Title	Practical aspects of video data transfer to network storage using 802.11 WLAN technology for low-end consumer digital cameras
Author(s)	Corcoran, Peter M.; Raducan, Ilariu
Publication Date	2003
Publication Information	P. Corcoran, I. Raducan, " Practical aspects of video data transfer to network storage using 802.11 WLAN technology for low-end consumer digital cameras", IEEE Transactions on Consumer Electronics, Vol. 49, No. 4, pp. 902-940, November 2003.
Publisher	IEEE
Item record	http://hdl.handle.net/10379/286

Practical Aspects of Video Data Transfer to Network Storage using 802.11 WLAN Technology for Low-End Consumer Digital Cameras

Peter M. Corcoran and Ilariu Raducan

Abstract - *The latest models of low-end consumer digital cameras have begun to support A/V streaming directly to memory card. This greatly enhances the utility of the camera to an end user. In this paper we investigate techniques to provide a similar data storage service from camera directly to a wireless (802.11 WLAN) network server. Based on the results of our investigations a server-side application prototype is implemented which can reliably handle up to 3 concurrent connections from client devices at QVGA MJPEG frame rates up to 20 fps.*

Index Terms — digital cameras, 802.11, WLAN, TCP/IP.

I. INTRODUCTION

There have been very rapid advances in the capabilities of consumer digital-still-cameras (DSC) over the past 3-4 years. Low-end models, with 1-2 Megapixel CMOS sensors, have recently begun to support A/V streaming directly to memory card. This turns a DSC into a low-cost video camera capable of capturing 12-15 minutes of QVGA video on 128 MB memory card. This greatly enhances the utility of the camera to an end user.

In this paper we investigate techniques to provide a similar streaming service from a camera directly to a wireless 802.11 network server. There are many potential applications for such a service. In the home environment it provides a user with a wireless 802.11 network with practically unlimited storage for capturing video within the home environment – a single R/W CD can hold an hour of video. By installing a wireless server appliance in the family car the utility of this service is further enhanced. In a commercial setting the visitors to a theme park or tourist attraction can be provided with a low-cost video camera and pay for the amount of video they actually record during their visit. On leaving the attraction they return the camera and receive a set of CD video recordings.

There are, naturally enough, certain problems associated with streaming data over a wireless network rather than simply writing directly to a file system on memory card. To understand these issues we illustrate, in ***Fig. 1*** below, the internal task structure of a DSC while streaming video to its memory card. This shows a main set of foreground tasks, commonly referred to as the *Viewchain*, which obtain and process image data from the CMOS sensor in real-time, storing the resulting image data in system RAM on completion of each task. As most low-end cameras currently support MJPEG image frames organized in either AVI or Quicktime file format we have used this as the basis for most of the work described in this paper. The present work could be easily modified to support video formats other than MJPEG. In this regard it is worth noting that the 1.6 Mbps data stream required for QVGA MJPEG at 20 fps would be more than adequate to support a full VGA MPEG4 video stream.

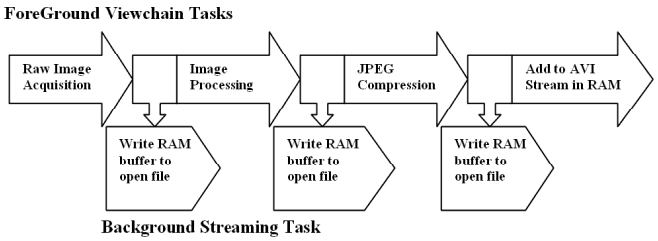


Fig. 1: Foreground and Background Tasks in a typical implementation for a consumer Digital Still Camera.

Most camera chipsets implement significant portions of the *Viewchain* tasks in dedicated hardware peripherals. As a result there are plenty of spare CPU cycles available to write data from system RAM to an internal memory card as a background system task.

In ***Fig. 2*** we illustrate how a video sequence can be written from the *viewchain* to system RAM as a circular memory buffer.

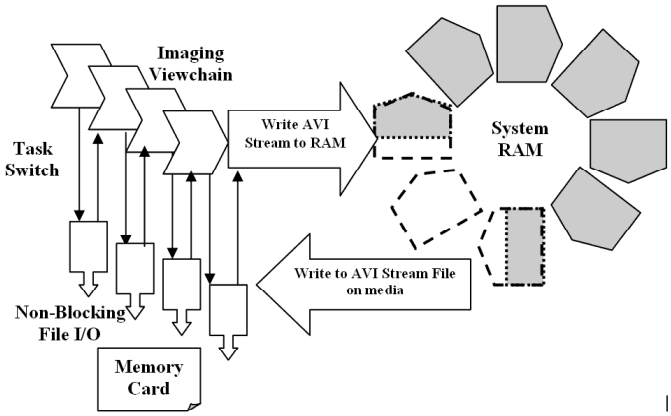


Fig. 2: Overview of Streaming Video to a CF/MMC card via a circular RAM buffer in the camera.

When a foreground task in the *Viewchain* is idle – this happens after data is sent to a hardware peripheral – control is passed to the background task which begins writing data from the RAM buffer onto the memory card in 512 byte sectors. After each sector write the background task checks that the foreground task is still idle and writes the next sector

Peter Corcoran *is with the Consumer Electronics Research Group, of the Electronic Engineering Dept., NUI Galway, Ireland.*

Ilariu Raducan *is with the Consumer Electronics Research Group, of the Electronic Engineering Dept., NUI Galway, Ireland.*

Manuscript received July 28, 2003

from the RAM buffer. Note that the file-system driver must be non-blocking for this form of co-operative multi-tasking to work.

II. Background Research

This paper deals with the core problem of transmitting data over a wireless 802.11, or WLAN connection using a reliable transport protocol. As we review the issues involved in more detail it was clear that there is a wealth of existing research in a number of areas which is very pertinent to this work. Consequently it was decided to review and collate some of the existing literature to provide the reader with a resource for further refinement of the research we present in this paper and to place the paper itself in a broader context.

A. WLAN Characteristics

In order to understand the problem we are investigating better it is useful to provide a short overview of the 802.11 MAC interface. The first point to note is that the 802.11 MAC has been designed to function in a very similar manner to a conventional Ethernet MAC. However, as it is a wireless medium we cannot offer true carrier sense multiple access with collision detection (CSMA/CD) but rather CSMA with *collision avoidance*. This is a subtle difference but, as we shall see, it is the key difference between the wired and wireless solutions.

Typically an 802.11 network will exhibit a forward error rate (FER) of less than 2% and, although the theoretical throughput is of the order of 11 Mbps the practical maximum throughput is 6 Mbps and drops to about 5.2 Mbps if a reliable transport protocol such as TCP/IP is used to transfer data [1, 2]. Network performance can be significantly worse, particularly if there are many nodes on the network. However some prior knowledge of the network topology allows the network to be tuned back towards optimal performance [3]. In our proposed applications for home or automotive use we do not envisage a multi-node network and these network scalability issues are not important. However for a multi-user/multi-node application such as theme-park video it is clear that network topology and the wired/wireless routing design will be critical.

It is also important to note that there are issues arising from the underlying PHY layer – the radio transceivers which underpin the functioning of the 802.11 network. For example in ref. [4] the authors describe how a 5dB RF power differential between two clients can lead to the stronger transceiver capturing an 802.11 radio channel to the practical exclusion of the weaker transceiver. We experienced a related problem where an older 802.11b network card had problems making a reliable connection with the network access point. It is not entirely clear if this was due to a faulty RF transceiver, or due to an early implementation of the 802.11 MAC firmware. In any event this "bad" 802.11 netwaork card slowed the other network clients significantly – by a factor of 30%+. This underlines the point that both network access points and client firmware must be kept up-to-date and that mixing and matching 802.11 equipment should be approached with caution.

B. Transport Protocols for WLANs

Now our first step in implementing our wireless video streaming was to choose a higher level protocol to transport the application data over the 802.11 MAC. There are various alternatives: the standard Internet protocols UDP [7] and TCP [8,10] must be considered; alternatively, other schemes support generic differentiated services [5,6,9] which can provide sophisticated QoS and operate with UDP and/or TCP.

The latter solutions are primarily attractive for large networks where a range of data services may be offered. However they require proprietary server-side solutions and, potentially, some client-side modifications to the underlying network transports and/or applications software. Now our main focus is to investigate solutions for low-end digital cameras. Solutions for such consumer appliances should be interoperable with existing standards and networking technologies in order to appeal to as broad a range of end-users as possible. In this context it does not make sense to add the overhead of proprietary network software if it can be avoided. Thus it was decided to focus our investigations on implementing our video stream using standard Internet protocols over the existing 802.11b WLAN MAC. A more detailed discussion of the background to this decision is given in *section II-E* of this paper.

Having decided to limit ourselves to a standard protocol the next issue was to decide if UDP might provide a suitable solution. Many multimedia applications have begun to rely on UDP as a means to a faster throughput with less overhead than is offered by the TCP protocol. However using UDP over a WLAN implies that we will have an FER of 1.5%-2.0% even on a direct point-to-point link. Again this becomes problematic because our application is focussed on low-end digital cameras. Currently such cameras implement video streams using MJPEG – that is, they combine a sequence of JPEG images to form the frames in a video sequence. This makes sense as these cameras will typically have JPEG compression implemented in the chipset hardware. Unfortunately MJPEG video streams are not very robust to data loss and the application will need to drop whole frames from the video, even with relatively low packet loss rates. The alternative, even if the JPEG decompression algorithms are adapted to compensate for missing data, is to have a significant number of frames with "blocking" or other error artifacts due to loss of JPEG data.

This leaves us to consider the TCP reliable transport protocol as our most likely choice for data transfer over the WLAN link. Now, although TCP is a reliable transfer protocol it does not provide any guarantees on the data transfer rates, has no support for priority mechanisms and is affected by the 2% *forward error rate* (FER) of WLAN. Fortunately even with low-end cameras we can expect to have a good-sized RAM buffer available at the client end. Typically at least 8MB–16MB is required by such a camera in order to process raw still images during the capture and acquisition process. This is generally enough to capture 30-

60 seconds of QVGA video at a nominal frame rate of 15 fps. By taking advantage of this RAM buffer to store video data until a TCP link is ready to transmit data we can avoid most of the complexities of trying to implement a *hard* real-time transfer of data over the TCP/802.11 link.

A useful discussion of TCP performance issues over WLANs is given in [8] and a useful comparison of several TCP auto-tuning techniques is given in [10].

C. TCP Algorithms and Congestion Control

The key problem which affects TCP connections over a WLAN is the effect that the FER of the wireless link has on TCP. In brief, packet loss over the wireless PHY is interpretted by TCP as network congestion. A useful description is given in [8]. To better understand this problem we will now give a brief overview of the best known TCP congestion control algorithms and review the current state of TCP implementions available in the field.

One of the major recent development efforts on Internet technology has been in the review and refinement of the existing TCP algorithms to incorporate greatly enhanced congestion control capabilities. Useful overviews are given in refs [11] and [12]. The original TCP algorithms in common usage were known as Tahoe and Reno. The basic strategy of both algorithms is to gently probe the network for spare capacity by linearly increasing the data transfer rate and exponentially decreasing it when congestion is detected. Congestion is interpreted as being the loss of a data packet. TCP Reno implements fast recovery to recover more efficiently from packet losses and was the predominant form of TCP until recently.

In 1995 a new algorithm known as TCP-Vegas was proposed [42]. This improves on Reno allowing more timely detection of packet loss and correcting the oscillatory behaviour of the Reno algorithm. The Vegas algorithm uses packet queuing delay to estimate network congestion in contrast to Reno which uses packet loss. More recently a number of enhancements were proposed to improve retransmission and recovery times for the Reno algorithm [16,17] which led to the proposal for NewReno [18]. An alternative scheme based on selective use of acknowledgement packets to determine packet loss rates was proposed in 1996 [15] and has subsequently become known as SACK. Finally, the most recent addition to the stable of TCP algorithms is TCP-Westwood [14] which purports to improve significantly on Reno and SACK, particularly for mixed wired/wireless network topologies.

A useful review and comparison of these, and other TCP modifications is given in [13]. A key point is that although the Vegas algorithm has optimal throughput when compared with other algorithms it interoperates poorly with Reno which is still the most widely used algorithm in the wild. In a practical network Reno-TCP connections will steal bandwidth from Vegas-TCP connections even though the latter is more efficient. We may also draw the conclusion from [13] that the actually throughput differences between each of these algorithms is of the order of ±5% under normal network conditions.

D. TCP and the 802.11 MAC

Given that TCP is likely to be the best choice for implementing our wireless video service we next consider how problems can arise between TCP and the 802.11 MAC layer and what solutions exist to solve, or at least minimize such problems. The key problem for most TCP algorithms was already discussed in *section II-C* of this paper. We will now discuss a range of related issues and modifications to TCP to enhance its performance over wireless 802.11 links.

The basic problem that TCP mis-interprets packet loss as network congestions has led to a proliferation of research in this area [25-41]. There are two principle approaches here: some authors focus on refining the TCP mechanisms for congestion control [25-34] whereas others focus on modification to the 802.11 MAC layer to provide better support for QoS or even real-time traffic transport [35 – 41]. What we can say about all of these modifications to TCP and the 802.11 MAC is that although they all improve or enhance the performance of the TCP/MAC interface there is no clear winner. Further, most of these solutions require modifications at either the client or the server end of a TCP connection, or require non-standard 802.11 MAC implementations. From the standards viewpoint the new 802.11g WLAN specifications will address some of the issues with the current MAC layer and will incorporate service differentation and QoS support. On the TCP side, the protocol continues to be refined and improved through the mechanism of IETF RFC documents. Thus we concluded that there are not sufficient advantages to any of these techniques to warrant their adoption in a production system.

In addition to this basic problem, another more subtle problem exists in muti-hop 802.11 networks [19,20,21,22,23]. In essence, an aggressive TCP sender can cause increased contention at the MAC layer which results in a significant increase in link failures. A solution is to reduce the maximum size of the TCP congestion window [19,22]. As this is a straightforward modification to the standard TCP set-up it should be considered and adopted where appropriate. Note that the fundamental cause of this instability is that the 802.11 MAC is not designed to support complex multi-hop network topologies. This, in turn, suggests that when designing a network topology for a theme part application it would be sensible to inter-link the wireless access points using a wired network infrastructure and only use a wireless link between network clients (cameras) and access points.

III Interim Conclusions and Feasibility Tests

In the preliminary presentation of this research [43] we identified a number of possible solutions for video data transfer to WLAN storage. As our research and understanding of the 802.11 MAC has progressed we found various practical problems in implementing these solutions. These are described briefly in the remainder of this section.

A. QoS Services

Although QoS or differentiated services support for TCP/IP over WLAN would be a very useful and effective means of simplifying the implementation of services such as our proposed network storage application, it is quite a complex task to implement this support. More importantly it requires fundamental modifications to the underlying 802.11 MAC DCF [37, 38, 41] or relies on the unimplemented PCF polling mechanism [40].

A soft QoS architecture based on end-to-end network feedback measurements is proposed in [40] and offers a MAC-independent approach to QoS and real-time services. However this research work is preliminary and is not suitable for practical implementations. In any case we expect that the on-going stardards activity of various 802.11 working groups will yield proper QoS support in the near future. Until that time any QoS solutions will require proprietary modifications to the underlying network protocols and do not offer a suitable production-ready solution.

B. Implementing PCF Functionality

Although no commercial access points implement the PCF functionality it was proposed that it might be possible to implement this by modifying the Linux drivers for some of the more current 802.11 chipsets. However our review of existing research in this field suggests that the centralized PCF protocol does not perform well in practical networks [44]. As a consequence it was decided to abandon further investigation into a means to implement the PCF functionality.

C. Real-Time UDP and Robust JPEG

This approach was already commented on in *section II-C*. It is true that UDP has the potential to offer an additional 15% bandwidth when compared with TCP based solutions [1,2,7]. However using UDP over a WLAN implies that we will have of the order of 1.5%-2.0% errors, even on a direct point-to-point link due to the forward error rate of the WLAN MAC [1,2]. Now because we are using MJPEG video this results in a significant number of frames with "blocking" or other error artifacts due to loss of JPEG data. Unfortunately the JPEG compression algorithms were not designed to handle loss of binary data in a robust manner and, even when the UDP packets stream is sequentially numbered and the packet size is kept small so that a missing data packet does not completely corrupt the JPEG image it is very difficult to achieve consistent frame-to-frame image quality.

Clearly if an MPEG4 video stream were used these problems with frame-to-frame image quality would be much less severe, but we must wait for the next generation of digital cameras before MPEG4 becomes the norm for in-camera video. Again, for a production-ready solution we cannot recommend the use of UDP based solutions.

D. Real-Time Linux

It was also proposed that using real-time Linux on the network server might help with improving the granularity of network timings and the management of multiple TCP/IP streams. In fact, as our research and current practical experience has taught us, the TCP/IP protocol implementation in the most recent versions of the Linux kernel – 2.4.20 and above - are already highly optimized and the TCP implementation is very finely tuned.

As the real-time Linux implementations which are available lag the main kernel development and as our initial experiments with the latest TCP/IP implementations suggest that the network performance is already close to optimal it did not make sense to further pursue this line of investigation.

E. Standard TCP/IP Implementations

We initially did some preliminary trials with a number of different TCP/IP implementations. These included FreeBSD, NetBSD and several varients of Linux. We finally settled on the latest Linux TCP because it is easier to configure and adjust dynamically than other TCP implementations. Also, the default configuration gives very close to optimal results.

Linux TCP is based on the NewReno algorithm [18] and in its default configuration also implements or provides support for selective acknowlege (SACK) [15], duplicate-SACK (DSACK) [RFC 2883], explicit congestions notification (ECN), forward-acknowledge (FACK), packet timestamps, window scaling, and protection against wrapped sequence numbers (PAWS) [RFC 1323]. Although the results described in [10] would suggest that the Linux TCP implementation is optimized for a large number of small connections we did not notice any significant difficulties or complications in running 3 relatively high-bandwidth TCP connections concurrently. It may be that the auto-tuning algorithm in Linux 2.4 has undergone revision since the work described in [10]. Alternatively, the client-side RAM buffer for video frames does allow us additional buffer-space at the application level and may help in overcoming some of the deficiencies of the Linux auto-tuning TCP algorithm.

IV System Architecture

Now the key idea of this paper is that rather than writing the A/V data stream from RAM to a memory card on the digital camera, we will instead write it to a wireless network connection which collects and re-assembles the data on a remote server. In this section the main software components of our system architecture are described.

As we do not have access to a prototype 802.11 digital camera the client application needs to closely mimic the higher-level functionality of such a camera. It is also useful to build the prototype client application with a view to later porting it onto an embedded system. Given that we chose to use the Linux TCP stack as a fundamental building block for our experiments it made sense to chose a scripting language with good embedded portability for application development. This led to the decision to use the Python scripting language.

It is based on a foundation of C libraries and once a working application has been prototyped it is relatively easy to convert the entire application into portable ANSI-C code.

The benefit of choosing such a scripting language is that we can also use it to prototype the server-side application. Our test system implements a relatively simple server program which creates a TCP socket for each video client. This is described in detail in *section IV-B* below.

A. Client Side Software

This is designed to run on any standard Linux platform which has an 802.11b PC-card network connection enabled. The structure of the application is given in Fig. 3 below.

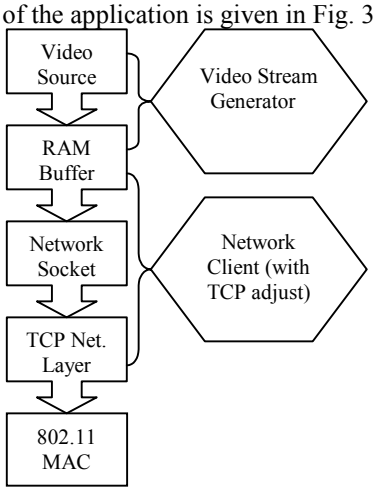


Fig. 3: Organization of the Client application emulating the functionality of 802.11b Video Camera.

The client application is divided into two principle components: (i) a video stream generator and (ii) a network client that grabs video frames from a memory buffer and transmits them over the WLAN link to the server application. This eliminates the need for local bulk storage other than the RAM buffer.

As a typical camera requires 16+MB of RAM storage in order to facilitate processing of large images we have provided a similar sized buffer in our client application. This allows about 45 secs of video to be buffered at 20 fps, *i.e.* the size restriction on the buffer is 900 frames.

B. Server Side Software

The server side software is slightly more complicated in that it must be able to handle multiple client connections. Fortunately we do not expect to have more than 3 clients at any one time as the bandwidth of 802.11b will not support more than 3 concurrent video streams at a frame rate of 15-20 fps. This means that we do not have to worry too much about scaling the server-side software for this class of application although the TCP sockets on the server side must be implemented using multi-threading so that one client connection does not "block" another.

The structure of the server-side application is given in ***Fig. 4*** below. Note that the *Connection Management Layer* is part of the application code which is responsible for setting up active TCP sockets to listen for video data from a particular client. In a full working implementation each digital camera would have a unique registration ID and would have to have its connection information entered at the server-end either manually, or though some form of network *plug and play* mechanism.

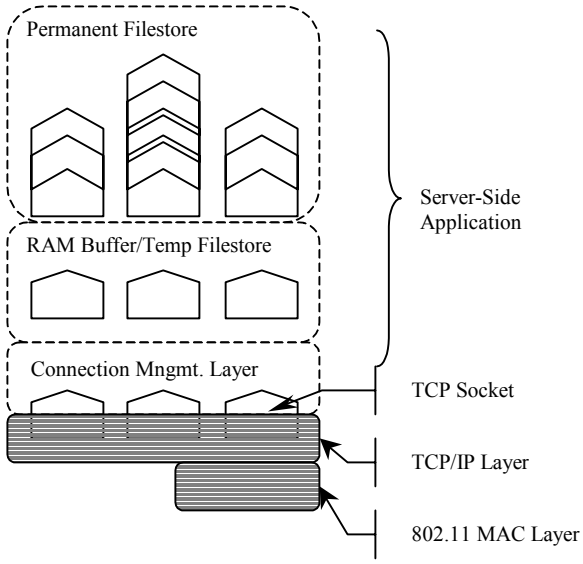


Fig. 4: Architecture of the Server-Side Application.

In our prototype application the server automatically creates a socket for each client on start-up. It then listens for connections from the relevant client and, when a client connection becomes active the incoming data stream is initially stored in a temporary file. After a connection has been correctly terminated this video stream will be written to permanent file storage and tagged with relevant metadata. In the event the connection is incorrectly terminated – either due to client-end failure, or network data loss and/or congestion – then the application performs an integrity check on the termporary file and, if possible, repairs the file. This recovered file is then tagged and written to permanent file storage.

This application can be run on any desktop computer but is also suitable for porting to a dedicated embedded system.

V EXPERIMENTAL SET-UP AND RESULTS

In this section we shall describe the experimental set-up used to run the tests of our video-WLAN application. Sample test results for 2 and 3 concurrent client video streams at a number of different frame rates are given. In each case we estimate the maximum frame rate which can be achieved before the client-side RAM buffer fills to overflow.

These tests were performed in a controlled environment with good reception. The main server application is run on a conventional desktop PC which is connected over a fast Ethernet link to a wireless access point – the main 802.11b WLAN network. In addition there are 3-4 client computers available at different locations in the lab, each with a 802.11b WLAN PC-card. Typically 3 of the clients will used to generate the client-side video streams. These are normally laptop-style computers which facilitates moving them to different locations in the lab should the need arise.

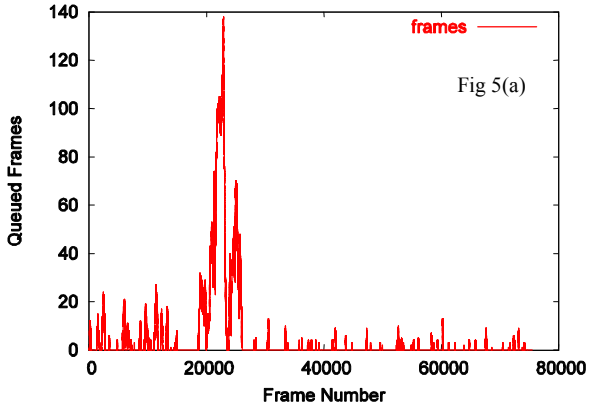
The other WLAN client is used to generate random network activity – browsing of Web pages, SSH or Telnet sessions and an occasional FTP transfer. The intention is to test to what extent random network activity may affect the video-WLAN application when the wireless bandwidth is close to saturation.

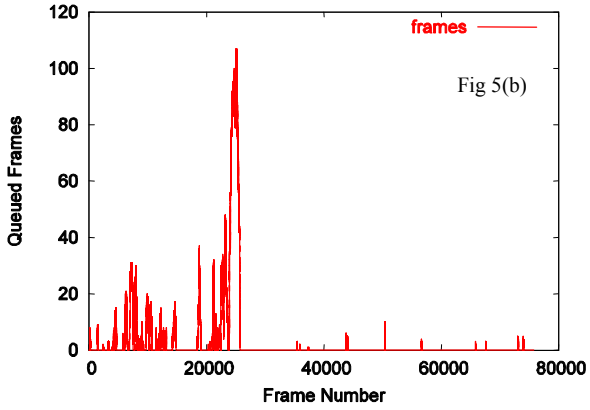
The video source was normally a data file read from the hard disk of each client computer. The test application could easily be adapted to handle any standard video source but in the interests of repeatability we chose to use the same video file throughout our tests. The raw video stream rate depends on the actual number of frames of video transmitted per second and is given as 80 kbps times the frame rate. For a 20 fps video stream, for example, we have a raw video bit rate of 1.6 Mbps.

A. Test Results for 3 Clients at 20 fps

A quick mental calculation will tell the reader that an 11 Mbps WLAN connection, with a maximum data rate of 5.5 Mbps can realistically only support 3 concurrent video streams. The questions which need to be answered concern the maximum number of frames per second that can be sustained over a reasonable time-frame and the error response and recovery times of TCP to lost packets.

Our initial tests were followed by a series of more detailed studies. Over a period of several weeks we experimented with transfer periods running up to 6 hours. We also varied the number of clients and the frame rates for different client groupings. The results have been reliable and repeatable. Our first sample sets of test data are presented in ***Fig 5*** for a group of 3 concurrent clients running at 20 fps.





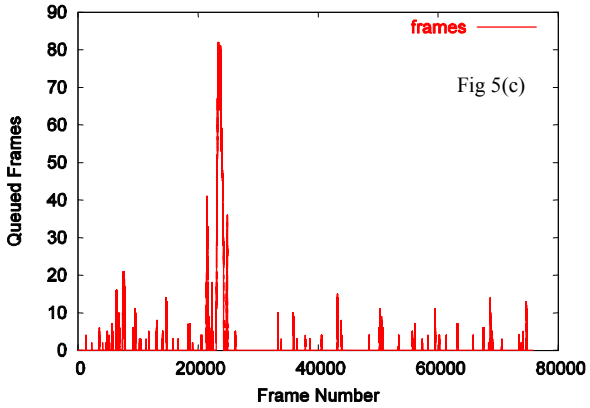


Fig. 5: Size of the Client-Side Video Frame Buffer for 3 Client Devices running at a Frame Rate of 20 fps: (a) Device 1, (b) Device 2 and (c) Device 3.

Note that at 20 fps none of the three client devices manages to queue more than 140 frames of video in its local RAM buffer. This behaviour is typical for 3 devices running at this frame rate. These example tests were run over 80,000 frames of video data, which translates to more than one hour of continuous operation.

We note that during the initial 15 minutes of testing that the local RAM queues of each device occasionally reached sizes of the order of 20-25 frames. During this period random browsing of Web pages was in progress on the 4th network client. At the end of this initial 15 minute period, corresponding to 0-20,000 frames of video data, we note that the local RAM queue of each of the client devices peaked at values ranging between 80 and 140 frames of video. This peaking was in response to a moderately sized FTP session being initiated on the 4th client. We note that TCP response and recovery were quite prompt for each client. Naturally, if multiple concurrent FTP sessions are initiated the queue sizes will degrade accordingly.

After the FTP session concluded we note that the remaining 40 minutes of this test cycle were very uneventful with none of the three client devices queuing more than 10 frames of video in the local RAM buffer.

B. Frame Rate Limitations for 3 Clients

The next step was to determine if we could achieve sustained transfer rates higher than 20 fps for 3 concurrent clients. Frame rates of 23 fps or higher are inherently unstable and lead to some or all of the clients saturating their RAM buffers.

The results for a frame rate of 22 fps are more repeatable and, although they can occasionally saturate – particularly in response to random network activity from the 4th node – they are normally quite stable over relatively long periods of time. Some example results are shown in ***Fig 6*** below.

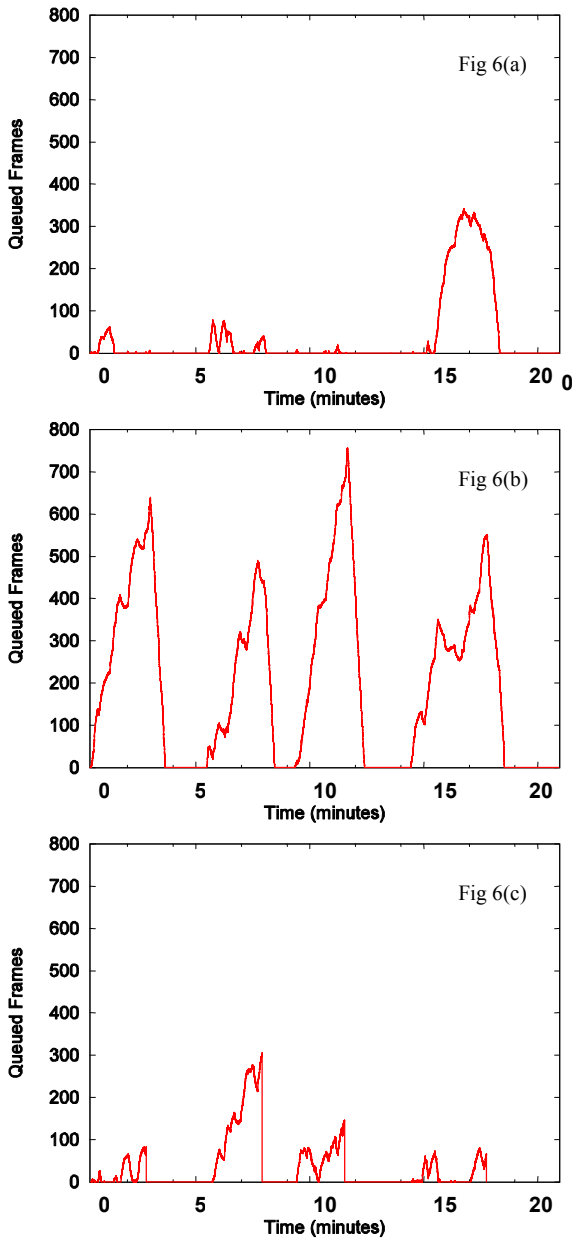


Fig. 6: Size of the Client-Side Video Frame Buffer for 3 Client Devices running at a Frame Rate of 22 fps: (a) Device 1, (b) Device 2 and (c) Device 3.

Two of the client devices manage quite well with frame buffer sizes staying below 300 frames of video – about 35% of total buffer capacity. However the third device becomes stuck in a periodic TCP recovery loop, reaching buffer sizes up to 770 frames of video. The periodic nature of the behaviour of this device is consistent with some of the issues identified with using TCP over WLAN links as discussed in *section II-D* of this paper.

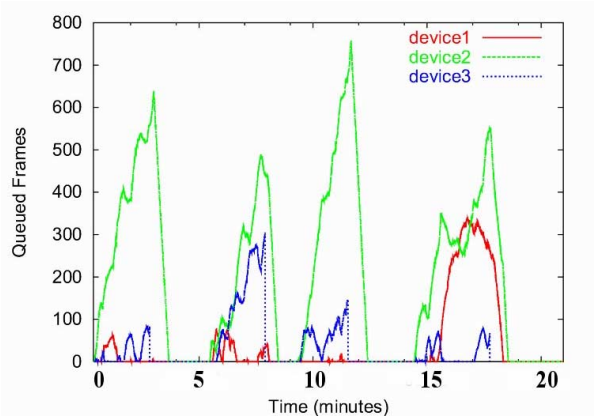


Fig. 7: Frame Buffers Data from Figs 6(a), (b) and (c) are superimposed in Fig 7 above.

In ***Fig 7*** we show the three frame buffer traces of ***Fig 6(a)-(c)*** superimposed on each other. This data set was generated without any network activity from the 4th node. Even so we note that one of the nodes – device 2 in this instance – behaves poorly. After initiating the connection of all three devices it enters, almost immediately into a TCP recovery cycle with its RAM buffer filling up to 75% of capacity in the first 3 minutes of operation.

Note that when the poorly performing device enters a TCP recovery phase it tends to force one of the other devices into a data build-up cycle in its RAM buffer. The reason for this is that, when in a steady-state transfer mode, TCP only requires an additional network overhead of 200 kbps on a 1.6 Mbps video stream, but when it enters recovery the total bandwidth requirements can almost double from 1.8 Mbps up to 3.0 Mbps during the initial recovery phase. Bandwidth usage will remain high – 2.0-2.5 Mbps - on this link until a full recovery is effected. This restricts the available bandwidth for the two remaining devices and at least one of them will be unable to fully satisfy the required data transfer rate for several minutes. Normally, if there is not a significant amount of additional network traffic, the behaviour will be stable and the three devices will clear their RAM buffers in a matter of minutes.

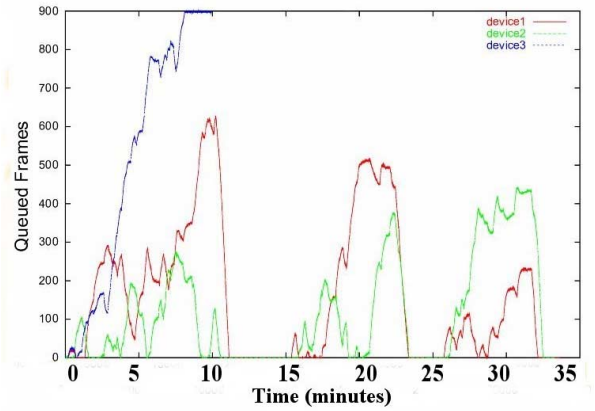


Fig. 8: Frame Buffers Data for 3 clients at 22 fps with concurrent Web browsing activity on the network.

Occasionally, however, a device may not recover. In ***Fig 8*** we show a case where the poorly performing device saturates its RAM buffer after about 9 minutes of operation. (Note that the poor performance of the device illustrated in ***Figs 7 & 8*** is mostly due to an early version of the WLAN firmware, but it serves to illustrate the unstable nature of the system performance at the 22 fps video rate.)

VI CONCLUSIONS

In this paper we studied the problem of transmitting multiple concurrent video streams from a client device to a server-side application implementing a prototype storage service for video clips. The goal of this research was to determine if it is practical to provide end-users with reliable wireless cameras which can take advantage of network storage for video clips.

Our initial tests suggest that for low-usage networks, such as a home WLAN, the solution can be practically implemented and provides reliable transfer of video data at 20 fps for up to 3 clients when the full 11 Mbps data rate of the WLAN is available. At lower data rates the video frame rate, or the number of connections must be reduced accordingly.

Furthermore, this can be achieved using standard TCP reliable transport protocol and, if the latest refinements to TCP are implemented then there is a relatively low loss of network bandwidth over the WLAN, provided the WLAN link is confined to a single node. Applications that require broader wireless coverage or extensive roaming capabilities lie outside the scope of the current work. However we remark that many consumer applications will not require such capabilities and for home networks or in-car WLAN systems the technology has great potential to add value to low-end consumer cameras and related appliances.

REFERENCES

[1] Net throughput with IEEE 802.11 wireless LANs, *Kamerman, A.; Aben, G.,* Wireless Communications and Networking Conference, 2000. WCNC. 2000 IEEE , Volume: 2 , 2000, Page(s): 747-752 vol.2

[2] Throughput performance of wireless LANs operating at 2.4 and 5 GHz , *Kamerman, A.; Aben, G.,* Personal, Indoor and Mobile Radio Communications, 2000. PIMRC 2000. The 11th IEEE International Symposium on , Volume: 1 , 2000 , Page(s): 190-195 vol.1

[3] IEEE 802.11 wireless LAN: capacity analysis and protocol enhancement , *Cali, F.; Conti, M.; Gregori, E.,* INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 1 , 29 Mar-2 Apr 1998 , Page(s): 142 -149 vol.1

[4] Unfairness and capture behaviour in 802.11 adhoc networks, *Ware, C.; Judge, J.; Chicharo, J.; Dutkiewicz, E.,* Communications, 2000. ICC 2000. 2000 IEEE International Conference on , Volume: 1, 2000, Page(s): 159-163 vol.1

[5] Introducing service differentiation into IEEE 802.11, *Aad, I.; Castelluccia, C.,* Computers and Communications, 2000. Proceedings. ISCC 2000. Fifth IEEE Symposium on , 2000, Page(s): 438 -443

[6] Differentiation mechanisms for IEEE 802.11, *Aad, I.; Castelluccia, C.,* INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 1 , 2001, Page(s): 209-218 vol.1

[7] Behavior of UDP-based applications over IEEE 802.11 wireless networks, *Arranz, M.G.; Aguero, R.; Munoz, L.; Mahonen, P.,* Personal, Indoor and Mobile Radio Communications, 2001 12th IEEE International Symposium on , Volume: 2 , Sep/Oct 2001, Page(s): F-72 -F-77 vol.2

[8] TCP Performance Issues over Wireless Links, *Xylomenos, G., and Polyzos, G. C.,* IEEE Communications Magazine, April 2001, Page(s): 51-58.

[9] Optimizing Internet flows over IEEE 802.11b wireless local area networks: a performance-enhancing proxy based on forward error correction, *Munoz, L.; Garcia, M.; Choque, J.; Aguero, R.; Mahonen, P.,* IEEE Communications Magazine , Volume: 39 Issue: 12 , Dec 2001, Page(s): 60-67

[10] A comparison of TCP automatic tuning techniques for distributed computing, *Weigle, E.; Wu-chun Feng,* High Performance Distributed Computing, 2002. HPDC-11 2002. Proceedings. 11th IEEE International Symposium on , 2002, Page(s): 265 -272

[11] A report on recent developments in TCP congestion control, *Floyd, S.,* IEEE Communications Magazine , Volume: 39 Issue: 4 , Apr 2001, Page(s): 84 -90

[12] Internet congestion control, *Low, S.H.; Paganini, F.; Doyle, J.C.,* IEEE Control Systems Magazine , Volume: 22 Issue: 1 , Feb 2002, Page(s): 28 -43

[13] TCP congestion control algorithms and a performance comparison, *Yuan-Cheng Lai; Chang-Li Yao,* Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on , 2001, Page(s): 523 -526

[14] TCP Westwood: congestion window control using bandwidth estimation , *Gerla, M.; Sanadidi, M.Y.; Ren Wang; Zanella, A.; Casetti, C.; Mascolo, S.,* Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE , Volume: 3 , 2001 , Page(s): 1698 - 1702 vol.3

[15] RFC 2018: TCP Selective Acknowledgement Options, *Mathis, M., Mahdavi, J., Floyd, S. and Romanow, A.,* Oct 1996.

[16] RFC 2001: TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery Algorithm, *Stevens, W.,* Jan 1997.

[17] RFC 2581: TCP Congestion Control, *Stevens, W. and Paxson, V.,* April 1999.

[18] RFC 2582: The NewReno Modification to TCP's Fast Recovery Algorithm, *Floyd, S. and Henderson, T.,* April 1999.

[19] Interactions between TCP and the IEEE 802.11 MAC protocol, *Rui Jiang; Gupta, V.; Ravishankar, C.V.,* DARPA Information Survivability Conference and Exposition, 2003. Proceedings , Volume: 1 , 2003, Page(s): 273 -282

[20] Revealing TCP unfairness behavior in 802.11 based wireless multi-hop networks, *Shugong Xu; Saadawi, T.,* Personal, Indoor and Mobile Radio Communications, 2001 12th IEEE International Symposium on , Volume: 2 , Sep/Oct 2001, Page(s): E-83 -E-87 vol.2

[21] Revealing TCP incompatibility problem in 802.11-based wireless multi-hop networks, *Shugong Xu; Saadawi, T.,* Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE , Volume: 5 , 2001, Page(s): 2847 -2851 vol.5

[22] Revealing and solving the TCP instability problem in 802.11 based multi-hop mobile ad hoc networks, *Xu, S.; Saadawi, T.,* Vehicular Technology Conference, 2001. VTC 2001 Fall. IEEE VTS 54th , Volume: 1 , 2001, Page(s): 257 -261 vol.1

[23] On TCP over wireless multi-hop networks, *Shugong Xu; Tarek Saadawi; Myung Lee,* Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE , Volume: 1 , 2001, Page(s): 282 -288 vol.1

[24] Simulation analysis of TCP performance on IEEE 802.11 wireless LAN, *Yong Peng; Haitao Wu; Keping Long; Shiduan Cheng,* Info-tech and Info-net, 2001. Proceedings. ICII 2001 - Beijing. 2001 International Conferences on , Volume: 2 , 2001, Page(s): 520 -525 vol.2

[25] Novel TCP congestion control scheme and its performance evaluation, *Xu, W.; Qureshi, A.G.; Sarkies, K.W.,* Communications, IEE Proceedings- , Volume: 149 Issue: 4 , Aug 2002, Page(s): 217 -222

[26] An effective way to improve TCP performance in wireless/mobile networks, *Fei Peng; Shiduan Cheng; Jian Ma,* EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security. IEEE/AFCEA , 2000, Page(s): 250 -255

[27] Robust TCP congestion recovery, *Haining Wang; Shin, K.G.,* Distributed Computing Systems, 2001. 21st International Conference on. , Apr 2001, Page(s): 199 -206

[28] Performance of TCP congestion control with explicit rate feedback: rate adaptive TCP (RATCP), *Kamik, A.; Kumar, A.,* Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE , Volume: 1 , 2000, Page(s): 571 -576 vol.1

[29] Additive increase adaptive decrease congestion control: a mathematical model and its experimental validation, *Grieco, L.A.; Mascolo, S.; Ferorelli, R.,* Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on, 2002, Page(s): 849 -854

[30] Additive increase early adaptive decrease mechanism for tcp congestion control, *Mascolo, S.; Grieco, L.A.,* Telecommunications, 2003. ICT 2003. 10th International Conference on , Volume: 1 , 2003, Page(s): 818 -825

[31] TCP with bandwidth estimation over wireless networks, *Martignon, F.; Capone, A.,* Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th , Volume: 3 , 2002, Page(s): 1422 - 1426 vol.3

[32] TCP-BuS: improving TCP performance in wireless ad hoc networks , *Dongkyun Kim; Toh, C.-K.; Yanghee Choi,* Communications, 2000. ICC 2000. 2000 IEEE International Conference on , Volume: 3 , 2000 , Page(s): 1707 -1713 vol.3

[33] TCP HACK: TCP header checksum option to improve performance over lossy links , *Balan, R.K.; Lee, B.P.; Kumar, K.R.R.; Jacob, L.; Seah, W.K.G.; Ananda, A.L.,* INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 1 , 2001 , Page(s): 309 -318 vol.1

[34] Improvements achieved by SACK employing TCP Veno equilibrium-oriented mechanism over lossy networks , *Chung Ling Chi; Fu Chengpeng; Liew Soung Chang,* EUROCON'2001, Trends in Communications, International Conference on. , Volume: 1 , 2001 , Page(s): 202 -209 vol.1

[35] Performance of reliable transport protocol over IEEE 802.11 wireless LAN: analysis and enhancement , *Haitao Wu; Yong Peng; Keping Long; Shiduan Cheng; Jian Ma,* INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 2 , 2002 , Page(s): 599 -607 vol.2

[36] Achieving efficient channel utilization and weighted fairness for data communications in IEEE 802.11 WLAN under the DCF , *Daji Qiao; Shin, K.G.,* Quality of Service, 2002. Tenth IEEE International Workshop on , 2002 , Page(s): 227 -236

[37] An adaptive retransmission scheme with QoS support for the IEEE 802.11 MAC enhancement , *Wen-Tsuen Chen; Bo-Bin Jian; Shou-Chih Lo,* Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th , Volume: 1 , 2002 , Page(s): 70 -74 vol.1

[38] Quality of service guarantee on 802.11 networks , *Srikant Sharma; Kartik Gopalan; Ningning Zhu; Pradipta De; Gang Peng; Tzi-cker Chiueh,* Hot Interconnects 9, 2001. , 2001 , Page(s): 99 -103

[39] End-to-end versus explicit feedback measurement in 802.11 networks, *Kazantzidis, M.; Gerla, M.,* Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on , 2002 , Page(s): 429 -434

[40] An error control scheme based on adaptation of polling schedule for real-time communication on IEEE 802.11 wireless LANs , *Junghoon Lee; Changik Kang; Mikyung Kang,* Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on , Volume: 3 , 2002 , Page(s): 1058 -1062

[41] Modified distributed coordination function for real-time traffic in IEEE 802.11 wireless LAN , *An-Tai Lin; Shie-Jue Lee,* Advanced Information Networking and Applications, 2003. AINA 2003. 17th International Conference on , 2003 , Page(s): 794 -797

[42] TCP Vegas: End to End Congestion Avoidance on a Global Internet, *Brakmo, L. S. and Peterson, L. L.,* IEEE J. Select Areas Commun, vol. 13, Oct. 1995 , Page(s): 1465 -1480

[43] Streamed Wireless Video for Low-End Consumer Digital Cameras, *Raducan, I. and Corcoran, P. M.,* IEEE International Conference on Consumer Electronics (ICCE) 2003, Conference Digest, June 2003

[44] Time Bounded Services and Mobility Management in IEEE 802.11 wireless LANs, *Kuo, W. K., Chan, C. Y., and Chen, K. C.,* Proc. IEEE Personal Wireless Communications Conference, 1997, Page(s) 157-161

AUTHOR BIOGRAPHIES

Peter Corcoran received the BAI (Electronic Engineering) and BA (Math's) degrees from Trinity College Dublin in 1984. He continued his studies at TCD and was awarded a Ph.D. in Electronic Engineering for research work in the field of Dielectric Liquids. In 1986 he was appointed to a lectureship in Electronic Engineering at NUI, Galway. His research interests include microprocessor applications, home networking and wireless networking technologies. He is a member of I.E.E.E.

Ilariu Raducan received his B.Eng. degree in Electronic Engineering from the Faculty of ElectroMechanical Engineering at the University of Craiova, Romania in 1994. He is currently completing an M. Eng. Sc. degree in the Consumer Electronics Research Group at the National University of Ireland, Galway. His research interests include embedded systems design, communication network protocols, and wireless data systems and their applications.