverables, we advocate that the state of the art video encoders like H.26x and MPEG have enabled the current revolution in multimedia streaming over the networks. However, in IoT based systems the computational capabilities, memory and power are severely limited on the streaming nodes. Thus, a very simple encoder needs to be run at the low power source with the encoded stream decoded on powerful machines at the receiving end. For this reason, we proposed to reduce video encoder complexity (both space and time), by the use of compressed sensing, while maintaining fairly low bit-rates (for transmission). In 2nd quarter, we did a comprehensive performance analysis of some of the popular techniques from the compressive video acquisition literature. The evaluation was done through a simulation study and the performance of the considered techniques was verified and profiled.

In this (5th) deliverable, we have designed and implemented the CS video acquisition and further compression in bare-metal C on the 168MHz STM32F4 Discovery board. We have used FPU, DMA and DSP library on the board in order to get optimal performance. The key idea behind this implementation was to make a hybrid codex, based on the theory of compressed sensing, while maintaining some useful features of existing codex, such as H.264.

The C language code for both of the above mentioned implementations of Green-RPL routing protocol and CBS video coding are given in the CD.

Chapter 1

Development of Green Multi-hop Routing Protocol

1.1 Introduction

Internet of Things (IoT) applications involve both scaler data and multimedia data acquisition and transmission. However, the challenges of provisioning multimedia devices over IoT are overlooked in current research activities. Thereby, leaving a gap to benefit from services and applications enabled by multimedia information. The multimedia devices in Internet of Multimedia (IoM) are envisioned to be small sized resource constrained devices. These devices are battery operated and deployed in large numbers communicating via wireless links. The unstable wireless channel conditions cause frequent packet drop and high bit-error-rate, referred to as Low-power and Lossy Networks (LLNs). Moreover, the acquired multimedia data, i.e. audio, video, and audio+video, impose stringent Quality of Service (QoS) requirements in terms of network bandwidth, delay, jitter, etc [1].

The IoT systems are based on LLNs demanding a lightweight energy efficient routing protocol. For this reason, IETF Routing Over Low-power and Lossy networks (ROLL) working group has recently standardized Routing Protocol for Low Power and Lossy Networks (RPL) [2], which is a proactive distance vector routing protocol for LLNs. RPL forms a tree like network topology by maintaining a directed acyclic graph (DAG). In RPL each sensor node chooses a preferred parent towards the root node based on specific routing policies. RPL uses multiple routing metrics and constraints while optimizing an objective function to select the best path. The standard provides the choice to select appropriate objective functions as per the application requirements, which makes RPL highly adaptive and dynamic. However, so far no optimizations have been made for RPL to support multimedia communication.

Information and Communication Technology (ICT) is one of the largest contributor towards the carbon dioxide (CO_2) emissions. Therefore, low-power and green communication communication systems have been promoted in recent studies to minimize carbon footprint emissions [3, 4]. Moreover, the multimedia traffic is bulky in nature and operating on high transmission rates. Thereby, the CO_2 emissions are significantly higher in multimedia networks or IoM. In heterogeneous wireless networks like IoM, the devices are equipped with different energy sources i.e. lithium batteries, solar cells, piezoelectric energy, etc. Therefore, green communication can be enabled by adopting routes with nodes equipped with green energy source in order to minimize carbon footprint emissions.

Previously, energy efficiency of RPL have been addressed for IoT systems based on scalar sensed data. However, to the best of our knowledge no RPL implementation is designed in prior studies that incorporates the QoS requirements for multimedia communication over IoM. Similarly, green communication has attracted lot of attention and motivated researchers to reduce CO_2 emissions, yet there is no RPL implementation to enable green communication. Therefore, in this work we design an energy efficient green routing protocol (Green-RPL) for IoM. The proposed Green-RPL routing protocol is an enhanced version of RPL in which a node chooses a preferred parent by considering a set of network metrics such as the delay constraint, battery consumption of potential parent nodes, type of energy sources along the route towards the root node, etc. In this way, the proposed Green-RPL routing protocol minimizes network carbon footprint emissions and energy consumption, while assuring application specific QoS.

1.2 Design specifications for Green-RPL

In an IoT system the sensor devices are deployed in large numbers and mostly these are battery operated. Thus, their operation is must be extremely energy efficient to prolong the network lifetime. With the help of a multi-hop routing protocol, the network nodes communicate with the sink node via different multi-hop paths. Note that not every path consumes same amount of energy in LLNs. Since, nodes can experience different channel conditions or the network topology can also contribute towards the number of transmissions a packet requires before it is successfully transmitted towards the sink node. However, if a routing protocol efficiently selects those paths which consume less energy then significant amount of energy can be saved.

Moreover, the multimedia communication is bulky in nature and require large number of packet transmission. Thus, the multimedia devices are more energy hungry and their energy efficient operation is very critical for longer network lifetime. Although many previous studies have addressed energy efficiency issue in resource constrained networks. However, in prior studies have not considered multimedia communication over the resource constrained devices in an IoT system. Although lot of work is done on green communication for various wireless networking technologies, yet enabling green communication over RPL-based IoT systems is not given any consideration.

In this work, we propose an energy efficient green routing protocol for resource constrained multimedia devices in IoM named Green-RPL. The proposed routing protocol is an enhanced version of the RPL protocol that is designed for IoT systems. In our proposed Green-RPL implementation the carbon footprints emission is minimized provided that the application delay requirement and energy efficiency is guaranteed. The delay bound for multimedia applications is pre-determined. For a video application the frames can be of different sizes yet the transmitter node must ensure that 25 frames are successfully transmitted in 1 sec. Similarly, the energy efficiency is ensured by considering the quality of the intermediate links towards the root node, the energy already consumed by the possible preferred parent node, and by evaluating the potential of the parent node to support traffic requirements for yet another child node. To evaluate a parent node as per these constraints and requirements, an optimization model for the proposed Green-RPL routing protocol is designed in the following part of this section. Among all the parent nodes of a specific sensor node, the solutions of the optimization problem gives the preferred parent.

Consider the wireless link between node α and node β is denoted by (α, β). In practical wireless networks the wireless link quality (Bit Error Rate) is time varying, thus the packet transmissions are affected by the wireless channel conditions. Various methods have been proposed in the literature to estimate the link quality, such as Received Signal Strength Indicator (RSSI), Link Quality Indication (LQI), Signal to Noise Ratio (SNR). The RFC-compliant RPL routing protocol implementation in Contiki-OS provides ETX metric to estimate the link quality which measures the expected number of transmissions required to successfully deliver a packet over a specific link. Let the estimated link quality metric between node α and node β is denoted by $\ell(\alpha, \beta)$.

We define the neighboring nodes of a node α as the nodes which are in the transmission range of node α, i.e. node α can transmit packets towards them. Let the set of neighbor nodes of node α is represented by $\delta(\alpha) = \{\delta_1, \delta_2, ...\}$. Since, signal power distributions are not uniform, thus it is possible that node α can listen to some node β, however node β may not listen to node α. So, we define another set of nodes, which are able to transmit packets to node α and represent it as $\vartheta(\alpha)$. Also, consider the rank of node α is denoted by $\phi(\alpha)$. As per the specifications provided by the IETF ROLL working group for RPL implementation, a node can be a parent node ρ for a node c, if it conform the following conditions:

$$\rho \in \delta(c) \tag{1.1}$$

$$c \in \delta(\rho) \tag{1.2}$$

$$\phi(\rho) < \phi(c) \tag{1.3}$$

Let the set of parent nodes of a node c is denoted by $\xi(c) = \{\rho_1, \rho_2, ...\}$. Thereby, the nodes in set $\xi(c)$ send DIO messages to node c who then choose one of these parent nodes as a preferred parent as per the objective function. In our proposed

Green-RPL routing protocol the preferred parent is chosen as per the solution of the optimization problem given below:

$$\min. \quad \Gamma(\rho) \tag{1.4}$$

$$s.t. \quad \rho \in \xi(c) \tag{1.5}$$

$$\Lambda(\rho) < \mu^d \tag{1.6}$$

$$\Omega(\rho) < \mu^\ell \tag{1.7}$$

$$\Psi(\rho) > \mu^b \tag{1.8}$$

$$\Upsilon(\rho) > \mu^i \tag{1.9}$$

Here $\Gamma(\rho)$, $\Lambda(\rho)$, $\Omega(\rho)$, $\Psi(\rho)$, and $\Upsilon(\rho)$ represent the cumulative path carbon footprints, cumulative path delay, cumulative path link energy, battery status, and idle time for the parent node (ρ), respectively. In this optimization problem, the objective is to minimize the cumulative carbon footprint emissions on all the links along the path from parent node ρ towards the root node. Therein, if when multiple parent nodes fulfill the given constraints then the node offering the most greener path will be selected as the preferred parent. The heterogeneous smart devices in a IoM network can be equipped with distinct energy source, thus emitting disparate amount of carbon footprints. Let the carbon footprints emitted by a link ℓ within the path from parent node ρ to root node is denoted by $cf(\ell)$, then $\Gamma(\rho)$ can be given as;

$$\Gamma(\rho) = \sum_{\ell \to 1}^{L} cf(\ell) \tag{1.10}$$

here L is the total number of links in the path. Moreover, the parent node ρ needs to fulfill some other constraints to enable application specific QoS and longer network lifetime. The first constraint is very basic that is the potential preferred parent node ρ should belong to set of parents of node c i.e. both ρ and c are neighbors of each other and rank of node ρ should be less than the rank of node c. The second constraint specifies the application QoS in terms of the delay bound that is the cumulative path delay for a data packet should not increase a predefined delay threshold μ^d. In LLNs a packet may experience different wireless channel conditions, thus undergo distinct packet delays per link. Thereby, if the delay induced by a link ℓ is denoted by $d(\ell)$, then the cumulative delay of the path can be given as

$$\Lambda(\rho) = \sum_{\ell \to 1}^{L} d(\ell). \tag{1.11}$$

Energy consumption of a node significantly depends upon the link quality of the path selected for packet routing. For example, if a path with poor link quality is selected then the probability of successful delivery of packet in single transmission attempt will be very low, which may result in several retransmissions before packet gets delivered to the destination, consuming significant amount of energy. Therefore, an important metric to consider while selecting a route is to evaluate the energy cost as per the quality of links along the path which can be given as

$$\Omega(\rho) = \bar{D} \sum_{\ell \to 1}^{L} \frac{ETX(\ell) \times P_t(\ell)}{\lambda(\ell)}. \tag{1.12}$$

Here $\bar{D}$ is the data packet size, while the $P_t(\ell)$ and $\lambda(\ell)$ is the transmit power and data rate of link ℓ, respectively. $ETX(\ell) = (\ell - \tau)\left(1 + \frac{1}{\tau}\right)$ is the expected number of transmissions required to successfully deliver the packet over link ℓ and τ is the probability of successful packet transmission.

The battery status of the parent node ρ is also a critical metric, since it can influence the network lifetime and traffic load on a particular node. Consider a scenario in which a single parent node is selected by multiple child nodes as a preferred parent for traffic forwarding, in this case the parent node will quickly drain its energy resources and network lifetime will be reduced. Similarly, selecting a parent node with low remaining energy will also result in route instability as the route will be required to change in a short time when the parent node battery dies. Therefore, we propose that the child node c only select a parent node ρ as the preferred parent if its available energy resources (battery level) is higher than a pre-defined threshold μ^b. If the maximum battery energy capacity and battery energy already utilized is given by E_c and E_u, respectively, then the battery status can be given as

$$\Psi(\rho) = \frac{E_c(\rho) - E_u(\rho)}{E_c(\rho)}. \tag{1.13}$$

Lastly, the $\Upsilon(\rho)$ constraint represents the amount of time the parent node ρ keeps its radio ON, yet no activity (transmission or reception) is done during this period. It is essential to know if a node has enough idle time to support a child node. For example, in a LLN the nodes operating on very low radio duty cycles can only support a limited number of child nodes within their radio ON period. For this reason, a parent node ρ should only be selected as a preferred node if its idle time is enough for supporting another child node, i.e.

$$\Upsilon(\rho) > \frac{t_{tx}(\rho) + t_{rx}(\rho)}{c\_n(\rho)}. \tag{1.14}$$

Here $t_{tx}(\rho)$ and $t_{rx}(\rho)$ represents the time parent node ρ spends in transmission and reception mode, respectively. While $c\_n(\rho)$ denotes the number of child nodes already supported by parent node ρ. This concludes our Green-RPL routing protocol design, which considers various constraints while selecting a parent node as a preferred parent, so that an energy efficient operation along with the minimization of carbon footprints emission can be guaranteed.

1.3 Development of Green-RPL

1.3.1 Implementation

The proposed Green-RPL routing protocol is implemented in Contiki v2.7. Contiki is a wireless sensor network operating system (OS) and consists of the kernel, libraries, the program loader, and a set of processes. It is the most popular operating system for IoT things and it has been vastly used by the research community for simulation and real-implementation of IoT based wireless networks. Contiki provides mechanisms that assist in programming the smart object applications. It provides libraries for memory allocation, linked list manipulation and communication abstractions. It is developed in C, all its applications are also developed in C programming language, and therefore it is highly portable to different architectures. The Contiki operating system provides modules for different tasks (layers).

Contiki-OS contains a complete communication stack that is proposed by the research community for the IoT. The RPL routing protocol functionality is divided into multiple files each performing one of the major tasks of routing operation. RPL routing protocol files and their functionality are as follows:

- rpl-dag.c is responsible for creating and interpreting the DODAG frames,
- rpl-timers.c contains all the timers for the transmission of DIO and other control packets,
- rpl-private.h contains the definition of routing parameters,
- rpl-of0.c implements the functionality of hop-count based objective function,
- rpl-mrhof.c implements the functionality of ETX based objective function,
- rpl-icmp6.c specifies the structure of formats of the control packets,
- rpl-ext-header.c implements the header configuration for the RPL control packets,
- rpl-conf.c contains the configuration settings for RPL such as which objective function is used and so on,
- rpl.c combines the whole functionality of RPL protocol by utilizing the functionality of other files.

This division of RPL routing protocol functionality makes it easier to program different objective functions and tuning the values of key parameters to modify the routing protocols performance.

To implement our Green-RPL routing protocol we need to add another objective function in the existing RPL implementation. The new objective function is named ENERGY_ROUTE. New routing metrics i.e., energy, idle_time, path_cf, are integrated in the rpl.h file in the rpl_metric_container structure. Moreover, already existing metrics i.e., etx, avg_delay_to_sink, will also be utilized in the Green-RPL routing operation. For energy and idle_time metrics, Contiki's build in energest model is employed. The energest model determines the instantaneous amount of time a node spent in listen, transmit, sleep, and idle states. Since, each network module designed by a specific vender draws a predetermined amount of current at each state. Therefore, using the energest model the energy utilized by a particular node can be computed. Moreover, the IoT wireless motes usually operate using specific batteries such as AA-battery with 2700mAh capacity. Thus, we combined the energest model with a given set of values for current, voltage, and battery capacity, in order to compute the energy consumed by a node and accordingly the energy consumed by the route by aggregating the energy of individual nodes. For this implementation the changes were made in energest.c file that is one of the system files for the Contiki-OS.

The local routing metrics calculated by a network node needs to be shared with its neighboring nodes. In RPL routing protocol the routing metrics are exchanged via the DIO control packets that is transmitted periodically at specific intervals. Besides default parameters, we integrated idle_time, path_cf, and route_energy, in the DIO packet. This information is received by the child nodes which evaluate the possible parent node using the constraints mentioned in the optimization model presented in previous section. If the constraints are satisfied then the path metric is calculated that is the path_cf (carbon footprints of the route via considered possible parent node). The implementation is done in calculate_path_metric() function in the rpl-mrhof.c file. Once the path metric for a possible parent node is calculated, then it is compared with the path metric of current parent node. If the new path metric in lower than the current parent node than the new parent is selected and vice versa.

1.3.2 Performance evaluation

Our implementation of the Green-RPL complied and loaded successfully to our project hardware module (Arduino due). However, for testing we require the Wi-Fi link at the node module, which is due in the next deliverable. Once the wireless link is enabled the Green-RPl implementation can be tested in real network. Therefore, for now we tested the performance of Green-RPL in the COOJA simulator of Contiki-OS.

The performance of Green-RPL is compared with existing objective functions i.e. OF0 and ETX. These objective functions select preferred parent to route traffic towards the root node by considering number of intermediate hops and the number of transmissions required to deliver the packet to the root node, respectively. The simulated network scenario consist of 20 nodes transmitting multimedia traffic towards the root node via multi-hop links. The multimedia content specifically

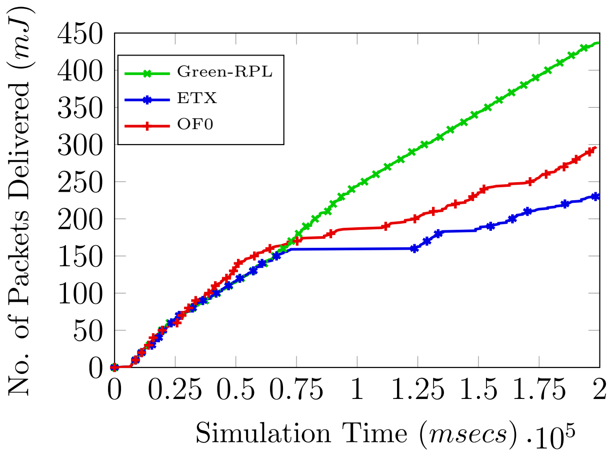


Figure 1.1: Number of packets delivered to the root node.

video content generate frames of different size at irregular intervals, thus packet generation rate is kept variable.

The network nodes are randomly distributed in an area of 200x200 meters experiencing a lossy channel model Unit Disk Graph Medium (UDGM)-distance-loss with a 2% packet loss. ContikiMAC is used as a radio duty cycling driver. As suggested in [5] DIO-Minimum-Interval, DIO-Interval-Doublings, and Radio-Duty-Cycling intervals are set to 12, 16, and 16, respectively. Half of the sender nodes are supposed to be equipped with greener energy sources (e.g. solar) and the other half have carbon emitting energy source. For comparison the network topology is kept same for all three cases, i.e. Green-RPL, OF0, ETX. The network simulation is run for 1 hour. Note that Contiki-OS provides IEEE 802.15.4 at the link layer which is proposed for the IoT communication stack. Thus, maximum packet size and data rate are restricted to 128 bytes and 250 kbps, respectively, with a packet generation rate of 25 pkts/sec. Certainly, the limited packet size and data rate of IEEE 802.15.4 questions its feasibility for multimedia communication. Yet the scope of this paper is restricted only to the performance of routing protocol and the parent selection mechanism in low-power and lossy multimedia networks. Thus these MAC layer issues are not relevant and the proposed Green-RPL mechanism can be adopted for any underlying MAC layer technology.

First, we compute the number of packets transmitted by nodes in each RPL implementation scenario. The total number of packets successfully delivered to the root node in a 20 node network is shown in Fig. 1.1. Network is simulated for 1 hour, however to present the results clearly we restrict the time window to a smaller duration. Our proposed Green-RPL algorithm provides a significant improvement in number of successful packet transmissions, as shown in Fig. 1.1. In ETX and OF0 objective function some nodes on key locations/positions are selected as preferred parent by multiple nodes, resulting a higher congestion on

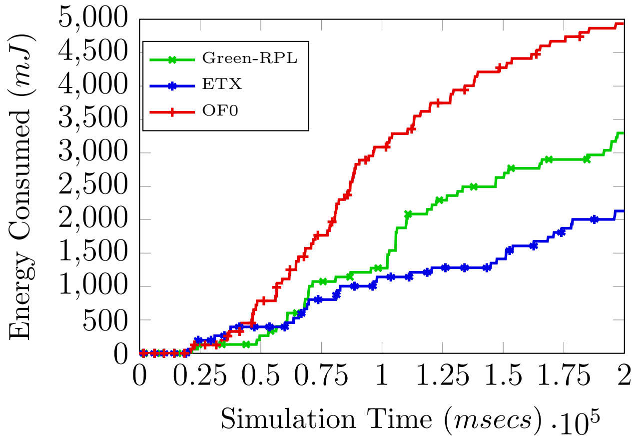


Figure 1.2: Cumulative amount of energy consumed with time.

the forwarding nodes. However, in our scheme preferred parent is selected only if it can support the traffic of another node. Thereby, a significantly high delivery rate is achieved in our proposed scheme.

To calculate energy Contiki-OS provides the timing information about different states of a node i.e. transmit, listen, etc. We computed energy as per TMote Sky specifications, voltage of 3.6 V and current values 21 mA, 23 mA, 0.21 mA, 1.2 mA, 2.4 mA, drawn in the transmit, listen, low-power-mode, idle, CPU states, respectively. The cumulative energy consumed by all the network nodes is shown in Fig. 1.2. Note that even though the energy consumption in our Green-RPL is higher than ETX, yet the number of packets transmitted in the same time by Green-RPL are significantly higher, see Fig. 1.1, thus energy consumption is higher. However, its energy efficiency can be observed by the per packet energy consumption which is 7.545 mJ for Green-RPL, 9.18 mJ for ETX, and 16.73 mJ for OF0. This shows that the proposed Green-RPL algorithm is eminently energy efficient as compared to ETX and OF0 objective functions.

Contiki-OS does not support heterogeneous energy source models. Thus, CO_2 emissions need to be inferred in a logical manner.We assume that the network contains equal number of possessing green or non-green energy source emitting no or some CO_2 footprints, respectively. The green nodes are randomly deployed in the network and identified by their IP addresses. We simulated multiple network scenarios with different network densities/topologies and observed the number of packets transmitted+forwarded by green nodes, in order to infer CO_2 emissions. Fig. 1.3 shows the normalized number of packets (ratio) sent by the green nodes in each case, which confirms that in each network scenario our proposed Green-RPL utilizes green nodes the most. Because, in ETX and OF0 the energy source of a node is not considered while selecting the preferred parent that may result in high CO_2 emissions.

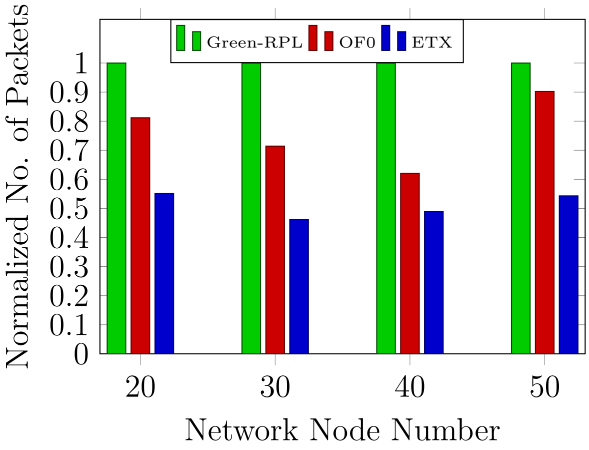


Figure 1.3: Normalized number of packets sent+forwarded by nodes possessing green energy source.

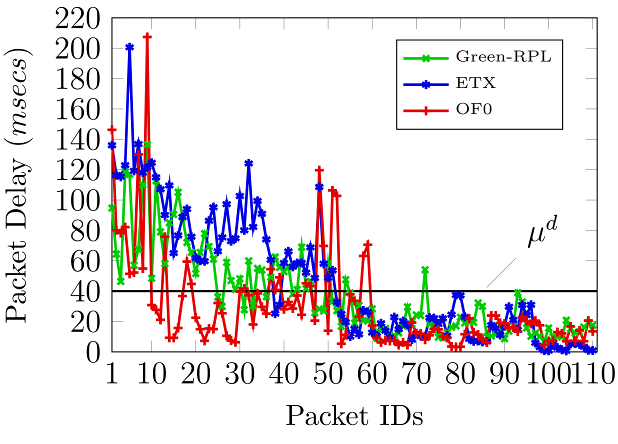


Figure 1.4: Per packet delay to get it delivered to the root node.

Finally, the per packet delay experienced by a packet is evaluated. Again a network of 20 nodes is simulated and the delay of individual packets sent by sender nodes towards the root nodes is computed. As shown in the Fig. 1.4, at the start of the network when network topology is not formed completely the delays are higher. However, as the complete RPL-DODAG is created optimal paths are selected based on the objective function in use and the delays get reduced. The simulation is run for 1 hour, yet to clear observation only limited time duration is presented. The average per packet delay in our proposed Green-RPL, ETX, and OF0, are 20.4 msecs, 47 msecs, and 26.2 msecs, respectively.

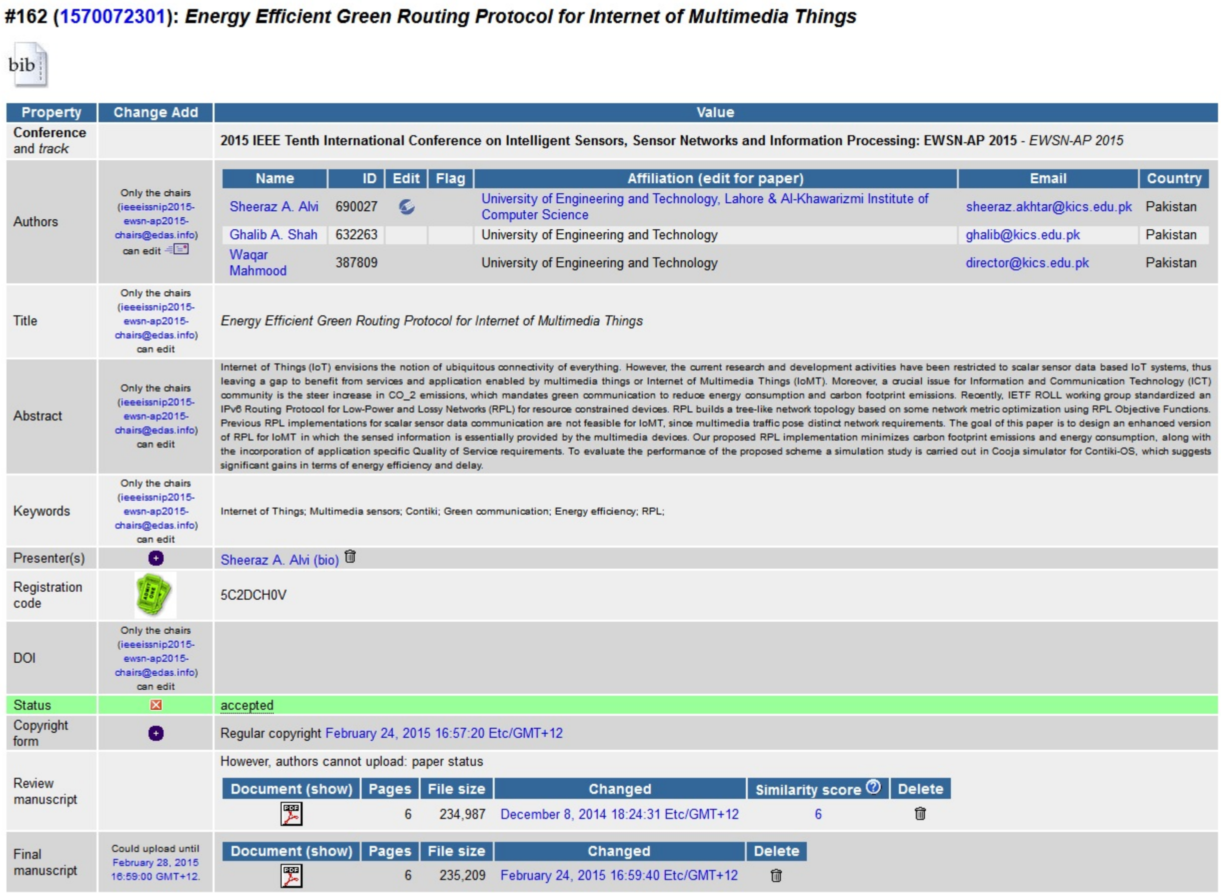
1.4 Conclusion

The current research and development activities for IoT systems have overlooked the incorporation of 'multimedia things'. Similarly, there have been no consideration of green communication or carbon footprints reduction using the RPL routing protocol. For this reason, in this deliverable an enhanced version of RPL for IoM is proposed, Green-RPL, in which the sensed information is essentially provided by the smart 'multimedia devices'. The proposed Green-RPL routing protocol minimizes carbon footprints emission and energy consumption, and supports application specific QoS requirements by considering various constraints while selecting routes towards the root node. To evaluate the performance of the proposed scheme a simulation study will be carried out in Cooja simulator for Contiki-OS, to evaluate its performance with the existing objective functions.

Chapter 2

International Conference Paper

A conference paper was written on the proposed Green-RPL routing protocol in the previous (4th) deliverable. The submitted paper is accepted for publication by the international conference titled 'IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing 2015'. Details can be seen in the Fig. 2.1:



#162 (1570072301): *Energy Efficient Green Routing Protocol for Internet of Multimedia Things*

bib

Property	Change Add	Value
Conference and *track*		**2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing: EWSN-AP 2015** - *EWSN-AP 2015*
Authors	Only the chairs (ieeeissnip2015-ewsn-ap2015-chairs@edas.info) can edit	(see authors table below)
Title	Only the chairs (ieeeissnip2015-ewsn-ap2015-chairs@edas.info) can edit	*Energy Efficient Green Routing Protocol for Internet of Multimedia Things*
Abstract	Only the chairs (ieeeissnip2015-ewsn-ap2015-chairs@edas.info) can edit	Internet of Things (IoT) envisions the notion of ubiquitous connectivity of everything. However, the current research and development activities have been restricted to scalar sensor data based IoT systems, thus leaving a gap to benefit from services and application enabled by multimedia things or Internet of Multimedia Things (IoMT). Moreover, a crucial issue for Information and Communication Technology (ICT) community is the steer increase in CO_2 emissions, which mandates green communication to reduce energy consumption and carbon footprint emissions. Recently, IETF ROLL working group standardized an IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) for resource constrained devices. RPL builds a tree-like network topology based on some network metric optimization using RPL Objective Functions. Previous RPL implementations for scalar sensor data communication are not feasible for IoMT, since multimedia traffic pose distinct network requirements. The goal of this paper is to design an enhanced version of RPL for IoMT in which the sensed information is essentially provided by the multimedia devices. Our proposed RPL implementation minimizes carbon footprint emissions and energy consumption, along with the incorporation of application specific Quality of Service requirements. To evaluate the performance of the proposed scheme a simulation study is carried out in Cooja simulator for Contiki-OS, which suggests significant gains in terms of energy efficiency and delay.
Keywords	Only the chairs (ieeeissnip2015-ewsn-ap2015-chairs@edas.info) can edit	Internet of Things; Multimedia sensors; Contiki; Green communication; Energy efficiency; RPL;
Presenter(s)		Sheeraz A. Alvi (bio)
Registration code		5C2DCH0V
DOI	Only the chairs (ieeeissnip2015-ewsn-ap2015-chairs@edas.info) can edit	
Status		accepted
Copyright form		Regular copyright February 24, 2015 16:57:20 Etc/GMT+12
Review manuscript		However, authors cannot upload: paper status
Final manuscript	Could upload until February 28, 2015 16:59:00 GMT+12.	

Authors:

Name	ID	Edit	Flag	Affiliation (edit for paper)	Email	Country
Sheeraz A. Alvi	690027			University of Engineering and Technology, Lahore & Al-Khawarizmi Institute of Computer Science	sheeraz.akhtar@kics.edu.pk	Pakistan
Ghalib A. Shah	632263			University of Engineering and Technology	ghalib@kics.edu.pk	Pakistan
Waqar Mahmood	387809			University of Engineering and Technology	director@kics.edu.pk	Pakistan

Review manuscript:

Document (show)	Pages	File size	Changed	Similarity score	Delete
	6	234,987	December 8, 2014 18:24:31 Etc/GMT+12	6	

Final manuscript:

Document (show)	Pages	File size	Changed	Delete
	6	235,209	February 24, 2015 16:59:40 Etc/GMT+12	

Figure 2.1: International Conference Paper

Chapter 3

Design and Implementation of Change-based Block Suppression (CBS) video coding technique

3.1 Introduction

Development of low complexity video coding techniques/algorithms has been accelerated by the expected increase in the up-link streaming from IoT (Internet of Things) enabled multimedia devices(IoM devices), and multimedia devices(such as mobile phones and camcorders) in general. Secondly, severe constraints on the cost, power, bandwidth and computational capability of IoM devices bar the usage of current video coding techniques(standards) such as H.265 and VP9 due to their high computational complexity on the encoder side. The high encoder complexity of H.265 and VP9 is because they are designed for down-link streaming applications. Thus, in order to match IoM device constraints, new video compression techniques, which shift the complexity from encoder to decoder, are desired.

In conventional video coding raw video data is acquired, at or above the Nyquist sampling rate, and is subsequently compressed. Though conventional video coding techniques such as H.265 and VP9 take advantage of the spatial and temporal redundancies inherent in video data(and achieve hundreds of times of compression), processing of such large amounts of raw data adds exponentially to the complexity of the encoder. Recently emergent techniques from the area of Compressive Sensing(CS), have given rise to a new generation of low complexity video encoders. Unlike traditional video coding, in compressive sensing based techniques, a small number of linear measurements of the scene are taken at sub-Nyquist rates, at the acquisition stage, before being passed on to the encoder. Thus, compression is inherent in the acquisition process.

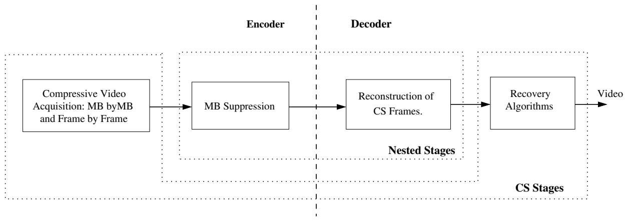


Figure 3.1: Nested video encoding-decoding.

3.2 CBS design

We aim at designing an adaptive video encoding/decoding scheme in which compressed sensed video data is further compressed within the CS domain. We have introduced a nested approach in which CS frames are further compressed(during and) after acquisition on the encoder side, and on the decoder side the full CS frames are reconstructed(using compression information) and then recovery algorithms are used for the reconstruction of video frames. The mechanism of further video compression should ideally be independent of the type of measurement matrix and the recovery technique used. Since temporal redundancy is exploited at block level, our technique is universal to all BCS based techniques. The video sequence is divided into groups of pictures(GoP), with eight frames in each GOP. The further compression has two parts: pre-acquisition block suppression based on block analysis at the start of each GoP; post-acquisition block suppression based on block analysis on a frame by frame basis. Using our block suppression technique, individual MB's don't need to be labeled. Instead, a two dimensional binary array(with one element representing one MB) is sent with each frame.

3.2.1 CS Video Acquisition Framework

Using the frame-by-frame, block-by-block CS Video acquisition framework [6, 7], we are able to obtain each frame, F_t, $t = 1, 2, ...$, which is divided, at the acquisition stage, into a two dimensional array, $I \times J$, of non-overlapping Macro-Blocks (MBs). Each MB consists of pixels, represented as a vectorized column of length N, $x_t(i, j) \in R^N$ where, $(i, j) = (1, 1), ..., (I, J), t = 1, 2, ...$ Therefore it is a temporal sequence (frames) of spatial raster scan sequences (MBs).

CS is performed by projecting $x_t(i, j)$ onto a $M \times N$ random measurement matrix Φ,

$$y_t(i, j) = x_t(i, j)\Phi \tag{3.1}$$

where Φ is a drawn from i.i.d. Gaussian random variables of zero mean and unit variance, M is the number of measurements and N is the number of pixels. A Randomized (scrambled) Hadamard Matrix[8] could also be a very good option for the random measurements. The resulting measurement vector $y_t(i, j) \in R^M$, is then passed to the Nested Encoder stage(see Figure 3.1). The GoP in our technique has two frame types, one Start frame S, where S_n is the first frame of the n^{th} GoP and seven P frames, where $P_m(m = 1, 2...7)$ is the m^{th} P frame of the current GoP. We have found the GoP size of 8 to be near optimal using empirical calculations. So the overall frame sequence looks like this:

$$S_1, P_1, P_2, ...P_7, S_2, P_1, P_2, ...P_7, ...S_n, P_1, P_2, ...P_7, ... \tag{3.2}$$

3.2.2 Change based Block Suppression(CBS)

At the heart of CBS design is the MB analysis block. Each MB, $S_1(i, j)$, of S_1 (start frame of first GoP) is passed on as it is, and also fed back as the initial updated MB, $y^{'}_{t-1}(i, j)$, for the MB analysis block(see Fig.3.3). This block performs two tasks. First, it compares the corresponding MB's, $y^{'}_{t-1}(i, j)$ and $P_t(i, j)$ (which belong to the first two frames of each GoP), to calculate the Sum of Absolute Difference(SAD) value, $s1$. The $s1$ value obtained from the comparison is then normalized and checked against eight ranges(of SAD values) to assign a pre-acquisition MB suppression pattern for the current MB of all frames in the current GoP.

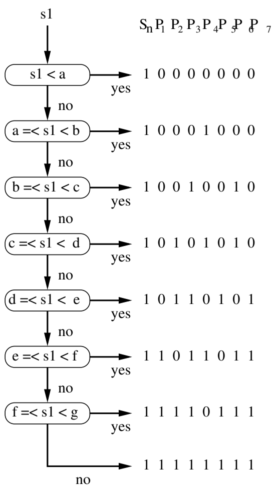


Figure 3.2: Pattern assignment.

The suppression pattern is passed to both post acquisition and pre acquisition MB suppression blocks. Since the suppression pattern for each MB location is determined using the corresponding MB's of the first two frames of each GoP, the suppression pattern bit for S is always 1 and the bit for P_1 is determined after acquisition, Therefore for any GoP, $S(i, j)$ is never suppressed using the suppression pattern and $P_1(i, j)$ is suppressed at the post acquisition stage. The SAD value ranges and the corresponding MB suppression patterns are shown in Fig.3.2.

At the pre acquisition MB suppression block, the pattern is used for suppression of $P_2(i,j)$ to $P_7(i,j)$ MB's before they are sensed, i.e the MB's for which the pattern has zero value are not sensed. MB $P_1(i,j)$ is already sensed so the post acquisition suppression block uses the second bit of the pattern for the suppression. In the second task, performed by the MB analysis block, $y^{'}_{t-1}(i,j)$ and $y_t(i,j)$ are compared, to find SAD value s2 for each MB as it arrives. For all MB's if the MB has not been suppressed(or is not to be suppressed in the case of $P_1(i,j)$), the value, s2, is compared to the adaptive threshold value, γ. If $s2 < \gamma$, the MB is suppressed, otherwise not.

Algorithm 1 Further Compression of CS Frames

Input: F_t, $F^{'}_{t-1}$, γ
Output:
  for $k = 0$ to $size(F_t)$ **do**
    if($F_t = P_t$)
      $s2 = SAD(F_t(:,k), F^{'}_{t-1}(:,k)$
    end if
    if($F_t = P_1$)
      $s1 = SAD(F_1(:,k), F^{'}_{t-1}(:,k))$
      $p_k = PatAsnmnt(s1)$
    end if
    $F^{'}_t(:,k) = PreAckSup(p_k)$
    $(I_t, F^{'}_t(:,k), \hat{F}_t(:,k)) = PostAckSup(p_k, s2, \gamma, F^{'}_t(:,k))$
  end for

The output frame $F^{'}_t$, contains all the unsuppressed, columnized MB's bundled together. The compression information is a bit-array, I_t of length equal to the total number of MB's in a frame. If an MB is suppressed the corresponding bit in, I_t is zero, otherwise it is one. I_t is updated by the Post Acquisition MB Suppression block (in Fig.3.3) at the end of the current MB processing.

3.2.3 Reconstruction and Recovery

Reconstruction is performed at the decoder and has two stages. First, the full CS frames are recovered and then recovery algorithms are used for recovery of video frames. The compressed frame received by the decoder has two parts. In the first part all the unsuppressed, columnized MB's are bundled together in the frame $\hat{F}_t$ such that each column, $\hat{F}_t(:,i)$, is a MB. The second part is a bit-array, I_t which contains the compression information, as explained above. So for reconstruction of CS frames we traverse through I_t and for every zero entry, we place a copy of the corresponding MB from the previous reconstructed CS frame, F_{t-1}, into the current reconstructed CS frame, F_t. Similarly for every non-zero(i.e. one) entry in

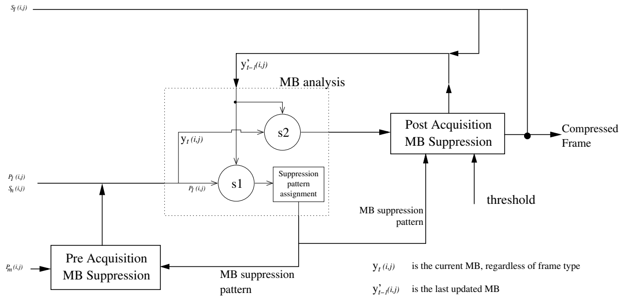


Figure 3.3: The Change based block Suppression Architecture(CBS).

I_t we place the next-up MB from $\hat{F}_t$ into the current reconstructed CS frame, F_t.

Once the current CS frame is reconstructed, the video frame can be recovered using any suitable recovery algorithm in the bases in which the signal of interest is sparse. The elements of the vectorized measurement vector are quantized individually by an 8-bit uniform scalar quantizer. At the decoder we have chosen to use total variation minimization(min-TV) [9, 10], for the recovery, due to its exceptional recovery performance. The reconstruction and recovery are decoder end processes, and we are doing this in MATLAB, for both design and implementation.

Algorithm 2 Reconstruction of CS Frames

Input: $\hat{F}_t,\ I_t$
Output:
 for $i = 0\ to\ size(I_t)$ **do**
  if($I_t(i) = 0$)
   $F_t(:, i) = F_{t-1}(:, i)$
  else
   $F_t(:, i) = \hat{F}_t(:, j)$
   $j{+}{+}$
  end if
 end for

3.3 Implementation of CBS video coding technique, on STM32F4 discovery Board.

3.3.1 Video Acquisition and Storage

We have used the OV7670 CMOS camera for acquisition of VERY low resolution (QQCIF), raw (YUV4:2:2) video at 30fps. The OV7670 camera provides an I2C (compatible) interface for configurations. A number of different configurable options are there, for example:

•VGA, QVGA, QQVGA, CIF and QCIF resolutions
•RGB raw, RGB 5:5:5, RGB 5:6:5, YUV 4:2:2,
•Up-to 30fps frame rate.
•Color correction options, e.tc.

The OV7670 provides an 8bit parallel data interface and a few other control pins compatible to the DCMI interface on the discovery board. The DCMI interface is a standard camera interface. When DCMI is used, the pixel data comes from the 8bit parallel data interface into the 32bit DCMI data register. Each pixel is of two bytes therefore two pixels are written to the DCMI data register at a time. And before the next two pixels arrive, the DCMI data register has to be read. Otherwise data will be lost. The data rate requirements for initial storage of the raw video frames are:

$$\begin{aligned} datarate &= framespersec \times pixelsperframe \times bytesperpixel \\ &= 30 \times (160 \times 120) \times 2 \\ &= 2.3MB/sec \end{aligned}$$

For instance, with the QQVGA resolution, one frame takes 37.5Kbytes of memory, so if the whole frame is stored in the SRAM for processing, and considering that the next frame will also be coming in (as well as the compressed frame), therefore the 192Kbyte SRAM will be stretched. The Kiel uVision5 compiler does not allow for creation of a buffer greater than 60Kbytes in the SRAM. So in order to store a frame in a single buffer we have gone for the lowest resolution, QQVGA. On the other hand, the write speed for the 1MB on-board Flash of the discovery board is 128KB/sec. At this write speed, the video frame cannot be stored in real time for processing. Therefore we have used an external SD card to store video data. The read write speed of SD cards depends on which class they belong. Class2 and class4 SD cards give data read/write speeds of 2MB/sec and 4MB/sec respectively. Class4 to class10 cards give data read/write speeds from 4MB/sec to 15MB/sec. Since our required data rate is more than 4MB/sec and we also have to do other read, write operations, so we have selected the class10 SD card, which is FAT32 filesystem compatible.

Data transfer from DCMI to SD card

How the data is transferred from the 32bit DCMI data register to the SD card has a few very important twists to it. Data has to be picked, continuously, from the DCMI data register at a very high frequency so the processor will remain unnecessarily busy. However the DMA (Direct Memory Access) controller on the STM32F4 facilitates the transfer of data from one location to another in embedded system without intervention from the central processor (CPU). Therefore we are using the DMA to transfer data from the DCMI data register. On the other hand there are two standard methods for writing data to the SD card, the uSD and the Fat FS. We have preferred the later because files saved under the Fat FS filesystem are directly interpreted on any windows OS. Since we have to perform sparce recovery of our video stream on some remote computer, the video data will be more easily understood if saved under the fat FS filesystem. However the Fat FS filesystem uses the physical layer interface (SDIO/SPI) with the ST card to transfer the data to the SD card so the DMA controller cannot be given the address of the SDIO port. Therefore we need a buffer in the middle where data is dumped by the DMA and then Fat FS write command transfers the buffered data to the SD card. This buffer is maintained in the SRAM due to it's fast read/write speed. The size of this buffer and a few more details are discussed in the next section.

Buffer size, and Block Columnization

The pixel data of each frame flows into the DCMI, pixel by pixel, in a raster scan fashion. For simplification we will only process the luma (Y) component, so we have omitted the chroma components by using binary AND and binary SHIFT operations (before the data is passed on by the DMA). The Fat FS also uses a DMA channel to read and write, to and from the SD card. We intend to use the end of frame interrupt to perform Block Coloumnization and write to the SD card before we get the next frame. However due to some memory and bus usage conflicts, for now we are taking the first 32×32 MB from each frame. For this we use the 32nd row interrupt to indicate that the first row of MB's is complete, we stop the rest of the frame from coming in, and place the first MB into a column-matrix (in the SRAM) using STM32F4 DSP library. The same process is repeated for 16 adjacent frames, and then DCMI is disabled, so that CS and further compression can be performed. The intuitive buffer size is the size of 32 rows of the frame i.e. 40KBytes. The total space taken by these 16 MB column-matrices is 64Kbytes, which leaves around 60Kbytes in the SRAM for the further processing of the frame. In the next section we discuss the video compression.

Compressed Sensing and Further Compression

At this stage we have 16 columnized MB's, where each is the first MB of 16 adjacent frames. Now we apply CS on each MB one by one and write the CS MB's back to the SD card using f_write function of Fat FS. The measurement matrix (i.i.d Gaussian Random matrix) is stored in the SD Card. For each measurement we read one row of the measurement matrix from the SD Card (using the f_read function), perform the floating point matrix multiplication using DSP library, and store the measurement in a CS-MB column-matrix. Implementation using the DSP library for matrix multiplication greatly reduces the computational complexity of the compression algorithm. Also the use of FPU for these calculations reduces the CPU usage by about 6-7 times. In this way each row of the measurement matrix is read one by one and the same procedure is repeated until the desired number of measurements are taken. The CS-MB column-matrix is then saved to the SD Card. The same procedure is repeated for all the MB's.

Now that we have 16 columnized CS-MB's, we retrieve CS-MB's (from the SD card) of neighboring frames to calculate SAD values. The SAD calculations are also performed using the DSP library in the FPU format, reducing the computational complexity of the further compression implementation. The SAD values are subsequently used for block suppression as discussed in section 3.2.2. The finally compressed frames are stored back into the SD card. Recovery of these frames (1 MB for each frame) is performed in the MATLAB environment on any WINDOWS system.

3.4 Conclusion

In the design and implementation of CBS video coding technique we have achieved significantly low bit-rates when compared to regular CS techniques. Also we have managed to keep complexity significantly low, both in the design and implementation of the CBS technique. So as a result of this work we have a base upon which we can build a more robust CS based camera node.

Bibliography

[1] P Venkataram and AP Shivaprasad. Wireless multimedia networks. *Journal of the Indian Institute of Science*, 80(3):187, 2013.

[2] T Winter, P Thubert, T Clausen, J Hui, R Kelsey, P Levis, K Pister, R Struik, and J Vasseur. Rpl: Ipv6 routing protocol for low power and lossy networks, rfc 6550. *IETF ROLL WG, Tech. Rep*, 2012.

[3] Aruna Prem Bianzino, Claude Chaudet, Dario Rossi, and J Rougier. A survey of green networking research. *Communications Surveys & Tutorials, IEEE*, 14(1):3–20, 2012.

[4] Xiaofei Wang, Athanasios V Vasilakos, Min Chen, Yunhao Liu, and Ted T Kwon. A survey of green mobile networks: Opportunities and challenges. *Mobile Networks and Applications*, 17(1):4–20, 2012.

[5] Hazrat Ali. A performance evaluation of rpl in contiki. 2012.

[6] Ying Liu, Ming Li, and Dimitris A. Pados. "Motion-Aware Decoding of Compressed-Sensed Video". *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY*, 23(3), MARCH 2013.

[7] Yusuke Oike and Abbas E.G. "CMOS Image Sensor With Per-Column ADC and Programmable Compressed Sensing". *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, VOL. 48, JANUARY 2013.

[8] Lu Gan, Thong T. Do, and Trac D. Tran. "Fast Ccomressive Imaging using Scrambled Block Hadamard Ensemble". In *Proc. EUSIPCO*, 2008.

[9] Z. Liu, H. Vicky Zhao, and A. Y. Elezzabi. "BLOCK-BASED ADAPTIVE COMPRESSED SENSING FOR VIDEO". In *Proceedings of 2010 IEEE 17th International Conference on Image Processing*, September 2010.

[10] Z. Liu, A. Y. Elezzabi, and V. Zhao. "Maximum Frame Rate Video Acquisition Using Adaptive Compressed Sensing". *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY*, 21(11), November 2011.