On the Forwarding Capability of Mobile Handhelds for Video Streaming over MANETs

Stein Kristiansen, Morten Lindeberg, Daniel Rodríguez-Fernández and Thomas Plagemann
Department of Informatics
University of Oslo, Norway
[steikr, mglindeb, dani, plageman]@ifi.uio.no

ABSTRACT

Despite the importance of real-world experiments, nearly all ongoing research activities addressing video streaming over MANETs are based on simulation studies. Earlier research shows that the limited resources of mobile handhelds, which are not modeled in most network simulators, can be a severe bottleneck. We study the capability of a modern handheld to perform one core task, which is the forwarding of video streams. We present end-to-end video quality and network measurements, along with an analysis of resource consumption. Our studies of the recent handheld Nokia N900 show that it can forward up to 3.70 Mbps. However, subjective video quality is compromised already at 3.35 Mbps, due to excessive delay. Our analysis unveils that direct memory access (DMA) relieves the CPU of forwarding overhead and that, due to the digital signal processor (DSP) support, additional coding overhead does not decrease the forwarding capacity. Finally, we find that power management impacts results considerably. It is possible to increase the forwarding capacity up to 27.4% by increasing the frequency of internal buses. Hence, our results demonstrate that the forwarding capacity is highly dependent on the internal state and activity of the device.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: PERFORMANCE OF SYSTEMS—*Reliability, availability, and serviceability*; H.4.3 [**Information Systems**]: INFORMATION SYSTEMS APPLICATIONS —*Communications Applications*

General Terms

Experimentation

Keywords

MANETs, Video streaming, Handhelds, Performance evaluation

MobiHeld 2010, August 30, 2010, New Delhi, India.

1. INTRODUCTION

Mobile handhelds such as PDAs and mobile phones are a part of everyday life for many persons. The technology has improved in such a way that they now can present and capture multimedia. Recent handhelds, such as the Google Nexus One, iPhone 3GS and Nokia N900, have built-in video camera(s) and a digital signal processor (DSP) which enables fast and efficient multimedia coding. The IEEE 802.11 interfaces of the devices can be used in both infrastructure and infrastructureless mode (ad hoc mode). The latter allows creating local networks using multiple cooperating devices without the need of any existing infrastructure. Such networks are often called mobile ad hoc networks (MANETs). One application domain, in which MANETs can be useful, is the creation of data communication services during rescue and emergency operations. In such operations, infrastructure might not exist (e.g., remote areas) or be partially or entirely destroyed (e.g., an earthquake). In such stressful environments, multimedia services, such as live video feeds or video conferencing, are expected to improve communication among rescue personnel. It should be noted that in this kind of application domain, there is a natural intensive for all participants to contribute with their resources.

There are many research challenges that need to be addressed in order to bring video streaming over MANETs to reality. MANETs inhibit dynamic and unreliable communication conditions, and generally offer significantly less bandwidth than wired networks. Also, the multimedia content has strict network demands, e.g., bandwidth, delay and jitter. The nodes, especially handhelds, may have limited resources, e.g., battery and processing power. Multimedia centric MANET routing protocols and adaptable video coding solve some of the numerous challenges. These do often not follow the principles of layering, making them incompatible with most existing technology and protocols. Also, they have not yet matured to the point that they are available to the public. Due to the complexity of implementing and running such solutions on real handhelds, experimental results are often obtained through simulations that simplify several aspects of the involved system.

Preliminary real-world experiments with handhelds indicate that their CPU might be a bottleneck for forwarding. Halvorsen et al. [2] revealed that almost 100% CPU is consumed at a Nokia 770 Tablet receiving and playing a 1 Mbps video stream. The high amount of CPU consumption induces that the node is not able to perform other tasks. This is important because, as mentioned, most research addressing the challenges of MANET video streaming use simulation, rather than real-world experiments. It should be noted that the major simulation tools used, such as ns-2[1], do not consider that the nodes themselves can be bottlenecks. This opens up questions regarding the realism of the obtained results. Therefore, we focus on analyzing the forwarding capabilities of recent mobile handhelds.

In this paper, we assess the feasibility of using new generation handhelds for the purpose of video streaming over MANETs. We have set up an experimental test-bed at a location with minimal wireless interference and carried out extensive performance evaluations of a Nokia N900 forwarding video streams in wireless ad hoc mode. Depending on the particular situation, a resource-constrained node may be required to perform several resource-demanding tasks simultaneously, such as forwarding and video coding. Hence, we also unveil how encoding and decoding impact forwarding.

The outline of this paper is as follows: In Section 2, we discuss challenges of conducting realistic performance analysis, and related work. Section 3 describes our experimental set-up. The experimental results are described in Section 4. Section 5 concludes and outlines future work.

2. REAL-WORLD EXPERIMENTS

Conducting realistic, repeatable and comparable performance evaluation of systems for video streaming over MANETs brings many challenges. The system is difficult to model due to the combination and complexity of the wireless network, protocols, video codecs and device drivers. For instance, node density and mobility will significantly affect the experimental results. For repeatability and comparability, such parameters, and even the wireless link conditions must be kept similar between experiments. This is often unrealistic for real-world experiments. Thus, for repeatable performance evaluations involving MANETs, network simulators such as ns-2 are typically used. However, most simulators do not model node resources, which have shown to have a potentially severe impact on performance. Therefore, real-world experiments are important for obtaining realistic results.

Real-world experiments can however be cumbersome and time consuming. During our experiments, we experienced various practical difficulties. Avoiding uncontrollable interference from nearby WLANs was a challenge on its own. Also, we experienced problems with the wireless ad hoc mode of the N900. Frequently, a kernel thread, seemingly "hanging," made the device incapable of transmitting or receiving any data over the wireless medium at all. Furthermore, the video and codec hardware drivers for the N900 are still under development. Simply changing codec settings on the N900 to utilize the DSP was problematic, due to the lack of documentation. During our experiments, the handheld sometimes rebooted or shut down, without any obvious cause, making it impossible to automate many subsequent experiment runs. We did not have time to investigate the cause of this further.

2.1 Related Work

Few real-world experiments study video streaming over MANETs. In [3], 300 KBps constant bitrate streams are transmitted from a laptop to one or more Zaurus SL-C1000 PDAs. The number of hops varies between one and four, and routes are maintained by running the OLSR [1] routing protocol. Their results indicate that stream reception at the intermediate nodes affects end-to-end packet loss considerably and that jitter and delay increases with hop-count.

[1] `http://www.isi.edu/nsnam/ns/`

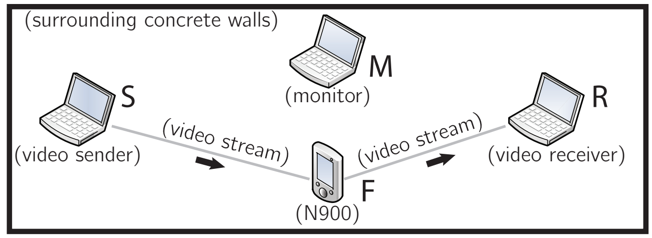

Figure 1: Schematic drawing of test-bed

The above paper does not report any node resource measurements. In [9], Xue et al. investigate CPU-consumption and intra-node delay on laptops forwarding video streams, running different operating systems. They generate video streams of different formats at rates ranging from 128 to 2000 Kbps. Hardware and operating systems significantly affect forwarding capacity, in certain cases incapacitating the forwarding laptop at bitrates as low as 512 Kbps. During forwarding, the packet spends a considerable amount of time being copied to and from the network card (e.g., up to 2.83 ms for 1500-byte UDP packets).

The most closely related work is presented in [2]. Here, an experiment is conducted in which video streams are pre-encoded at the bitrates 200, 500 and 1000 Kbps. These are streamed across a 2-hop MANET consisting of three Nokia 770 handheld devices. OLSR is used to establish and maintain routes. The measurements indicate that CPU is the bottleneck both for decoding and forwarding. With a single stream, no more than 1000 Kbps can be decoded at the receiver. The intermediate node is able to forward 12, six and three streams with bitrates of 200, 500 and 1000 Kbps, respectively, with a video quality that the authors consider acceptable.

What distinguishes our work from [2, 3, 9] is first of all that they utilize 11 Mbps physical net bitrate, while we utilize 54 Mbps. While [9] differs by the use of laptops, [3] and [2] are more related since they investigate handhelds. However, the N900 is based on a newer generation architecture, which we found makes more extensive use of direct memory access (DMA) to relieve the CPU from the packet copying overhead during network transmission. Finally, none of the efforts above investigate how video coding affects forwarding capacity.

3. EXPERIMENT DESIGN

Our experiments are designed to unveil the capabilities of one of the handhelds of today, a Nokia N900, to forward video streams encoded with the state-of-the art H.264 codec. Furthermore, we investigate how the additional tasks of encoding and decoding affect its forwarding capacity.

Set-Up: Figure 1 shows our test-bed surrounded by concrete walls which block out most interference from outside WLANs. Nodes S and R act as the sender (generating streams) and the receiver (consuming streams), respectively. They are responsible for imposing the forwarding workload on F (the N900). Since our primary goal is to unveil forwarding capability, rather than the effect of routing, a static route through F is set up before we conduct the experiments. Node M is within transmission range of all nodes, and promiscuously monitors and stores all transmitted wireless network traffic for later analysis. Since we study the performance of F, we want to avoid other potential bottlenecks in the end-to-end path. Therefore, nodes S, R and M are laptops which are more powerful than F.

In the first experiment (`FW`), we investigate only the forwarding capability of F. In target scenarios, nodes might also produce or consume additional video content. In the second experiment (`FWENC`), we investigate how forwarding is affected when F, in parallel, has to encode a live video feed from the back camera and stream it towards R. In our final experiment (`FWDEC`), instead of encoding, F decodes a stream from S. For all experiments, we repeat each run five times, across which we present the average measurement values. The standard deviations between runs are not presented, as they are found to be negligible.

Workload: The stream content is the clip entitled "foreman" obtained from Video Traces Research Group[2], in 352x288 resolution (CIF). It has a duration of 12 seconds, and contains one section with low motion (an interview) and one with high motion (pivoting of the camera). Due to its widespread use in literature, using this clip makes our results comparable with existing results. Also, due to the instability of our testbed, we prefer a clip of short duration to minimize the probability of unsuccessful experiment runs. Close investigation of raw measurement data unveils relatively stable performance throughout the experiment runs, suggesting that our results are valid also for clips of longer durations. The clip is pre-coded using H.264 [8], at the bitrates presented in Table 1. The resulting video consist of I and P frames, with a group of pictures (GOP) size of 30. From the lowest to the highest bitrate values, the encoded video ranges from what we consider to be acceptable to high quality. In Table 1, we also present the average peak signal to noise ratio (A-PSNR) of the encoded video before transmission. According to the possible PSNR to mean opinion score (MOS) conversion from [5], the values in Table 1 fit within "Good" to "Excellent" MOS ranges.

We use RTP to transmit the streams, where each packet contains no more than one video frame. A frame can however span several packets if required. Maximum application layer packet size is set to 1400 bytes, to avoid fragmentation at the layers below. To unveil how F handles the forwarding of multiple streams for each bitrate, we increase the number of streams to reach an aggregate bitrate of up to 5 Mbps at the application layer. We selected this range of bitrates based on preliminary experiments to investigate how F performs while subject to workloads up to, and beyond its forwarding capacity. To exclude the wireless medium as a potential bottleneck, we performed preliminary experiments with M as the intermediate node, obtaining overall higher throughputs than with the N900. We need a higher number of low bitrate streams (e.g., 128 Kbps) to reach the same aggregated bitrate as with high bitrate streams (e.g., 1024 Kbps). Note that this results in a higher packet rate for low bitrate streams. In `FWENC` and `FWDEC`, the additional encoded and decoded video at F has a bitrate of 512 Kbps, yielding what we consider to be acceptable perceptual quality. This bitrate is kept constant across all runs to clearly indicate how the encoding/decoding of acceptable video affects performance during varying forwarding workloads.

Metrics: We measure end-to-end delay, throughput and packet loss rate. As a measure on video quality, we present the A-PSNR for each video stream after transmission, in order to be comparable with previous work.

M monitors and logs all the ongoing network transmissions during each run. This is to obtain an indication of how much packet forwarding through F contributes to end-to-end delay. We calculate intra-node delay as $\delta T = T_t - T_r$, where T_t is the time at which an 802.11 MAC-frame is transmitted from F, and T_r the time at which it was received by F.

We also measure the consumption of network bandwidth and CPU at F during the experiments. The latter to see whether the CPU is still the bottleneck, as reported in [2]. Finally, interrupts are logged for obtaining an overview of hardware activities.

3.1 Hardware Description

The N900 is the most recent Linux based handheld from Nokia. It includes a TI OMAP 3430 [7] multimedia applications processor, making it more powerful than its predecessors (e.g., the Nokia 770 used in [2]). The OMAP 3430 integrates several components including a general purpose CPU (600 MHz ARM Cortex-A8), a DMA, and a DSP (430MHz TI C64x+). The latter relieves the CPU of multimedia encoding and decoding. The DSP supports the video codecs H.263, MPEG4 and H.264 at up to 720p HD resolution. The DMA controller is used for memory-to-memory, memory-to-periferical and periferical-to-memory data transfers. The memory consists of 256 MB SDRAM, and 768 MB NAND flash memory. The N900 includes a camera in the back and one in the front, with resolutions of 848x480 and 640x480, respectively. The wireless interface is a TI WL1251 which conforms to the 802.11b/g specifications, in our case operating at 54 Mbps physical layer net bitrate. Note however that the obtainable MAC-layer throughput is considerably lower, see e.g., [4].

The higher bandwidth enabled by 802.11g and the DMA support are beneficial for data forwarding, while the DSP support is beneficial for multimedia encoding and decoding. Such features are becoming prevalent in today's handhelds.

The nodes S and R are over provisioned with 2.6 GHz Intel Centrino Duo CPUs and 3 GB of RAM, and M has a 2.2 GHz Intel Core Duo CPU and 2 GB of RAM.

3.2 Software Description

In our experiments we use a N900 that runs Maemo[3] 5 3.2010.02-8 on a 2.6.28-omap1 Linux kernel. Maemo is a slimmed version of Debian GNU/Linux. Its multimedia application support is based on the GStreamer multimedia framework[4] version 0.10. The GStreamer support in the N900 includes a plug-in[5] that provides access to the DSP. This is used in `FWENC` and `FWDEC` for coding. The measurements of CPU and memory utilization are obtained by using `sar`, included in the sysstat 7.0.0-4osso4. These are measured once per second. The resource consumption of running `sar` has shown to be neglectable (< 0.5 % of the CPU).

[2] `http://trace.kom.aau.dk/`

[3] `http://maemo.org/`

[4] `http://www.gstreamer.net/`

[5] `http://code.google.com/p/gst-dsp/`

Table 1: Stream Workload

Bitrate	PSNR after Enc.	Size of I-Frames (bytes)	Size of P-Frames (bytes)	No. of Streams
128 Kbps	31.83 (±1.38)	5664.56 (±2544.00)	490.90 (± 346.22)	{1, 4, 8, 12, ..., 40}
256 Kbps	35.15 (±1.37)	11229.22 (±4934.37)	1039.14 (± 773.88)	{1, 2, 4, 6, ..., 20}
512 Kbps	38.20 (±1.46)	19273.89 (±7117.66)	2263.41 (±1729.21)	{1, 2, 3, 4, ..., 10}
1024 Kbps	40.05 (±1.65)	29568.78 (±8792.11)	4877.93 (±3636.67)	{1, 2, 3, 4, 5}

Nodes S, R and M run Ubuntu GNU/Linux 9.10 on a 2.6.31-19 Linux kernel. The EvalVid framework [5] is used for the evaluation of the quality of video transmitted over our test-bed. This tool computes the packet loss rate, the end-to-end delay and the PSNR. In our experiments, we use the provided tool `mp4trace` to stream the pre-encoded video clips across the network. The tool `etmp4` combines the output from `mp4trace` with the original raw video clip, the encoded video clip and `tcpdump` traces retrieved from S and R to obtain the above mentioned metrics. The methods used by EvalVid require all clocks on the involved devices to be synchronized. Thus, we make use of NTP [6] to synchronize the clocks of S, R and M. The clock skew during each experiment run is logged and found to be negligible. M uses `tshark`[6] in order to capture the traffic received and forwarded by F.

4. RESULTS

In this section, we present results from experiments `FW`, `FWENC` and `FWDEC`. We start by presenting the end-to-end delay and the A-PSNR at the receiving node. Since F is the bottleneck, we thereafter look into the actual rates at which data is transmitted from the N900 (i.e., forwarded). Finally, we present CPU measurements and an analysis of the obtained results.

The top plot in Figure 2 shows how average end-to-end delay increases with increased aggregated bitrate. These plots show an initial linear increase reaching up to 20 to 200 ms, depending on the experiment and bitrate of the individual streams. N refers to the number of streams given in Table 1. When N exceeds certain values, we observe a sudden significant increase in delay due to saturation of F. It appears that 200 ms is the boundary between pre- and post-saturation measurements. We define N_{max} to be the maximum number of streams allowed before delay exceeds 200 ms. Values for N_{max} are presented in Table 2. We see higher values for low bitrates streams (i.e., with smaller packets) than for high bitrate ones. Notice also that in `FWENC`, we are not able to saturate F with the selected workload.

The middle plot in Figure 2 shows A-PSNR. We see that it stays close to the one of the encoded video (see Table 1) as N increases up to, and somewhat beyond N_{max}. After this point, lower bitrate streams (128 and 256 Kbps) suffer a drastic decline in quality, while the high bitrate ones remain at a high level. The A-PSNR decreases due to packet loss, presented in the bottom plot in Figure 2. We see that packet loss occurs only with low bitrate streams, in our experiments reaching up to 40.3% (FW), 21.9% (FWENC) and 21.8% (FWDEC) for 128 Kbps streams. This is because a high number of streams are required to obtain the same aggregate bitrate as with the high bitrate streams, resulting in very high packet rates. Before saturation, however, packet loss is negligible in all cases, never exceeding 0.05%.

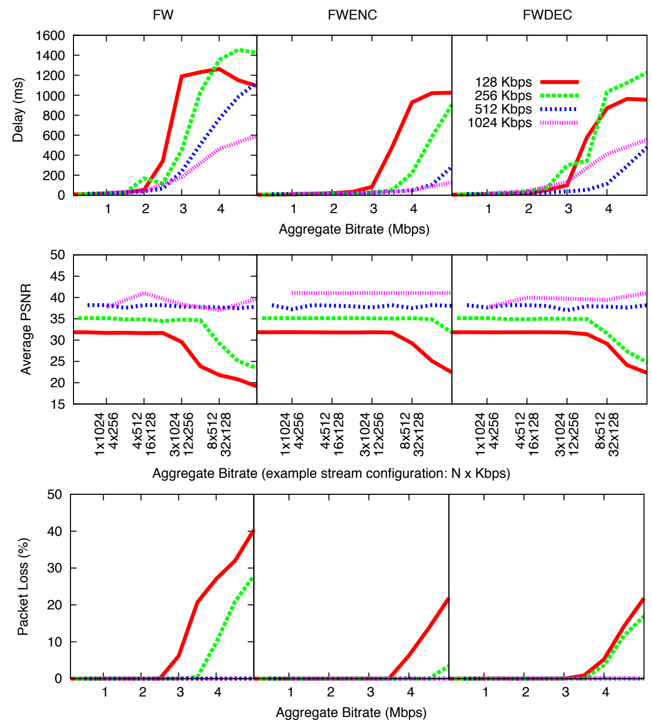

Figure 2: End-to-end delay, A-PSNR and packet loss.

The top plot in Figure 3 presents the traffic transmitted from F with increasing aggregate bitrate, to indicate the forwarding capacity of the N900. Note that for `FWENC`, this includes the extra stream encoded and streamed by F. This stream has a bitrate of 0.44±0.03 Mbps, and should be deducted when calculating the maximum forwarding capacity FW_{max}. This value has already been deducted in Table 2, where we present FW_{max} for all experiments. Note that the transmitted data measured by `sar` also includes packet headers, resulting in slightly higher bitrates than what is transmitted from the application. We observe higher rates for `FWENC` and `FWDEC` than for `FW` (explained in the analysis). As expected, higher values are obtained with higher bitrate streams, due to the larger packet sizes. Although FW_{max} is larger than with N_{max} streams, the overall quality of experience is compromised when forwarding more than N_{max} streams due to excessive delay.

[6] `http://www.wireshark.org/`

Table 2: Results (* max not reached)

N_{max}(Amount)	FW	FWENC	FWDEC
128 Kbps streams:	16	24	24
256 Kbps streams:	8	14	10
512 Kbps streams:	5	9	8
1024 Kbps streams:	3	> 5 *	3
FW_{max}(Mbps)	**FW**	**FWENC**	**FWDEC**
128 Kbps streams:	2.83	3.62	3.24
256 Kbps streams:	3.33	4.66	3.83
512 Kbps streams:	3.53	5.33	3.67
1024 Kbps streams:	3.70	> 5.56 *	3.80

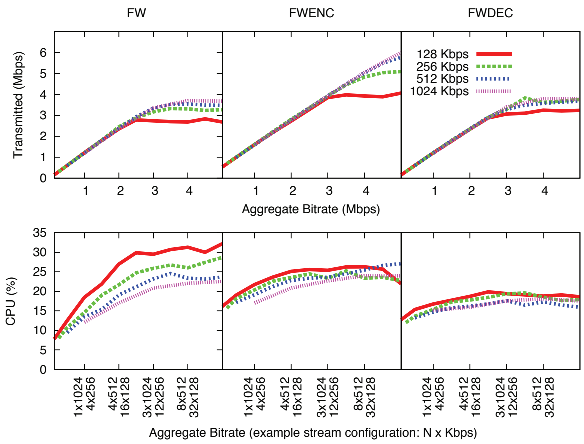

Figure 3: Forwarding bitrates and CPU.

Analysis: Our results show significantly lower values for throughput than what is supported by the wireless medium, indicating that the N900 is the bottleneck. Looking at the CPU measurements (at the bottom in Figure 3), we never observe an average utilization above 35%, indicating that the CPU is not the bottleneck as in [2, 9]. Investigation of the interrupt frequencies indicates that the N900 uses DMA for transmission of packets between the NIC and memory. The CPU nevertheless has to handle the hardware interrupts from the NIC, along with other tasks (e.g., routing). Since the computational requirement of these tasks is not dependent on packet size, CPU usage is dependent only on packet forwarding rate, resulting in a higher load when handling many low bitrate streams. Comparing the three experiments in Figure 3, we also find an overall lower CPU utilization in `FWENC` and `FWDEC`. Since F performs the additional tasks of encoding and decoding, one would expect the opposite. We found that in order to preserve energy, the N900 reduces the CPU frequency during longer periods of inactivity. In `FW`, no actions are performed in addition to the forwarding, causing the CPU frequency to drop to 250 MHz. During `FWENC` and `FWDEC` the CPU frequency remains at 600 MHz and 500 MHz, respectively. Hence, CPU utilization is relative to the current CPU frequency, which must be considered to obtain an accurate indication of the actual CPU consumption.

The drastic increase in delay upon saturation can be explained by retransmissions at the MAC layer. Our laptops were configured with a maximum retransmission limit of seven, which can not be altered with the utilized hardware.

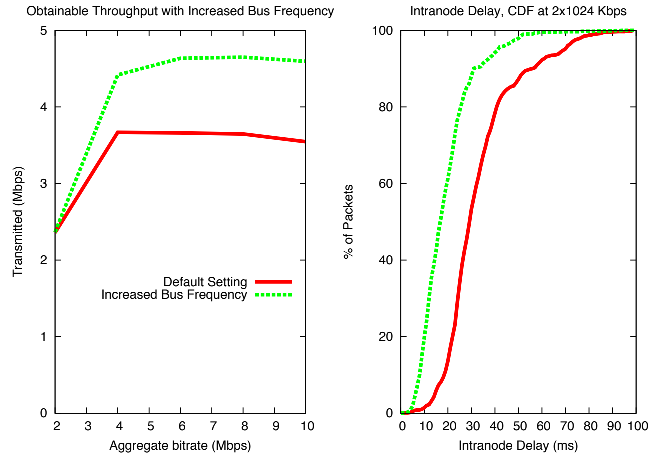

Figure 4: Increased bus frequencies.

As F is not able to forward packets at the rate they arrive, S retransmits packets until they are captured by F, causing an increased delay at S. Upon saturation, up to more than 1 second of video is backlogged at S, resulting in the excessive delay observed in the results.

FW_{max} is lower in `FW` and `FWDEC` than in `FWENC`, due to the power management mechanisms of the Nokia N900 [7]. In addition to changes in the CPU frequency, the frequency of the buses connecting the NIC with the memory, the DMA controller and the CPU depends on the currently selected voltage level. By default, a low level is selected to preserve battery. However, when the camera is activated, the provided software requires a higher throughput from these buses, in turn increasing the bus frequencies at the cost of higher power consumption. In order to maximize throughput, we implemented a kernel module utilizing the Linux-embedded power management QoS API to improve hardware performance. This causes increased bus frequencies at the cost of increased battery consumption. To the left in Figure 4, we present results from two additional experiments indicating the throughput gain provided by loading our module. S transmits two to ten 1024 Kbps streams. The results show that FW_{max} increases by 1.01 Mbps (27.4%), approaching that of `FWENC`. The cumulative distribution function (CDF) to the right in Figure 4 presents intra-node delay, i.e., the time F consumes to forward the packets, from the runs with two 1024 Kbps streams. In these runs, intra-node delay decreases by more than 10 ms after loading the module. Since F requires less time to forward each packet, more of them can be forwarded per time unit.

5. CONCLUSION AND FUTURE WORK

A core element in streaming over MANETs is the nodes' capability to do forwarding. We investigate the capability of a Nokia N900, to forward video streams of varying rates, while possibly at the same time encoding or decoding an additional video stream. By default, the N900 is capable of forwarding up to three 1024 Kbps CIF streams ("Excellent" quality according to [5]), or up to 16 128 Kbps ("Good" quality) streams without quality loss. Thus, we obtain lower aggregated bitrates for lower quality streams. This confirms the existing common belief that small packets are inefficient with regard to obtainable throughput. In experiments with added coding overhead, the forwarding capacity is surprisingly increased to 24 128 Kbps streams, and the N900 is not saturated with five 1024 Kbps streams. This is explained by the power saving mechanisms on the N900 by default lowering CPU and bus frequencies to preserve battery. With the additional coding overhead, these frequencies are automatically increased, positively affecting forwarding capacity. In an additional experiment, we obtain up to 1.01 Mbps (27.4%) improvement by explicitly requesting higher bus frequencies.

Since DMA is used for packet copying, CPU consumption never exceeds 35%, even at maximum forwarding rate. Since the CPU is no longer the bottleneck (as in [9, 2]), forwarding by itself cannot saturate the CPU. Hence, it should be capable of carrying out additional tasks required within a MANET, such as routing and handling user space applications. Furthermore, the additional overhead of encoding, decoding and streaming does not affect forwarding performance negatively, since this is mostly handled by the hardware support of the DSP.

Although the N900 outperforms its predecessors, we obtain a considerably lower throughput than the one supported by the wireless medium. This questions the reliability of existing simulation studies where node resources are not considered.

Future Work: We aim to locate the exact throughput bottleneck within the Nokia. Also, performing multi-hop experiments remains as future work. We also plan to investigate the effect of mobility and dynamic routing. In addition, we aim at comparing these results with those from simulations to obtain a measure on how the restrictions in node resources cause deviations from the simulation results. Finally, proper studies of how power management affects battery consumption are left as future work.

6. ACKNOWLEDGMENTS

This work is funded by the VERDIKT program of the Norwegian Research Council, through the DT-Stream project (no. 183312/S10). The authors would like to thank Hans Vatne Hansen, Piotr Srebrny and Sebastien Mondet for valuable comments and feedback.

7. REFERENCES

[1] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, 2003.

[2] M. Halvorsen, T. Plagemann, and M. Siekkinen. Video Streaming Over MANETs: Reality or Fiction? In *MobiMedia '08, Proceedings of the 4th International Mobile Multimedia Communications Conference*, 2008.

[3] F. Jan, B. Mathieu, and D. Meddour. Video streaming experiment on deployed ad hoc network. In *3rd Int. Conf. on Testbeds and Research Infrastructure for the Development of Networks and Communities. TridentCom '07.*, 2007.

[4] J. Jun, P. Peddabachagari, and M. Sichitiu. Theoretical Maximum Throughput of IEEE 802.11 and its Applications. In *NCA '03, Proceedings of the Second IEEE International Symposium on Network Computing and Applications*, 2003.

[5] J. Klaue, B. Rathke, and A. Wolisz. EvalVid - A Framework for Video Transmission and Quality Evaluation. In *13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, 2003.

[6] D. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305, 1992.

[7] Texas Instruments Inc. *OMAP34xx Multimedia Device, Silicon Revision 3.1.x, Technical Reference Manual*, 2010. Last accessed, May 2010.

[8] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Tech.*, 13(7):560 –576, 2003.

[9] P. Xue and S. Chandra. Revisiting multimedia streaming in mobile ad hoc networks. In *NOSSDAV '06: Proceedings of the 2006 international workshop on Network and operating systems support for digital audio and video*, pages 1–7, New York, NY, USA, 2006. ACM.