Performance Evaluation of Raptor Codes in TCP/IP-based Wireless Networks

Fanoro Mokesioluwa, Mjumo Mzyece and Guillaume Noel
French South African Institute of Technology (F'SATI)
Department of Electrical Engineering
Tshwane University of Technology,
Private Bag. Box X680, Pretoria 0001, South Africa
Email: adeniyicall@gmail.com, MzyeceM@ tut.ac.za,noel_gpa@yahoo.com

Abstract- **This paper evaluates the performance of Raptor codes on Transport Control Protocol/Internet Protocol (TCP/IP) packet-based wireless networks. Performance metrics such as packet loss rate and recovery rate are studied closely. This is carried out by building a functional module in a network simulator 2 (ns2) environment that models the Raptor transmitter and Raptor receiver. At completion of this, the standalone platform for the development of the actual Raptor module as an ns2 compatible protocol is implemented. The results show that packet recovery rate increases to over 60% when Raptor code was integrated into the transport layer.**

Index Terms— ALFEC, UDP, Raptor code, ns2, packet recovery.

I. INTRODUCTION

Wireless networks are becoming increasingly important in global communications. Transport Control Protocol/Internet Protocol (TCP/IP) protocols have been useful for providing support for packet-based traffic on wireless networks. Notable among such TCP/IP protocols are the User Datagram Protocol (UDP) and the Transport Control Protocol (TCP). UDP relies on a connectionless communication mechanism. However, TCP is a connection-based protocol. While UDP offers lower overheads, it does not perform as well as far as error correction and reliably delivery is concerned. UDP does not add any extra functionality to the Internet Protocol in the network Layer.

Although the reliability of UDP is negatively affected by the lack of connection management capabilities and rate control mechanisms [1], higher transmission rates and significant amount of resources saved in a networking environment by UDP are among its most attractive attributes. Furthermore, many real-time delay-sensitive applications such as Voice over Internet Protocol (VoIP), Video on Demand (VOD), Internet Protocol Television (IPTV) adopt UDP as a transmission protocol. Therefore, improving the performance of UDP is important for delay sensitive applications as high throughput and moderate latency enhances the deployment of multimedia applications.

The concept of rateless code [2, 3] , aids in ensuring reliability for media streaming services which utilise an unreliable transport layer protocol like UDP. Rateless codes have been proposed as a powerful Forward Error Correction (FEC) mechanism. Raptor codes are fountain codes generating as many encoding symbols as desired on-the-fly from the source symbols of the source block. Raptor codes, an advanced Application layer Forward Error Correction (ALFEC) code, have been standardised for file streaming and download delivery. They have since been deployed in the 3rd Generation Partnership Project (3GPP) Multimedia Broadcast Multicast Services (MBMS) standard for broadcast file delivery and streaming services, the Digital Video Broadcasting – Handheld (DVB-H) Internet Protocol Datacast (IPDC) standard for delivering IP services over DVB networks, and Internet Protocol Television (DVB-IPTV) standard for delivering commercial TV services over an IP network, DVB-H, Digital Multimedia Broadcasting (DMB), Integrated Services Digital Broadcasting - Terrestrial (ISDB-T) and Media Forward Link Only (MEDIA FLO) [4-6].

In this paper, we evaluate performance of UDP, Luby transform (LT) and Raptor codes over wireless networks in terms of packet loss, recovery rate and throughput. The efficiency of Raptor codes is clearly seen as packet loss in UDP is minimized and the recovery rate of UDP increased considerably. This work focuses on streaming services. Specifically,

1. We clearly specify and introduce the wireless network error transmission model in our TCP/IP based wireless network. This error model is needed to test the efficiency of Raptor codes.
2. We state the design goals and built our standalone platform which represents the Raptor encoder and decoder. The Raptor encoder ensures that the source symbols are transformed into a potentially unlimited number of encoded symbols while ensuring that the Raptor decoder recovers the maximum number of packets at the receiver end. Our simulations show that compared to the use of the LT code, the original class of practical fountain codes, our design provides a better recovery rate.

This paper is organized as follows. An overview of the system design is presented in Section 2. In Section 3 based on the adopted system design, we explain the methodology employed and introduce clearly the performance metrics studied. In Section 4, we present simulation results that show the actual performance of the system design. Finally, we present our conclusions and future work in Section 6.

II. RELATED WORK

Several efforts have been made by various researchers to improve the performance of UDP. For instance, in [4], a hybrid transmission that uses TCP and UDP to send packets is used as well as an FEC transmission scheme. However, the delay introduced at the point of establishing connection with the receiver is detrimental to multimedia applications that involve streaming of data. The approach used by [7] in tackling bursty packet losses was quite different in that it combines the Real-time Transport Protocol (RTP) with the Partially Reliable extension of Stream Control Transmission Protocol (PR-SCTP). However, the success of the scheme was at the expense of delay introduced into the system, which is bad for streaming applications.

The authors in [8] exploit the low overhead of UDP by proposing a scheme which makes use of retransmission in UDP. At the end of the experiment, it was discovered that UDP has a lower delay rate and a lower packet loss rate compared to TCP. Similarly, the authors of [9] proposed Reliable Dynamic Buffer UDP (RBUDP) in the wireless network aimed at improving the performance of UDP. RBUDP transmits control information using the TCP protocol while data information is transmitted using the UDP protocol. The sender and the receiver make use of the buffer size field in ensuring the correct arrangement of the packets. The performance is evaluated in terms of network throughput and delay. However, the constant sending speed on which the model is based on, makes the buffer of the receiver overflow, thus causing terminal congestion.

The authors in [10] developed a Conditional Retransmission Enabled Transport Protocol (CRETP) to improve the transmission reliability and real-time delay in Networked Control System (NCS). It employs a conditional retransmission mechanism and acknowledgements while keeping the timeliness of data transfer in mind. However, the retransmission could introduce additional delay which could impact negatively on time-critical applications.

While most of the studies focused their attention on exploiting and implementing the key strengths of TCP, which are acknowledgements and retransmissions, we would be applying an application layer FEC to improve the performance of UDP.

III. SYSTEM DESIGN

In this design, we choose UDP as our transmission protocol for transmission at the transport layer. UDP as a transport layer protocol is useful in avoiding retransmission and delay. However, it does not provide reliability in datagram delivery. If more reliability is needed with UDP, measures should be put in place to incorporate such reliability techniques above the transport layer in the case of UDP. In order to achieve a reasonable amount of reliability in UDP, techniques such as Automatic Repeat reQuest (ARQ) and the Forward Error Correction (FEC) [11] are considered.

However, since UDP is aimed at transmitting data to a large number of receivers, the retransmission scheme which Automatic Repeat reQuest (ARQ) was built on was considered as being less efficient. This spurred the invention of Fountain code in the late 1990's [12]. Raptor codes, a type of fountain codes [13] will be applied above the UDP layer.

In order to achieve the set goal of introducing reliability to UDP, Raptor codes will be implemented as an Application Layer Forward Error Correcting Code (ALFEC). The application layer will send packets through the Raptor encoder and appropriate processing, such as the introduction of redundancy, will be carried out on the packets before it is sent to the UDP transport layer. Once this is accomplished, it will be sent to the lower layers for onward transmission to the receiver. This is depicted in Figure 1.

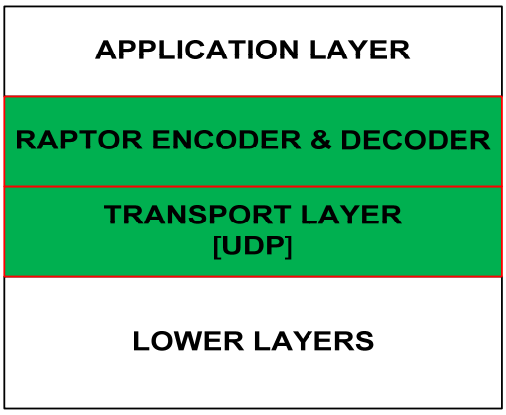

Figure 1: System design for Raptor code

The Raptor code as an Application Layer Forward error correction code will be discussed below.

A. *Raptor code*

Assuming the message k bits are of the Galois field $GF(2)$ order, let $x \in \mathbb{F}_2^{\,k}$ be the vector of message symbols and let $z^{[1:k]} = G_{LT}^{[1:k]}.G_C x$ be the vector of the first k encoded symbols, where G_c is a $k \times k$ generator matrix of the Raptor code termed precode C, while $G_{LT}^{[1:k]}$ is a generator matrix, formed by the first k rows of the Luby Transform (LT) generator matrix. To ensure the systematic nature of the Raptor code, we run a scheduling algorithm to see the possibility of decoding the input symbols.

Thus, $G_R = G_{LT}^{[1:k]}.G_c$ is a quadratic Raptor encoding matrix. If this matrix follows the inverse-encoding principle, we can set $\hat{x} = G_R^{-1}.x$ to be the new input processed by the Raptor encoder. Thus, the original transmitted message x retains its systematic property alongside the repair or redundant symbols' introduced to produce an intermediate symbol. Once this is achieved, the intermediate symbols are passed through the LT encoder, resulting in a set of encoded symbols *z*.

The degree distribution used in generating the LT encoder at the final stage will be required at the first stage of the decoding process specifically at the code constraint processor. This is depicted in Figure 2.

The code constraint processor is made up of multiple codes. Prominent among them are the Low Density Generator

Matrix (LDGM) code and the gray code. Others are Luby Transform code and the identity matrix as well as a matrix consisting zeros. A component of this LDGM includes a full circulant matrix and a partial circulant matrix. In the construction of the LT code; the degree distribution plays a critical part in the design of the generator matrix.

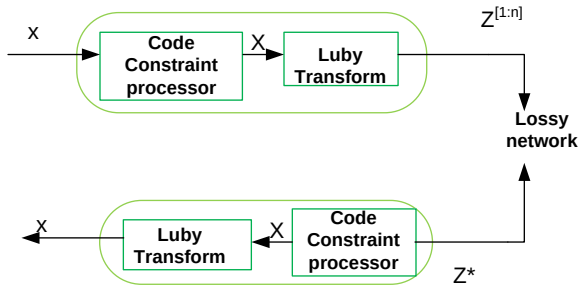

Figure 2: Block diagram of Raptor encoder and decoder

The constituent code of Code constraint matrix is represented by Figure 3 where G_{LDPC}, I_S,Z,G_{HALF},I_H,G_{LT} are the generator matrix for low density generator matrix code, identity matrix of size *s*, zero matrix of size *s*h*, Gray code, identity matrix of size *h* and the generator matrix of LT code.

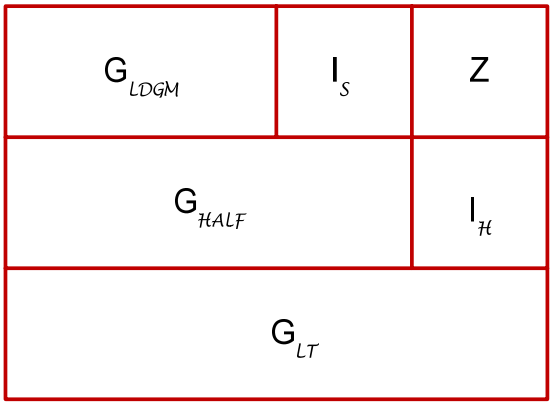

Figure 3: Block diagram of Code Constraint processor

The degree distribution is based on the Robust Soliton Distribution (RSD) invented by [14]. RSD is based on ideal soliton distribution (ISD). It was introduced because of the stochastic fluctuations experienced when the degree distribution is generated using the ISD. RSD ensures that most of the packets have a low degree (degree one) so that the decoding process can be started and continue. Similarly, it limits the number of addition operations involved in the encoding and decoding process. Thus, the first k degree distribution will be used in modeling the G_{LT} needed in the code constraint matrix. RSD can be computed according to Equation (1).

$$\tau(d)=\begin{cases}\frac{S}{K}\frac{1}{d} & for\ d=1,2,....,\frac{K}{S}-1\\ \frac{S}{K}\log\left(\frac{S}{\delta}\right) & for\ d=\frac{K}{S}\\ 0 & for\ d>\frac{K}{S}\end{cases} \quad (1)$$

Where δ denotes the probability of decoding failure imposed by the lack of degree one packet, K is the number of source packet, d is the degree , S is the expected number of degree one.

The code constraint matrix at the decoder side is used for the recovery of the intermediate bits. Once the LT decoder is applied to the intermediate bits, the source symbol k is recovered. Of these k source symbols, the first $\hat{k}$ correct symbols are chosen as the sent message. The relationship between the source symbol, x and the intermediate symbol $\hat{x}$ is represented below:

$$H\hat{x}=0,$$
$$G_{LT}^{[1:k]}\hat{x}=x \quad (2)$$

IV. METHODOLOGY

In this work, the abstraction of the Raptor code module as an ns2 protocol was implemented. We took an advantage of the scenario generation and extensibility of the ns2 platform in introducing new C++ classes.

This affords us an opportunity to use Raptor codes as an Application Layer Forward Error Correction (ALFEC) reliability mechanism in this simulation. Figure 4 depicts the model used in the simulation process. The sending node represented by the colored circle transmits using the unicast mode to the receiving node of the RPAGENT represented by the rectangular node.

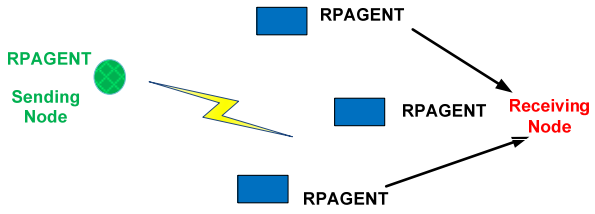

Figure 4: Block diagram of simulation Model

A. *Standalone simulator*

While building the standalone simulator, two new packet structures were defined in which information necessary for the communication of transmitting and receiving parties were put in place. ns2 initializes the header structure during the construction of the simulation object.

- The header structures are hdr_RPData and hdr_RPReport. Attributes described in the new structure hdr_RPData include block size, block identity (block id.), degree of symbol as well as the packet length. The header structure incorporates

their access methods and operations needed for each attribute. All these attributes are necessary in defining the packet flow.

- In order to send the encoder packet, a timer is responsible for the invocation of the SendRpDataAgent, which is important for the scheduling and execution of events during the simulation. When developing the functional module, RaptorReports was used as a reporting agent to the sender. Nodes in ns2 that transmit encoded packets use the RPAgent. This is defined with its constructor, attributes and access method.
- At the creation of RaptorDataAgent, a linked object was created which aids the realization of an instance in Otcl. Once there is the execution of the simulator environment, SendRpReportAgent and SendRpDataAgent are linked to the Otcl object and invoke the two main objects, reporter_s_ and Sender_. These allow the communication of the OTcl and C++ interface. SendRpPacketDataTimer is of great importance to the SendRpDataClass as it prompts it on the timing of the packets to be sent.
- Every new instance of SendRpReportAgent links with TCL space and communicates with SendRpDataClass, which plays the important function of reporting the status of the information sent from the receiver end. SendRpDataClass contains a direct reference to RPDataBlockSend. This is necessary for keeping track of the size of data sent with regard to the input block. Each instantiation of RPDataBlockSend also contains its own reference to RPEncoder object, which is necessary class for the generation of Raptor encoder packets.

B. Functional Module.

The functional module is made up of two parts: the Raptor transmitter and Raptor receiver. The Raptor transmitter shows the process of data transmission to the receiving side. Two protocols namely Raptordata and Raptorreport were built. In the transmitting side, Raptor encoder initiates the packet creation process.

Once the encoding process is completed, `RaptorSymbol` serves as storage for encoded packet. A reference is placed in the `RaptorSymbol`, useful in the recovery of encoded packet to start the decoding process. The Raptor receiver is initialized when the reference from the `RaptorSymbol` is called. The messages are extracted from the packet and the decoding process begins. Once the decoding process is completed, a control packet is sent confirming the successful decoding of the message and communicating to the transmitter, the beginning of the next phase of the input data encoding and transmission.

The following classes were built into the implementation of the Raptor encoder and decoder:

IncEncoding: In the incremental encoding stage, the packet to be passed onto the receiver is encoded by the code constraint matrix. This matrix is made up of a combination of six separate compartments. This is discussed in section III. It must be noted that the same matrix is required at the receiver end, necessary for the starting of the decoding.

LTEncoder: After the intermediate encoding, the LTencoding takes place. In this stage, the intermediate messages are picked randomly using the degree distribution and are XORed together. It is stored up with its number of bits encoded; degree distribution, Symbol length and length of the message .These are all stored up in `Raptor Symbols`.

LTSymbol: The functions `Amatrix`, as well as the encoded message are stored up in the `RaptorSymbol`. It serves as the proxy server – link between the encoding side and the decoding side of Raptor code. The `AMatrix` is a full representation of the code constraint processor. A reference for the `AMatrix` is placed in the `RaptorSymbol` object directly. This reference allows the receiving party to immediately use the same object that is generated by the encoder.

IncDecoding: The code constraint matrix that is made using function `AMatrix`, `DegreeDistro` that represents the degree distribution as well as the encoded message is stored up in the function `RaptorSymbol`. This is called once the IncDecoding process is to commence. A subset of the `AMatrix` is used during the decoding operation such that the set of constraint introduced in the generation of the `AMatrix` help in the recovery of the intermediate message.

LTDecoder: The LTDecoder is similar to the operation carried out by the LTEncoder. The LTDecoder is used in the recovery of the original packet sent through the network.

C. Simulation parameters

We conducted our performance analysis using the ns2 simulator, version 2.34 which provides an IEEE 802.11 wireless channel. In all our simulations, 10 mobile nodes are randomly placed in the 500m * 500m area.

The Raptor encoder carries out some pre-processing on the packet to be sent before it was sent to the UDP. The payload was segmented into a group of 1456 bytes. This was computed based on the fact that the UDP packet header is 8 bytes , the Internet Protocol version four (IPv4) is assumed to be 20 bytes . Lastly, we made an assumption that some percent of the information needed at the receiver node has been stored up in the `RaptorSymbol`, thus reducing the size of the Raptor packet header to 16 bytes [15] ,thus we arrived at 1456 bytes .This implies that the packets are evenly divided.

Each session of the simulation ran for 1000 seconds. Similarly, we introduced randomness into our simulation by specifying a seed value of 10,000. This aids the computation of the average values needed for analysis of results. The symbol size is represented by varying values of k while the value of free parameter, c and the probability of decoding failure imposed by the lack of degree one packet was denoted by d.

These values were selected based on [16]. These two parameters were optimized enough for the degree distribution of the Robust Soliton distribution, enhancing the overall performance of the code. A sender transmits a message and the receiver receives the same message measuring: Number of Lost packets and Number of packets received

This is carried out in conjunction with the three scenarios studied: UDP, UDP+LT, UDP+Raptor. In aiding the successful implementation, an error model was introduced which help in simulating packet loss of different rates. The error model imposes error on packet transmission, resulting in packet drops. In order to evaluate the performance of Raptor code, the trace file generated were analyzed using Matlab scripts. The two performance metrics studied are recovery rate and throughput.

The recovery rate is the rate at which the ALFEC code helps in the recovery of the data lost. The parameter that verifies the packet recovery rate of Raptor code is the employed coding degree distribution. In this case, the Robust Soliton Distribution is employed. The recovery rate can be computed as:

$$Packet\ recovery\ rate = \frac{Packets\ recovered}{Total\ packets\ sent} \times 100$$

Where packets recovered refer to the number of packets successfully decoded while the total packet sent is the number of encoded packets sent to the receiver.

The throughput is the amount of data exchanged per second between a source and destination in the network.

The throughput can be computed by:

$$Throughput = \frac{Number\ of\ packet\ delivered \times size\ of\ packet\ (bytes) \times 8}{Total\ Duration\ of\ Simulation}$$

V. RESULT AND DISCUSSION

The simulation concept and details presented in section III and IV allow us to simulate the performance of Raptor codes in the wireless network.

Figure 5 shows the recovery pattern of the ALFEC. This was computed by comparing the amount of message received using the UDP, UDP+LT, UDP+RAPTOR.

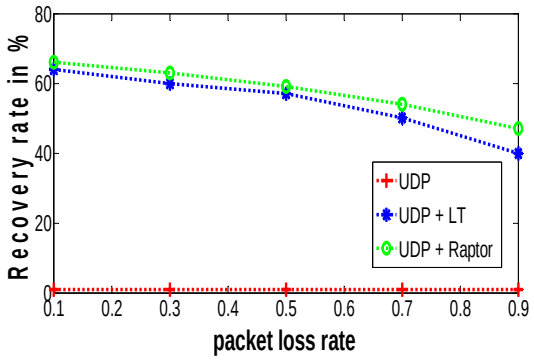

Figure 5: Plot of recovery rate vs. packet loss rate

From Figure 5, we notice clearly that there is a significant decrease in the recovery rate of UDP+LT compared to UDP+Raptor from a packet loss rate of 0.5 to 0.9. UDP+Raptor has an average of 60 % compared to the 54 % which is applicable to the UDP+LT code. This result is corroborated by [17] which ascertains the fact that raptor code has a higher decoding probability compared to LT code. A higher amount of redundant packets will be required by LT code when compared to Raptor code in the recovery of the packet. While the above statement is true, the number of packets recovered is a function of the decoding algorithm. In this case, incremental Gaussian Elimination (IGE) [18]decoding was employed in the first decoding stage .

The plot of the throughput comparing UDP, UDP+LT and UDP+Raptor is shown in Figure 6. The performance of UDP in terms of its throughput falls from 20Mbps to 8.2Mbps when evaluated. This is due to the absence of any error correction mechanism in UDP. This poor performance became obvious as the packet loss rate increases. However, the same cannot be said about UDP+LT and UDP+Raptor codes, whose throughput dropped as low as 13.8Mbps and 17Mbytes respectively. With the result, it can be discovered that the effect of Raptor code on the improvement of the performance of UDP is significant. This improvement is due the code constraint matrix, which ensures the generation of a systematic encoded symbol. In this process; all the source symbols are among the encoding symbols that can be generated. Through this method, the unique properties of the source symbol are kept intact. This method of generating a redundant bit is through the Xor process. This is the key to the enhanced performance of UDP when Raptor code is applied.

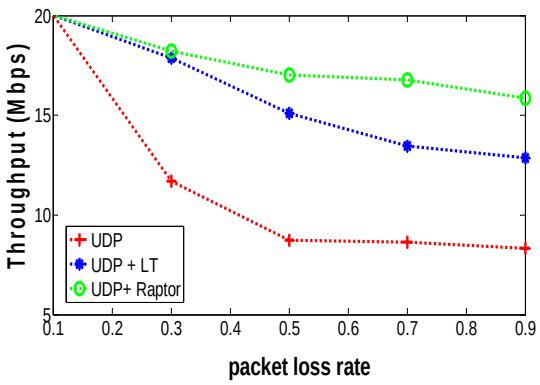

Figure 6: Plot of throughput vs. packet loss rate

VI. CONCLUSION AND FUTURE WORK

An assessment of the performance of Raptor codes over packet-based wireless networks was carried out. We considered the situations where Luby Transform codes and Raptor codes were used in augmenting UDP. Luby transform code and Raptor code were applied above the transport layer. Both codes were applied as an ALFEC Code. Numerical results showed that the Raptor codes perform well recovering over 60 percent of the lost packets when compared to the UDP. This is due to the effect that the precoding stage has on the performance of Raptor code. This study is subject to further work, as other metrics will be considered. Future work will investigate the tradeoffs between the redundancy ratio and the overhead employed in ensuring such a recovery rate when Raptor codes are used.

REFERENCE

[1] A. Shokrollahi and M. Luby, "Raptor codes," *Foundations and Trends in Communications and Information Theory*, vol. 6, pp. 213-322, 2009.

[2] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551-2567, 2006.

[3] A. Shokrollahi, "Theory and application of Raptor code," *Springerlink*, vol. 3, pp. 59-89, 2009.

[4] Z. Chen, J. F. Barnes, and B. Bodenheimer, "Hybrid and forward error correction transmission techniques for unreliable transport of 3D geometry," *Multimedia Systems*, vol. 10, pp. 230-244, 2005.

[5] M. Luby, T. Stockhammer, and M. Watson, "IPTV systems, standards and architectures: Part II-Application layer FEC in IPTV services," *Communications Magazine, IEEE*, vol. 46, pp. 94-101, 2008.

[6] T. Stockhammer;, A. Shokrollahi, M. Watson, M. Luby, and T. Gasiba, "Application Layer Forward Error Correction for Mobile Multimedia Broadcasting," in *Handbook of Mobile Broadcasting:DVB-H,DMB,ISDB-T and Media FLO*, ed Boca Raton,Florida: CRC Press, 2008, pp. 239-280.

[7] T. Maeda, M. Kozuka, and Y. Okabe, "Reliable Streaming Transmission Using PR-SCTP," in *Ninth Annual International Symposium on Applications and the Internet (SAINT '09)*. Seattle,U.S.A, 2009, pp. 278-279.

[8] C. Dong Kwon, C. Sang Hun, A. Jeong Gyun, K. Yong Sik, E. Jong Hoon, and K. Young Il, "A UDP-based protocol for mobile robot control over wireless Internet," in *Second International Conference on Robot Communication and Coordination. (ROBOCOMM '09).* , 2009, pp. 1-4.

[9] L. Wang and Z. Wan, "Performance Analysis of Reliable Dynamic Buffer UDP over Wireless Networks," in *Second International Conference on Computer Modeling and Simulation (ICCMS '10)*, Sanya,China, 2010, pp. 114-117.

[10] Y. Zhaojuan, R. Yongmao, and L. Jun, "Performance evaluation of UDP-based high-speed transport protocols," in *IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS'11)*, Beijing China, 2011, pp. 69-73.

[11] J. C. Moreira and P. G. Farrell, *Essentials of error-control coding*: John Wiley & Sons, 2006.

[12] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *Information Theory, IEEE Transactions on*, vol. 47, pp. 569-584, 2001.

[13] William E. Ryan and L. Shu, *Channel Code:Classical and Modern*, 1st ed.: Cambridge University Press, 2009.

[14] M. Luby, "LT Codes," in *Proceeding of the 43rd Annual Symposium on Foundations of Computer Science(FOCS 02)* Vancouver, BC, Canada, 2002.

[15] C. J. Botha, H. C. Ferreira, and W. A. Clarke, "The effect of Distributed packet loss on LT codes," presented at the Southern Africa Telecommunication Networks and application Wild Coast Sun,Durban, 2008.

[16] P. Cataldi, M. P. Shatarski, M. Grangetto, and E. Magli, "Implementation and Performance Evaluation of LT and Raptor Codes for Multimedia Applications," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP '06)* California, USA, 2006, pp. 263-266.

[17] I. Attarzadeh, S. H. Ow, and A. Barati, "Proposing a Novel Algorithm to Improve Security in Wireless Sensor Networks Using Data Coding " in *Advances in Computer, Communication, Control and Automation*. vol. 121, Y. Wu, Ed., ed Berlin Heidelberg: Springer 2012, pp. 573-581.

[18] T. Mladenov, S. Nooshabadi, and K. Kiseon, "Efficient Incremental Raptor Decoding Over BEC for 3GPP MBMS and DVB IP-Datacast Services," *IEEE Transactions on Broadcasting*, vol. 57, pp. 313-318, 2011.

Fanoro Mokesioluwa received his B. Eng degree in Electrical and Electronic Engineering Department of the Federal University of Technology, occur in 2007. He is presently studying towards his dual Masters (M.sc and M. Tech) with F'SATI and Tshwane University of Technology. His research interests include TCP/IP protocol, Network coding and wireless networks.