Android Multi-Hop Video Streaming using Wireless Network

Shylaja.B.R shylaja.b.r@gmail.com

Abstract

Modern world has deep penetration of smartphones Which provides an greater range of multimedia content processing like video capturing , audio processing etc. The processing of multimedia content is becoming an dominant form of information that people produce and consume on a daily basis.

This Application supports an live sharing of video content to an multiple users who is multi-hops away . It is based on peer-peer communication between mobile phones i.e; The Live Information captured by the mobile phone sensors like camera, microphone with person who is multiple hops away , could share his video content to multiple users.

Keywords : Android , encoding and decoding , H.264,Mobile adhoc-networks, wireless networks.

I. INTRODUCTION

Currently a majority of smart phones are equipped with both hardware that supports real-time video processing and ad-hoc wireless communication between peers and this allows real-time video streaming over multiple wireless hops between peer devices. Phones within communication range of each other automatically establish a wireless link creating a client mesh network (ad-hoc network of devices). Each phone in the client mesh network is able to produce/consume video and also acts as a relay to forward video to its next hop neighbours.

Peer-to-peer video streaming from the cameras on smart phones to people nearby allows users to share what they see. Such streaming can be used in a variety of applications, in particular in various social network applications including sharing unforgettable moments with friends that can be multiple wireless hops away, cooperative fieldwork (providing video sharing for
teams distributed in a small area.

In this paper we present a wireless multi-hop video streaming application for the Android based mobile devices (including Android based tablet PCs). This application allows users to capture live video using camera on mobile devices, and to share this video feed with people nearby using a free-of-charge wireless mesh network. Routing protocols can be installed to facilitate the multi-hop communication to go beyond a single hop. Fig. 1 shows an example application scenario in which a person streams live video of a concert to friends nearby over the wireless mesh network of phones, without being caught by the expensive mobile phone bill. We evaluate the presented peer-to-peer video streaming application in a variety of experiments. In these experiments we show feasibility of peer-to-peer video streaming for various generations of Android phones and also evaluate performance of various video encoding and decoding schemes.

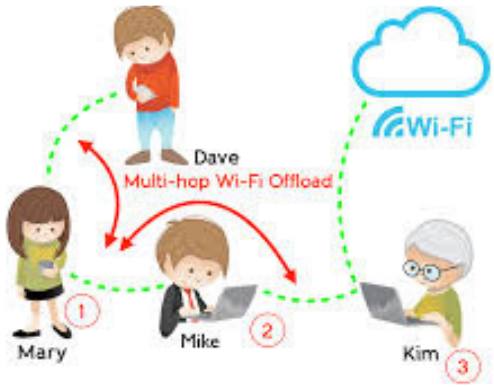

Fig 1. Application Scenario.

We proposed a wireless multi-hop video streaming application for the Android based mobile devices (including Android based tablet PCs). This application allows users to capture live video using camera on mobile devices, and to share this video feed with people nearby using a free-of-charge wireless mesh network. RTP can be installed to facilitate the multi-hop communication to go beyond a single hop. A person can stream live video of a concert to friends nearby over the wireless mesh network of phones, without being caught by the expensive mobile phone bill. This can entertain own or group of people in remote location where internet connections are not available. It can be used in defense projects where camera views can be shared to backend teams to be prepared when front team is on work.
The figure 2.0 shows a peer-to-peer (P2P) technology mobile user, the ability to share real time video that they are watching or transmitting to other authorized users.

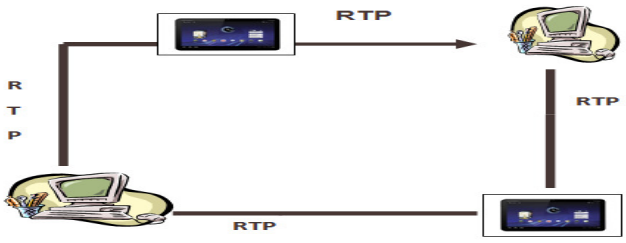

Fig 2. Peer –to-Peer Communication using RTP

II. THE BUILDING BLOCKS

1.1 Overview of Android System

The figure 3.0 shows the Android system architecture, which consists of the Linux kernel with device drivers and the Android runtime environment (along with a number of libraries) that support interactions between the Linux kernel and the high-level application framework. The application framework released in a bundle as the Android SDK [5] provides high-level Java interfaces for accessing the underlying resources, such as camera and WiFi .

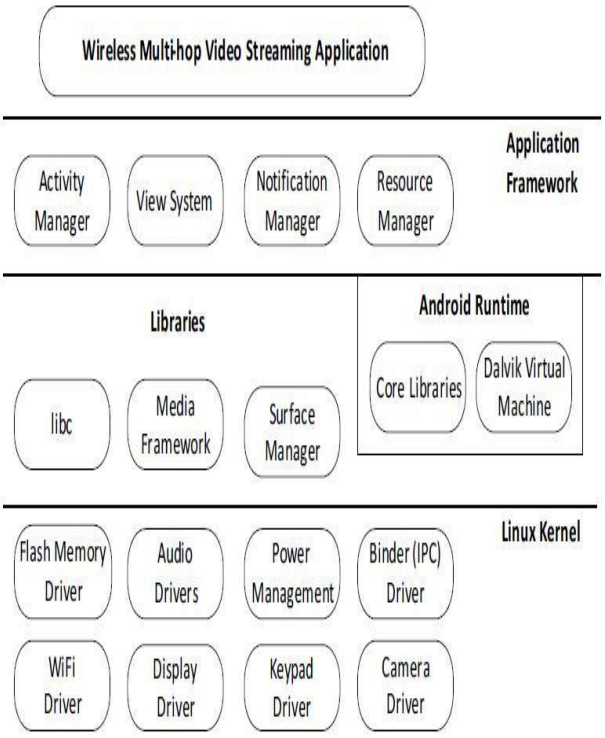

Fig 3. Android System Architecture.

1.2 Codecs and Method of Video Encoding

Video encoding and decoding is an essential aspect of any video streaming application. There are many ways by which a video can be encoded or decoded. We describe two widely used video coding techniques which are implemented in our application module.

• *Intraframe encoding* is the simplest form of encoding. It treats every frame as an individual image to encode. This method is resilient against lost frames due to each frame having enough information to create an entire image.

• *Interface encoding* uses two types of frames, i.e., the key frames and predicted frames, for better compression ratio. The key frame contains complete information to create an image, whereas the predicted frames only contain the differences between frames thus previous frames are required for their successful decoding.

III. DESIGN AND IMPLEMENTATION

1.1 Architectural Design

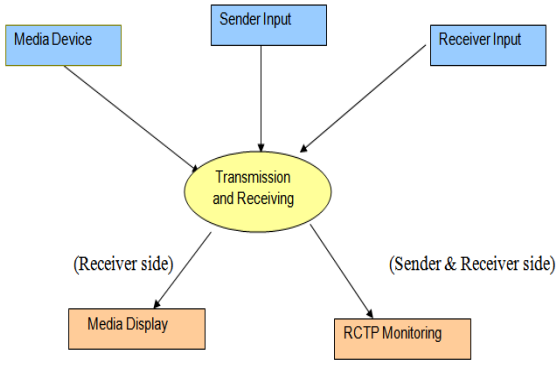

Fig 4. Architecture of the Project

The Mobile Application is broken down into two major subsystems, the mobile sender would send an video content through an common wireless media using RCTP Monitoring protocol . This Application allows sharing live information captured by mobile phone sensors with persons that might be multiple wireless hops away. The video streaming is based on peer-to-peer communication between mobile phones.

1.2 Decomposition Description

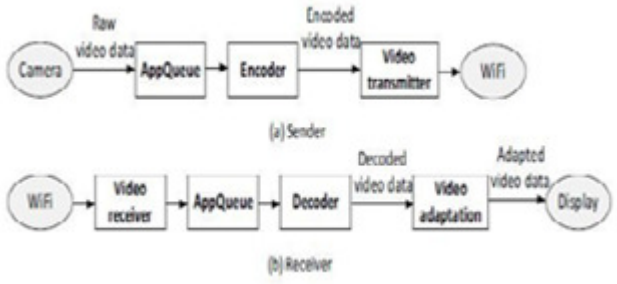

Fig 5. Process of Encoding and Decoding

The mobile application subsystem is divided up into a three layered architecture; it has a user interface, application, and device layer. Each layer has its own interface that other layers can use to interact with it. The user interface layer contains an observer object and updates its data, using data from the observable application layer, via the observer pattern. The application layer handles threads and messages from the user interface layer messages send them to the device layer. The device layer handles the interactions with the hardware, all the features of the phone necessary for the application, including but sending video streaming over wi-fi, and ports to send and receive data to and from the other Android phone. Interface layer will handle Video encoder/decoder module.

1.3 Video Streaming using H.264 encoding and Decoding

H.264 is an industry standard for video compression, the process of converting digital video into a format that takes up less capacity when it is stored or transmitted. Video compression (or video coding) is an essential technology for applications such as digital television, DVD-Video, mobile TV, videoconferencing and internet video streaming. Standardizing video compression makes it possible for products from different manufacturers (e.g. encoders, decoders and storage media) to inter-operate. An encoder converts video into
a compressed format and a decoder converts compressed video back into an uncompressed format. Recommendation H.264: Advanced Video Coding is a document published by the international standards bodies ITU-T (International Telecommunication Union) and ISO/IEC (International Organization for Standardization / International Electro technical Commission). It defines a format (syntax) for compressed video and a method for decoding this syntax to produce a displayable video sequence. The standard document does not actually specify how to encode (compress) digital video – this is left to the manufacturer of a video
encoder – but in practice the encoder is likely to mirror the steps of the decoding process. Figure 6 shows the encoding and decoding processes and highlights the parts that are covered by the H.264 standard.

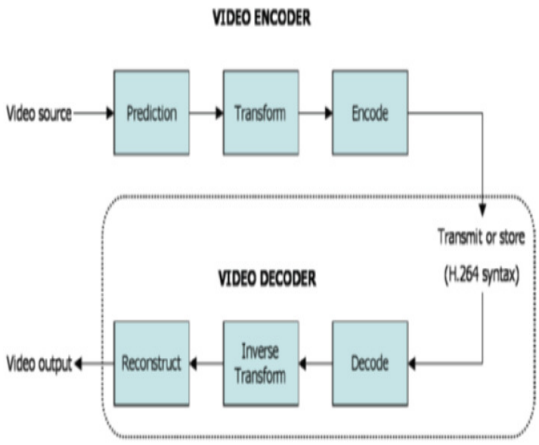

Fig 6: Video Encoder and Decoder

1.4 Encoding process Prediction

Prediction of the macro block based on previously-coded data, either from the current frame (intra prediction) or from other frames that have already been coded and transmitted (inter prediction).

1.5 Transform and quantization

A block of residual samples is transformed using a 4x4 or 8x8 integer transform, an approximate form of the Discrete Cosine Transform (DCT). The transform outputs a set of coefficients, each of which is a weighting value for a standard basis pattern.

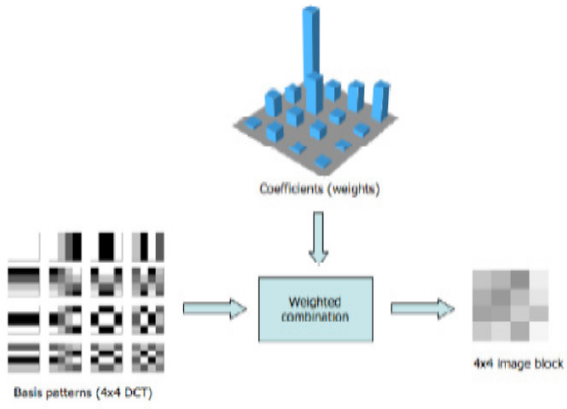

Fig 7. Transform and Quantization.

1.6 Decoder Process Bit Stream Decoding

A video decoder receives the compressed H.264 bit stream, decodes each of the syntax elements and extracts the information described above (quantized transform coefficients, prediction information, etc).

1. Rescaling and inverse transform

The quantized transform coefficients are re-scaled. Each coefficient is multiplied by an integer value to restore its original scale2. An inverse transform combines the standard basis patterns, weighted by the re-scaled coefficients, to re-create each block of residual data.

2. Reconstruction

For each macro block, the decoder forms an identical prediction to the one created by the encoder. The decoder adds the prediction to the decoded residual to reconstruct a decoded macro block which can then be displayed as part of a video frame.

1.7 Multi-hop Setup

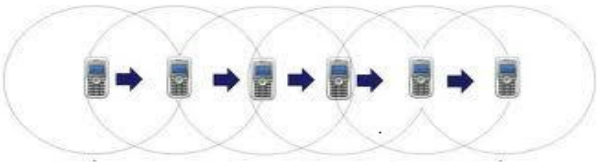

Fig 8 Multi-Hop

In multi-hop wireless networks, as shown in fig 8 communication between two end nodes is carried out through a number of intermediate nodes whose function is to relay information from one point to another.

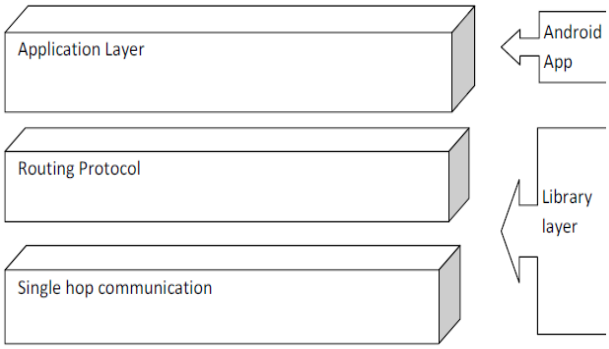

Fig 9. Placement of packages at each Hop

1.8 Real Time Streaming Protocol (RTSP) and RTP

It is a network control protocol designed for use in entertainment and communications systems to control streaming media servers. The protocol is used for establishing and controlling media sessions between end points. Clients of media servers issue VCR-like commands, such as *play* and *pause*, to facilitate real-time control of playback of media files

from the server. Options, Describe, Setup, Pause, Record, Announce, Teardown, Redirect, Set Parameter.

Real-time Transport Protocol (RTP) defines a standardized packet format for delivering audio and video over IP networks. RTP is used extensively in communication and entertainment systems that involve streaming media, such as telephony, video teleconference applications, television services and web-based push-to-talk features. RTP is used in conjunction with the RTP Control Protocol (RTCP). While RTP carries the media streams (e.g., audio and video), RTCP is used to monitor transmission statistics and quality of service (QoS) and aids synchronization of multiple streams. RTP is originated and received on even port numbers and the associated RTCP communication uses the next higher odd port number. RTP is one of the technical foundations of Voice over IP and in this context is often used in conjunction with a signaling protocol which assists in setting up connections across the network. The main processes are identified as under:

[1] . Adhoc net work /Wi-Fi setting need to be done with IP/Port

[2] . We need to establish the connection between android mobile user and Receiver User

[3] . When application initiate , User has to launch request and start streaming

[4] . It will verify camera is connected and then after process Raw Video .

[5] . It Uses H.264 Encoder/Decoder for this process

[6] . Once the streaming start, video will process in encoding with the help of H.264 encoder.

[7] . Once Encoding is done and it will go in network with the help of RTSP/RTP

[8] . Receiver will receive the video and display on their device after the decoding.

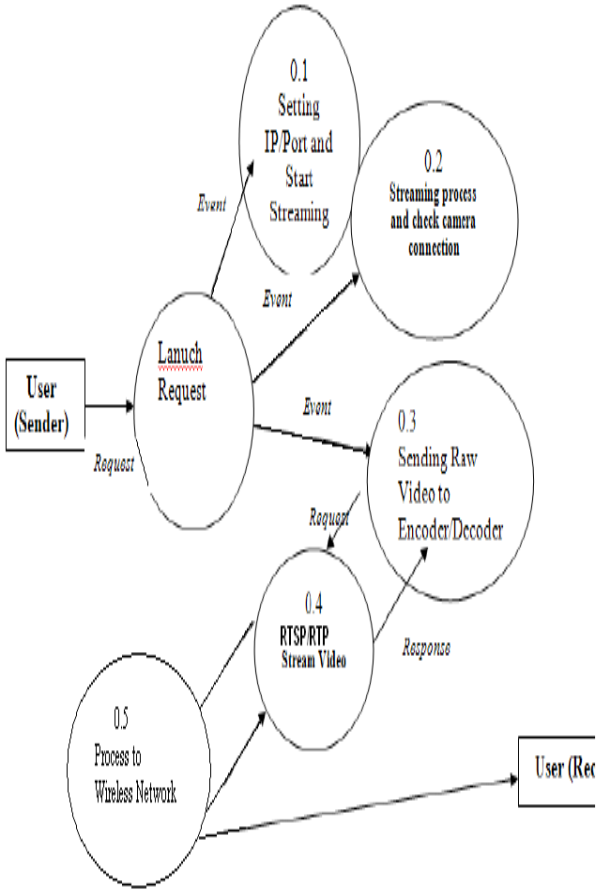

Fig 10 . RTSP & RTP Process

1.8.1 RTP TRANSMIT MODULE

In this module, we use RTP protocol since it is best effective for Live media transmission. The output should be transmitted in JPEG/RTP or JPEG format so that the transmission is made easy and faster. The JMF API is used to read the source and convert it to JPEG/RTP or RAW_RTP packet data. Also, we need to specify the port through which we are going to transmit the Data to the requests. Create a processor for the specified media locator. Create an RTP session to transmit the output of the processor to the specified IP address and port no.

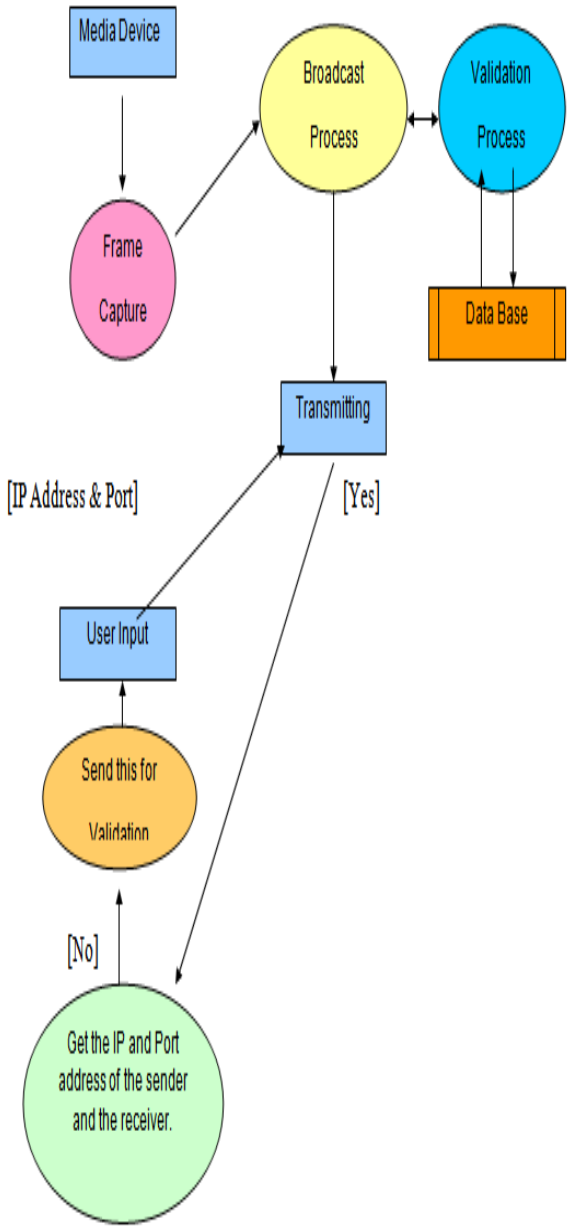

Fig 11. Dataflow Diagram of RTP Transmitter

1.8.2 RTP RECEIVE MODULE

The output that's being transmitted by the sender system has to be captured by the program and be viewed on the Live panel. The panel that is received may be in JPEG/RTP or RAW_RTP. It depends on how the transfer takes place. We need to specify the IP from which Media data are received, the port number through which the server is transmitting the data. It allows receive media streams containing any number

of tracks from a number of different hosts/transmitter on the network simultaneously.

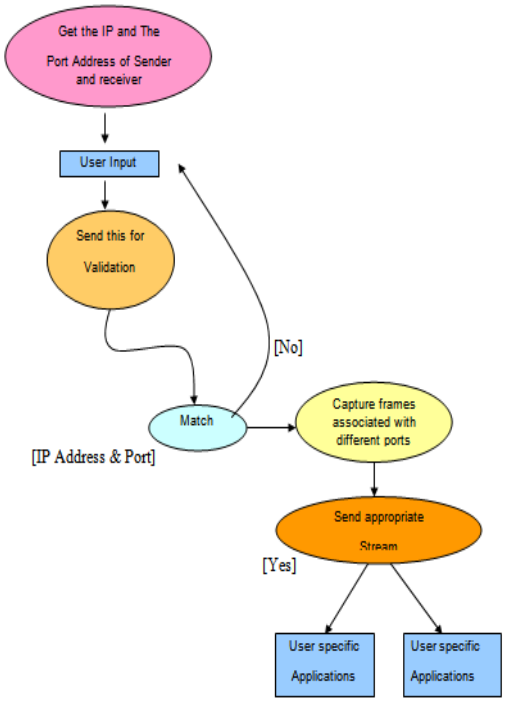

Fig 12 . Dataflow diagram of RTP Receiver.

IV. EVALUATION

In this section, we present an evaluation of the video streaming application in a number of experimental scenarios and discuss the results.

1.1. Experiment devices and setup

Downloading the SDK Starter Package

The SDK starter package is not a full development environment—it includes only the core SDK Tools, which you can use to download the rest of the SDK components (such as the latest Android platform).

Installing the ADT Plugin for Eclipse

Android offers a custom plugin for the Eclipse IDE, called Android Development Tools (ADT), that is designed to give you a powerful, integrated environment in which to build

Android applications. It extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, debug your applications using the Android SDK tools, and even export signed (or unsigned) APKs in order to distribute your application. Downloading the ADT Plugin, Configuring the ADT Plugin, Updating the ADT plugin and Adding Platforms and Other Components .

We deploy the application on each of these phones. Fig. 13(1,2) shows a screen shot of the live view of the application and its menu bar.

• *Start Streaming* triggers the actions to broadcast information to the multi-hop network that the node is a video content provider, gather raw video frames from the camera, encode these video frames using a particular codec, and finally stream out these encoded video frames.

• *Request Video* starts the discovery process to search for video content providers within the network; following that, users can select from a list of providers the node from which they will receive video streams. Due to the use of the standard Android development kits, the application can be easily deployed on all of these mobile.

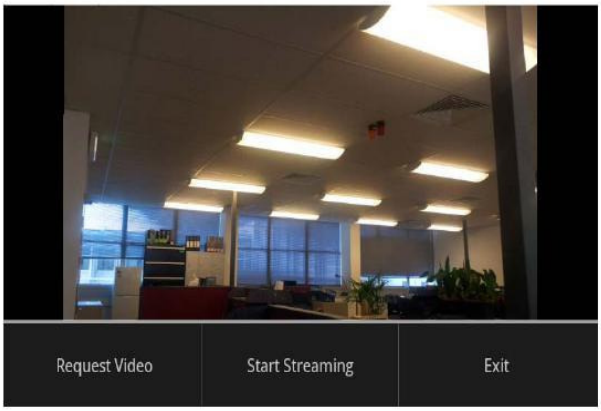

Fig 13.1 UI Design

Fig 13.2 UI Layout Design

1.2. Performance of video codecs and coding

In a live video streaming application, the performance of video encoding and decoding is an indication of the feasibility of the application. The speed of en/decoding has
impact on perceived quality of the video and in turn affects the usability of the application. Having tested our application on newer generation phones, we repeat the same set of experiments for the other two models of mobile phones.

Fig. 14 shows a comparison in encoding time for different models of mobile phones using the aforementioned codecs and encoding techniques. There is no surprise to see that the HTC Dream, which is the first generation of Android phones with a much lower resource profile, performs much worse than the Samsung Galaxy S2.

In the worst case when using H.264 with intraframe encoding it takes up to around 100 ms to encode a video frame. However, according to [10] 100 ms encoding time should be able to produce video streams at around 10 fps, which is still a fairly good quality of a video stream. Fig. 14 also shows that we can reduce the encoding time by using other codecs or encoding techniques, even on the first generation of Android phones.

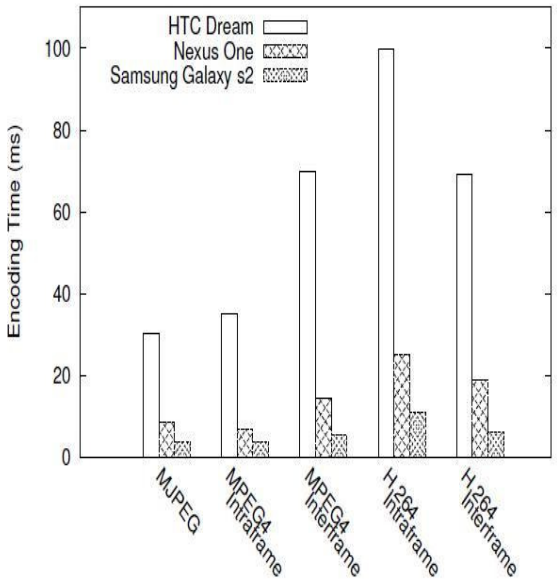

Fig 14. Comparison of encoding time using Akiyo Sequence

V. Conclusion & Future Work

A wireless multi-hop video streaming application for Android mobile phones application allows users to capture live video feeds using the mobile phone camera and to share these feeds with people who might be multiple wireless hops away. The video feeds are shared using wireless client mesh network (ad-hoc network) established between mobile phones. Thus the video streaming does not rely on a traditional network infrastructure (such as the cellular), therefore it is a free-of-charge communication. Such a multi-hop video streaming can be used in a variety of application domains including social networking.

In this paper we presented an evaluation of the prototype application feasibility of the multi-hop video on three generations of Android phones (with different resource capabilities). We showed that even after five wireless hops, our application still can handle video streams with high quality. For future work we are planning to extend the evaluation test to study the application performance within a larger network.

We are also consider that developing a richer user interface with additional features, such as implementing multicast over multiple hops and allowing users to record video
contents on local SD cards while streaming or forwarding. Introducing a digital rights management protocol that can protect the shared contents between sources to destination transmission.

VI. References

References listed below contain all the technical publications, text books, web sites etc. used during the project:

[1] . ITU. Statistics on global mobile subscriptions. http://www.itu.int/newsroom/press releases/2010/06.html.

[2] . C. Quick. (2009) With smartphone adoption on the rise, opportunity for marketers is calling. http://blog.nielsen.com/nielsenwire/online mobile/with-smartphoneadoption-on-therise-opportunity-for-marketers-is-calling/.

[3] . H. L. Cycon, T. C. Schmidt, G. Hege, M. Wahlisch, D. Marpe, and M. Palkow, "Peer-topeer videoconferencing with h.264 software codec for mobiles," in Proc. Int. Symp. a World of Wireless, Mobile and Multimedia Networks WoWMoM 2008, 2008, pp. 1–6.

[4] . Qik. www.qik.com

[5] . Android development sdk http://developer.android.com.

[6] . Optimized link state routing protocol (olsr). http://www.ietf.org/rfc/rfc3626.txt.

[7] . P. Hu, W. L. Tan, R. Wishart, M. Portmann, and J. Indulska, “Meshvision: an adaptive wireless mesh network video surveillance system,” Multimedia Systems,vol.16,pp.243–254, 2010, 10.1007/s00530-010-0191-z.

[8] . Q. Huynh-Thu and M. Ghanbari, “Temporal aspect of perceived quality in mobile video broadcasting,” Broadcasting, IEEE Transactions on DOI 10.1109/TBC.2008.2001246, vol. 54, no. 3, pp. 641–651, 2008. 787.

[9] . Mohanjeet Singh, D.S Dhaliwal and Neeraj Garg, “Searching and Streaming of Multimedia Content in P2P Overlay Network”, International Journal of Computer Engineering & Technology (IJCET), Volume 3, Issue 2, 2012, pp. 433 - 438, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.

[10] . Srikanth T.N. and Prabhudeva S, “Explicit Study on Security Issues in Multimedia Streaming in Peer to Peer Network”, International Journal of Computer Engineering & Technology (IJCET), Volume 3, Issue 2, 2012, pp. 588 - 602, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.

[11] Asokan M, “Android Vs iOS – An Analysis”, International Journal of Computer Engineering & Technology (IJCET), Volume 4, Issue 1, 2013, pp. 377 - 382, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.