Tavarua: A Mobile Telemedicine System Using WWAN Striping

by

Jennifer Carlisle

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2007
[June 2007]

Author ....
Department of Electrical Engineering and Computer Science
May 11, 2007

Certified by ....
John V. Guttag
Professor, Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Tavarua: A Mobile Telemedicine System Using WWAN Striping

by

Jennifer Carlisle

Submitted to the Department of Electrical Engineering and Computer Science
on May 11, 2007, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Tavarua is a platform designed to support mobile telemedicine systems over wireless wide area networks, WWANs. It utilizes network striping and other complementary techniques to send uni-directional near real time video and audio data streams from a mobile node to a stationary location.

The key technical challenge is transmitting high-bandwidth, loss-sensitive data over multiple low-bandwidth, lossy channels. We overcome these challenges using dynamic adjustment of the encoding parameters and a novel video encoding technique (grid encoding) that minimizes the impact of packet losses.

Using five WWAN interfaces, our system reliably and consistently transmits audio and diagnostic quality video, with median PSNR values that range from $33.716dB$ to $36.670dB$, with near real-time latencies.

We present a study of the characteristic behavior of WWANs, and a description of our system architecture based in part on the lessons gleaned from that study. Through a set of experiments where we transmit video and audio data from a moving vehicle we evaluate the system, focusing on consistency, reliability, and the quality of the audio and video streams. These experiments demonstrate that we can transmit high quality video and audio in varying conditions and even in the presence of hardware failures.

Thesis Supervisor: John V. Guttag
Title: Professor, Electrical Engineering and Computer Science

This is dedicated to my family,

who have always been my foundation.

Acknowledgments

I owe a deep gratitude to my advisor, John Guttag. Despite displaying a glaring lack of judgment in being a Yankees fan, John is the best advisor I could have hoped for. He provided encouragement when needed, advice when necessary, and allowed me the freedom to chart my own path. For all these things I am profoundly grateful.

Asfandyar Qureshi served as both a co-advisor and a co-conspirator in this process. His endless reserve of patience for my networking and Linux questions as well as his terrific sense of humor during our seemingly endless hours of driving experiments made this journey a treasured experience. We will always have downtown Orlando. I would also like to thank Vladimir Bychkovsky and Jakob Eriksson for their help in wrangling the Soekris boxes and demystifying Debian scripts.

I am indebted to Barbara Liskov, who provided me support and mentorship. Dina Katabi, for all our late night conversations. Karen Sachs, who was always there for support and advice. A great appreciation to my band of brothers that I've had the privilege to know these past two years. Hariharan Rahul, my gym-buddy and fellow ice cream addict. Daniel Abadi, who always made sure I wasn't too busy to go play softball. Nate Kushman, and our always random and entertaining conversations. Daniel Myers, for turning me into a bike junkie and helping me survive Advanced Algorithms. James Cowling, for being the Dre to my Snoop. Finally I would like to thank the staff and students of SEED Academy, who provided me with an experience unlike any other.

Above all I thank my family, who are my greatest blessing.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Tavarua is a platform designed to support mobile telemedicine systems that utilize network striping over wireless wide area networks, WWANs, to send uni-directional near real time video and audio data streams from a mobile node, e.g. an ambulance, to a stationary location, e.g. a hospital.

WWANs are not ideally suited for our application. However, since our application must be mobile and cover a large geographical area, we are left with no other current option. To deploy our own infrastructure would be economically infeasible, even between a few cities, not to mention restrictive, as it would preclude us from taking advantage of the rapid advances in commercial networks. The other option is to use the 802.11 network. While this network has a low loss rate and low latency, it is also under provisioned for the areas that we need most, the highways, bridges and tunnels that ambulances travel to transport patients. Additionally these networks are short range and the connection time as the ambulance travels between two different networks is long, on average about 1-2 seconds. This would cause frequent interruptions in our data streams, which would at best be frustrating to the user, and in most cases preclude its use as a diagnostic tool.

WWANs are our best option, however these networks have drawbacks that need to be overcome in order to use them effectively. The drawbacks are that the channels have low throughput, a high loss rate, high packet round-trip-times (RTTs), and are highly variable. To combat the issue of low throughput we need to connect to many WWAN interfaces simultaneously. To accomplish this we developed Tribe, a protocol that emulates each remote interface as a local interface, allowing us to be simultaneously connected to multiple interfaces. In conjunction with that component we leverage Horde, a previously developed network striping middleware that allows us to easily and efficiently stripe data over dissimilar WWAN channels. This virtual channel provided by Horde and Tribe can provide adequate throughput since the only limit to the number of simultaneously transmitting channels is what the network can sustain. If more bandwidth is required, one can simply add more interfaces until the network is saturated.

To mitigate the effects of the high loss rate, we use a grid video encoding technique that segments each frame into several smaller subframes. Each of these subframes are encoded and decoded independently, so losses or corruptions in one subframe are localized and do not effect the rest of the frame. Because of the high packet RTTs, we are unable to mask losses with retransmissions without sacrificing latency. This makes the use of localizing losses even more vital, without grid encoding a single packet loss would corrupt the entire frame. In situations where near real-time interaction is not required, we allow the user to switch to a higher latency mode that does utilize retransmissions.

While different variations of mobile telemedicine systems have been previously explored, we know of no other system that can provide diagnostic quality audio and video with a near real-time latency. At the University of Maryland, they have developed a system that wirelessly transmits segments of recorded video to help detect ischemic stroke in-route to the receiving hospital[37]. This system not only operates under considerable latency, first the segment is recorded, then transmitted, but the received video is only eight frames per second.

Another project at the University of Massachusetts, Amherst also uses the idea of sending data over a WWAN channel[34]. However, they only use one WWAN interface and unoptimized congestion control. Because of this, they cannot send all data streams simultaneously and the video quality is not good enough to be considered a diagnostic tool. Additionally, as the video approaches three frames per second, the average delay exceeds three seconds.

With a near real-time latency, Tavarua can transmit high quality audio and video, even at frame rates as high as 25 frames per second. WWANs are highly variable and can fluctuate dramatically. To absorb these fluctuations in available bandwidth we dynamically update the quantization, or Q, parameter that controls the quality at which the frame is encoded, thus controlling the sending rate for our data streams for a fixed frame rate. Through matching our data transmission rate to the value of the last available bandwidth we are able encode video at the highest Q value while not overburdening the network, which would cause losses. It is through our novel channel aggregation component and network striping techniques, as well as our innovative video encoding techniques that we are able to transmit diagnostic quality audio and video from a mobile unit to a stationary location.

1.1 Motivating Application

Over the last decade, the increasing availability of data communications has led to a dramatically increased interest in new applications of telemedicine. We will discuss two such potential applications, trauma care and the transport of critically ill neonates, and the requirements derived from assessing the needs of the physicians and the emergency medical transport team.

1.1.1 Trauma Care

The leading cause of death and disability for Americans under the age of 45 is trauma. Despite this, there are no Level 1 Trauma Centers, which are specialized hospitals that are equipped to handle all types of traumas, in most of the counties in the United States. As of 2002, there were only 190 Level 1 trauma centers in the entire United States[65]. Therefore, trauma patients experience long transport times without physician supervision under the care of in-ambulance teams that have limited capabilities. These long transport times lengthen the time to treatment which has been found to be the critical factor in saving life, limb and brain functions.

In the treatment of trauma victims, establishing effective communication between the emergency medical transport team and the in-house trauma team is challenging. The physical distance and pressure of providing emergent care often prevents a detailed and effective information exchange from one team to the other. A system that allows real-time audio and visual linkage of the two teams could lead to improved outcomes for the patient, since therapies could begin while in transit.

This would also improve the time to treatment once the patient arrives at the hospital. Currently, the physicians have little information about the patient prior to their arrival at the hospital. Concomitantly, when this information exchange is to take place, the patient has also been wheeled in front of the physician, moving the physician's focus from the information provided by the medical transport team to the patient directly. Thus, the information culled about the patient in the ambulance is often been ignored and the physician begins the patient evaluation from scratch, wasting precious minutes as work is being repeated. If the physicians could receive patient information prior to their arrival at the hospital, they would not repeat the initial examination and therefore shorten the time to treatment. Additionally, if the patient were to require special facilities, e.g., an operating room, or personnel, e.g., a neurosurgeon, that determination and the arrangements for such facilities and people could be made while the patient was in-transit, decreasing the amount of time a patient would have to wait prior to surgery.

1.1.2 Neonatal Care

As medicine has become increasingly specialized, treatment centers that can handle critically ill patients have become fewer and further between. This dichotomy between specialized treatment centers and local hospitals is evident in the care of neonatal patients. While many babies are still born in local hospitals, these hospitals are often not properly qualified to care for these babies should something go seriously awry. When these emergencies arise, the critically ill neonates must be transported to specialized treatment centers, often located quite a distance from where these neonates were born.

Transporting these critically ill neonates long distances demands a transport team trained to handle the particular needs of neonatal patients. Unfortunately, most medical transport teams are not qualified to assist neonatal patients. These factors contribute to the number of newborn's with developmental problems and to the number of newborn deaths.

As conveyed to us by neonatal specialists, many of the critical procedures that could be done while the patient is in transit are simple; all medical transport teams are qualified to preform them. However, noticing that these procedures need to be done takes a specialist who can read the subtle cues from the neonates, e.g., coloring, vital signs, and activity. Tavarua would bridge such a gap, providing the transport team with the trained eye of a specialist, remotely monitoring from the specialized treatment center that will be receiving the patient.

1.1.3 Requirements

Building an application that meets these needs requires a mobile communications system that simultaneously handles real-time bi-directional video, on the order of $500kbps$ with a minimal amount of latency, and bi-directional, reliable audio. Bi-directional video would allow for a richer communication as the doctor could demonstrate procedures for the medical transport team or tasks for the patient to perform. This system must also be adaptable to sudden changes in bandwidth, moving gracefully between what is acceptable in high bandwidth areas, specifically video encoded at a high quality level and a high frame rate, and what is possible in low bandwidth areas. Additionally, it must also give the physician the ability to select a higher latency video stream so that they can choose to view smooth video when the ambulance is in a location where the network is experiencing an abundance of losses. Through a few simple controls, like adjusting the frame rate, or selecting high or low latency video, the physician can tailor the application to the specific medical needs, even as those needs change over time.

1.2 Contributions

After we determined that the only feasible network to use was the 3G WWANs provided by cellular telephony companies, we needed to know the characteristic behavior of these channels. We could find no large scale study of these networks in the public domain; the only available data was cellular carrier advertising that provided best case throughput numbers and coverage maps that presented binary information regarding whether or not coverage existed in that particular area. While not a rigorous study, this thesis provides the largest known study of these WWANs, spanning several days and multiple cities.

Tavarua provides a scalable infrastructure that can be used to study the potential usability of telemedicine in real world environments. Additional components, e.g., cameras or WWAN interfaces, can be incorporated into the system easily. The original prototype used one camera and three PCMCIA interfaces. To perform some recent experiments we easily added an additional camera and two Rev-A USB WWAN interfaces. Adding the camera was a matter or plugging it in, while the upgraded WWAN interface took an additional twenty minutes to fine tune the congestion control parameters in the middleware. However, this fine tuning of parameters only needs to be done when a new class of card is added to the system. An identical WWAN interface would take minutes to incorporate into the infrastructure. WWAN technology is improving rapidly, with that in mind we designed a system that benefits from these upgrades, e.g. from EV-DO to the current state-of-the-art Rev-A[50, 75], by simply plugging in the latest piece of hardware and not requiring any modifications to our existing code-base.

The main contribution of this thesis is the construction of this system and the novel technologies incorporated to improve deficiencies in performance that were observed through running the system. Through our study of the network we realized that the throughput of WWANs is highly variable. To improve the video quality of our system and to decrease the packet loss rate we utilized feedback from our networking middleware to dynamically update the quality at which we encoded the video to match the available bandwidth of the network. Figure 1-1, discussed in more detail in chapter 5, shows how effective we are in matching the encoding rate to the allowed bandwidth of the network. Maintaining this small gap between the allowed rate and the encoding rate allows us to absorb drops in the network throughput or spikes in the encoded frame size, brought on by high motion scenes, without incurring losses. This attention to minimizing the loss rate is a primary factor in our ability to consistently attain diagnostic quality video.

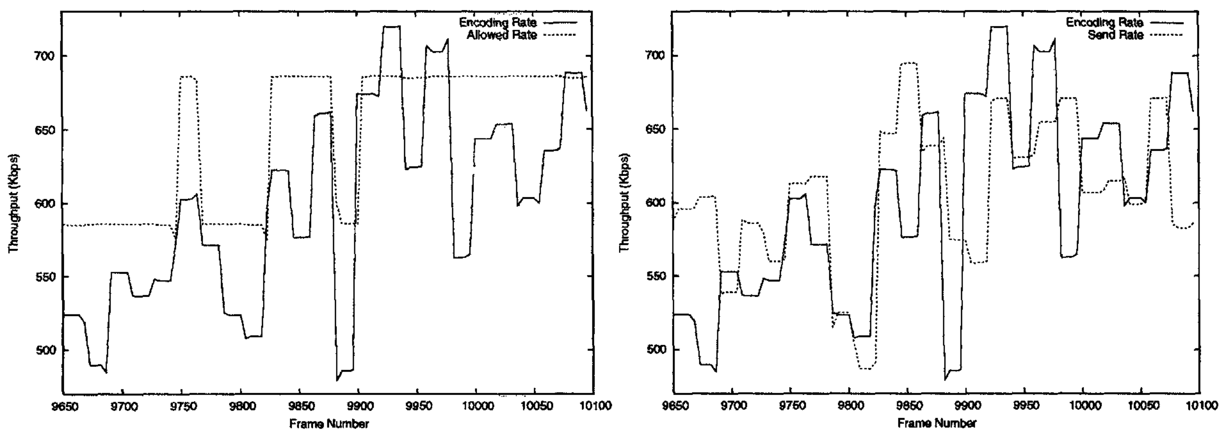


Figure 1-1: The figure on the left compares the bitrate at which the video was encoded to the feedback from the network of how much we were allowed to send. The goal is to consistently keep a buffer between the encoding rate and the allowed rate, which will probe for more bandwidth but not incur losses. Because of buffering, the encoding rate can exceed the allowed rate, as occurs around frame 9925, if only for a short duration. The figure on the right compares the encoded bitrate and the send rate. Prolonged periods where the encoding rate is above the send rate is indicative of non-transmitted, and thus not received, data.

While carefully managing the encoding rate did improve the quality of the video and reduce the loss rate, we were still receiving an unacceptable number of partial and corrupted frames. We noticed that if an intra-coded frame, whose size was roughly 16 packets experienced even a single packet loss, the entire frame was corrupted and would have to be discarded, which dramatically lowers the quality of the decoded video. This led us to develop and apply a technique of grid encoding where the frame is segmented into smaller subframes that independently encode and decode data. Not only does this isolate the effects of loss, but through staggering when these independent frames begin encoding we also smooth out the send rate. At each frame, a smaller number of subframes are producing intra-coded frames, dividing the cost of sending the large intra-coded frames over the entire group of pictures (GOP). Figure 1-2 shows how this technique has reduced the burstiness of this stream, reducing the number of losses that occurred through overburdening the network with sudden data spikes.

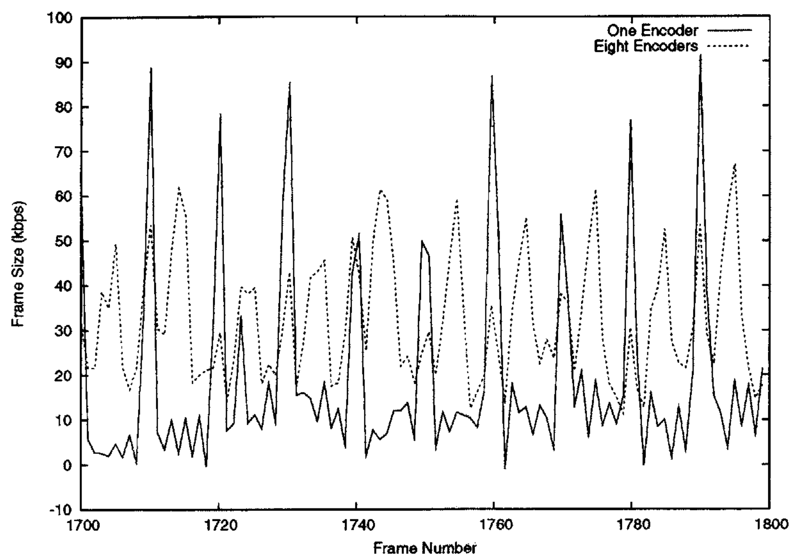


Figure 1-2: The sizes of each frame for a grid of one encoder and for a grid of eight encoders.

These techniques as well as an architecture that emphasizes consistency and reliability have resulted in a system that sends diagnostic quality audio and video streams from a mobile unit even in the presence of drastic network declines, e.g., bridges or tunnels, or hardware failures. As discussed further in chapter 5, while moving through a tunnel we were able to transmit video with an average peak-signal-to-noise-ratio (PSNR) of $32.5dB$, a value above diagnostic quality, even as we experienced losses up to 80%. A PSNR value in the range of $30dB$ to $40dB$ is considered quite good. Through a series of experiments driving over a wide geographical area, we are able to consistently provide diagnostic quality video, yielding median PSNR values that range from $33.716dB$ to $36.670dB$. Additionally, because we utilize opportunistic retransmissions of the audio packets as well as take advantage of the priority flags within Horde, even as the loss rate increases when the vehicle is no longer stationary, the PSNR of the audio stream remains the same.

1.3 Thesis Organization

Chapter 2 provides background and the results of a study of the end-to-end performance for multiple WWAN providers. Chapter 3 discusses related work in the fields of WWANs, multimedia streaming, and telemedicine applications. Our system architecture, including expositions on the networking layers and the audio, video and telemetry subsystems is discussed in chapter 4. An evaluation of the performance of the Tavarua system, in terms of losses, throughput, and the quality of the received video, is presented in chapter 5. Finally, in chapter 6, we present our conclusions as well as directions for future work.

Chapter 2

Network Characteristics

2.1 Introduction

Wireless Wide-Area Network (WWAN) channels provide pervasive mobile internet connectivity, but their end-to-end performance has not been well studied. In order to intelligently design our system we needed to gain insight into the characteristic behavior of WWAN channels. We therefore conducted a number of experiments, measuring end-to-end performance for multiple WWAN providers. Our experiments were conducted over three days, in two cities. Given the small sample size, we are cautious in drawing broad conclusions, however, our data is extensive enough to provide a high-level characterization of the WWAN links we tested. We are unaware of any other public-domain study of these networks as extensive as the one presented in this thesis.

Background

In most urban areas, there are a large number of public carrier wireless cellular channels providing mobile connectivity to the Internet, many of these carriers providing overlapping coverage. Also, there is some technological diversity among the WWAN providers (Verizon [16] and Sprint [15] have deployed CDMA2000 1xRTT/EV-DO [50, 75] based networks, while Cingular [6] has a GPRS/EDGE [51] based network).

Individual WWAN channels provide little in the way of quality of service guarantees. Achievable average throughput can deviate significantly from advertised peak rates. Additionally, service providers optimize for the downstream link, so the upstream bandwidth offered by these channels is far less than the advertised rates might lead one to believe. These WWANs are also dogged by high and variable round trip times, occasional outages, and considerable burstiness.

The performance at any given time depends on a multitude of factors, including, the underlying wireless technology (e.g., CDMA or GPRS), the spatial placement of the WWAN interface relative to the provider's base-stations, competition with other users in the WWAN cell, and artifacts arising from physical occlusion and electromagnetic interference.

The most advanced WWAN technology presently deployed in the US is the one in use by both Sprint and Verizon's Code Division Multiple Access 1xRTT and EVolution-Data Optimized (CDMA 2000 1xRTT, CDMA 2000 1xEV-DO, and CDMA 2000 Rev-A) networks [50, 75, 92]. These standards are often collectively referred to as 3G CDMA. EV-DO is an evolution of the 1xRTT specification, and is optimized for the efficient transfer of data. Rev-A is a further optimization that provides higher data transfer rates. All experiments in this chapter utilize EV-DO. The advertised rates for EV-DO data transfer are $2.4Mbps$ for downstream (base station to mobile user) and $153kbps$ for upstream transfers (mobile user to base station). In order to mask air channel losses, these networks make aggressive use wireless of coding schemes, specifically turbo codes, and wireless link layer retransmissions [50, 92].

Experimental Methodology

To provide realistic estimates for cellular service performance, we conducted three sets of experiments, two of which were conducted in Orlando, FL and one in Boston, MA. The two Orlando experiments were conducted on consecutive days, and the Boston experiment was conducted a couple of months later.

In these experiments, we tested for available upload throughput, packet round-trip-times, and loss characteristics. We attempted to isolate and individually analyze the effects of vehicular speed and geographical location. The goal of these experiments was to determine how well multiple WWAN interfaces would perform, if they were transmitting simultaneously in close proximity, inside a moving vehicle.

All experiments consisted of driving through a large area constantly transmitting data from every active WWAN interface, to a host on an wired ethernet in Boston. The data consisted of 1024-byte UDP packets, and sending rates were limited by a congestion control algorithm. The packets were sent from Linux laptops, using PPP connections over the WWAN interfaces.

When we have comparable data between the Boston or the Orlando experiments, we will present results from the Boston data set, as it is more recent and utilizes the optimized congestion control algorithm. Some experiments were only conducted in Orlando and as such, that data will be presented. The Orlando experiments occurred in March, 2006 and the Boston experiments were conducted that following summer. In April, 2007, while we were conducting our system evaluation we upgraded three of our interfaces to Rev-A. While the per channel throughput increased, the other aspects of the network remained consistent with what we saw from the EV-DO interfaces. Therefore this study remains useful as a high level overview of the behavior of these channels, even as providers race to upgrade their networks.

The remainder of this section provides an overview of the three experiments.

Orlando-1 The first set of experiments, in Orlando, used four interfaces from three different providers: two EV-DO enabled interfaces from Verizon, one of which is slightly older and has different hardware than the other Verizon card used, an EV-DO interface from Sprint, and an EDGE based interface from Cingular. This set of experiments was our first thorough investigation of how speed, location and possible cross-channel interference would affect throughput. Cingular, the one EDGE interface tested, provided unacceptable performance for our purposes, possibly due to spotty coverage in the area we were testing. We abandoned the Cingular interface part-way into the experiment. For the rest of this thesis our focus will be on CDMA networks.

Orlando-2 It is not immediately clear how many co-located WWAN interfaces one can operate simultaneously before experiencing cross-channel interference in either the air channel or further down stream.

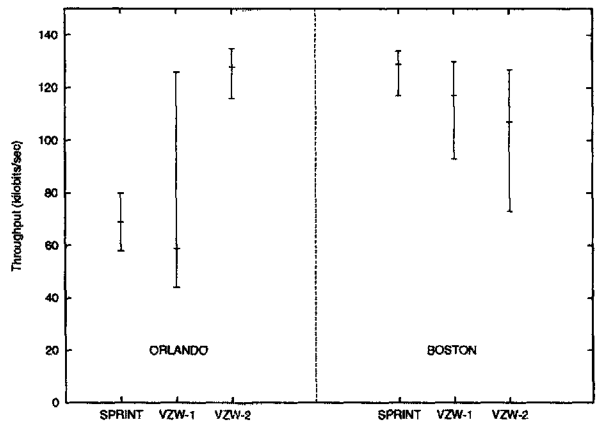
To shed some light on this question, the second set of experiments explored carrier scalability using four EV-DO enabled interfaces from Verizon. Verizon was chosen for this experiment because it had performed the most reliably with the highest throughput in the earlier Orlando experiment.

Boston The final set of experiments, in Boston, used three interfaces from two different providers: two EV-DO enabled interfaces from Verizon, one of which is slightly older and has different hardware than the other Verizon card, and an EV-DO interface from Sprint. In this experiment we wanted to explore two things: understanding and reducing the loss rate and analyzing the effect of vehicular speed on throughput. A high average loss rate is unacceptable. We needed to lower the loss rate as well as understand the nature of these losses in order to offset their effects in our application.

To reduce the loss rate we changed the congestion control algorithm that we had used in Orlando. The new algorithm uses some ideas from TCP Vegas [27] about interpreting elevated packet round-trip-times as indicators of congestion. This allows the algorithm to avoid most of the systematic probing losses incurred by a TCP-like AIMD scheme.

2.2 Throughput

The per-channel upload throughput was relatively low on average and varied significantly during our experiments. The network experiments in this section refer to UDP throughput, using 1024-byte packets, and ignoring IP/UDP overhead. Figure 2-1 shows the upload throughput distributions from two experiments. In the Orlando experiment there is a striking difference between the two Verizon interfaces. This is likely due to the differences in hardware between the older $PC5520$, shown as VZW-1, and the newer $KPC650$, VZW-2.



Interface	Median	Mean	StDev
Sprint	$65kbps$	$64kbps$	$17kbps$
Verizon-1	$100kbps$	$95kbps$	$30kbps$
Verizon-2	$125kbps$	$120kbps$	$18kbps$

Orlando

Interface	Median	Mean	StDev
Sprint	$130kbps$	$121kbps$	$28kbps$
Verizon-1	$115kbps$	$107kbps$	$29kbps$
Verizon-2	$105kbps$	$97kbps$	$35kbps$

Boston

Figure 2-1: Upload throughput distributions in two experimental data sets. The graph shows the median, upper and lower quartiles. Throughput was averaged over two-second windows.

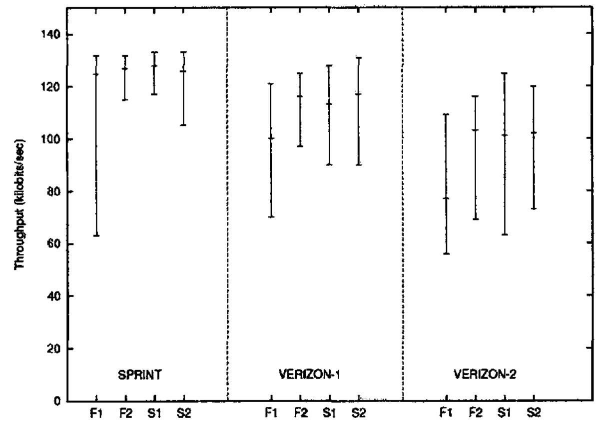


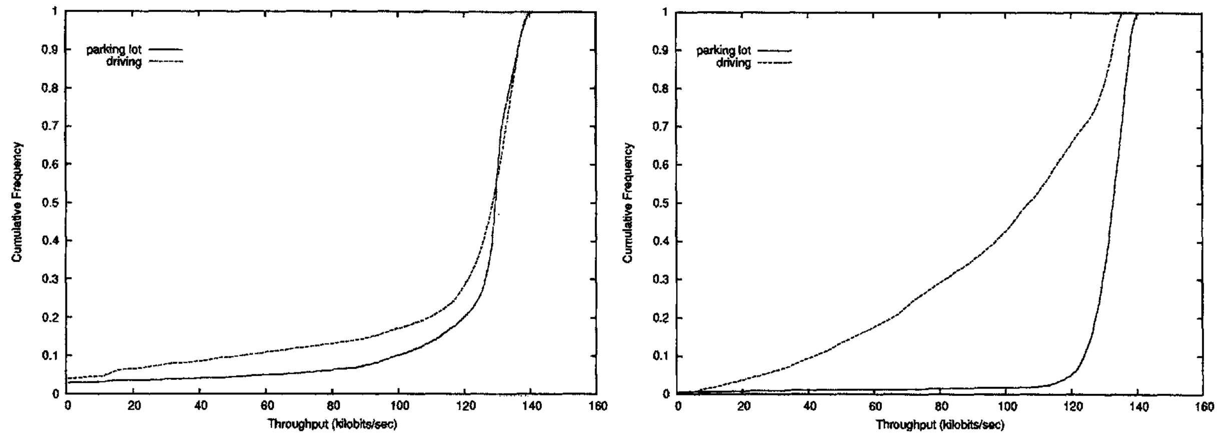
Figure 2-2: Upload throughput distributions in four experiments conducted driving the same loop but with different speeds. In F1 and F2 the vehicle was moving fast; and in S1 and S2 the vehicle was moving very slowly.

The throughput achieved during the Boston experiment were significantly different from those achieved during the Orlando experiment, with both the Sprint and older Verizon interfaces showing marked improvement. A different congestion control algorithm was used in Boston, which may be a factor. However, since the second Verizon interface showed a slight decline between the Orlando and Boston experiments, the differing coverage maps, i.e., the way in which coverage varies based on geographical location, is likely the dominant factor giving rise to the variation. Figures 2-5 and 2-6 show approximations to the true coverage map based on our measurements.

Average data rates are often far below advertised rates. According to the CDMA2000 1X standard, the WWAN interfaces we were using are capable of delivering peak packet data rates of 153 kilobits-per-second. At our peak, we failed to achieve even 140 kilobits-per-second. On average for each interface, the highest sustained rate was around 120 kilobits-per-second, while the lowest in Orlando was around 60 kilobits-per-second.

Baseline Variation

Achievable throughput varies when stationary, and can vary much more when moving about a city. This variation is not due to vehicular motion, as demonstrated in figure



(a) Sprint **(b) Verizon-2**

Figure 2-3: Throughput distribution when stationary compared to the distribution for the overall Boston experiment. The graphs show the upload throughput CDF's for two interfaces.

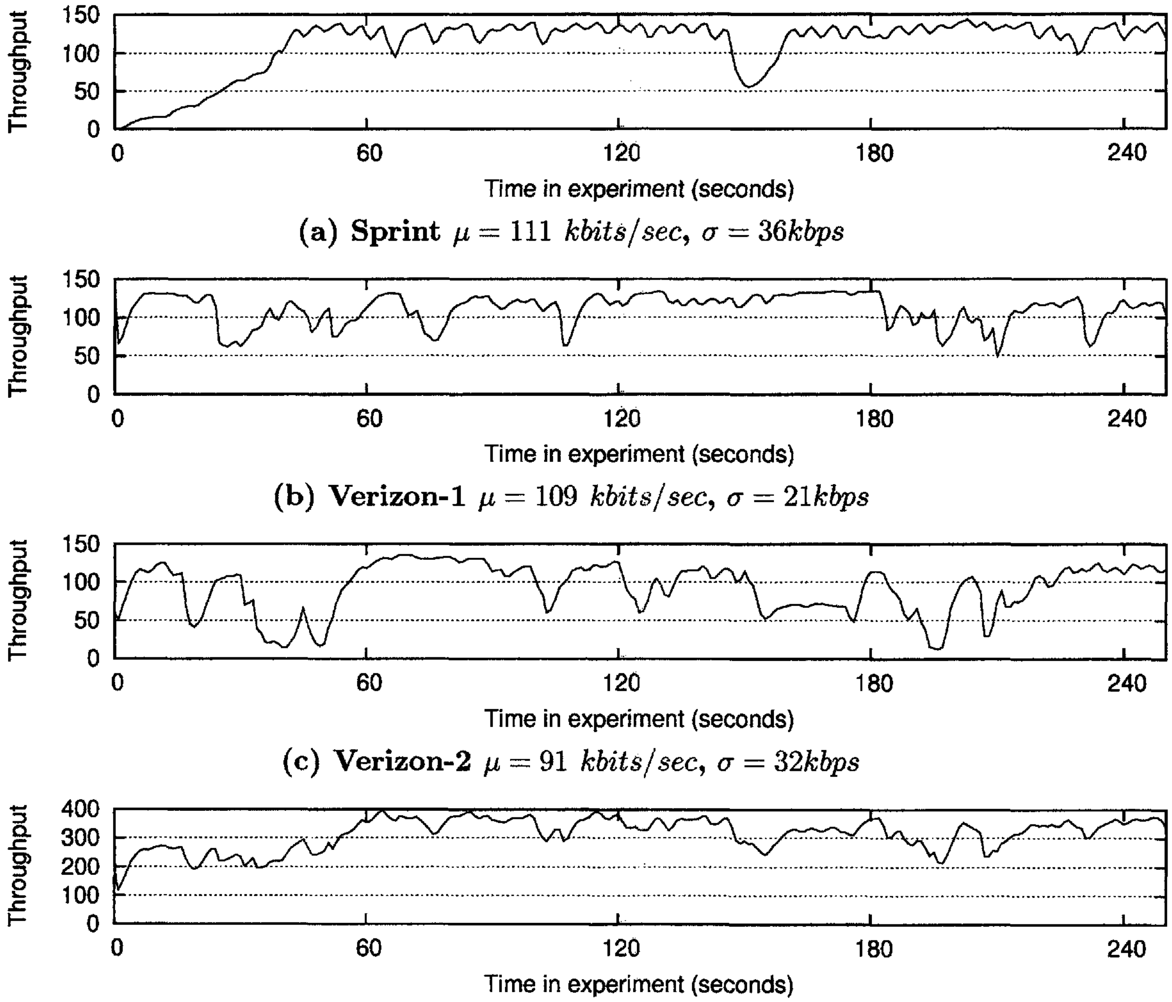
2-2, but likely due to a lack of uniformity in the coverage area, caused by base-station occlusions or entering a lower provisioned cell. Figure 2-3 compares these two situations, using data from the Boston experiment. Each graph in figure 2-3 plots the upload throughput CDF during a long period of time spent stationary in a parking lot (a location with good signal); and the CDF of the entire Boston experiment.

Both interfaces had lower variation when stationary. When moving, the Verizon interface (figure 2-3b) experienced a significant change in behavior. In contrast, the variation on the Sprint interface did not change much. Given that the two interfaces were co-located this seems to imply that the coverage provided by Verizon in the area we measured is less-uniform than the coverage provided by Sprint.

Variation in Time

Figure 2-4 shows how throughput varied over time during part of the Boston experiment. The vehicle was moving relatively fast, on average approximately 50mph, during this period. This snapshot is from the middle of a long-running experiment. The ramp-up at the start of the Sprint graph is recovery from a short-lived disconnection.

As indicated in the aggregate throughput graph, figure 2-4d, the use of multiple



(a) **Sprint** $\mu = 111$ $kbits/sec$, $\sigma = 36kbps$

(b) **Verizon-1** $\mu = 109$ $kbits/sec$, $\sigma = 21kbps$

(c) **Verizon-2** $\mu = 91$ $kbits/sec$, $\sigma = 32kbps$

(d) **Aggregate** $\mu = 306$ $kbits/sec$, $\sigma = 64kbps$

Figure 2-4: An example of the dynamic variation in available upload throughput while moving in a vehicle.

interfaces provides a considerable smoothing effect.

The Vehicle's Speed

We were interested in determining if our vehicle's speed would significantly impact WWAN channel performance as suggested by the simulated study in [25]. In order to do this, we constrained our location and ran multiple experiments at multiple speeds. We chose a five block loop in the Back Bay area of Boston, and drove the loop four times: twice traveling close to the speed limit, 30 mph, and twice traveling very slowly, around 5 mph.

Figure 2-2 shows the upload throughput distribution for each loop. There is no clear difference between the slow loops, during which the vehicle was barely moving, and the faster loops.

All the WWAN channels performed their worst during the `F1` experiment. `F1` was the first experiment we conducted. At this point we had just entered the geographical area in question. It is possible that performance during `F1` was handicapped due to a start-up cost (e.g., base-station handoff etc). `S1` was the second experiment we conducted; `S2` the third; and `F2` was the fourth.

Geographical Variation

Figures 2-5 and 2-6 are the measured coverage maps for Boston and Orlando, respectively. These maps imply that the coverage varies significantly with geographical location.

2.3 Latency

For large packets, round-trip-times (RTTs) were high (averaging around $600ms$). They were also highly variable when the vehicle was moving to different locations, perhaps due to base-station occlusions, handoffs, etc. Figure 2-7 shows how 1024-byte packet round-trip-times were distributed for the WWAN interfaces we tested.

The distributions for the different CDMA interfaces, even those from different providers, are similar, in both cities. There is more variation in the round-trip-times from the Orlando experiments. Congestion control may have been the dominant factor here, as an older congestion control algorithm was being used during the Orlando experiments. The older algorithm tried to send as much data as possible, sending at levels the exceeded the available bandwidth of the network. Because the network itself tries to prevent these losses, link-layer retransmissions were increased, yielding higher packet round-trip-times [50].

Figure 2-8 shows examples of how RTTs varied from packet to packet on one of the

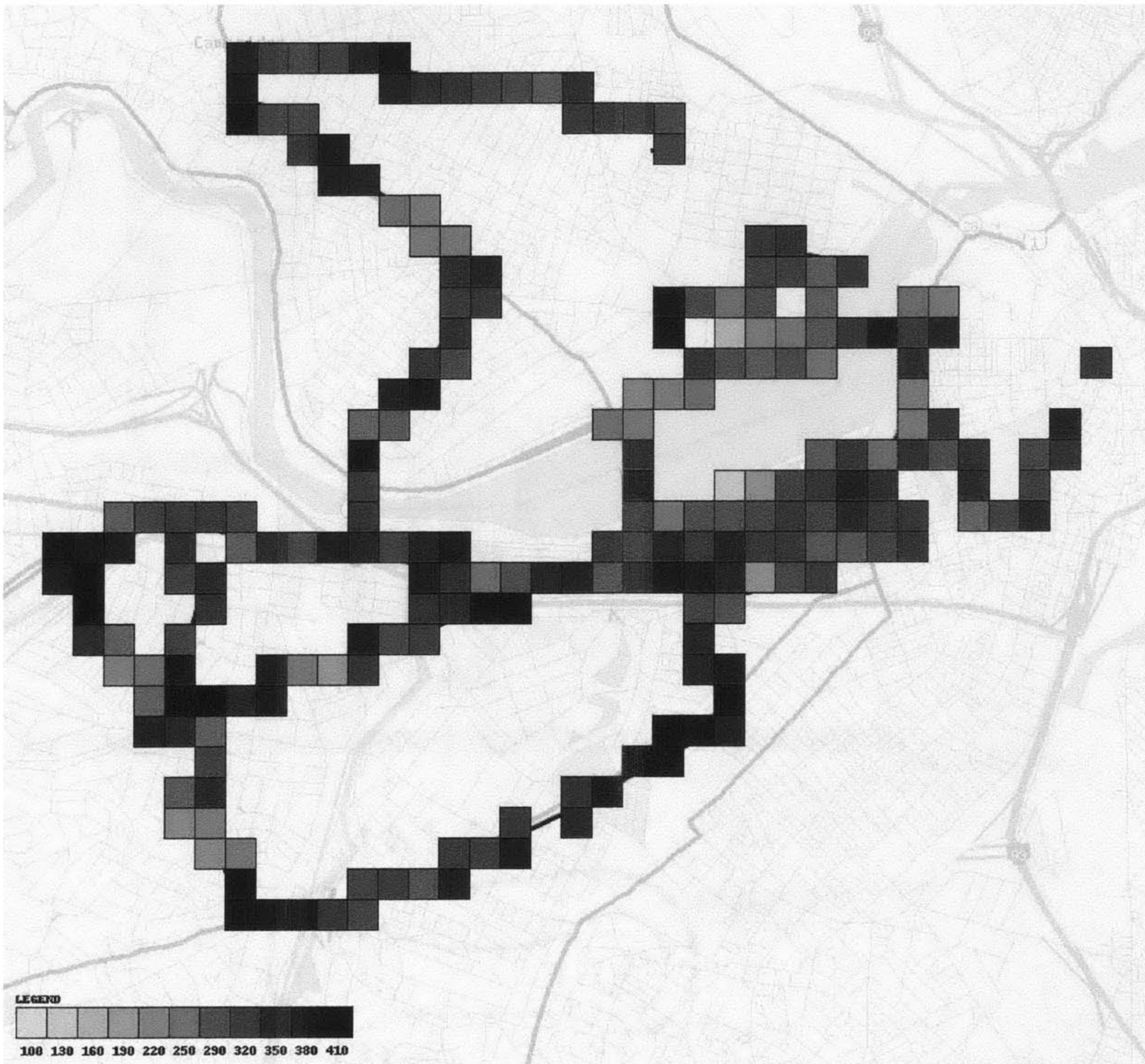


Figure 2-5: Measured coverage map for Boston. Darker squares indicate regions of higher throughput.

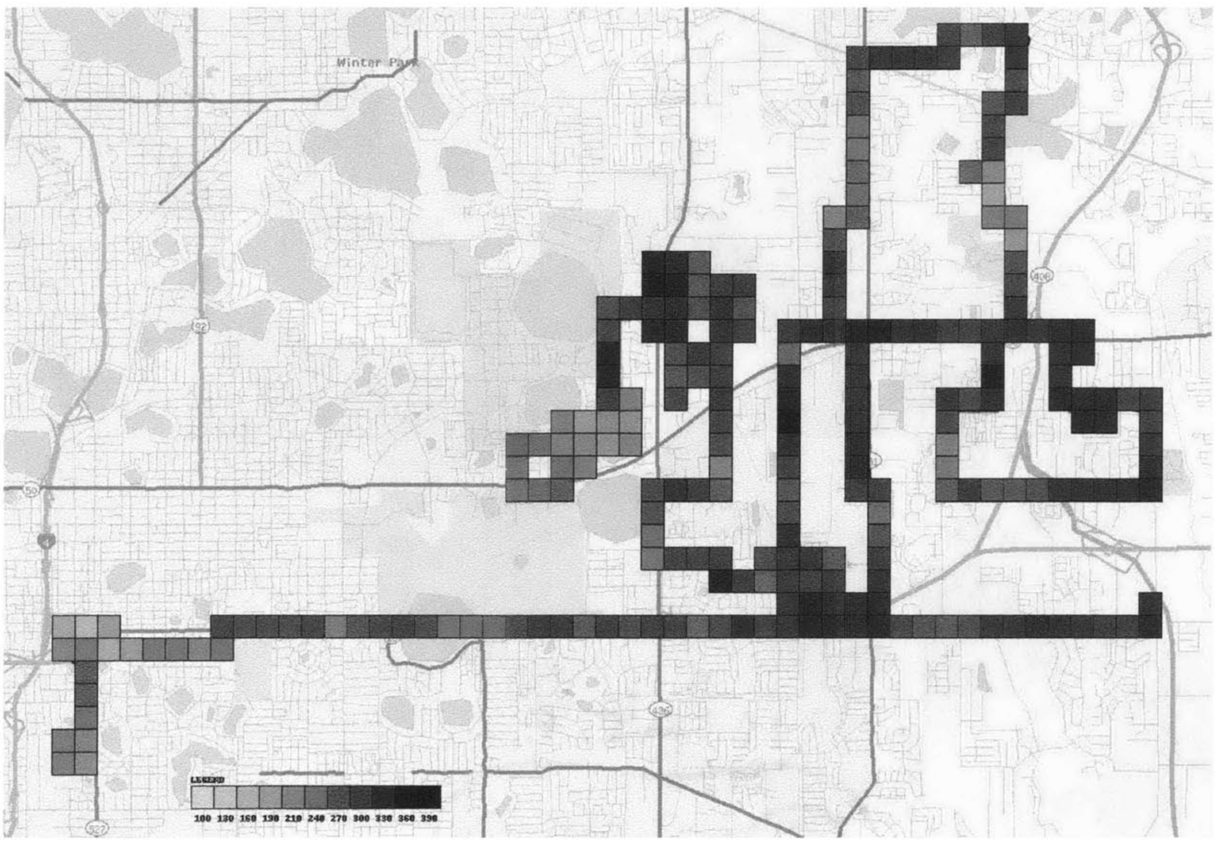
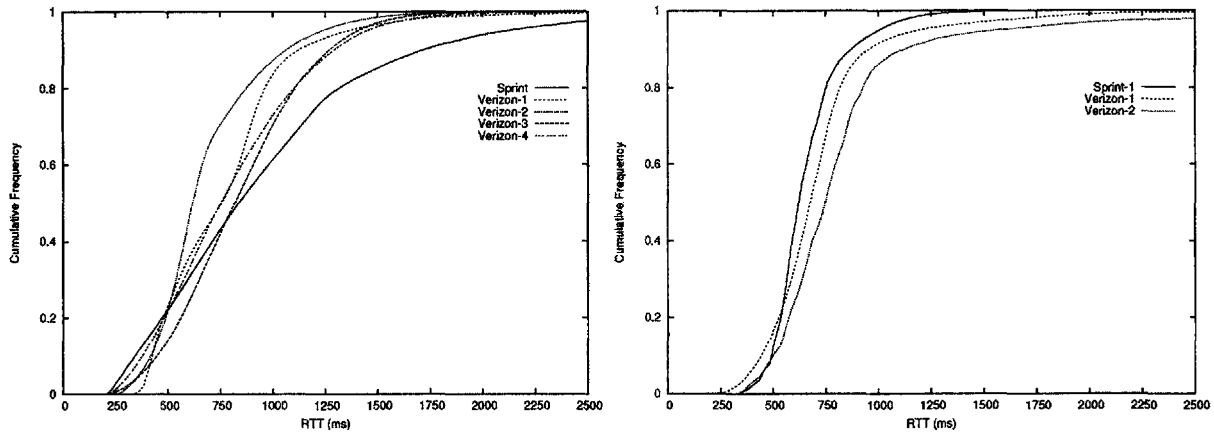


Figure 2-6: Measured coverage map for Orlando. Darker squares indicate regions of higher throughput.



Interface	Median	Mean	StDev
Sprint	845ms	979ms	655ms
Verizon-1	755ms	779ms	431ms
Verizon-2	715ms	793ms	317ms
Verizon-3	825ms	843ms	321ms
Verizon-4	755ms	806ms	380ms

(a) Orlando

Interface	Median	Mean	StDev
Sprint	615ms	657ms	180ms
Verizon-1	685ms	718ms	338ms
Verizon-2	745ms	844ms	563ms

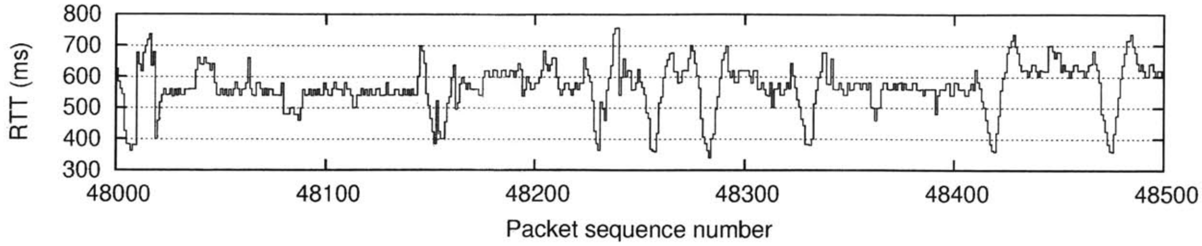
(b) Boston

Figure 2-7: Measured packet round-trip-time distributions (measured CDFs).

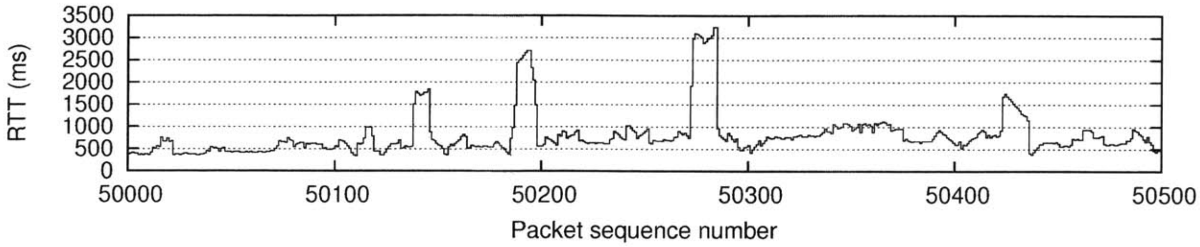
interfaces. During 'good' periods, see figure 2-8a, RTTs are relatively stable around $570ms$. However, during some periods, see figure 2-8b, RTTs become elevated, close to $700ms$, with some spikes in excess of $3s$.

Packet Size

Packet size directly affects the expected round-trip-time for a packet. Larger packets tend to have higher round-trip-times. Figure 2-9 shows packet RTT distributions for different packet sizes in a stationary experiment conducted in Boston. As packet sizes are increased, there is an increase in both the median RTT and the variance.



(a) 'Best'-case behavior $median = 565ms,\ \mu = 567ms,\ \sigma = 74ms$



(b) Elevated RTTs $median = 685ms,\ \mu = 795ms,\ \sigma = 485ms$

Figure 2-8: Two examples of the packet-to-packet variation in round-trip-times (on Verizon-2, in Boston).

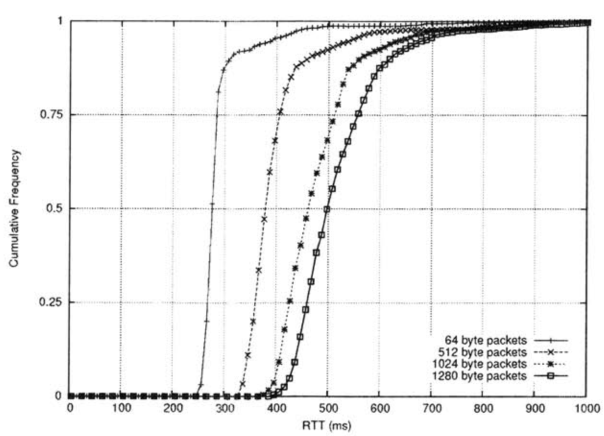


Figure 2-9: Packet round-trip-time distributions for ICMP ping packets of different sizes. Increasing packet size leads to significantly larger average RTTs. There was no motion in these experiments.

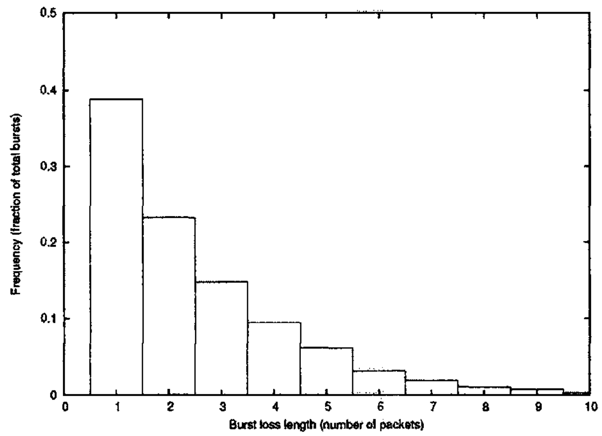


Figure 2-10: The measured frequency distribution of packet-loss burst lengths for a single interface (Verizon-2, Orlando). The other interfaces had similar distributions both in Orlando and in Boston. The distribution implies an exponential PDF.

2.4 Losses and Disconnections

Losses

The CDMA interfaces were able to mask most wireless losses, therefore packet losses were mostly due to network congestion. With unoptimized congestion control algorithms, packet losses were as high as 10%. With TCP-style AIMD this high loss-rate is unavoidable: the congestion window was relatively small, and a loss was guaranteed in every cycle. By optimizing the congestion control algorithm (e.g., using elevated RTTs as indicators of congestion), we were able to reduce packet losses to 0.5% when stationary. When moving, loss rates rose significantly, but mostly stayed below 5%. This elevated loss is likely due to a decline in network bandwidth as we move from one location to another. If the next location has lower available bandwidth, losses will occur until the congestion control can throttle back the sending rate to match the available rate at the current location.

Packet loss bursts were mostly single-packet losses. Figure 2-10 shows how frequent different burst lengths were for an interface, during the Orlando experiments.

Disconnections

A *disconnection* is a period during which an interface is not able to upload any data. During our experiments, every interface was constantly attempting to upload data, throttling itself by using a congestion control algorithm.

Even though individual disconnections could be reasonably long (e.g., 15 seconds), no interface spent a significant fraction of the total time in a disconnected state. The Boston experiment lasted well over two hours. The Sprint interface was disconnected for 3.5% of the experiment. Each of the Verizon interfaces were disconnected for less than 0.5% of the experiment. Moreover, disconnections were not correlated in time across the different interfaces.

2.5 Carrier Scalability

In order to examine how well performance scaled, we simultaneously transmitted from four co-located interfaces from the same provider, three of which were the same hardware, one was an older model, in Orlando. Figure 2-11 shows how upload throughput varied during this experiment. The graphs show the smoothed average throughput over about 3 hours of continuous operation.

Throughput

In this experiment, the four interfaces were able to sustain average upload data rates of around 75 *kbits/sec* each. In contrast, during our earlier experiment in Orlando (see figure 2-1), with only two Verizon interfaces, one interface was able to achieve an average of around 130 *kbits/sec*.

We were able to increase the average aggregate throughput by a factor of $1.5x$ by doubling the number of Verizon interfaces. This implies diminishing returns from adding additional interfaces from the same provider. This is likely due to contention for shared resources among the interfaces.

Correlation

We were interested in whether these four Verizon interfaces exhibited correlated behavior. Of the four, three interfaces were highly correlated with each other, in terms of achieved upload throughput. This is shown by the matrix of correlation coefficients[1], figure 2-11f. Losses and packet round-trip-times were not correlated in time across the interfaces.

The uncorrelated interface consisted of a different version of the hardware; the others were comparable. This may explain the difference in behaviors. When we upgraded our system to include the Verizon Rev-A $USB720$ and the Verizon Rev-A $PC5750$ we observed that the USB interface did not correlate strongly with the Verizon EV-DO $KPC650$, but there was a strong correlation between the USB interface and the $PC5750$. This leads us to believe that the problem lies in the firmware or the hardware, not that two interfaces have saturated the network.

Unlike figure 2-4, the graphs in figure 2-11 do not show raw throughput signals. We used a low-pass filter to smooth out the throughput signal, so our correlation analysis would ignore high frequency variations.

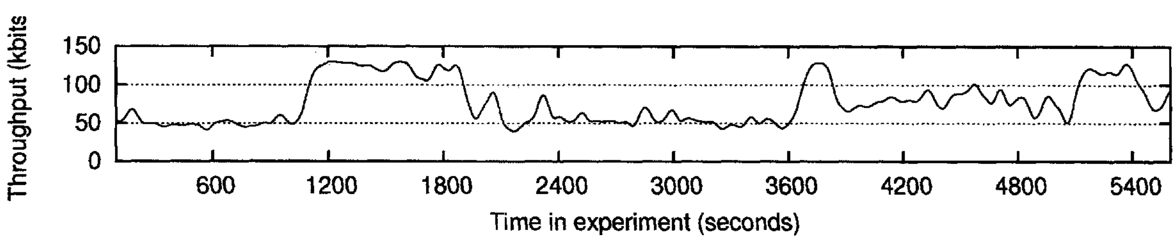
Summary of results

Our analysis shows that in practice, the upload throughput seen by an interface varies considerably. However, contrary to our expectations, which were derived from simulation based network studies [25], we found no correlation between the vehicle's speed and achieved throughput. It seems that geographical location is the dominant factor leading to variation. This geography-based variation can be attributable to any of several factors including distance to base station, topography, occlusion by other structures, and electro-magnetic interference.

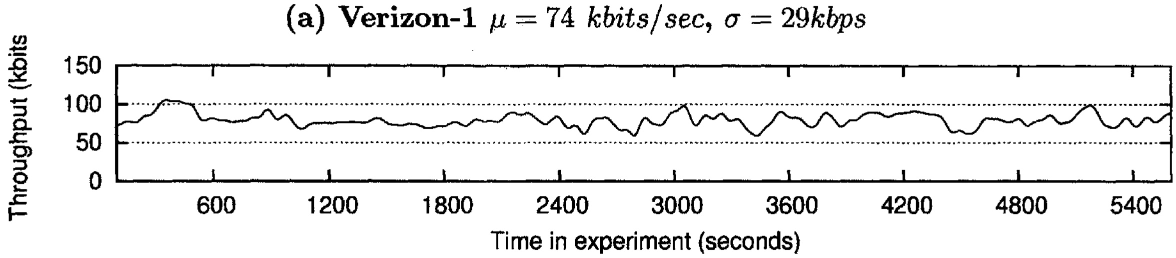
Disappointingly, the peak upload throughput for an interface never reached the

[1] The correlation coefficient of the two signals x and y is defined as:

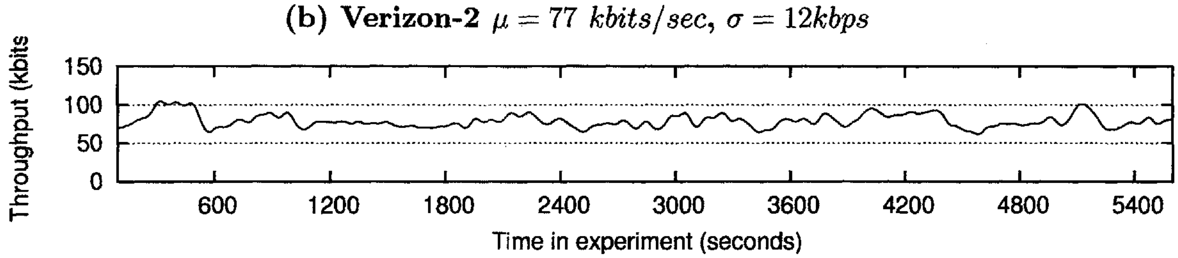
$$correlation_{x,y} = \frac{covariance_{x,y}}{\sqrt{covariance_{x,x} \cdot covariance_{y,y}}}$$



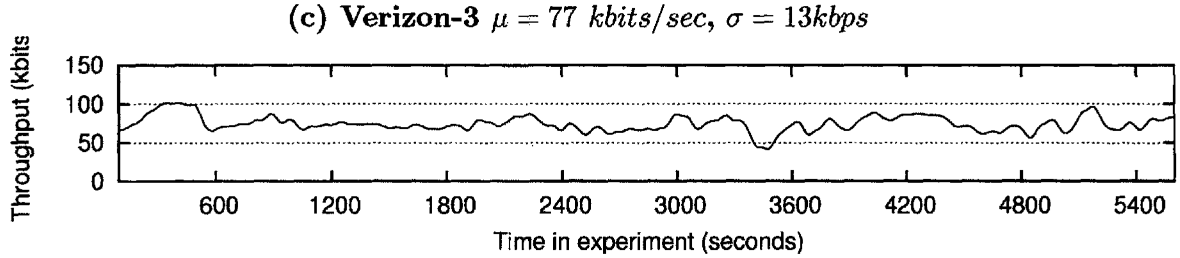
(a) **Verizon-1** $\mu = 74\ kbits/sec$, $\sigma = 29kbps$



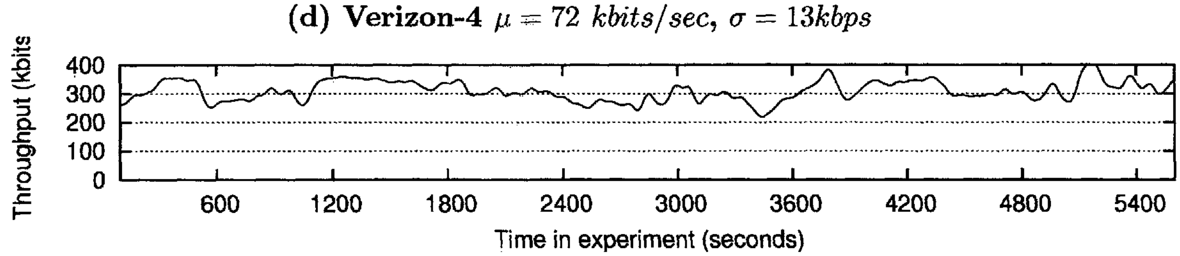
(b) **Verizon-2** $\mu = 77\ kbits/sec$, $\sigma = 12kbps$



(c) **Verizon-3** $\mu = 77\ kbits/sec$, $\sigma = 13kbps$



(d) **Verizon-4** $\mu = 72\ kbits/sec$, $\sigma = 13kbps$



(e) **Aggregate** $\mu = 302\ kbits/sec$, $\sigma = 50kbps$

	Verizon-1	**Verizon-2**	**Verizon-3**	**Verizon-4**
Verizon-1	1.00	0.11	0.12	0.15
Verizon-2	0.11	1.00	**0.94**	**0.90**
Verizon-3	0.12	**0.94**	1.00	**0.92**
Verizon-4	0.15	**0.90**	**0.92**	1.00

(f) **Correlation coefficients for the throughput signals.**

Figure 2-11: Upload throughput (smoothed) on four interfaces from the same provider, in Orlando, and the correlation coefficients for these signals. Verizon-1 used older hardware than the others. The other three interfaces exhibited correlated throughput behavior.

rate promised by network providers (never exceeding $140kbps$). We were also disappointed that the evidence suggests that the throughput provided by interfaces from the same provider can be strongly correlated in time.

Our scalability experiments made it clear that there is a point of diminishing returns for co-located network interfaces for the same carrier. Doubling the number of interfaces (from two to four) led to only a 50% increase in throughput.

The measured packet round-trip-times (RTTs) for reasonably large packets were consistently high (around $600ms$) with high variance ($\sigma \approx 350ms$), even when the CDMA channels were otherwise well-behaved. Periods of elevated RTTs were not uncommon, with RTT spikes as high 3.5 seconds. Fortunately, the RTTs were not correlated across interfaces. Taking advantage of this to ensure that long RTT's don't adversely effect the user experience is an important issue in the design of our interactive video system.

We experienced only a small number of disconnections during which an interface was unable to upload any data. Furthermore, these periods were not correlated across interfaces, suggesting that the diversity provided by multiple interfaces can be used to enhance reliability.

Chapter 3

Related Work

This chapter will cover previous work on aspects explored in the Tavarua system. We will discuss prior attempts at characterizing WWANs, other projects that performed video streaming over WWANs, and provide a background into the current state of telemedicine applications. We will conclude by discussing Horde, the network striping middleware on which Tavarua is built.

3.1 Wireless Wide Area Networks

Since the invention of CDMA technologies several efforts have been made to quantify their performance. Largely these undertakings have focused on theoretical analysis, or relied on simulators to model these channels [44, 95, 64, 62, 68, 25]. These studies have concentrated mainly on analyzing link capacity, ignoring issues such as burst losses or variation in packet round-trip-times. Additionally, TCP-based analysis has been done [31] with simulators that address rate variability, but have stopped short of conducting real-world measurements. There have been a few attempts at measuring the actual nature of CDMA networks, notably [63], that logged measurements using one interface from a stationary location.

Other projects have also explored the idea of exploiting the use of diversity in WWAN channels to increase reliability. A successful example of a project of this nature is Microsoft Research Lab, Cambridge's MAR project [83]. MAR focuses on interfacing with existing internet applications, using the diversity solely for reliability and does not try to aggressively push the bounds of these independent interfaces to maximize bandwidth.

Congestion control algorithms are an integral part of any network. However, not all networks are created equally. The requirements of an 802.11 network using TCP are drastically different from a 3G network using UDP. Optimizing the congestion control algorithm for WWANs has been previously explored [86, 29]. However, these algorithms were only tested on simulated WWAN channels, instead of a deployment of actual WWAN interfaces.

A possible way to estimate signal strength, which is an indicator of available bandwidth, is through querying which base station a device is connected to and looking up its GPS coordinates. Locations of cellular base stations have been logged extensively in many European countries [13]. In the United States, the FCC [7] provides some information about base stations, however this information is extremely coarse. Only SIDs are logged, which in the best case only narrows down the location to a sizable region, and only antennae large enough to be registered by the FCC are used, which could ignore many cell sites especially in urban areas.

3.2 Video Streaming over Wireless Networks

As WWAN technology steadily improved, it became seen as another platform to stream multimedia data[32]. This project utilizes provider diversity to increase transmission reliability, however, they achieve this by sending packets redundantly over these independent channels. This effort and other multimedia applications that utilize the cellular network for data transmission are largely focused on the bandwidth-rich downlink channel, while our target applications demand solutions for accommodating transmissions using the impoverished uplink channel.

The problems that arise in sending delay-sensitive packets over unreliable IP links have also been studied. In [33] the idea of network striping to send multimedia data is addressed, but only for the case of constant transmission rates. Because of the mobile nature of our application, we need striping protocols that flexibly adapt to the available network.

Approaches such as multi-description video streaming [89] also address the issues of packet and compression losses by sending layers of video each of increasing quality.

3.3 Telemedicine Applications

There exist several telemedicine applications that incorporate technology into patient care. The tele-stroke program at Massachusetts General Hospital (MGH) [10] uses dedicated, stationary tele-conferencing facilities to connect MGH with several distant sites. This allows stroke specialists at MGH to evaluate patients who have been brought to one of the remote sites. Radiologists and dermatologists often use the 802.11 network to send high quality photographic images used for remote consults [73].

As satellite technology emerged, so did early mobile telemedicine applications[76, 52, 85, 72]. Because these applications relied on satellite technology, they were dogged by high packet round trip times, making them non-interactive. Additionally, the high cost of utilizing such networks precluded their widespread use.

With the advent of cellular wireless wide area networks (WWAN) there was an economically feasible infrastructure upon which to build mobile telemedicine applications. Many applications simply transmit telemetry data, such as ECG and heart rate[56, 58, 70, 42, 74]. Others, such as the AMBULANCE project[77, 60, 38] transmitted still images as well as telemetry. These efforts operated under low bandwidth conditions, since they only used one WWAN interface, and high loss rates because of unoptimized congestion control.

Using custom hardware, researchers at the University of Maryland, developed a system to aid in diagnosing ischemic stroke in-route to the hospital by sending video clips of patients engaging in stroke skills tests[37, 49]. Even with incurring the cost of developing specialty hardware and not restricting themselves to the real-time domain, they still only achieve video rates of eight frames per second.

At the University of Massachusetts, Amherst they have developed a mobile telemedicine system with commodity hardware that transmits data over a single WWAN interface[34]. Since they use a single interface, the maximum bandwidth available for them to transmit the telemetry, audio, and video streams is on average $70 - 80kbps$. This yields a frame rate ranging from $4.2fps$ when only video is transmitted and $1.5fps$ when all three streams are being transmitted. Additionally, in order to achieve these frame rates they reduce the resolution to 160x120. Also, when the video stream alone reaches three frames per second, the average latency begins to exceed three seconds. If the frame rate should increase to a rate that would provide smooth motion, it would follow that the latency would increase drastically.

3.4 Horde: Network Striping

Tavarua is built on top of Horde[82], a network striping middleware.

Network striping takes data from a single source channel, sends it in some order over a set of channels, and, if appropriate, reassembles the data in the correct order at the other end. A great deal of work has been done on network striping [19, 40, 67, 83, 87]. Most of this work is aimed at providing improved scheduling algorithms under the assumption that the underlying links are relatively stable and homogeneous.

In our environment the underlying links are neither stable nor homogeneous. Moreover, the data streams in our telemedicine system have distinctly different service needs. Notably, the video stream is highly sensitive packet losses and jitter while the audio stream is less so.

In many cases, researchers have decided to make the striping layer invisible to applications. Applications are written as if there is a single large transmission channel. However, when striping heterogeneous data streams over a multi-provider set of WWAN channels, the manner in which the striping middleware decides to schedule the transmission of application packets can have a large influence on observed packet latencies, stream loss rates, and throughput. This, in turn, can have a large impact on the utility the application layer derives from network service. This suggests that it is important to allow applications some control over the way in which striping is done.

One approach to doing this involves application-specific code to perform the striping. This is often the method chosen by multi-path video streaming applications [84, 21, 24]. However, this can lead to complicated application-level code, and typically incorporates implicit assumptions about the actual network channels.

Horde provides transparency into the multiple, heterogeneous application data streams, providing network performance characteristics to the application as well as using this information about the specific channels to efficiently stripe the data. We use Horde because it allows control over how the data should be sent, for example sending audio data as a higher priority than video data, without the burden of handling low-level channel management issues.

Chapter 4

System Architecture

Figure 4-1 shows the structure of the Tavarua system.

The hardware in Tavarua consists of two main components, a "sender" placed on the ambulance, and a "receiver" residing at a stationary command center, probably in a hospital. The component on the ambulance consists of a PC, multiple lightweight routers, WWAN interfaces, a bluetooth microphone headset, and video cameras.

The software in Tavarua consists of a collection of applications (video streaming, audio communications, telemetry transmission, etc), supported by networking middleware.

The networking middleware provides network striping capabilities to the applications. Application data is distributed over many network interfaces. The middleware arbitrates between different applications, and optimizes for each application's QoS requirements. The middleware also handles low-level issues related to the network interfaces (e.g., congestion control, disconnections and reconnections). An abstract striping interface is exposed to applications.

4.1 Design Goals

The set of applications is derived from the needs of our mobile telemedicine project. The video server has, by far, the highest network bandwidth requirements. The video server is designed to exploit the availability of multiple channels, and must be able

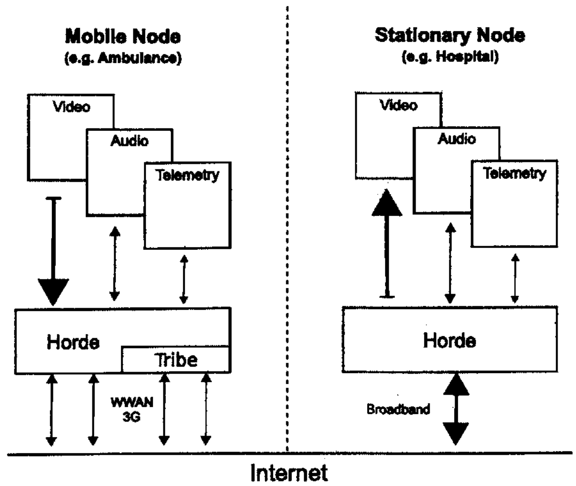


Figure 4-1: Overview of the Tavarua system.

to deal with network instability (both in terms of varying bandwidth and varying packet latencies). A separate application handles bi-directional voice communications. Although the voice channel does not need to use network striping to overcome bandwidth limitations, it does take advantage of the multiple network channels to increase the reliability of the audio link. Finally, a third application, yet to be built, will handle the relatively low-bandwidth telemetry.

System reliability is an important issue. Tavarua is part of a multi-city multi-disciplinary research project. In the coming months, we expect to deploy Tavarua as part of a months-long medical study. During this period, the system needs to operate with high availability, and with minimal interaction with its engineers. In order to ensure reliability, we have taken several steps to harden the system. The system is structured as a set of restartable components, keeping everything in user-space, and taking advantage of process-level isolation.

In the following sections, we discuss aspects of the Tavarua system in more detail. We begin with *Tribe*, a network sub-system, and then address the video, audio, and telemetry applications.

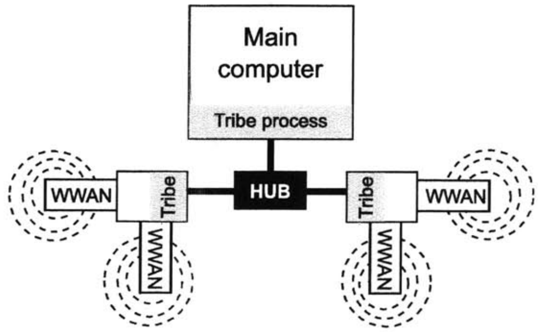


Figure 4-2: The Tribe protocol allows Horde to transparently connect to an arbitrary number of PCMCIA WWAN interfaces.

4.2 Tribe: Connecting to Many Networks

Our network experiments imply that meeting our minimum bandwidth requirements using today's networks will require operating at least five WWAN interfaces simultaneously, and perhaps as many as ten, depending on how many of the cards are Rev-A or EV-DO. Some of the interfaces are PCMCIA cards, while the remainder are the new generation of USB interfaces. We considered using PCI-to-PCMCIA bridges to connect the necessary network interface cards to a single machine. In the end, however, we chose a more scalable software-based approach, using conventional off-the-shelf hardware. Figure 4-2 shows the structure of this solution.

The main computer is connected to a network of lightweight routers over a local high-speed ethernet. Each router box handles forwarding packets to and from the main computer. In the present implementation, each box is a `net4521` embedded computer, from Soekris Engineering [14] and can have a maximum of two WWAN interfaces.

The *Tribe* protocol runs over the local ethernet. The protocol allows the main computer to keep track of active WWAN interfaces (as they come up and go down, in response to call disconnections and reconnections), and to remotely manage and restart services on the routers as necessary.

On the primary computer, Tribe emulates each active remote interface as a local interface. This emulation is accomplished using a standard `TUN/TAP` Linux kernel module. `TUN` and `TAP` are virtual kernel network drivers where `TUN`, abbreviated from network tunnel, is used for routing IP packets while TAP creates a virtual ethernet interface. Together they allow user-space applications to send and receive packets from an associated device. Packets sent to a TUN interface managed by Tribe are transparently tunneled over the local ethernet to the appropriate Soekris box, before being sent out over a WWAN interface.

With Tribe in place, Horde has the ability to treat the remote WWAN interfaces as local interfaces. Tribe masks the additional complexity associated with managing the remote interface cards.

In addition to not requiring specialized hardware this solution also scales well. If additional bandwidth is desired and more WWAN interfaces are needed, we simply add more Soekris boxes or USB hubs.

Tribe is not Horde specific. It can support any Linux application that uses TCP, UDP, or ICMP. Like Horde, Tribe runs entirely in user-space, requiring only a small set of elevated privileges, such as raw socket capabilities.

4.3 Video Subsystem

Our telemedicine application will support multiple cameras and provide a number of image related services including high resolution still images, real-time video, non-real-time video (when network connectivity will not support real-time video), and functions such as pan, tilt and zoom. The most challenging of these is providing high quality real-time video.

We first discuss the basic approach used for video encoding/decoding. The key problem to be solved is robustness in the face of packet losses and limited bandwidth.

Video Encoding/Decoding

Our video streaming application is built using the ffmpeg code-base [8]. ffmpeg is an open-source project that provides a feature-rich audio/video codec library. We use the H264 codec. Several video codecs were empirically evaluated at different frame rates and different bit-rates. H264 was chosen because throughout our experiments

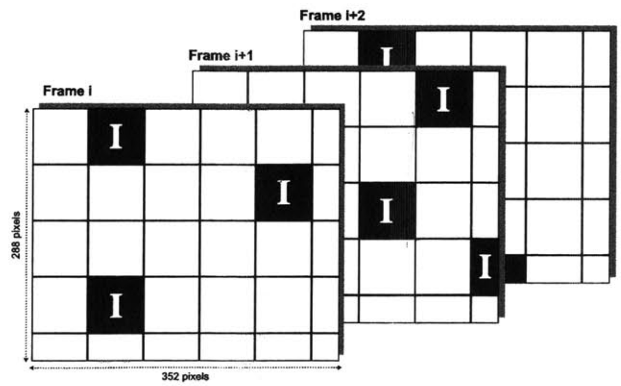


Figure 4-3: A grid of `x264` video encoders is run, with the encoders out of phase with each other. At most three encoders produce I-frames for any given frame. The I-frame pattern is randomly selected.

it was observed to achieve the best quality at low bit-rates. We are using the `x264` encoder [18], which is an open-source implementation of H264.

Unfortunately, the standard `x264` encoder is not ideally suited to support video streaming in the face of restricted bandwidth and packet losses. The I-frames produced by the encoder are quite large and therefore span many packets, even at low bit-rates. For a `CIF`[1] video encoded at $300kbits/sec$, more than sixteen UDP packets may be needed to hold a single I-frame. A single packet loss can corrupt an I-frame. In our experience, the `ffmpeg/x264` decoder does not handle partial I-frames well, and often crashes when asked to decode these partial, or corrupted frames. We do not want to rely on retransmissions to mask losses because of the large packet round-trip-times in our system. Furthermore, given our limited bandwidth, adding forward-error-correction coding is not an appealing option.

To construct a video encoding resilient to packet losses, we use the following approach:

- Each frame is segmented into a grid of subframes. An example grid is shown in figure 4-3. To attain the best compression, the width and height of these subframes must be divisible by 16 because of the macroblock sizes.

- Each subframe is encoded and decoded independently of the other subframes.

[1] 352 pixels wide, 288 pixels high.

This requires grids of `ffmpeg/x264` encoders and decoders.

- Some of the subframes of each frame are intra-coded (i.e., treated as I-frames) and others are encoded as either P-frames or B-frames. In figure 4-3, these P-frames and B-frames are represented by the blank subframes.

The encoded subimages are linearized as follows:

$$..., D_j^{(0,0)}, D_j^{(1,0)}, ..., D_j^{(5,0)}, D_j^{(0,1)}, D_j^{(1,1)}, ..., D_j^{(5,4)}, D_{j+1}^{(0,0)}, ...$$

Where $D_j^{(x,y)}$ is the component for frame j from the encoder at (x, y) in the grid. Each $D_j^{(x,y)}$ is an Application Data Unit (ADU). In Horde, ADUs are the smallest data objects. We use the definition of smallest object to represent subframes. Therefore, the video stream Horde sees consists of the above sequence.

The grid of video encoders is run so that encoders are out of phase with each other. Each encoder begins at a different frame depending on the random seed pattern that was generated. With a constant group of picture (GOP) size, this causes them to operate out of phase with each other. With 8 encoders and a GOP of 10 frames, at most one subframes of the original frame are intra-coded.

By creating more but smaller independently decodable I-frames, this encoding dramatically reduces the number of packets needed to transfer an I-frame component. Most I-frame components fit in a single packet. Consequently, the amount of received data that needs to be discarded because of a packet loss is also dramatically reduced. Furthermore, a packet loss causes a localized corruption in part of the video, rather than corrupting the entire frame, and any other subsequent frames whose decoding depends on the missing data.

The grid approach also eliminates bitrate spikes usually associated with each I-frame. Without grid encoding, during the course of each GOP the network would be overburdened at every I-frame and under utilized at P-frames and B-Frames. This pattern contributes to burst losses on the very packets we value most, those that contain I-Frames. With grid encoding, for every frame, the network has to transfer some I-frame components, some P-frame components, and some B-frame components.

This amortizes the cost of sending an I-frame, smoothing out the transmission rate, improving packetization and decreasing the amount of burst losses.

In order to avoid the correlated losses of I-frame components, we use Horde to tag these components with the same correlation group G_I. Additionally, a Horde loss threshold objective can be used for every ADU in the video stream, in order to avoid high-loss channels.

Network Bandwidth Variation

Because this is a mobile system, the video application must deal with the frequent changes in available bandwidth, as discussed in chapter 2, and still provide near real-time video. Variation is inevitable for a variety of reasons including congestion control, network interface disconnections, vehicular location, and competition with other users for base-station resources.

In an effort to mitigate the impact of variations in bandwidth, we dynamically adapt the encoder's Q parameter [88], which controls the bit-rate at which the video is encoded. This value ranges from 1-51, and the higher the value for Q the higher the compression. So, for a constant frame size, as Q increases, the size of the encoded frame decreases. Each increment of Q corresponds to 12.5% more compression of the frame.

Horde provides feedback to the application telling it the maximum available data transmission rate for the last sending period. Given this information and the encoded size of the previous frame, the video server can attempt to calculate the appropriate Q parameter value at which to encode the video.

Unfortunately, the values of the two main factors that control the success of data transmission are unknown. We only know the bandwidth that was available from the last sending period and the encoded size of the previous frame. We do not know the available bandwidth of the current sending period or the encoded size of the current frame. This presents us with a control loop problem. We want to ensure that the encoded frame size closely matches but does not exceed the available bitrate, but we experience some delay in learning if we achieved our objective.

Therefore we have an additive-increase additive-decrease, AIAD, approach to adjusting the Q parameter that is both responsive to sudden spikes in the encoding rate and drops in the available bandwidth. The algorithm uses two values. The first is the *total encoded frame size* of the current GOP, which is a running summation of the previous encoded frame sizes of the current GOP. The second is the *total target frame size* of the current GOP, which is a summation of the previous target frame sizes. We calculate this by dividing the previous allowed bitrate by the current frame rate. This approach is based on the following five rules, the first two apply per frame, the remaining three apply per GOP.

- To be responsive to spikes in the encoding rate or drops in the network's available bandwidth, for each frame we calculate the difference between the total encoded frame size and the total target frame size. If the difference is greater than 2 KB, then we increase Q to lower the encoded bitrate.

- We also want to be responsive to spikes in the network's available bandwidth. Therefore for each frame, if the total encoded frame size is 4 KB smaller than the total target frame size, we decrease Q to raise the encoded bitrate.

At the end of each GOP the ratio between the total encoded frame size and the total target frame size is calculated. Through empirical observation, we have determined ranges of values where the ratio should be updated.

- If the is less than 0.7, we decrease Q by 2.

- If the is greater than 0.7 and less than 0.9, we decrease Q by 1.

- If the ratio is greater than 1.25 we increase Q by 1.

Adjusting Q appears to be more of an art rather than a science. Many variables, such as the changing bandwidth or the amount of motion in the scene, effect the encoded frame size. In turn, this encoded frame size effects what the congestion control algorithm returns as the allowed bitrate of the network. Ask for too little and the system will go into a negative feedback cycle where it believes that there is very little bandwidth in the network, decreasing the quality of the video. Ask for too much and the network will supply it, at the expense of the loss rate. These values were found through numerous trials. If the ranges are wider, the Q value oscillates wildly lowering the quality of the video. If the ranges are smaller, Q converges to its maximum value.

Multiple Cameras

At the beginning of a session, the */dev* directory is probed for video devices. For each device discovered, that device is initialized and added to a vector of cameras. We allow the user to switch between the multiple cameras at any time and as often as they like. Along with the video stream from the selected camera, we send thumbnail images as a preview of the information that the other cameras contain.

Dynamic Resynchronization

In periods of high loss or high latency, the receiver will consume more frames than it receives from the sender. If this pattern continues the playback buffer at the receiver will completely drain, leaving the user with poor quality video containing motion artifacts and partial frames. At this point we have two options, rebuffering or resynchronization. If we were to rebuffer, we would wait a specified period receiving new data as well as receiving old data that was waylaid due to elevated packet RTTs. This old data could complete frames that were previously only partial filled, marking them as valid frames to be displayed. For example, if the frame rate was $20fps$, and the rebuffering period was until 20 valid frames were received, the user would still expect the same amount of latency, one second, after this rebuffering operation took place. However, if any of the old partial frames were filled in, they would be played back as well once the streaming continued, thus increasing the latency.

Instead of rebuffering, we resynchronize by clearing the queues of the sender and the receiver. Thus discarding old partial frames, whose inclusion would cause the latency to grow unbounded with each rebuffering event. Because this is an interactive system, we have an obligation to the user to not adjust the latency without their consent. Through resynchronization we can ensure that we maintain a constant latency between the sender and the receiver.

If the receiver has experienced a series of empty or partially filled frames it will send a resynchronization request to the sender. When the sender receives a resynchronization request it advances the frame number to a much larger number, so the playback at the receiver will not be affected if there are late arriving or out of order packets with frame numbers larger than the frame that triggered the request. Then the sender clears out its internal video stream queue by draining all of the data from the encoders, selecting a new random seed pattern for the grid and re-starting each encoder on an intra-coded frame. Finally, it sends an acknowledgment to the receiver containing the new frame number. Once the receiver gets the resynchronization acknowledgment from the sender, it updates the frame number and refills the video playback buffer. To the user, the screen remains fixed on the last frame it had displayed and displays a message in the status bar of the player stating that a resynchronization event is taking place.

Buffering

When Tavarua starts, its playback buffer is filled. The amount that it is filled depends on the amount of latency the user can withstand, which is represented in the requested size of the buffer and the maximum buffering time. It stops when the buffer contains enough valid frames or if it has buffered for the maximum time. The buffer is also refilled at every resynchronization request.

We have three modes for the buffer, and we allow the user to switch between these modes. The first is a real-time mode with subsecond latency. This mode has only been evaluated in a stationary setting. The second is a near real-time mode, where the requested buffer size is 2.5 times the current frame rate and the buffering time is 2.5 seconds. The third mode has a much larger buffer, 5.5 times the current frame rate, and a longer buffer time of 5.5 seconds. When latency can be tolerated and the channel is lossy, we allow the user to enable retransmissions in order to mask losses. With retransmissions and a larger buffer, the user can attain near perfect video quality, without experiencing artifacts of motion or viewing partial frames.

4.4 Audio Subsystem

Our telemedicine application must provide a reliable audio communication stream between the mobile ambulance and the stationary hospital site. Compared to the video stream, the audio stream uses significantly less bandwidth, approximately $11kbps$.

We first discuss the basic approach used for audio encoding/decoding. The key problem to be solved is synchronization between the audio and video streams as well as ensuring a reliable audio link between the clients.

Audio Encoding/Decoding

The audio streaming application is built using the `alsa`, Advanced Linux Sound Architecture, library [1]. `alsa` is a user space library that allows us to capture audio streams directly from the hardware devices. For encoding/decoding we use the `speex` code-base, an open-source library that encodes and decodes audio data and is specialized for voice communication.

It is imperative that the audio stream and the video stream be well synchronized, even in the face of packet loss. To ensure that they are always synchronized we have two streams within our Horde session, one for the transfer of video data, one for the transfer of audio data. These streams will capture data at each clock cycle, encode them at send them to the receiver. By teasing apart the streams and capturing data at regular, known intervals we can tag this data so that when it is decoded it can be played back appropriately.

Because we are striping this data across multiple channels, we increase the reliability of the data being sent with respect to disconnections. Even when there are multiple disconnections, if at least one interface is transmitting data the packets will be sent, freed of the dependence on one provider or one interface. To ensure that the audio stream is always transmitted, we use Horde's priority option to give the audio packets a higher priority than all other packets. This guarantees that as long as the available bandwidth is greater than the encoded audio stream, around $11kbps$, all of the audio packets will be sent.

4.5 Telemetry Subsystem

The current devices used by medical transport teams, with whom we are collaborating, to monitor the physiological signals of a patient is a proprietary, file-based system. Therefore, the data stream from the device is not capable of real-time streaming; the patients vital signs are written to a file, which then must be opened, read and sent to the receiver. We chose not to provide an interface to a proprietary system. In this implementation we focus a camera on the screen of this device and transmit the physiological signals as a video stream. While it is an adequate first approximation, admittedly, this is not an acceptable long-term solution.

Chapter 5

Performance Evaluation

The key question addressed in this chapter is, are we capable of streaming diagnostic quality audio and video from a moving vehicle in the face of a variable, lossy network? In addressing this question, we also address a number of subsidiary questions. Will have the same success consistently, absorbing day-to-day fluctuations in the available networks, dynamically adapting to provide high quality audio and video? Is the system is resilient in the presence of hardware failures and service provider failures? How does performance vary with respect to loss rate? Etc.

In section 5.1, we describe a baseline evaluation conducted from a stationary location to evaluate the effects of varying the frame rate and the resolution on the quality of the received video. In section 5.2 we assess the quality of the audio stream, and show that we were able to mask the additional losses that occur when the sender is no longer stationary. In the next two sections we shift our focus to the video stream, examining how well we overcome two factors, packet loss and variable bandwidth, that hinder video quality. We then examine the reliability of the system by running an experiment where we simulate hardware failures and areas of zero network coverage by manually removing interfaces, then re-inserting them while the system is running with no outside intervention save manipulating the physical hardware of the interfaces. Consistency is demonstrated though a set of experiments conducted from a moving vehicle over a period of five days. Seven such trials took place, one a ninety minute drive through a large section of suburban Massachusetts, the other six were loops through Cambridge and downtown Boston. The average duration of these trials was roughly forty minutes.

5.1 Baseline Evaluation

Streaming video on a channel that is lossy and highly variable requires some compromises in the sender side quality to achieve the best results on the receiving end. To better understand this we conducted a series of experiments to select the appropriate resolution and frame rate. These experiments were done in a stationary environment. The resolution experiment was roughly three minutes of video of a moving digital clock and a photograph. The frame rate experiment was also three minutes of video, the first half was lateral motion, swiveling in a chair, and the second half was a simulated stroke scales test, featuring broad movement of the arms in all directions.

Given the low bitrate constraints of our network, a high resolution image passed to the encoder does not translate into a higher quality image at the receiving end. This is due to the encoder being forced to choose which of the bits to keep and which to discard and approximate later. When the capture resolution increased, the peak-signal-to-noise ratio (PSNR) fell. PSNR is a metric of audio and image quality that denotes the difference between two signals, the original uncompressed signal and the one that arrives and is decoded at the receiver. The higher the PSNR value, the more the two signals are alike, implying a higher quality video or audio stream. For a video stream that has been compressed a good range is between $30dB$ and $40dB$. Table 5.1 shows the median, mean and standard deviation for three different resolutions. All resolutions exceeded $40dB$, and there was not a definitive choice as to which was the best. Therefore we capture at 320x224, for our experiments because it performs as well as the slightly higher resolution while allowing us to transport fewer bits.

Table 5.2 shows a striking demarcation between rates up to $18fps$ and rates $20fps$ and above. This demonstrates that not only is $20fps$ perceptually better than lower frame rates because it provides smoother motion, but its PSNR value is also much higher. The smoothness likely helps, as there are smaller changes between frames at

$20fps$ and $18fps$, and therefore dropped subframes will negatively affect the quality $18fps$ video more than the $20fps$. There is not such an improvement when moving from $20fps$ to $25fps$, despite the bandwidth cost of sending an additional five frames every second. This implies diminishing returns on increasing the smoothness of the video. Therefore we stream our video data at $20fps$

Resolution	Median	Mean	StDev
320 x 224	45.016	44.432	2.621
352 x 288	45.226	44.923	1.112
480 x 360	42.149	42.152	1.035

Table 5.1: Median, mean, and standard deviation in PSNR values for the different capture resolutions

Frame Rate	Median	Mean
$12fps$	41.043	41.483
$18fps$	40.260	41.402
$20fps$	45.016	44.432
$25fps$	45.783	45.370

Table 5.2: Median and mean PSNR values for the different frame rates. The standard deviation is not shown as the distribution is not Gaussian.

5.2 Audio Evaluation

Akin to the video stream, the audio stream is also very sensitive to packet loss. Figure 5-1 is a segment of the PSNR for the audio stream during a stationary experiment. The sharp dip in the graph illustrates the effect of a short burst loss on the quality of the received audio stream, as noted by the drastic decline in the PSNR.

Table 5.3 summarizes the results of our audio subsystem. These results show that the measures we took to mitigate the effects of loss in chapter 4 were very effective as the median PSNR for the stationary and for the motion experiments are nearly identical. The two trials having a similar PSNR distribution indicates that we do quite well in handling additional losses, and only fail when there is a burst loss where no packets make it to the receiver.

Due to the bi-modal pattern of speech, silence or vocalization, the distribution of the PSNR values is not Gaussian. Also, because the standard deviation was extremely high with respect to the mean, we are further convinced that PSNR is not an adequate measurement. A different metric, that values word clarity over simple signal clarity should be used instead.

Because we used a low quality microphone headset and we sample at $8000Hz$, our PSNR values are much lower than the quality of a compact disc or the radio. Simply put, every word spoken on the sender can be deciphered by the listener at the receiver. Some words are clipped, some are convoluted with noise, but all are intelligible. Currently it is consistently of the quality of a cell phone conversation where both parties are located in a good coverage area. However, because the PSNR values were similar in both trials it indicates that the quality of the equipment and the low sampling rate, not the loss rate, are the main reasons that we experience a low PSNR value for our audio stream.

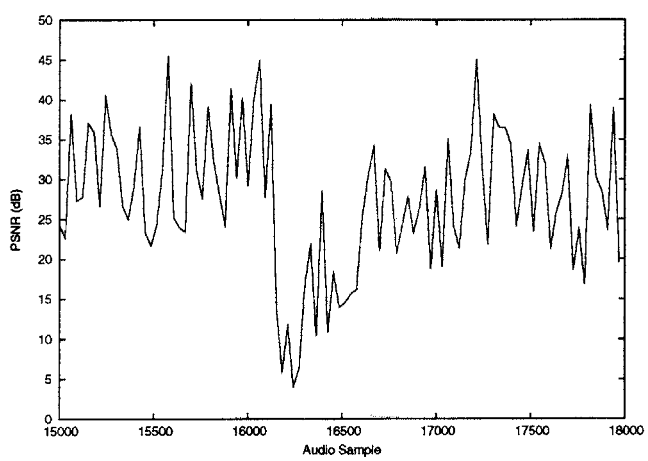


Figure 5-1: The impact of a packet burst loss on audio PSNR.

Trial	Median	Mean	StDev
Stationary	25.296	25.571	7.633
In Motion	24.759	25.205	7.346

Table 5.3: Median, mean, and standard deviation in PSNR values for the two audio trials.

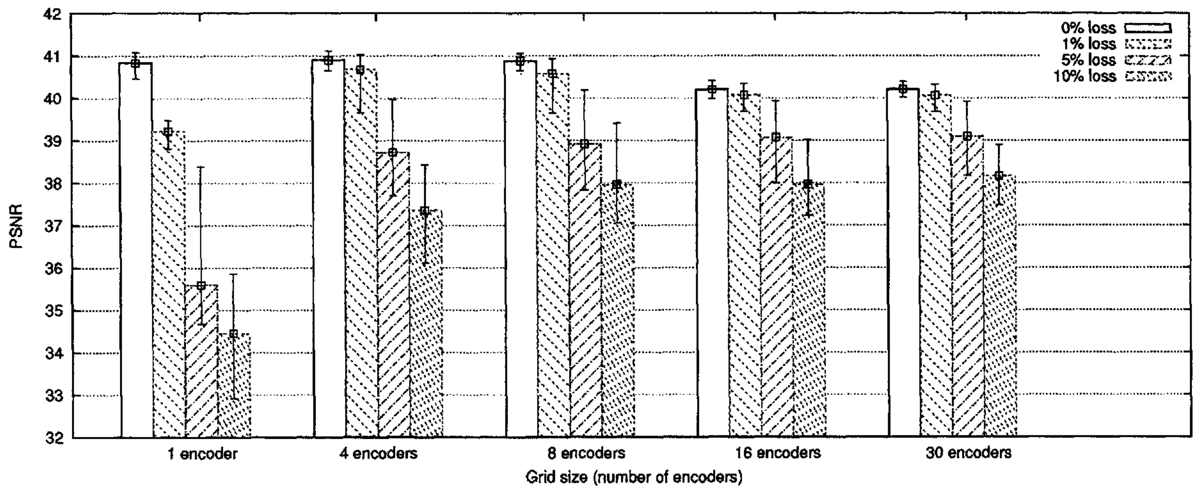


Figure 5-2: The impact of packet loss rates on the peak-signal-to-noise-ratio (PSNR) of decoded video for different types of grids.

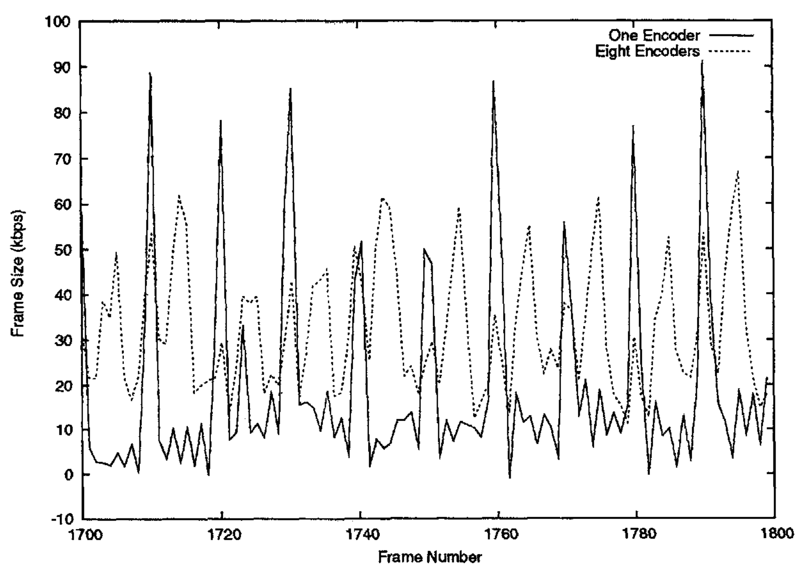


Figure 5-3: The total image frame size for a grid of one encoder and for a grid of eight encoders. The spikes that occur on the one encoder correlate with I-frames.

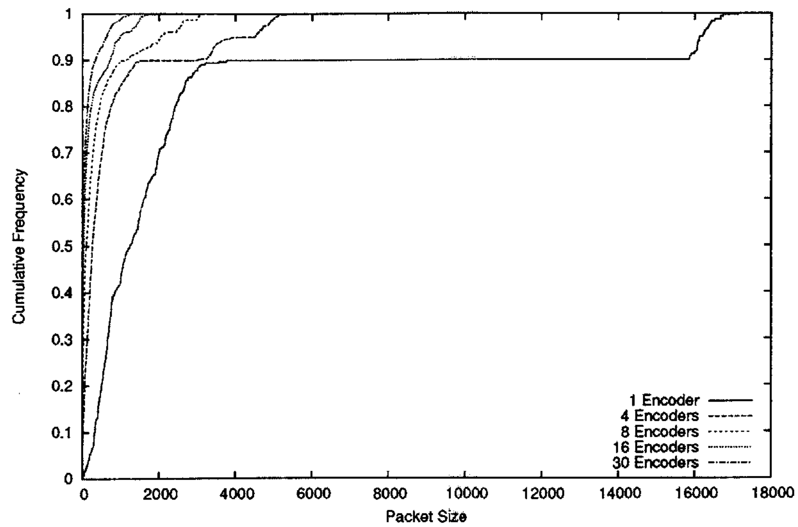


Figure 5-4: The distributions of ADU sizes for different types of grids.

5.3 Packet Loss Resilience

When transmitting encoded video over a network, packet losses can cause corruptions in the decoded video. We argued in chapter 4 that our grid-encoder would be resilient to packet losses. In this section we provide evidence for this claim by evaluating the impact of simulated packet losses.

We started by selecting five grid sizes: a single encoder; four encoders; eight encoders; sixteen encoders; and thirty encoders. This was done to gauge the performance of different grid sizes. For each grid size, we ran multiple experiments with simulated random packet losses. `silent`, a video segment chosen from the standard video repository, was encoded at a constant bitrate of around 500*kbps* and we simulated losses at 1%, 5% and 10% of total packets. Figure 5-2 shows the results of our experiments, showing the PSNR for the decoded video in each experiment. This experiment was conducted at an early point in our development. Our current PSNR's are considerably better than these but the relationships depicted in figure 5-2 still hold.

When there are no packet losses, the 4-encoder and 8-encoder grids performed about the same as the single encoder. The 16-encoder and 30-encoder grids performed slightly worse. The problem is that working with smaller subframes adversely impacts compression. Therefore, when moving from the 8-encoder grid to the 16-encoder grid, we had to reduce the encoders' Q, in order to keep the bitrate roughly constant, leading to the signal degradation.

When a single encoder was used, increasing packet losses caused a sharp decrease in PSNR. In this case, when part of an I-frame was lost, all the packets for that I-frame (as many as 18) had to be discarded. This also caused subsequent frames dependent on that I-frame to become corrupted.

The 4-encoder case demonstrates that even a grid composed of relatively large sub-images yields a dramatic improvement. In this case, 1% packet losses did not significantly impact video quality.

Moving from four encoders to eight yields a less dramatic improvement. Adding even more encoders results in diminishing returns.

The quality improvement provided by the grid-encoder is due to several factors. First, smaller ADU's span fewer packets, so a single packet loss does not cause a large number of already received bytes to be discarded. Second, losing packets causes localized image corruptions, rather than corrupting entire frames. For instance, losing a packet from a 8-encoder grid's video stream would at most corrupt one of the subframes, leaving the rest intact. Losing a packet from a single encoder's video stream would corrupt the entire frame. Finally, because the grids are encoding frames out of phase, not all encoders in the grid are transmitting I-frames simultaneously. Figure 5-3 shows how increasing the number of encoders can spread the cost of sending an I-frame across an entire GOP by decreasing the size of each frame and thus smoothing out the send rate.

Figure 5-4 shows the distribution of ADU sizes for each type of grid. Recall that in our video stream ADU's are encoded frames. Note the dramatic difference between the single encoder and 4-encoder cases. In contrast, all multi-encoder grids have relatively similar distributions.

5.4 Dynamic Network Adaptation

WWAN bandwidth is highly variable. If one were always to encode the video at the same rate, the video stream would experience a large amount of losses in periods where the network was behaving poorly. Conversely, if the network was behaving well the system would not be optimally utilizing the bandwidth from the network, thus settling for lower quality video. Moreover, frames that contain more motion cause an increase in the encoding rate. Since the exact behavior of the network and the size of each frame are unknown, it is not possible to pick an optimal encoding rate *a priori*.

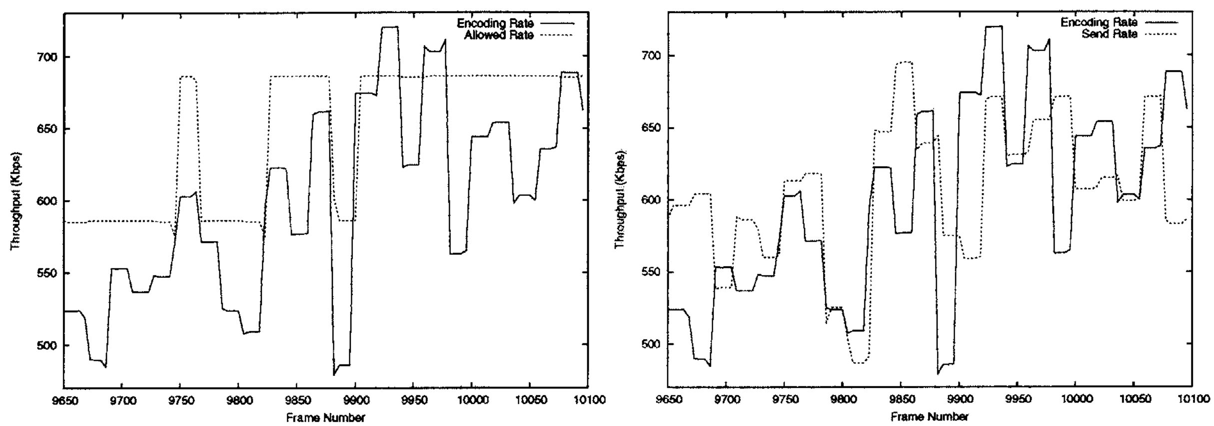
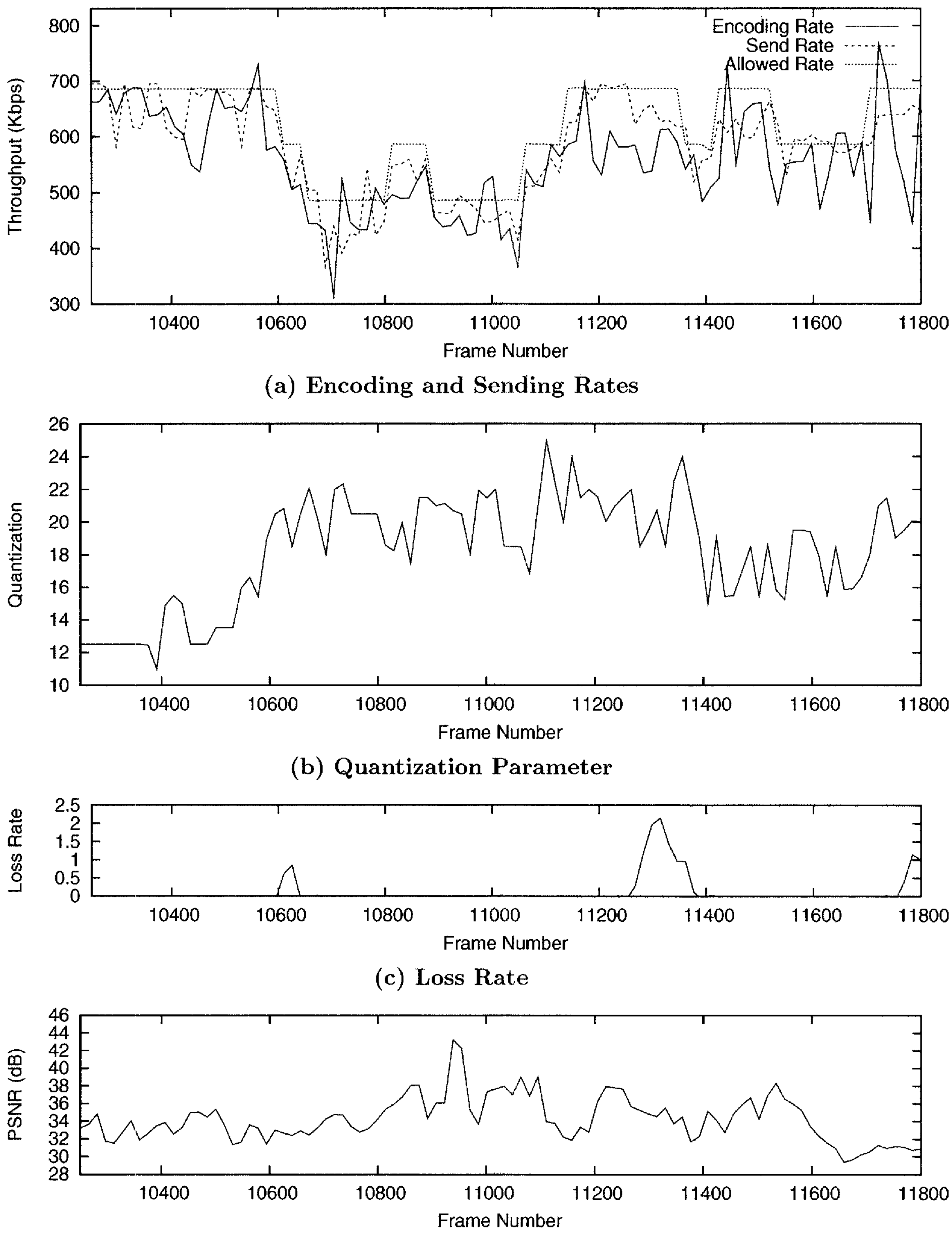


Figure 5-5: The figure on the left compares the bitrate at which the video was encoded to the feedback from the network of how much we were allowed to send. The goal is to consistently keep a buffer between the encoding rate and the allowed rate, which will probe for more bandwidth but not incur losses. Because of buffering, the encoding rate can exceed the allowed rate, as occurs around frame 9925, if only for a short duration. The figure on the right compares the encoded bitrate and the send rate. Prolonged periods where the encoding rate is above the send rate is indicative of non-transmitted, and thus not received, data.

Therefore we select our encoding rate dynamically, and leave a buffer between the encoding rate and the current allowed rate of the network in order to absorb sudden spikes in the size of the frames or drops in the available bandwidth of the network. Figure 5-5 illustrates this idea of a maintaining a gap to absorb spikes while consistently keeping the encoding rate below the allowed bitrate.

Figure 5-6 serves as a demonstration of how Tavarua reacts to normal bandwidth fluctuations. As described in chapter 4, the encoding rate is adjusted by applying a set of rules based on comparisons between the actual frame sizes and a calculated target frame size. We note that even though the throughput dropped close to $400kbps$ over the span of 100 frames, during that period the PSNR was above $32dB$ and the loss rate never exceeded 1%. This demonstrates that even with highly variable bandwidth, we still provide diagnostic quality video.



(a) **Encoding and Sending Rates**

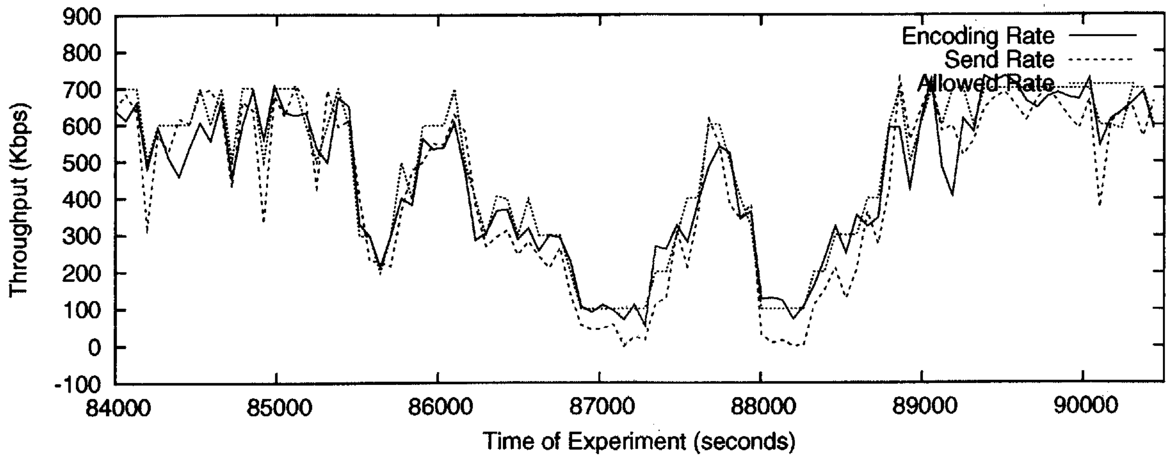
(b) **Quantization Parameter**

(c) **Loss Rate**

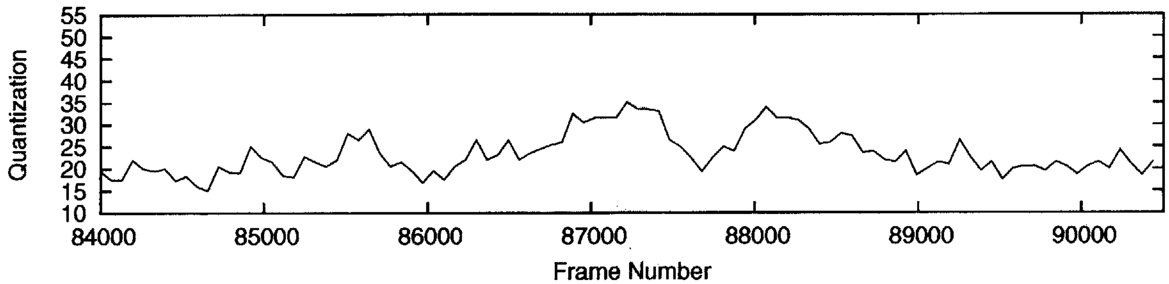
(d) **PSNR**

Figure 5-6: An example of Tavarua's adaptations to typical network fluctuations. Due to buffering, compare the frame numbers in the network graphs to the PSNR value approximately 100 frames to the right.

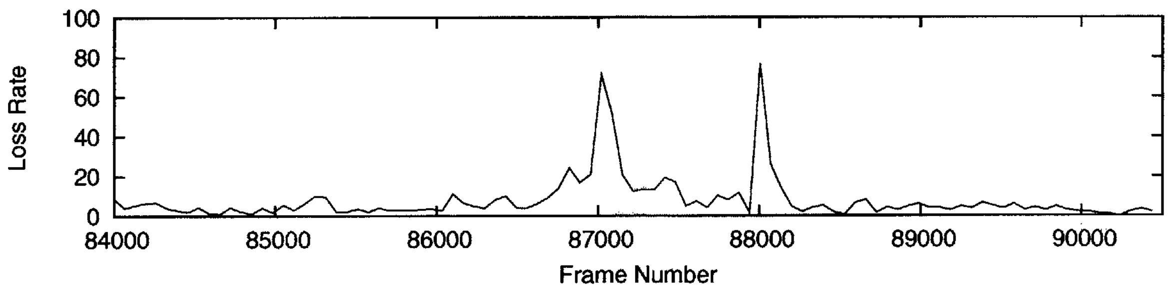
Because ambulances travel through tunnels and on bridges, we examine the quality of the video in a period of drastic bandwidth fluctuation. Figure 5-7 is a segment prior to, during, and exiting a tunnel. Despite elevated loss rates, one with a short peak at just below 80%, the video quality remained high even during the period inside the tunnel, remaining above $32.5dB$. Therefore even in periods of high sustained loss and where the bandwidth drops quickly and without warning our video is still diagnostic quality.



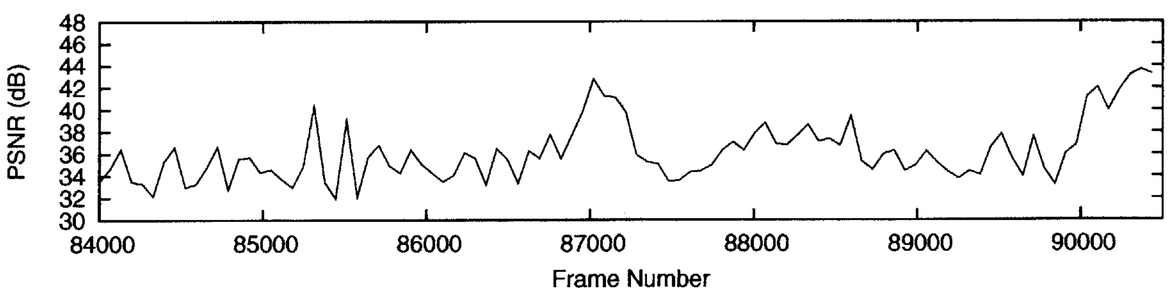
(a) **Encoding and Sending Rates**



(b) **Quantization Parameter**



(c) **Loss Rate**

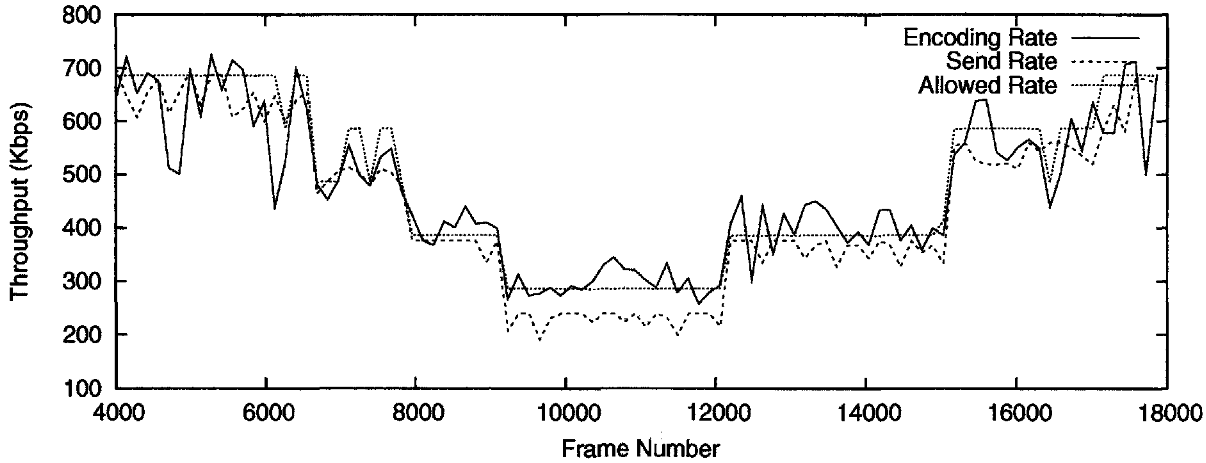


(d) **PSNR**

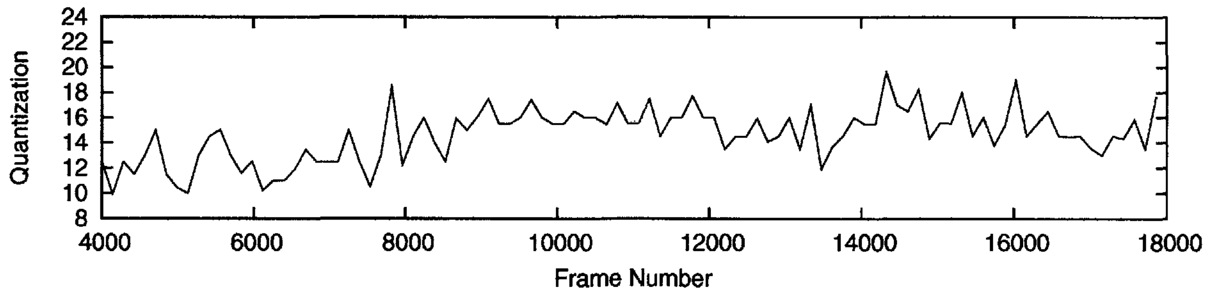
Figure 5-7: An example of Tavarua's adaptations to drastic network fluctuations, e.g. entering a tunnel. Due to buffering, compare the frame numbers in the network graphs to the PSNR value approximately 100 frames to the right.

5.5 Reliability and Consistency

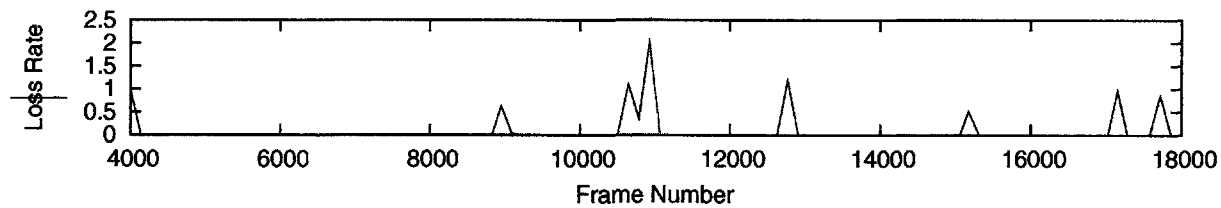
Over an extended period of time hardware will fail or the sender will enter network areas beyond where a particular cellular provider guarantees coverage. Therefore Tavarua must be able to cope with interfaces disconnecting, as well as the possibility of them attempting to reconnect at a later time. To simulate these events we conducted an experiment from a stationary location where we systematically removed all locally connected interfaces one at a time. Once they were removed each was added back one at a time. Currently, Tribe does not support this feature for the `TUN` interfaces, so the two PCMCIA interfaces remained active throughout the experiment. Figure 5-8 shows the effects of such an experiment on the loss rate and the PSNR as well as how well the encoding rate is updated to account for sudden drops in bandwidth. Throughout the experiment the PSNR value is consistently above $40dB$. This is due to timely updating of the Q parameter, witnessed by the slight spike in Q at around frame 8000 when the second interface is disconnected.



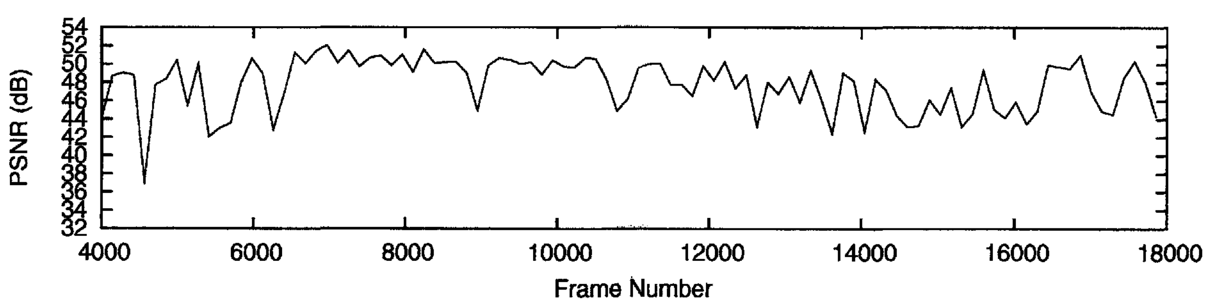
(a) **Encoding and Sending Rates**



(b) **Quantization Parameter**



(c) **Loss Rate**



(d) **PSNR**

Figure 5-8: An example of Tavarua's adaptations to hardware failures. In this experiment, three of the interfaces were sequentially removed, then re-inserted. Due to buffering, compare the frame numbers in the network graphs to the PSNR value approximately 100 frames to the right.

Tavarua must consistently provide diagnostic quality audio and video streams. The other experiments have shown specific cases where the system copes with high loss, hardware failures, and a highly variable network, as well as overcomes drastic variations, e.g. entering a tunnel. To prove that this system can achieve these results on a consistent basis, without crashes and without any engineering oversight beyond starting up the sender and the receiver applications, we ran seven trials over the course of five days. The results of those trials are summarized in table 5.4 showing the median, mean, and standard deviation of the PSNR for each of the trials as well as the median loss rates. Throughput distributions over the seven trials are shown in figure 5-9. From table 5.4 and figure 5-9, it appears that the most dominant factor in achieving high PSNR values is to minimize the loss rate rather than maximize the throughput. The throughput in **Boston-3** far exceeded the other trials, however so did its median loss rate, resulting in the lowest PSNR value.

The longest trial, referred to as **Suburbs**, was 90 minutes in length covering a geographic area from Massachusetts General Hospital through areas of suburban Massachusetts. We chose this route because many of the suburban areas we drove through, such as Lynn, transport their trauma patients to Massachusetts General Hospital, a Level 1 trauma center. Additionally this route was selected because it included both bridges and tunnels as well as a stretch of highway. The highway portion was able to satisfy our curiosity that Tavarua does indeed perform well at high speeds. No abnormal drops in throughput or increases in loss rate occurred when the vehicle traveled at speeds of $70mph$ for an extended period of time.

The remaining six trials were loops from Cambridge though an urban location, Boston's Back Bay neighborhood. The average speed was much lower, but these experiments were conducted through a wide variation in weather patterns, from clear to cloudy to raining, none of which had an impact on the throughput or the loss rates. All of these trials exceeded the $30dB$ PSNR expectation for diagnostic quality, proving that we can perform consistently well and reliably no matter the conditions.

Trial	Median PSNR	Mean PSNR	StDev PSNR	Median Loss Rate
Suburbs	35.046	35.258	2.469	2.934
Boston 1	35.558	35.727	2.105	2.005
Boston 2	36.170	36.230	2.634	2.001
Boston 3	33.716	34.271	2.290	4.056
Boston 4	35.552	35.975	2.635	< 1
Boston 5	36.670	37.814	4.509	< 1
Boston 6	36.650	36.550	2.895	< 1

Table 5.4: Median, mean, and standard deviation in PSNR values and the median percent loss rate for all of the trials.

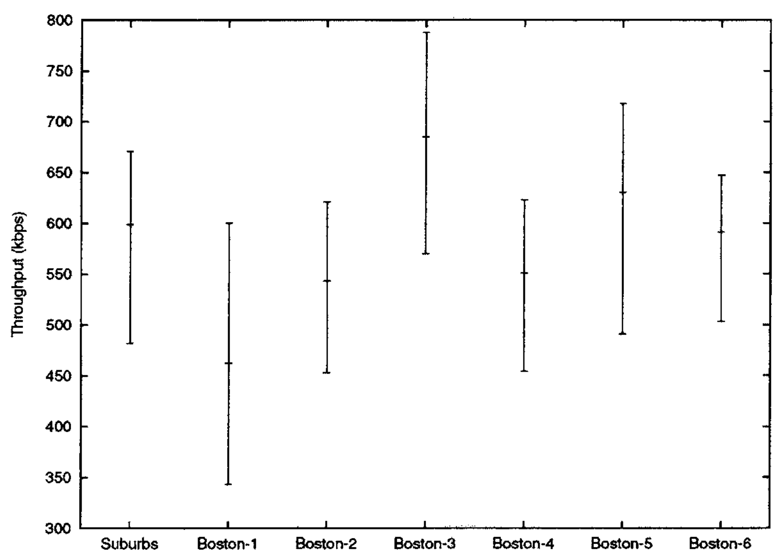


Figure 5-9: Upload throughput distributions in seven experimental data sets. The graph shows the median, upper and lower quartiles. Throughput was averaged over two-second windows.

5.6 Subsecond Latency

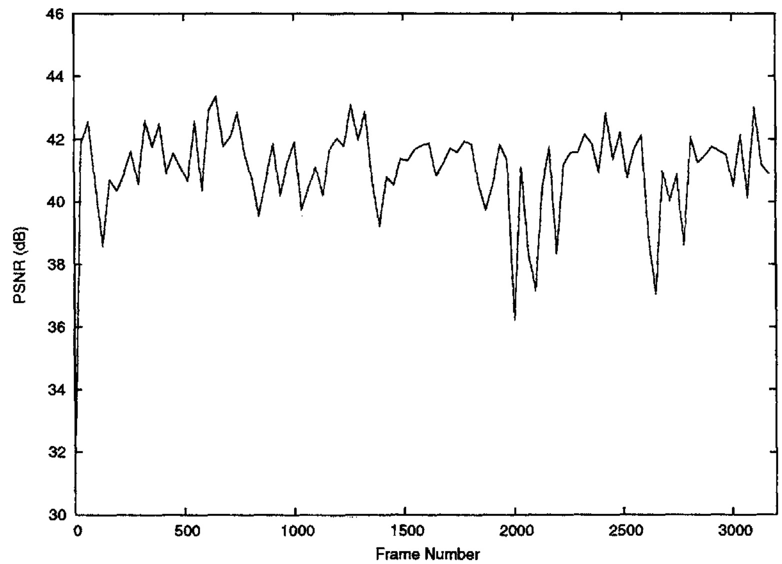


Figure 5-10: PSNR values for a stationary, subsecond experiment.

Ideally, the system would be real-time, with latencies less than one second, and to a first approximation in a stationary setting, we have achieved this with a median PSNR of $41.271dB$. We integrated this into our system by decreasing the buffer size and throttling the congestion control back aggressively, capping it at $200kbps$. Figure 5-10 shows the PSNR for a stationary experiment with a latency consistently less than one second. Dips in the PSNR correspond to bursts of *artifacts*, which are subframes that have not received the current frame either due to a packet loss or because the packet RTTs was longer than the latency we were able to tolerate.

When the rate was capped much lower, $100kbps$, *artifacts* still appeared. This leads us to believe that the loss rate is not the cause of these *artifacts*, and that it is likely due to elevated packet RTTs. When the vehicle is in motion, the packet RTTs are longer than when the vehicle is stationary. For subsecond latency to work effectively while the vehicle is in motion, packet level analysis needs to be done to determine the nature of these elevated rates.

Chapter 6

Conclusion

We first summarize the contributions presented in this thesis, and then present some directions for future work.

6.1 Summary

We have presented Tavarua, a mobile telemedicine system that utilizes WWAN striping and complimentary techniques to reliably and consistently transmit diagnostic quality video, with median PSNR values that range from $33.716dB$ to $36.670dB$, and serviceable audio.

WWAN Study In order to design our system properly we conducted a battery of experiments to characterize the behavior of 3G cellular WWANs. These experiments spanned several days and multiple cities. While not rigorous enough to draw broad conclusions, these data sets are extensive enough to formulate a high-level characterization of these networks, and contain information not previously available through public sources.

Infrastructure We have built an infrastructure upon which the potential usability of mobile data streaming applications can be studied in real-world environments. Our design is flexible and modular, additional hardware, e.g., cameras, microphones, or

WWAN interfaces, can be integrated easily.

Dynamic Adjustment of Encoding Rate WWAN bandwidth is highly variable and the size of encoded frames can vary, e.g., moving from a high motion scene to a low motion scene. For these reasons we dynamically adjust the rate at which we encode the video in order to maximize our use of the available bandwidth, without incurring losses by overburdening the channel. Even when the bandwidth drops drastically, e.g. entering a tunnel, and the loss rates spike up to 80%, our PSNR never drops below $32.5dB$

Grid Encoding Video streams are incredibly bursty, with large I-frames that span up to 18 packets followed by much smaller P-frames. Moreover, if one of the I-frame packets were lost then the frame would be corrupted and the entire group of pictures (GOP), roughly half a second of video, would be discarded. By segmenting the frame into smaller, independent subframes and initializing these encoders out of phase, we can smooth out the sending rate by amortizing the cost of sending an I-frame over an entire GOP. Also, because these subframes are independent, corruptions are localized to the subframe where the loss was experienced, ensuring that single or small burst losses do not cause us to discard an entire GOP. Because of that we are able to sustain loss rates of around 1% without a significant decline in quality, and as shown in the evaluation chapter, last three trials had median loss rates below one percent once the congestion control was fully refined for the Rev-A interfaces.

6.2 Future Work

While Tavarua provides the infrastructure for a mobile telemedicine system, there are areas that need to be explored before it is deployable.

Audio Quality As noted in the system evaluation, the audio quality was not up to our expectations. This was not because of packet losses or bandwidth issues, but rather because of choices of equipment and sampling rate. Experimentation with various microphones as well as sampling rates to determine the optimal configuration should result in a large boost in audio quality.

Telemetry Subsystem To extract raw, real-time physiological signals from the devices on the ambulance would be of great benefit to the physicians. Currently the device does not lend itself to this application, though one could imagine that if Tavarua were to be adopted an additional device that does provide real-time streaming could be included as well.

Bi-Directional Video Recently, our colleagues at Massachusetts General Hospital, impressed upon us the need for bi-directional video. Giving the physician the ability to gesture and demonstrate would make for a richer communication. Often, nuances in how a task should be performed are easily shown, yet are difficult to communicate through voice alone. This video stream will be able to take advantage of the fully provisioned downlink channel, which provides much higher throughput.

User Interface Currently, the user interface consists of a video player window that in addition to the video and audio streams displays system statistics. Key bindings to switch frame rates, select cameras, and choose the buffer size are the only external controls. As it is, this is not acceptable for use by physicians. There needs to be collaboration with physicians to assess their needs and develop a design that allows this system to be an intuitive extension of their abilities.

Subsecond Latency The most technically challenging as well as the most useful element of future work is to reduce the latency of the system to the subsecond level when the sender is not stationary. We determined that the key difficulty is likely high packet RTTs rather than an effect of the loss rate. Statistical analysis on the characteristic behavior of these elevated packet RTT needs to be conducted to develop techniques to circumvent those issues.

6.3 Conclusions

We have shown that it is possible to stream high quality audio and video from a mobile unit under a variety of network conditions, and at different levels of latency. Because of Tavarua's modular, extensible design, we know that improvements in the cell phone networks will result in direct improvements of our streaming capabilities as evidenced by our own recent upgrade of two interfaces from EV-DO to Rev-A.

A system that is this reliable and that consistently provides high quality data streams can be utilized in many different domains outside of the realm of telemedicine. Moreover, even with cobbled together hardware, all the components of the Tavarua sender could fit into a briefcase or a backpack. The ability to portably stream data from any location covered by a cellular network opens the door to many applications. For example, mobile news reporting, mobile tele-conferencing, or disaster response teams could also utilize this technology to stream live visual information back to the central command center.

Bibliography

[1] Advanced Linux Sound Architecture.
http://www.alsa-project.org/.

[2] Advantages of CDMA2000.
http://www.cdg.org/technology/3g/advantages_cdma2000.asp.

[3] American Telemedicine Association.
http://www.atmeda.org/.

[4] ASTRA project.
http://nms.lcs.mit.edu/.

[5] BARWAN: the Bay-Area Wireless Network.
http://daedalus.cs.berkeley.edu/.

[6] Cingular Wireless.
http://www.cingular.com.

[7] FCC: ULS Daily Transaction Files.
http://wireless.fcc.gov/cgi-bin/wtb-transactions.pl.

[8] FFmpeg Multimedia System.
http://ffmpeg.sourceforge.net.

[9] HIPAA Compliance Standards.
http://www.hhs.gov/ocr/hipaa.

[10] Massachusetts General Hospital Stroke Service.
http://www.stopstroke.org.

[11] MIT class: Discrete Time Signal Processing.
http://web.mit.edu/6.344/www/.

[12] MPLAYER: the linux video player.
http://www.mplayerhq.hu/.

[13] Sitefinder Mobile Phone Base Station Database.
http://www.sitefinder.radio.gov.uk/.

[14] Soekris Engineering. http://www.soekris.com.

[15] Sprint Wireless.
http://www.sprint.com.

[16] Verizon Wireless.
http://www.verizonwireless.com.

[17] Video Test Sequences.
http://media.xiph.org/video/derf/.

[18] x264.
http://developers.videolan.org/x264.html.

[19] Hari Adiseshu, Guru M. Parulkar, and George Varghese. "A Reliable and Scalable Striping Protocol". In *SIGCOMM*, pages 131–141, 1996.

[20] D. Andersen, D. Bansal, D. Curtis, S. Seshan, and H. Balakrishnan. "System support for bandwidth management and content adaptation in Internet applications". In *Proceedings of 4th Symposium on Operating Systems Design and Implementation, USENIX*, pages 213–226, October 2000.

[21] J. Apostolopoulos and S. Wee. "Unbalanced Multiple Description Video Communication using Path Diversity".

[22] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz. "A comparison of mechanisms for improving TCP performance over wireless links". *IEEE/ACM Transactions on Networking*, 5(6):756–769, 1997.

[23] Deepak Bansal and Hari Balakrishnan. Binomial Congestion Control Algorithms. In *IEEE Infocom 2001*, Anchorage, AK, April 2001.

[24] Ali Begen, Yucel Altunbasak, and Ozlem Ergun. "Multi-path Selection for Multiple Description Encoded Video Streaming". In *IEEE ICC*, 2003.

[25] Qi Bi and Stan Vitebsky. Performance analysis of 3G-1X EVDO high data rate system. In *IEEE Wireless Communications and Networking Conference*, pages 389–395, March 2002.

[26] James C. Brailean, Richard P. Kleihorst, Serafim Efstratiadis, Aggelos K. Katsaggelos, and Reginald L. Lagendijk. "Noise Reduction Filters for Dynamic Image Sequences: A Review". In *Proceedings of the IEEE*, pages 1272–1292, September 1995.

[27] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP vegas: New techniques for congestion detection and avoidance. In *SIGCOMM*, pages 24–35, 1994.

[28] Joel Cartwright. "GPRS Link Characterisation". http://www.cl.cam.ac.uk/users/rc277/linkchar.html.

[29] R. Chakravorty, J. Cartwright, and I. Pratt. "Practical Experience With TCP over GPRS". In *IEEE GLOBECOM*, 2002.

[30] Benjamin A. Chambers. "The Grid Roofnet: a Rooftop Ad Hoc Wireless Network". http://citeseer.nj.nec.com/chambers02grid.html.

[31] Mun Choon Chan and Ramachandran Ramjee. Tcp/ip performance over 3g wireless links with rate and delay variation. In *MobiCom '02: Proceedings of the*

8th annual international conference on Mobile computing and networking, pages 71–82, New York, NY, USA, 2002. ACM Press.

[32] Julian Chesterfield, Rajiv Chakravorty, Ian Pratt, Suman Banerjee, and Pablo Rodriguez. "Exploiting Diversity to Enhance Multimedia Streaming over Cellular Links". In *IEEE INFOCOM*, March 2005.

[33] Gene Cheung, Puneet Sharma, and Sung-Ju Lee. "Striping Delay-sensitive Packets over Multiple Burst-loss Channels with Random Delays". In *IEEE International Symposium on Multimedia*, 2005.

[34] Y. Chu and A. Ganz. "A Mobile Teletrauma System Using 3G Networks". In *IEEE Transactions on Information Technology in Biomedicine*, volume 8, December 2004.

[35] D. Clark and D. Tennenhouse. "Architectural Consideration for a New Generation of Protocols". In *ACM SIGCOMM*, 1990.

[36] A.J. Crawford, H. Denman, F. Kelly, F. Pitie, and A.C. Kokaram. "Gradient based dominant motion estimation with integral projections for real time video stabilisation". In *International Conference on Image Processing*, pages 3371–3374, October 2004.

[37] J. Cullen, W. Gaasch, D. Gagliano, J. Goins, and R. Gunawardane. "Wireless Mobile Telemedicine: En-route Transmission with Dynamic quality-of-service Management". In *TRW, Inc.*, 2001.

[38] G. R. Curry and N. Harrop. "The Lancashire Telemedicine Ambulance". In *Telemedicine and Telecare*, volume 4, pages 231–238, 1998.

[39] Constantinos Dovrolis, Parameswaran Ramanathan, and David Moore. "Packet-dispersion techniques and a capacity-estimation methodology". *IEEE/ACM Trans. Netw.*, 12(6):963–977, 2004.

[40] J. Duncanson. "Inverse Multiplexing". *IEEE Communications Magazine*, pages 34–41, April 1994.

[41] L. Eledath, J.and McDowell, M. Hansen, L. Wixson, A. Pope, and G. Gendel. "Real-time fixation, mosaic construction and moving object detection from a moving camera". In *Fourth IEEE Workshop on Applications of Computer Vision*, pages 284–285, October 1998.

[42] M. Elena, J. M. Quero, C. L. Tarrida, and L. G. Franquelo. "Design of a Mobile Telecardiology System Using GPRS/GSM Technology". In *EMBS/BMES*, October 2002.

[43] Dawson R. Engler, M. Frans Kaashoek, and James O'Toole Jr. "Exokernel: an operating system architecture for application level resource management". 1995.

[44] Gilhousen K.S et al. On the capacity of a cellular CDMA system. In *IEEE Transactions on Vehicular Technology*, pages 303–312, May 1991.

[45] N. Feamster and H. Balakrishnan. "Packet Loss Recovery for Streaming Video". In *the 12th International Packet Video Workshop*, April 2002.

[46] Nicholas G. Feamster. "Adaptive Delivery of Real-Time Streaming Video". citeseer.ist.psu.edu/443706.html.

[47] Nick Feamster. Adaptive Delivery of Real-Time Streaming Video. MEng Thesis, Massachusetts Institute of Technology, May 2001.

[48] P. Fredette. "The Past, Present, and Future of Inverse Multiplexing". *IEEE Communications Magazine*, pages 42–46, April 1994.

[49] D. M. Gagliano. "Final Report: Mobile Telemedicine Testbed Health Applications for the National Information Infrastructure". In *National Library of Medicine*, 1998. Project N0-1-LM-6-3541.

[50] Vijay K. Garg. IS-95 CDMA and cdma 2000: Cellular/PCS Systems Implementation. Prentice Hall.

[51] Vijay K. Garg. "Wireless Network Evolution: 2G to 3G". Prentice Hall Communications.

[52] V. Garshnek and F. Burkle. "Telemedicine Applied to Disaster Medicine and Humanitarian Response: History and Future". In *32nd Hawaii Int. Conf. System Sciences*, 1999.

[53] Ekram Hossain and Nadim Parvez. Enhancing tcp performance in wide-area cellular wireless networks: transport level approaches. pages 241–289, 2004.

[54] H. Hsieh, K. Kim, and R. Sivakumar. "a Transport Layer Approach for Achieving Aggregate Bandwidths on Multi-homed Hosts". In *ACM MOBICOM*, pages 83–94, 2002.

[55] H. Hsieh, K. Kim, Y. Zhu, and R. Sivakumar. "A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces. In *Mobicom*, pages 1–15, 2003.

[56] R. H. Istepanian. "Performance Analysis of a GSM-based Mobile Telecardiology System". In *Telemedicine and Telecare*, volume 5, pages 129–130, 1999.

[57] M. Jain and C. Dovrolis. "End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput", 2002.

[58] P. Johnson, D. C. Andrews, S. Wells, S. de Lusignan, J. Robinson, and M. Vandenburg. "The Use of a New Continuous Wireless Cardiorespitory Telemonitoring System by Elderly Patients at Home". In *Telemedicine and Telecare*, volume 7, pages 76–77, 2001.

[59] D. Katabi, M. Handley, and C. Rohrs. "Internet Congestion Control for Future High Bandwidth-Delay Product Environments", August 2002.

[60] E. Kyriacou, S. Pavlopoulos, A. Berler, M. Neophytou, A. Bourka, A. Georgoulas, A. Anagnostaki, D. Karayiannis, C. Schizas, C. Pattichis, A. Andreou,

and D. Koutsouris. "Multi-purpose HealthCare Telemedicine Systems with Mobile Communication Link Support". In *Biomedical Engineering Online*, volume 2, 2003.

[61] M. P. LaMonte, Y. Xiao, P. Hu, D. Gagliano, M. N. Bahmouth, R. D. Gunawardane, C. F. Mackenzie, W. R. Gaasch, and J. Cullen. "Shortening time to stroke treatment using ambulance telemedicine: TeleBAT". *Journal of Stroke*, 2004.

[62] W.C.Y. Lee and D.J.Y. Lee. CDMA system capacity analysis. In *IEEE Transactions on Personal, Indoor and Mobile Radio Communications*, September 2001.

[63] Youngseok Lee. Measured tcp performance in cdma 1xev-do network. In *Passive and Active Measurement Conference*, 2006.

[64] Zhuyu Lei, D.J. Goodman, and N.B. Mandayam. Location-dependent other-cell interference and its effect on the uplink capacity of a cellular CDMA system. In *IEEE Transactions on Vehicular Technology*, pages 2164–2168, July 1999.

[65] Ellen J. MacKenzie, David B. Hoyt, John C. Sacra, Gregory J. Jurkovich, Anthony R. Carlini, Sandra D. Teitelbaum, and Jr Teter, Harry. National Inventory of Hospital Trauma Centers. *JAMA*, 289(12):1515–1522, 2003.

[66] L. Magalhaes and R. Kravets. "MMTP: Multimedia multiplexing transport protocol", 2001.

[67] Luiz Magalhaes and Robin Kravets. "Transport Level Mechanisms for Bandwidth Aggregation on Mobile Hosts". In *ICNP*, 2001.

[68] Vladimir Marbukh. A framework for performance evaluation and optimization of an emerging multimedia ds-cdma network. In *WOWMOM '01: Proceedings of the 4th ACM international workshop on Wireless mobile multimedia*, pages 55–64, New York, NY, USA, 2001. ACM Press.

[69] Yasuyuki Matsushita, Eyal Ofek, Xiaoou Tang, and Heung-Yeung Shum. "Full-frame video stabilization". In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 20–25, June 2005.

[70] G. G. Mendoza and B.Q. Tran. "In-home Wireless Monitoring of Physiological Data for Heart Failure Patients". In *EMBS/BMES*, volume 3, pages 1849–1850, October 2002.

[71] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall. "MPEG Video Compression Standard". 1996. Chapman and Hall, International Thomson Publishing.

[72] J. Navein, A. Fisher, J. Geiling, D. Richards, J. Roller, and J. Hagmann. "Portable Satellite Telemedicine in Practice". In *Telemdicine and Telecare*, volume 4, pages 25–28, 1998.

[73] AM Oakley, MB Duffill, and P. Reeve. "Practising dermatology via telemedicine". In *The New Zealand Medical Journal*, pages 296–299, August 1998.

[74] O. Orlov, D. Drozdov, C. Doarn, and R. Merrell. "Wireless ECG Monitoring by Telephone". In *Telemedicine Journal of e-Health*, volume 7, pages 33–38, 2001.

[75] R. Parry. CDMA200 1xEV-DO. In *IEEE Potentials*, pages 10–13, October/November 2002.

[76] C. Pattichis, E. Kyriacou, S. Voskarides, M. Pattichis, R. Istepanian, and C. Schizas. "Wireless Telemedicine Systems: An Overview". In *IEEE Antennas and Propagation Magazine*, volume 44, pages 143–153, April 2002.

[77] S. Pavlopoulos, E. Kyriacou, A. Berler, S. Dembeyiotis, and D. Koutsouris. "A Novel Emergency Telemedicine System Based on Wireless Communication Technology – AMBULANCE". In *IEEE Information Technology in Biomedicine*, volume 2, pages 261–267, December 1998.

[78] Jeffrey M. Perloff. "Microeconomics 3rd edition" . Addison Wesley Publishing Company.

[79] Asfandyar Qureshi. "Flexible Application Driven Network Striping over Wireless Wide Area Networks". MEng Thesis, Massachusetts Institute of Technology, March 2005.

[80] Asfandyar Qureshi, Jennifer Carlisle, and John Guttag. Tavarua: Video Streaming with WWAN Striping. In *ACM Multimedia 2006*, Santa Barbara, CA, October 2006.

[81] Asfandyar Qureshi and John Guttag. "Horde: Separating Network Striping Policy from Mechanism". In *ACM MOBISYS*, 2005.

[82] Asfandyar Qureshi and John Guttag. Horde: Separating Network Striping Policy from Mechanism. In *3rd International Conference on Mobile Systems, Applications, and Services (Mobisys 2005)*, Seattle, WA, June 2005.

[83] Pablo Rodriguez, Rajiv Chakravorty, Julian Chesterfield, Ian Pratt, and Suman Banerjee. Mar: a commuter router infrastructure for the mobile internet. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 217–230, New York, NY, USA, 2004. ACM Press.

[84] E. Setton, Y. J. Liang, and B. Girod. "Multiple Description Video Streaming over Multiple Channels with Active Probing". In *IEEE International Conference on Multimedia and Expo*, 2003.

[85] M. T. Shusuke, S. S. Takashima, L. F. Yuichiro, M. Minoru, H. Kazuhisa, and H. K. Asakura. "The Mobile Hospital – An Experimental Telemedicine System for the Early Detection of Disease". In *Telemedicine and Telecare*, volume 4, pages 146–151, 1998.

[86] P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan. WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks. University of Illinois at Urbana-Champaign.

[87] Alex Snoeren. "Adaptive Inverse Multiplexing for Wide-Area Wireless Networks". In *IEEE conference on Global Communications*, pages 1665–1672, 1999.

[88] Thomas Stockhammer and Miska M. Hannuksela. "H.264/AVC video for wireless transmission". In *IEEE Wireless Communications*, August 2005.

[89] Xiao Su and Benjamin W. Wah. "Multi-Description Video Streaming with Optimized Reconstruction-Based DCT and Neural-Network compensations". In *IEEE Transactions on Multimedia*, May 2001.

[90] Joshua A. Tauber. "issues in building mobile-aware applications with the Rover toolkit". Master's thesis, Massachusetts Institute of Technology, May 1996.

[91] Michael C. Toren. tcptraceroute: an implementation of traceroute using TCP SYN packets. http://michael.toren.net/code/tcptraceroute/.

[92] Andrew J. Viterbi. *CDMA principles of spread spectrum communication*. Addison-Wesley, 1995. ISBN: 0201633744.

[93] Zhou Wang, Ligang Lu, and Alan C. Bovik. "Video Quality Assesment Based on Structural Distortion Measurement". In *Signal Processing: Image Communication*, pages 121–132, February 2004.

[94] P.H. Westerink, R. Rajagopalan, and C.A. Gonzales. "Two-pass MPEG-2 variable-bit-rate encoding". *IBM Journal of Research and Development*, 43.4, 1999. http://www.research.ibm.com/journal/rd/434/westerink.html.

[95] T.C. Wong, J.W. Mark, K.C. Chua, J. Yao, and Y.H. Chew. Performance analysis of multiclass services in the uplink of wideband CDMA. In *International Conference on Communication Systems*, pages 692–696, November 2002.

[96] Y. Xiao, D. Gagliano, M. P. LaMonte, P. Hu, W. Gaasch, R. Gunawadane, and C. Mackenzie. "Design and Evaluation of a Real-Time Mobile Telemedicine System for Ambulance Transport". *Journal of High Speed Networks*, pages 47–56, September 2000. http://hrfp.umm.edu/Papers/2000/mobiletelemed.pdf.