FINDING PERCEPTUALLY OPTIMAL OPERATING POINTS OF A REAL TIME INTERACTIVE VIDEO-CONFERENCING SYSTEM

BY

WEE HONG YEO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Adviser:

Professor Benjamin W. Wah

ABSTRACT

This research aims to address issues faced by real time video-conferencing systems in locating a perceptually optimal operating point under various network and conversational conditions.

In order to determine the perceptually optimal operating point of a video-conferencing system, we must first be able to conduct a fair assessment of the quality of the current operating point in the system and compare it with another operating point to determine if one is better than the other in terms of perceptual quality. However at this point in time, there does not exist one objective quality metric that can accurately and fully describe the perceptual quality of a real time video conversation. Hence there is a need for a controlled environment to allow tests to be conducted in and in which we can study different metrics and identify the best trade-offs between them.

We begin by studying the components of a typical setup of a real time video-conferencing system and the impacts that various network and conversation conditions can have on the overall perceptual quality. We also look into different metrics available to measure those impacts.

We then created a platform to perform black box testing on current video-conferencing systems and observe how they handle the changes in operating conditions. The platform is then used to conduct a brief evaluation of the performance of Skype, a popular commercial video conferencing system. However, we are not able to modify the system parameters of Skype.

The main contribution of this thesis is the design of a new testbed that provides a controlled environment to allow tests to be conducted to determine the perceptual optimum operating point of a video conversation under specified network and conversational conditions. This testbed will allow us to modify certain parameters, such as frame rate and frame size, which were not previously possible.

The testbed takes as input, two recorded videos of the two speakers of a

face-to-face conversation and desired output video parameters, such as frame rate, frame size and delay. A video generation algorithm is designed as part of the testbed to handle modifications to frame rate and frame size of the videos as well as delays inserted into the recorded video conversation to simulate the effects of network delays. The most important issue addressed is the generation of new frames to fill up the gaps created due to a change in frame rate or delay inserted, unlike as in the case of voice, where a period of silence can simply be used to handle these situations.

The testbed uses a packetization strategy designed on the basis of an uneven packet transmission rate (UPTR) and that handles the packetization of interleaved video and audio data; it also uses piggybacking to provide redundancy if required. Losses can be injected either randomly or based on packet traces collected via PlanetLab. The processed videos will then be pieced together side-by-side to give the viewpoint of a third-party observing the video conversation from the site of the first speaker. Hence the first speaker will be observed to have a faster reaction time without network delays than that of the second speaker who is simulated to be located at the remote end. The video of the second speaker will also reflect the degradations in perceptual quality induced by the network conditions, whereas the first speaker will be of perfect quality. Hence with the testbed, we are able to generate output videos for different operating points under the same network and conversational conditions and thus able to make comparisons between two operating points.

With the testbed in place, we demonstrate how it can be used to evaluate the effects of various parameters on the overall perceptual quality.

Lastly, we demonstrate the results of applying an existing efficient search algorithm used for estimating the perceptually optimal mouth-to-ear delay (MED) of a Voice-over-IP(VoIP) conversation to a video conversation. This is achieved by using the testbed designed to conduct a series of subjective and objective tests to identify the perceptually optimal MED under specific network and conversational conditions.

To my parents, for their love and support.
To my girlfriend, for being there for me.

ACKNOWLEDGMENTS

I would like to thank my adviser, Prof. Benjamin W. Wah, for his guidance, patience and sharing of his knowledge. I also wish to thank him for providing me with the resources and support that are vital to the work accomplished.

I would like to thank my family in Singapore for their support and understanding.

I also would like to thank my girlfriend Yan Lin for her faith in me, for always being there for me through good times and bad times.

I would like to thank my senior Batu, whose work this thesis was built upon.

Last, but not least, I wish to thank all my friends who have accompanied me through this journey.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ACR	Absolute Category Rating
ACELP	Algebraic Code-Excited Linear-Prediction
AVI	Audio Video Interleave
CCR	Comparative Category Rating
CE	Conversational Efficiency
CDF	Cumulative Distribution Function
CMOS	Comparative Mean Opinion Score
CS	Conversational Symmetry
F2F	Face-to-Face
FPS	Frames per Second
FPTR	Fixed Packet Transmission Rate
IETF	Internet Engineering Task Force
ITU	International Telecommunication Union
IP	Internet Protocol
JND	Just Noticeable Difference
LC	Loss Concealment
LOSQ	Listening-Only Speech Quality
LR	Loss Rate
MED	Mouth-to-Ear Delay
MOS	Mean Opinion Score

MTU	Maximum Transmission Unit
NALU	Network Abstraction Layer Unit
PC	Personal Computer
PDF	Probability Distribution Function
PESQ	Perceptual Evaluation of Speech Quality
POS	Play-out Scheduling
PSNR	Peak-Signal-to-Noise-Ratio
PSTN	Public Switched Telephone Network
QoE	Quality of Experience
QoS	Quality of Service
RIFF	Resource Interchange File Format
RTP	Real-time Transport Protocol
SSIM	Structural Similarity Index
TCP	Transmission Control Protocol
UCFLR	Unconcealed Frame Loss Rate
UDP	User Datagram Protocol
UPTR	Uneven Packet Transmission Rate
VBR	Variable Bit Rate
VQM	Video Quality Metric

CHAPTER 1

INTRODUCTION

1.1 Motivations

Communication has always been an integral part of human society. Having face-to-face conversations with each other is ideal but not always achievable due to distance and time constraints. These limitations have led to the development of conventional circuit-switched telephone systems that allow long distance calling. With the evolution of the Internet, Voice-over-IP (VoIP) protocols have been designed and have since started a newer and inexpensive way of making a phone call. However, the system is significantly different from that of public switched telephone networks (PSTNs), as the route between the two hosts is not fixed and could suffer a much larger variance in degree of losses and delays. Thus different and more advanced measures must be in place to conceal the errors in order to ensure high quality is perceived by the parties at the ends.

However PSTNs and VoIP technologies are still not sufficient to complete the experience of a face-to-face conversation. This is because being able to see each other and observe the other party's body language is a large part of the conversation experience. Hence in recent years, with the improvements in Internet infrastructure, which offers higher bandwidths and improvements in encoding techniques that reduce the bandwidth required for video, video-conferencing technology has become much more affordable and more commonly used by people around the world.

There currently exist various types of videoconferencing software, such as Skype, MSN Messenger Live and GoogleTalk, that allow users to make free video calls over the Internet. Such software makes video-conferencing technology more accessible to the masses and has been steadily gaining popularity. Such software can run on a wide range of platforms, from specialized

hardware to common personal computers and small mobile handheld devices. These platforms vary greatly in terms of available bandwidth, processing power and latency.

Video-conferencing technology has a higher bandwidth demand and delay than VoIP, thereby making it even more important to identify an optimum operating configuration for the system. An optimum configuration allows us to achieve high perceptual quality based on the available resources.

1.2 Overall Goals

The goal of this research is to be able to identify the perceptually optimal operating point corresponding to specific network and conversation conditions. The perceptually optimal operating point is defined as a set of operating conditions that yield the highest perceptual quality under the defined limitations and conditions.

This is achieved by first looking into the overall architecture of a video-conferencing system and the parameters that define its operating environment. Various effects of degradation and how they can be measured using various metrics are also investigated. A platform is then set up to perform black box testing on current video conferencing systems and evaluate their reaction to changes in the operating environment.

The main contribution of this research is on the design of a new testbed that provides a controlled environment that allows a series of tests to be conducted. These tests can be used to locate the perceptual optimum operating point corresponding to specific network and conversation conditions.

The effectiveness of the new testbed is then demonstrated in conducting a series of subjective tests that allow us to observe how modifying certain parameters will affect the overall perceived quality.

Using this capabilty, we have applied an existing algorithm [1] to effectively estimate the perceptually optimal mouth-to-ear (MED) delay of a VoIP conversation to that of a video conversation and present the results.

1.3 Real Time Video-Conferencing System

As the main goal of this research is to identify the perceptually optimal operating point of a real-time video conferencing system, it is essential to understand the main characteristics of the system.

a) *Multiple objective quality metrics.* Many objective quality metrics have been designed to address the problem of evaluating the quality of a multimedia system. International organizations, such as the International Telecommunication Union (ITU) and the Internet Engineering Task Force (IETF), have also been created to verify and provide recommendations for such metrics, . However, at this point in time, no one metric can easily be calculated and yet fully represent the quality of the system by a simple number. Instead, there are numerous metrics which individually attempt to represent some parts of the system [1].

b) *Constrained resources.* Real time video-conferencing systems are one of the most demanding multimedia applications. Conventional multimedia systems such as phone applications only involve processing of audio data. Video streaming applications such as YouTube can tolerate a much higher degree of delay as there is no interactivity requirement. Video-conferencing systems must be able to send both video and audio data over a data network with relatively small amounts of delay to allow interactive conversations. The data network is often not ideal. Limited bandwidth and varying degrees of latency will have huge consequences on the overall perceived quality. Furthermore, packet transmissions between the two clients may be erroneous or even lost. Hence in order to determine the optimal operating point of the system, it is necessary to understand the constraints that the system operates under.

c) *Communication scenario.* The kind of conversation between the two speakers plays an important role in determining the best operating configuration for a video conferencing system. If the speakers have frequent interactions with lots of body gestures, most probably they will have a low tolerance for delay. Otherwise, the speakers may perceive each other as non-responsive. In the event of excessively long delays, double-talks may occur where both speakers speak at the same time. As the speakers can see each other, slow response to body gestures, such as a simple hand wave or nod, will have an adverse impact on the overall conversational experience.

d) *Control system parameters.* In order to handle the imperfections of the

network, control schemes have to be in place to provide redundancy and buffering. These control schemes each have their own configurable parameters. The set of parameters need to be dynamically adjusted at run time with inputs from measurements that describe the current conditions. The set of achievable parameters forms the operating curve.

e) *Trade-offs among different objective metrics on overall perceived quality.* As mentioned earlier in paragraph (a), there are multiple objective metrics that describe the quality of a part of the system. However, none of them can effectively capture the overall perceived quality. Under different network and conversational environments, trade-offs must be made among the metrics, as maximization based on one metric may result is severe degradation in another. As these trade-offs are not well defined, it is difficult to use these metrics to locate the best perceptually optimal operating point. Hence, it will be essential to perform subjective tests to study the trade-offs among the metrics and find a relationship that is configurable at run time.

1.4 Evaluations of Interactive Video Quality

In order to locate the perceptually optima operating point, there is a need for some measure of quality that allows comparison between two points on an operating curve. Evaluation of interactive video quality can be broadly categorized into two kinds: objective and subjective.

Objective metrics deal with some kind of measurements of the video or audio signal. These objective metrics can be further broken down into smaller categories, namely Full Reference (FR) Methods, Reduced Reference (RR) Methods and lastly No Reference (NR) Methods [2]. Full reference methods need to have access to the original and the degraded version of the sample tested. They will tend to provide the most accurate measure of quality because they have a base case available for it to compare the degraded version against. Examples of FR methods are Perceptual Evaluation of Speech Quality (PESQ) for speech and Video Quality Metric (VQM) for video. Both are recommendations by ITU, listed under ITU-T P.862 and J.144, respectively. These are the metrics chosen to be used in this research to augment subjective tests [3, 4].

NR metrics are less accurate than FR and RR, but they do not require

access to the original data. Thus these metrics can be deployed at run time to provide an estimate of the quality of the video and speech. Examples of such metrics are ITU-T Recommendation P.563 for speech and measurements of degree of blur or blockiness in the video stream [5, 6, 7].

There are three major difficulties of reducing all aspects of a video conversation into one dimension that can be represented by a single metric. Firstly, a video conversation is different from listening to music or watching a movie. It involves interactions with another person on the remote end. The conversation itself plays a huge part in determining the experience of the video conversation. Most of the metrics do not take this into consideration at all. Different kinds of conversations also place different emphasis on the different metrics. For example, audio will not be that important if a video call is set up to show the other party an object of interest.

Secondly, the relationship between each metric to the overall perceived quality has a non-linear relationship even if all other conditions are constant. This is due to the human's ability to perceive quality. For example, if the audio and video quality falls below a certain threshold that is unacceptable to the user, any lower does not make a difference, but having a slight improvement may greatly increase the utility.

Thirdly, the trade-offs between the metrics on the overall perceived quality are not well defined. For example, if we have excellent video quality but we are hearing a lot of static noise in the conversation from the other party. The perceived quality of the video conversation will still be low, as we cannot hear what the other person is saying. However if the audio is clear and the video is not that clear, the user may still feel that it is acceptable as he can still convey the message mainly through speech. As a result, single dimension metrics do not accurately evaluate the overall quality.

Subjective metrics require subjective evaluations to be performed by human subjects. These metrics will provide the highest accuracy if conducted over a sufficiently large test group to account for errors in human judgment. ITU P.800 [8] recommendation describes how to obtain MOS_{CQS} (subjective conversational quality) that provides subjective evaluations of conversations. Even though these tests provide very high accuracy, it is impossible to conduct them at run time. Hence offline tests have to be conducted to obtain information that will then be used to guide the selection of the operating point at run time. However subjective metrics are very time consuming and

resource intensive to conduct; thus it will not be possible to apply them to cover all possible network and conversational conditions.

Subjective evaluations are usually conducted in a controlled environment with subjects shown one video conversation followed by the other and asked to rate the quality using an *absolute category rating* (ACR). The *mean opinion score* (MOS) is then obtained by calculating the algebraic mean of the score of the subjects given to that same video conversation [9].

However as mentioned, there are inherently many possible network and conversational conditions that can be tested and it is not possible to have subjective tests conducted to cover all of them. Thus it will be helpful to utilize certain observations to reduce the number of subjective tests conducted. Human perception has limited sensitivity. Two operating points that yield output quality very close to one another cannot be differentiated. Hence there will be no need to compare operating points that lie within this condition, defined as the *just noticeable difference* (JND). There are also thresholds of perfect and intolerable qualities. Similarly, we do not need to waste time conducting subjective tests in those regions. With these constraints, we can reduce significantly the number of tests that need to be conducted [10].

Evaluation of conversational quality for video conferencing systems is a largely unexplored field. Not only is it difficult to conduct large numbers of subjective tests under different conditions, it is difficult to recreate the same conversational condition under different network conditions. This is because two people making the same conversations at two different times will have different speeds at which they talk and react to each other. Furthermore, the body language will also be different.

1.5 Problems Studied

The aim of a video conferencing system is to provide a good face-to-face conversation experience between the two parties over the network. Being able to operate at the perceptually optimal operating point will allow the system to achieve that aim within the constraints of the operating environment. In order to achieve the main goal of finding the perceptually optimal operating point, we break the problem down into several steps.

Firstly, we begin by looking into the architecture of a video conferencing

system and by studying its various components. We also study the various network and conversational conditions and their effects on the overall output quality.

The next problem we have studied is how current video conferencing systems react to changes in network conditions, in order to provide us an idea of how they are determining their operating configurations. This is achieved by creating a test platform to perform black box testing. By varying the network conditions and comparing the inputs with the outputs, we can see how the system under test reacts accordingly.

We then address the problem of testing one conversation over different network conditions, as it is not practical to make a new recording for each possible network condition. Hence, we have developed an algorithm that can generate different versions of the same video conversation for different operating points.

We also study how video and audio are packetized together to ensure that they are received on time to be played and how losses are injected into the data streams. A network simulator is then built that allows us to conduct subjective tests of a video conversation over different network conditions.

Lastly, we address the problem of the large number of subjective tests that need to be conducted to locate a good operating point. To achieve that, we have extended an existing algorithm that can locate the perceptually optimal mouth-to-ear delay for a voice conversation with minimal batches of subjective tests and apply that to video conversations by adding a metric on video quality.

1.6 Contributions of This Research

The first contribution of this thesis is the identification of a set of objective measures that relate to the perception of quality of a video conversation.

The second contribution of this thesis is the design of a platform that allows the study of how existing systems react to network changes using black box testing.

The third and major contribution of this thesis is the design of a new testbed that allows evaluation of the same conversation at different operating points. This testbed allows us to determine which of the two operating points

is preferred, which is used to guide the search for locating the perceptually optimal operating point.

The fourth contribution is the observation of how modifying various parameters of the system will affect the output quality.

The last contribution is the successful application of the efficient search algorithm that estimates mouth-to-ear delay for voice conversation to that of a video conversation.

1.7 Outline of the Thesis

The background of a video conferencing system is presented in Chapter 2. The testbed for black box testing on existing systems is presented in Chapter 3. Skype, an existing video conferencing software is chosen as an example for the test platform, and its evaluation results are also presented in Chapter 3. In Chapter 4, we present the design of the new testbed. Within the chapter the algorithm used to generate different versions of a conversation under different network conditions is also presented. We also show the packetization strategy that is implemented in the network simulator. In Chapter 5 we show the results of using the testbed that we have set up, to study the various effects of different operating parameters. In Chapter 6, we present the results of the application of an efficient search algorithm used to estimate perceptually optimal mouth-to-ear delay of an interactive video conversation under different network and conversation conditions. In Chapter 7, we present our conclusions and plans for future work.

CHAPTER 2

BACKGROUND

2.1 Video-Conferencing System Architecture

The architecture of a video conferencing system consists of hardware, software, and network components.

Hardware Components for video conferencing range from personal computers equipped with webcams to high-quality specialized hardware and high-resolution cameras (e.g. Polycoms [11]). With advances in technology, mobile devices such as smartphones have also been equipped with Internet capability and hence are able to use the video conferencing software available. The hardware components play a huge role in determining the operating conditions of a video conferencing system. For example, a piece of specialized hardware will most likely be able to process and send video and speech captured at higher frame rate and frame size, compared to a smartphone.

Software Components refer to the video conferencing software used. Examples of current video-conferencing software are Skype and MSN Messenger Live. These software perform the same fundamental function of allowing two parties to hold a video conversation over the Internet. They encode the video captured by the hardware into a data stream format. This is determined by the codec chosen. Despite the many different codecs available today, they share common purposes of reducing the number of bits required to send data over the network and adding some level of error resiliency. Often codecs offer different quantization levels to be used to determine the bit rate. As video conferencing deals with both video and speech, two different kinds of codecs are required.

- *Speech Codecs* are designed to convert speech captured in raw format into bit streams using certain modulation techniques to allow transmission over a network. Simple speech coding techniques such as ADPCM sample speech signal at a fixed period and apply a certain level of quantization. ITU-T Recommendations G.726 and G.729 are some examples of such speech codecs [12, 13]. More advanced speech codecs such as ITU-T Recommendations G.722.2 use Algebraic Code-Excited Linear-Prediction (ACELP) coding that try to apply a speech model to the signal [14]. These newer techniques tend to offer better quality, but often require a higher bit rate.

- *Video Codecs* compress the large number of bits that make up a raw video in order to allow video frames to fit into a much smaller bit budget. The size of the bit budget is constrained by the amount of bandwidth available. For example, a 10s raw video can easily occupy as much as 1GB of space. After compression it only takes up less than 1KB. Common techniques are to encode the video into I-frames and P-frames. I-frames are keyframes which are broken into macroblocks, where each macroblock contains color information. These I-frames contain most of the color data and hence should be encoded with high quality and redundancy where possible. P-frames are the frames that exist in between I-frames. P-frames only contain the difference between the previous frame and the current frame. As a result P-frames are a lot smaller than I-frames and hence allow us to greatly reduce the overall size of the video. For videos that do not have many changes in the scene, such as video-conferencing videos, often the video sequence will consists of only a few I-frames and the rest will be P-frames. Examples of video codecs are ITU-T recommendation H.263, H.264 and proprietary codec VP7 by On2 Technologies used by Skype [15, 16, 17].

Network Components of the system come into play after the video and speech data have been encoded by the codec. Here the bit stream output from the codec is packetized and transmitted over the network. The network can be made up of specialized cellular networks, catered specially for video conferencing communication, or it could be the Internet. The network protocol chosen by the system will also determine its robustness and responsiveness.

For example, if the RTP/UDP protocol is chosen rather than RTP/TCP, we will expect more losses but more responsiveness. This is due to the fact that UDP protocols do not ensure in-order delivery but TCP does. However to achieve that, TCP may require that several packets to be sent back and forth before the first data chunk is received. Using TCP may give us fewer packet losses, but longer delay compared to UDP, due to the retransmissions [18, 19, 20].

The quality of the network that the video-conferencing system runs on plays a huge role in determining the quality of the received video and speech. The network may induce delays and losses that will have a detrimental effect on the perceived quality. Playout scheduling and loss concealment schemes have since been implemented to manage delay effects and conceal losses.

Playout Scheduling (POS) schemes aim to buffer irregular packet arrivals (jitter) in order to achieve smooth playback of the video and speech frames. Video-conferencing systems commonly employ such schemes at the receiver with a fixed MED. The value of the MED determines how much buffer is available before a data chunk is considered to be late. If the data frame is delayed longer then the predetermined MED, it will be considered lost, even though it is received eventually, as the next frame will have to be played by then. However, this value cannot be increased indefinitely, as it will slow down the conversation and have a negative impact on interactivity. The value of the MED may be adjusted according to the fluctuations in the network. Being able to accurately estimate this value will ensure minimal losses and yet preserve the high level of interactivity that a face-to-face conversation will have.

Loss Concealment (LC) schemes are essential in the deployment of video conferencing systems. Even with POS schemes and a good estimate of the MED, some losses are inevitable if the network conditions are bad or the jitter increases past the buffers available. Losses that cannot be concealed at the lower layers will propagate to the decoder and cause degradation in speech and video. One simple loss concealment strategy is piggybacking. Piggybacking involves sending redundant copies of a data chunk. The redundant copies are spread over multiple packets. Hence as long as one of the packets is received, the data will still be received. However, there is a limit

to the number of redundant copies that can be sent. As every copy is placed in a following packet, each of the data frames in each of the later packets will incur a delay equivalent to the time between the two packets sent. When the delay gets larger than the MED at the receiver side, then there is no point in sending it, as it will be marked as lost by the POS scheme, even if it is successfully received. Furthermore, video conferencing is bandwidth intensive. By sending multiple copies of a packet, the total effective bandwidth available will be reduced accordingly. This will result in a gradual decline in the overall quality even if all data sent is received, due to the higher level of quantization required.

2.2 Conversation Dynamics and Quality Metrics

In a face-to-face conversation between two parties, each of the two parties takes a turn to speak and to listen. During the time where the speaker and listener switch roles, there is a period of silence. This period of silence is defined as *mutual silence (MS)*. A conversation thus contains alternating segments of speech and silence [10].

In the ideal face-to-face environment, both parties have a common perception of the conversation: a speech segment followed by a silence duration that is identically perceived by both parties. However when the same conversation is conducted over a network, the conversation suffers delays, jitters and losses. Each party of the conversation will then have a different perception. When there are delays in the video conversation, the MSs perceived by the parties in the conversation will be alternating short and long silences between turns.

The quality of a video conversation is determined by three factors: the received video quality, the listening-only speech quality (LOSQ) and the delay incurred from the mouth of the speaker to the ear of the receiver (MED).

Objective Speech Quality Metrics. In Chapter 3 of B. Sat's dissertation [10], whose work this thesis is built upon, there is a survey of various recommendations for the measurement of LOSQ. They are the *Perceptual Evaluation of Speech Quality* (PESQ) (ITU P.862) [3], *ITU G.114* [21], the *E-Model* (ITU G.107) [22] and *ITU P.562* [23]. According to [10], PESQ is a

better choice than the other metrics due to its high correlation to subjective MOS results in various IP telephony applications.

G.114 is not chosen, as it is only concerned with delay rather than speech quality. The E-Model is not used as it assumes the independence and additivity of degradations due to LOSQ and delay. This assumption has been shown not to be true in [10]. P.562 is not adequate as it was not designed to model the packet switch-network conditions that Internet communication systems are based on. Hence based on similar reasons, in this research work, PESQ is chosen as the metric for the measurement of LOSQ. However as mentioned in [10], PESQ needs to be augmented with interactivity metrics that capture the effects of delay.

Objective Video Quality Metrics. The received video quality can be measured by various objective metrics. In this research, because the original video is available, FR metrics are chosen as they give higher accuracy than NR or RR. However, during real time implementations, FR metrics will not be suitable.

In [24], various objective existing video quality metrics have been studied, namely *Peak-Signal-to-Noise-Ratio* (PSNR), *Video Quality Metric* (VQM), *Moving Pictures Quality Metric* (MPQM), *Structural Similarity Index* (SSIM) and *Noise Quality Measure* (NQM). Comparisons were made between the various metrics, and it is found that VQM has the best correlation to subjective measurement results. VQM performance has also been acknowledged by ITU in its J.144 recommendation [4]. Furthermore VQM has a specific model for video-conferencing systems. Hence in this research, VQM is selected as the measure of received video quality. The source code for its implementation is also available at [25].

However, VQM faces certain limitations. Similarly to PESQ, it is not able to capture the interactivity in conversations and hence needs to be augmented by other metrics. It is also not able to compare two video sequences of different frame rate and frame size. Hence different objective metrics or subjective evaluations have to be used to measure the effects of different frame rates and frame sizes.

Human response delay and mutual silence. In Figure 2.1 adapted from [26], we see the point of view of participant B. The *human response*

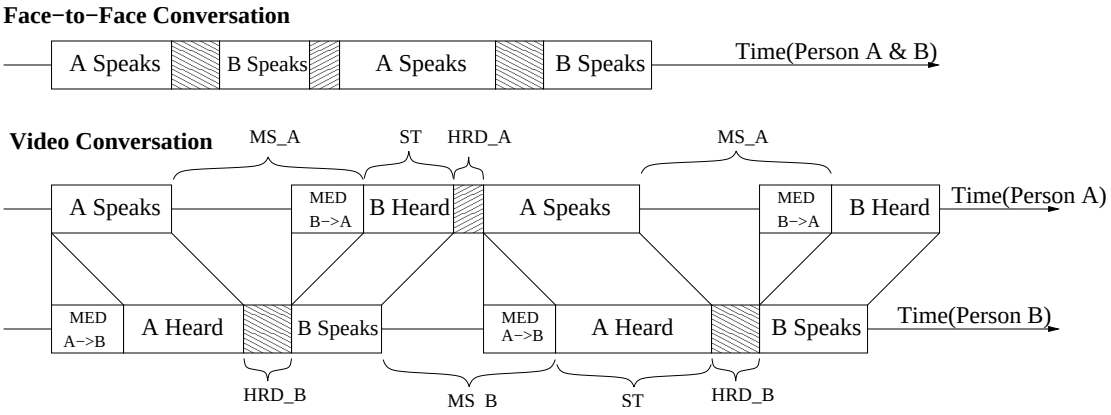


Figure 2.1: Conversational dynamics in a face-to-face and two-party interactive video conversation setting.

delay of B is defined in [26] as the time that B takes to determine that A has finished talking and for B to begin to respond (HRD_B). HRD_B is perceived to be longer by participant A due to the effects of network delays. In [26], MS_A^j is defined as the *mutual silence* before the j^{th} single-talk speech segment (ST_j) is spoken/heard. This MS_A^j corresponds to the time taken for A to witness B's response to him. $MED_{A,B}^j$ is then defined in [26] to be the MED between A's mouth and B's ear for transmitting ST_j from A to B. The relationship between MS, HRD and MEDs is listed in Equation 2.1, taken from [26]. Even though the equations are derived for a VoIP conversation, they are also applicable to a video conversation

$$\begin{aligned} MS_A^j &= MED_{A,B}^{j-1} + HRD_B^j + MED_{B,A}^j, \\ MS_A^{j+1} &= HRD_A^{j+1}, \\ MS_B^j &= HRD_B^j, \\ MS_B^{j+1} &= MED_{B,A}^j + HRD_A^{j+1} + MED_{A,B}^{j+1}. \end{aligned} \tag{2.1}$$

When having a video conversation, the participants do not have a clear perception of MED as they do not know exactly when the other party starts talking. However by observing the indirect effects caused by the introduction of MED, such as MS, the participants can realize the existence of MED. Hence each participant will experience an asymmetry in the conversation dynamics, when he realizes that the remote party seems to have a slow reaction to what he is saying. This asymmetry leads to a degradation in the efficiency of the conversation and perceived quality [27].

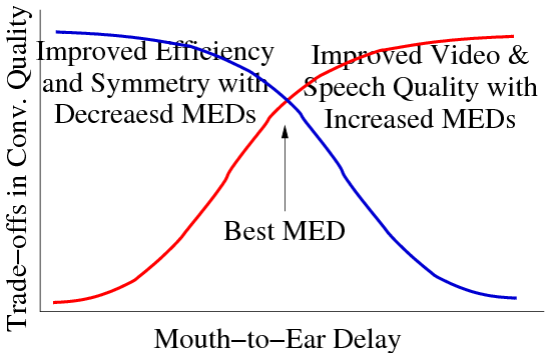


Figure 2.2: Trade-offs considerations.

However, we cannot improve the conversation quality by just reducing MED because by having a MED that is too small, packets will begin not arriving on time and be assumed lost, which will inherently lead to a degradation in speech and video quality. A short MED will indeed improve the symmetry and efficiency of the conversation. However, the speech and video quality will suffer. In Figure 2.2 adapted from [28], we show the trade-offs between delay and quality under a specific network and conversational condition and identify a suitable MED that offers the best trade-off. The location of this optimal MED also depends on the turn-switching frequency [27, 29] and on the network and conversational conditions [30]. Different network and conversational conditions will have different operating curves and hence their optimal MEDs will differ.

Hence, apart from PESQ and VQM that capture the one-way speech and video quality, there is a need to incorporate other metrics to capture the effects of delay on conversational dynamics. Two such metrics are used in this research: *conversation symmetry (CS)* and *conversation efficiency (CE)* [28].

Conversational symmetry. Conversational symmetry (CS) is defined in [28] as follows:

$$CS_A = \frac{\max_j MS_A^j}{\min_j MS_A^j}, \qquad CS_B = \frac{\max_j MS_B^j}{\min_j MS_B^j}. \tag{2.2}$$

CS_A, for example, is defined as the ratio of the maximum and the minimum MSs experienced by A. CS_B is defined in a similar manner.

In a face-to-face conversation, the CS is purely dependent on the HRDs of each of the participants. However, in an interactive video conversation, CS is affected by the network delays. Assuming each participant has similar HRD, the ideal CS will be 1. However, due to the effects of network delays, CS will be skewed. For example, with reference to Figure 2.1, B is only heard by A after 2*$MED + HRD$. Thus with a larger MED, B will be seen to respond much slower to A, and this effect can be captured by CS. Note that CS is heavily dependent on the HRDs of the participant. If their HRDs are not similar in the first place, having more delays may improve CS. Nevertheless, it does reflect the amount of delay in the system and its effect on interactivity.

One possible negative impact of having an excessively skewed symmetry is that if A perceives B to be reacting slowly, he may tend to slow himself down. As a result, the conversation takes longer to complete.

Conversational efficiency. Conversational efficiency (CE) mainly aims to capture how long it takes to complete a conversation compared to when there is no delay. Basically, with more delay introduced into the system, a conversation will take longer to complete, which is an indication of decrease in interactivity. Similarly to CS, CE is dependent on HRDs. If the HRDs are short, a slight increase in MED will cause a significant fall in CE. If HRDs are long, a slight increase in MED may not have any significant impact.

CE is defined in [28] as the ratio of the duration of time, in which the participants are actively speaking or listening, to the total duration of the call:

$$CE = \frac{\text{Total Speaking Time} + \text{Total Listening Time}}{\text{Total Time including Silence}}. \tag{2.3}$$

The ideal CE will be close to 1. In a face-to-face conversation, the HRDs are very small when compared to the time spent speaking.

2.3 Effects of Network Imperfections on Perceptual Quality

Double talk. This effect is discussed in [28] and shown in Figure 2.3 reproduced from [10]. When there is a large spike in the amount of delay in the system and the MED is not adapted accordingly, there will be a series of

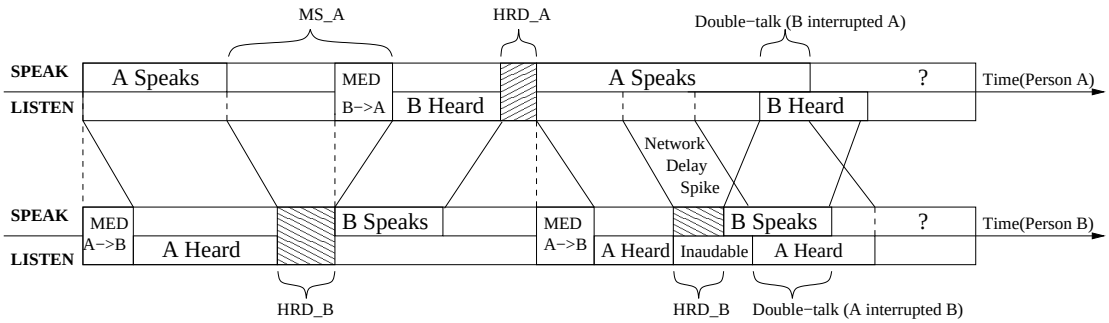


Figure 2.3: The occurrence of a double-talk due to a lack of adequate system reaction to network-delay spikes.

consecutive losses that cannot be concealed by the LC strategy. As a result, the communication may temporarily break down between the two parties. At the end of his talk spurt, User A will wait for User B to respond. However, User B may not have received what User A has said and proceeds to repeat his last message. Similarly, User B may repeat himself after waiting for some time for User A to respond. However, at the end of the spike, the packets start getting through. We may well end up in a situation where both users are talking and interrupting each other. This will result in confusion, and time will need to be spent to sort it out. As a result, the overall conversation quality will be adversely affected.

Out of Sync Audio and Video. In a video-conferencing system, audio and video data has to be captured at the same time by the physical components of the system. They are then encoded and packetized together for delivery over the network. Packets sent can contain data chunks that have audio data or video data. How the audio and video data are packetized varies from system to system and depends on the packetization strategy in place. Audio and video data received at the encoder will then be unpacketized and sorted accordingly in time for the playout scheduler. If some packets containing audio data are lost, while those that contain video data are received for the specific frame to be played, video will appear to be playing smoothly but the audio is not. If the next piece of audio data received is not played at the appropriate time, it may lead to audio and video being out of sync. The most obvious negative observation will be seeing the speaker's lips moving in a way not consistent with the audio speech heard [31].

Video Frozen. Video and audio data chunks are interleaved and packed into packets before they are sent over the network. As the network is far from ideal, with packets being dropped and large fluctuations in the delay, losses may not always be concealed and will be propagated up to the codec layer.

When speech frames are lost completely, they can be easily replaced by additional periods of silence, which may not always be perceived by the user to be a system failure. However, in the case of video, if video frames are lost due to system failure, the person in the video will appear to stop moving completely [32]. The video image will be seen as being frozen in time, which is a much more detectable effect. This is one of the obvious indications of network degradation, which leads to a drop in perceptual quality. Hence frozen videos should be prevented as much as possible. One common solution is to implement redundancy using piggybacking, although as mentioned in the loss concealment paragraph in Section 2.1, piggybacking has its limitations.

Unsightly Visual Effects. When the video stream data suffer unconcealable losses or errors, the decoder will try to make compensation and attempt to conceal these losses. One such technique is to use motion-compensated temporal prediction [33], where we replace a missing macroblock with the macroblock at the same location from the previous frame. However, the concealment is not perfect if there is motion in the scene, such as the movement of lips or blinking of eyes. When multiple macroblocks or frames are lost, the loss concealment techniques may not be able to conceal the excessive losses. As a result, unsightly visual artifacts will appear [2]. If an I-frame is lost, possible ugly effects may include color bleaching. If errors occur during the decoding of the macroblock, we will witness effects of blockiness and bluriness. If P-frames are lost, macroblocks that are affected by the motion vectors will appear incorrectly. Examples of blur, blockiness and color bleaching effects are shown in Figure 2.4.

2.4 Summary

In this chapter, we have presented the basic architecture of a video-conferencing system and its various components. We have also examined the conversation

dynamics of a video conversation, as well as the various metrics that can be used to measure the quality of a video conversation. Lastly, we have introduced the various effects that network imperfections can have on perceptual quality.

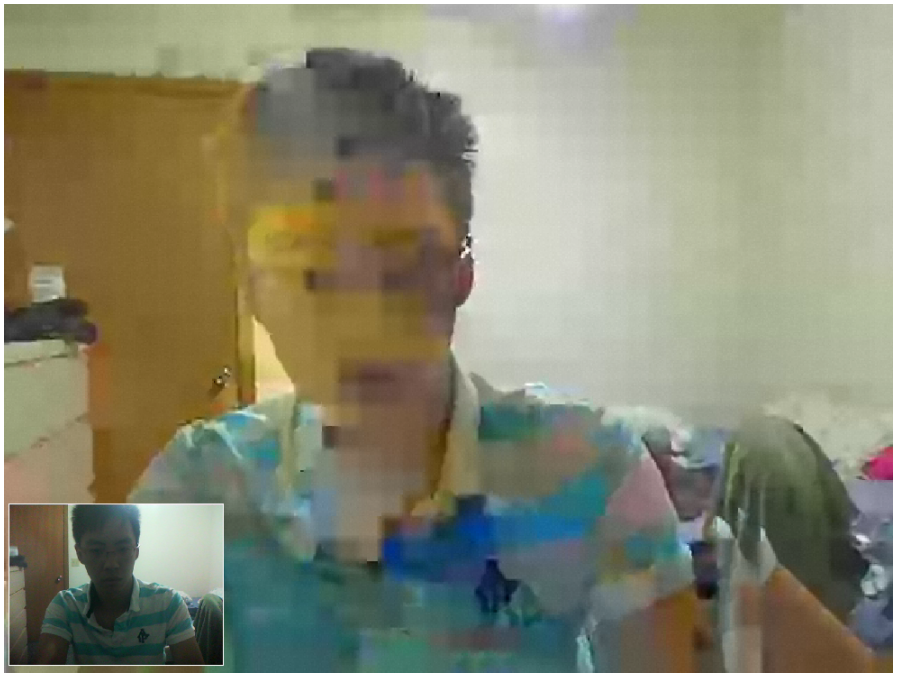


Figure 2.4: Illustration of blur, blockiness and color bleaching effects. By comparing the main image to the smaller image on the bottom left corner, we can see the main image is blurred. Squarish chunks appear in the main image indicating blockiness. The color of the door can be seen to bleach onto the face of the person in the video.

CHAPTER 3

BLACK BOX TESTING OF CURRENT VIDEO-CONFERENCING SYSTEMS

In this chapter we present our setup that can be used to evaluate existing video-conferencing systems. A similar study has been done previously on an older version of Skype for the Linux Operating System in [34]. The work presented in this chapter is not aimed to show that the setup is better than existing setups. It seeks to demonstrate that we have created a working platform that can be used to evaluate the performance of existing video conferencing systems.

3.1 Test Platform Setup

The test platform consists of the following:

- Two client PCs, each running a version of the video-conferencing software, a video capture device and one wired high speed Ethernet port to connect to the router PC
- One router Ubuntu PC running eireshark and a network simulator [35]. The router PC is equipped with three wired high speed Ethernet ports. The three ports are used by the router PC to connect to the client PCs and one active Internet connection.

The two client PCs will connect to each other through the router PC. Hence, when the two clients call each other, the router PC after proper configurations will be able to intercept the packets received and divert them to a queue in the kernel that is accessible by the network troll program. The troll will then perform the necessary simulated network effects such as delays, losses, errors, duplication or reordering. After which, the troll forwards the modified packets to their intended recipient. The setup is shown in Figure 3.1.

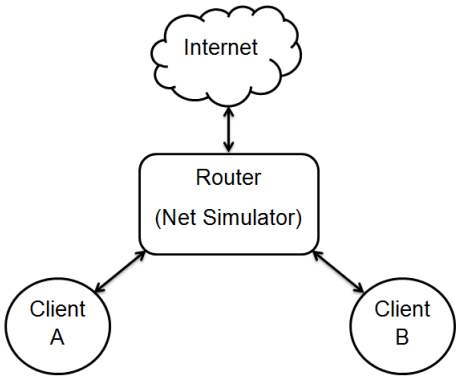


Figure 3.1: Test platform setup.

Initial Setup at the Router PC. At the router PC, the iptables have to be set up correctly to allow packets to flow between the two connected clients. Iptables is a user level program that can be used to configure rules for the routing of packets in the Linux kernel and must be accessed using root privileges. In Ubuntu 10.04 LTS, this can be easily done by the GUI by setting each of the network interfaces to be *shared to other computers*. Some modifications to the iptables may still be necessary for systems with firewalls in place.

Initial Setup at the Client PC. After the connection to the router PC is completed, we ensure that the client PCs have access to the Internet and make sure that they are able to ping each other. Once that is verified, we can open the video-conferencing software and perform a test call between the two clients.

Running Wireshark on Router PC. During the test call, we run Wireshark [36] on the router PC to monitor the traffic between the two clients. This will show us detailed information on the packets being sent between the two video-conferencing clients, such as source and destination IP addresses and ports. The detail content of each packet can be viewed too. Refer to Figure 3.2 for a screenshot of Wireshark. This information is then used in the setting up the network simulator.

Setting up Network Simulator. The network simulator requires access to the intercepted packets. Hence a rule in the iptable has to be set up, such that the kernel does forward these packets immediately. The rule that has

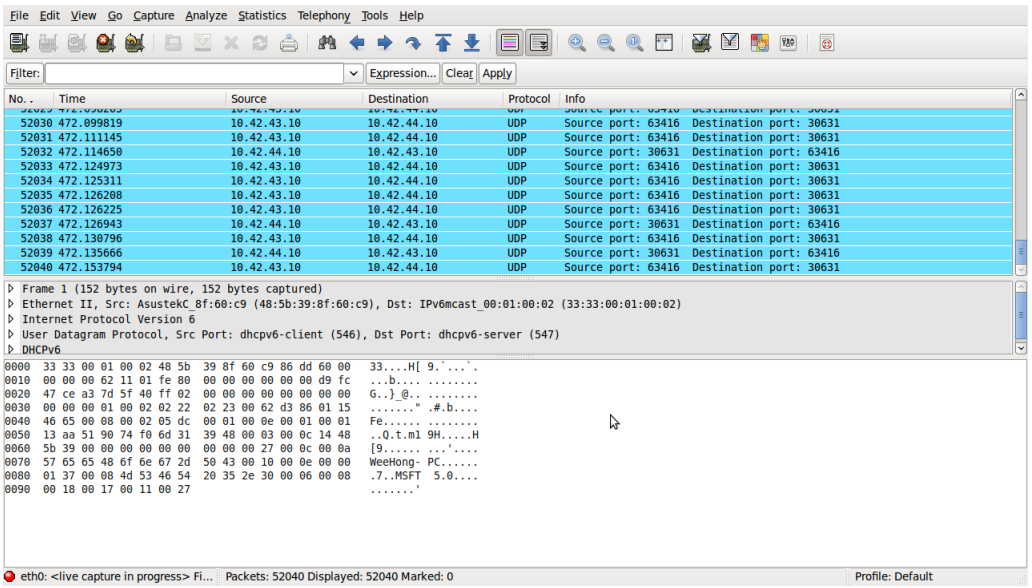


Figure 3.2: Screenshot of Wireshark showing the packets being sent between the two clients.

to be set up needs to specify that all packets between the two clients (they can be represented by IP addresses) are to be placed in a system queue. Once they are placed in the system queue, the troll will perform whatever modifications necessary and forward them at the right time based on its settings.

An example will be: *sudo iptables -I FORWARD 1 -s 10.42.43.10 -p udp –source-port 63416 -j QUEUE*. This rule means to send all packets from 10.42.43.10, with UDP protocol and source port 63416 to the system queue which the network troll will process.

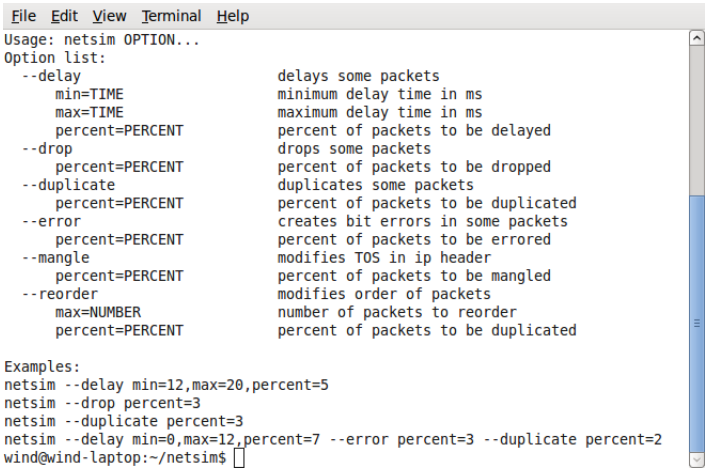


Figure 3.3: Options offered by the network simulator.

Usage of the Network Simulator. The network simulator offers many possible options to allow the tester to set the network conditions desired. Refer to Figure 3.3 for the options. The network troll needs to be executed with administrator privileges, as it has to access the network queue. Once the iptables have been set up correctly, start the troll with the settings desired, and it will vary the network conditions accordingly.

Using Recorded Videos and Capture Received Video. In the event that recorded videos are to be used instead of a live feed from a webcam, a software called Fake Webcam is available to read raw video stored in an AVI file. For Skype, a software called "IMCapture for Skype" is able to extract the received video and audio streams, but not at the frame rate captured. Similar software is available for other video-conferencing software [37]. Once the received video is captured, video quality metrics can be used to compare the sent and received videos. However, these software are not fully developed at the time of this research. They are currently only able to record at a lower frame rate than what is sent. Hence, for this research the tests are carried out using webcam streams, and measurements are extracted from the Call Technical Info provided by Skype.

Modifications needed for use with Generic Video-Conferencing Systems. The only modifications needed for this setup with any generic video-conferencing system is in the collection of information on the quality of received video stream. Skype conveniently provides us with a Call Technical Info function that does this. As long as a similar function is available for the video-conferencing system under test, this setup will be viable.

3.2 Evaluation of Skype using the Test Platform

Using the test platform, we have conducted a series of experiments on how Skype, a popular video conferencing software, reacts to changes in network conditions. Screenshots of the results of the experiment can be seen in Figures 3.4 - 3.29. The experiments are not exhaustive and are conducted to show that the test platform is working correctly. The network effects are only applied to the **receiving** channel.

Overview of Skype. Skype is a free video-conferencing software that allows users to make free voice and video calls over the Internet [38]. It uses On2 Technologies' VP7 video codec and a range of audio codecs, one of them being G.729. The evaluation is done on Skype version 4.2.0.187.

Measurement of Call Quality using Call Technical Info and Subjective Measurements. Skype clients are equipped with a function called *Call Technical Info*. The call technical info gives us information such as the identity of the callers, the amount of packets received and sent, the percentages of packet errors, the measured network delay and the measured network jitter. It even tells us the frame rate and frame size of packets sent. Information about what codec is used is also displayed. From the vast amount of information that Skype is able to provide in the Call Technical Info, it is safe to assume that the software either performs some degree of estimation of the network parameters, or it sends auxiliary information alongside the video and audio data to allow such measurements to be carried out. In this research, the information presented by the Call Technical Info is used to determine how Skype reacts to changes in network conditions.

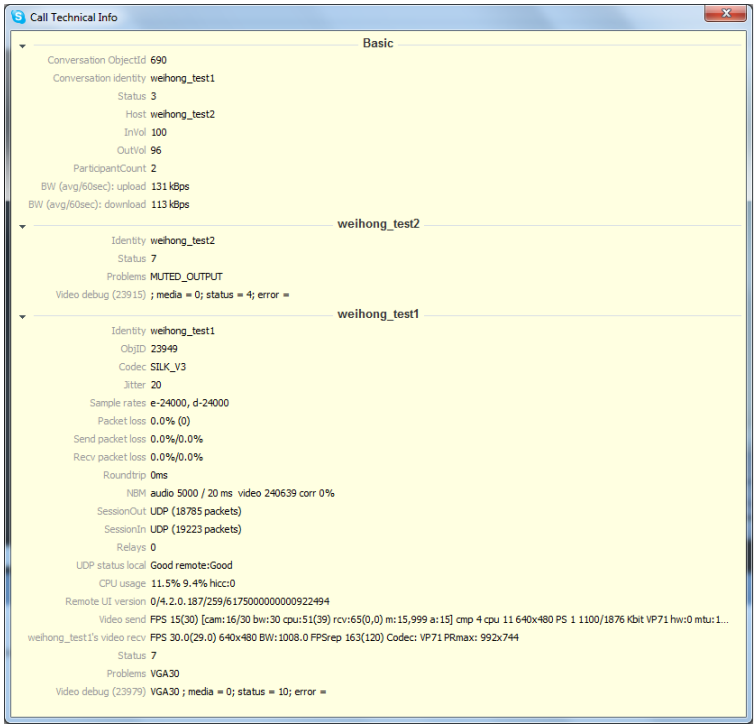
Figure 3.4: Skype screenshot showing perfect quality.

Subjective evaluation of the video and audio streams is also conducted. Only one subject is used for the subjective evaluations due to resource limitations, and hence the results are not statistically significant. However, the observations still provide us with useful insight on the limitations of the mea-

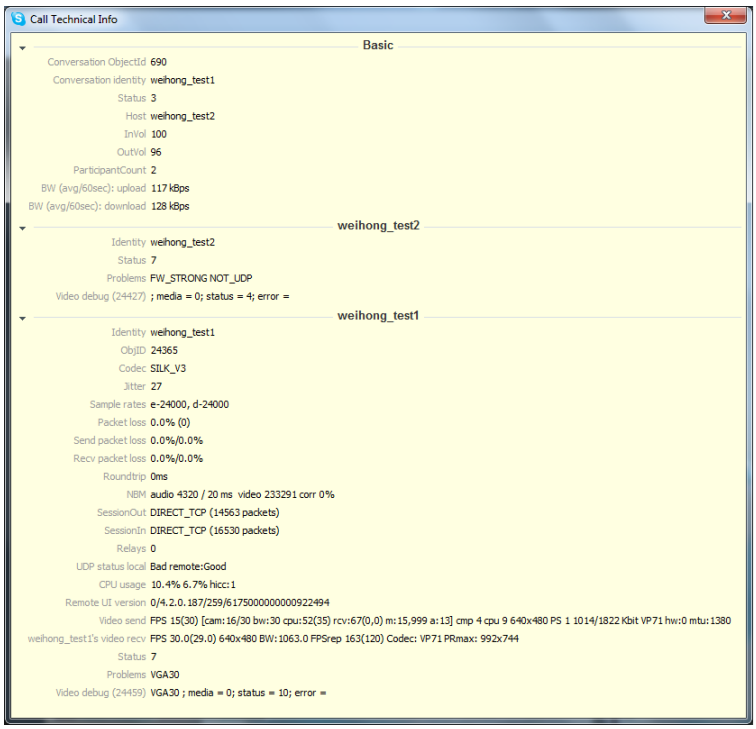
surements shown in the Call Technical Info. Screenshots of the video stream are shown in the results together with the Call Technical Info in Figures 3.6 - 3.29.

A screen shot of a perfect video quality call is shown in Figure 3.4. Its corresponding Call Technical Info is presented in Figure 3.5. It has been observed that Skype uses both UDP and TCP protocol, with preference to the UDP protocol under good conditions. However, when UDP is not available, Skype will switch over to TCP.

All measurements are taken after the call setup is completed and stabilized.



(a) UDP



(b) TCP

Figure 3.5: Call Technical Info for perfect quality shown in Figure 3.4, under two different network protocols.

Reactions to Packet Drops. To investigate the reaction of Skype to packet drops, we vary the packet drop percentage from 5% to 10%, 15% and 20%. The screenshots of the results and their Call Technical Info are shown in Figures 3.6 - 3.8.

With packets being dropped, some speech and video information will not be received. It is observed that Skype switches from UDP to TCP when packets are being dropped. The switch in protocol seems to improve the call quality, as even with 10% packet drop, the call quality is not significantly affected. This improvement in quality is believed to be due to TCP requesting for retransmission of the lost packets, which can be seen by looking at Figures 3.6(a) and 3.7(a). Both images are seen to be relatively clear.

However, this results in an increase in the jitter ($\sim$ 400 to 600ms) experienced by the system seen in Figures 3.6(b), 3.7(b) and 3.8(b), where the jitter value increases from 451 to 669. The video and speech is observed to be more jittery with the increase in packet drop percentage. Sometimes video freezes are also observed.

When the packet drop percentage reaches 15%, the amount of packet drops seems to be too much for the system to conceal. The image starts to become blurred as seen in Figure 3.8(a).

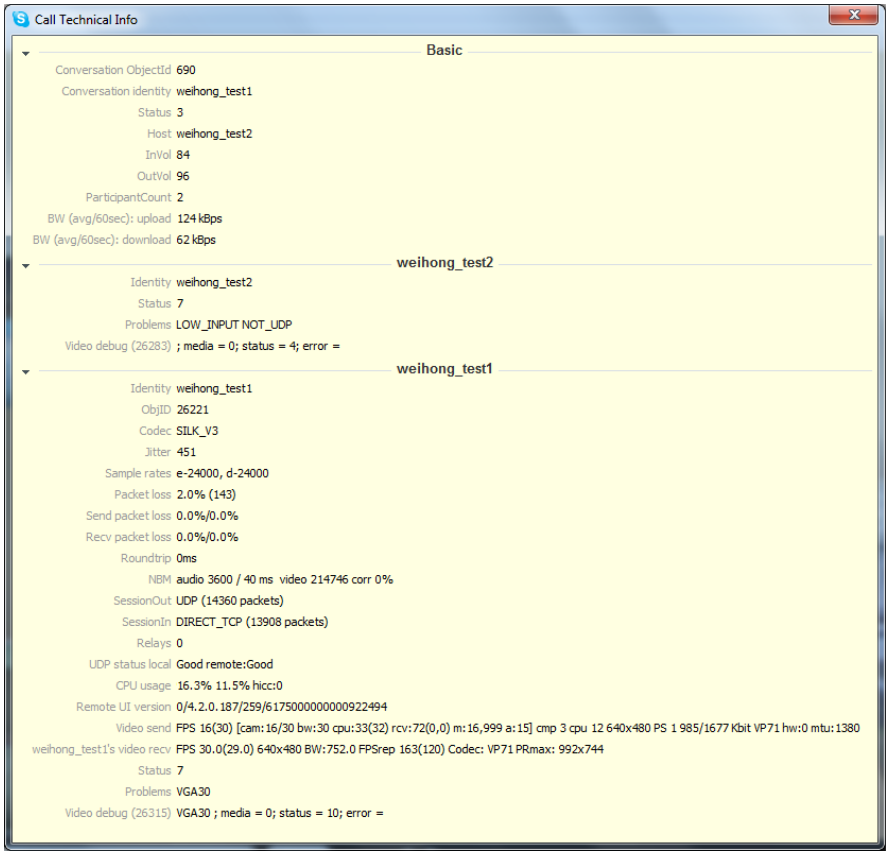
It is noticed that the call also takes longer to be set up between the two clients. The call set-up time is measured using a timer. The timer starts when the call is initiated, and stops when the call is received on the remote end. Under low losses (5%), the time to set up a call takes less than 1 s. When it reaches 15% losses, the time needed to set up a call can take up to 5 s.

When the packet drop percentage reaches 20%, the call was dropped by the system.

The bandwidth available is seen to fall with the increase in the packet drop rate. In Figure 3.6(a) the download bandwidth is 62 kBps for 5% packet drop rate and drops to 6 kBps in Figure 3.8(a) for 15% packet drop rate.

The frame rate of the video received is also seen to fall from 30 FPS to 11 FPS in Figure 3.6(a) - 3.8(a).

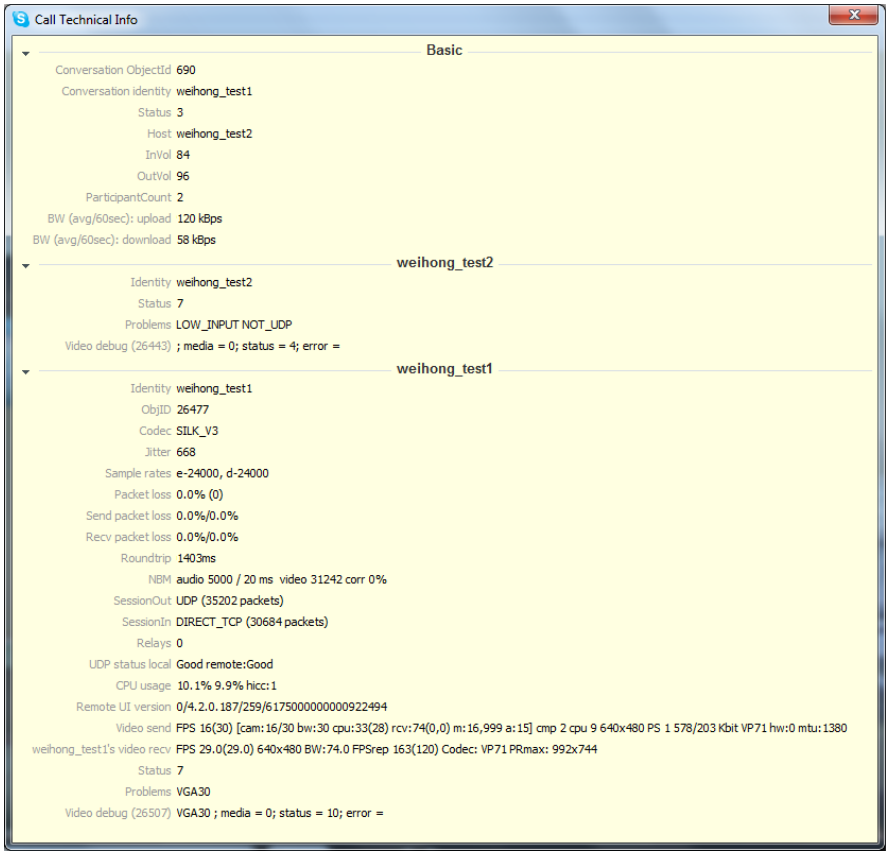
(a)



(b)

Figure 3.6: Screenshots showing Skype Video (a) and Call Technical Info (b) for 5% packet drop. Packet drop percentage is not accurately reflected by packet loss statistics. Video is received at 30 FPS and VGA size. SessionIn protocol changed to TCP. Jitter value is 451 ms. Inbound bandwidth is 62 kBps.

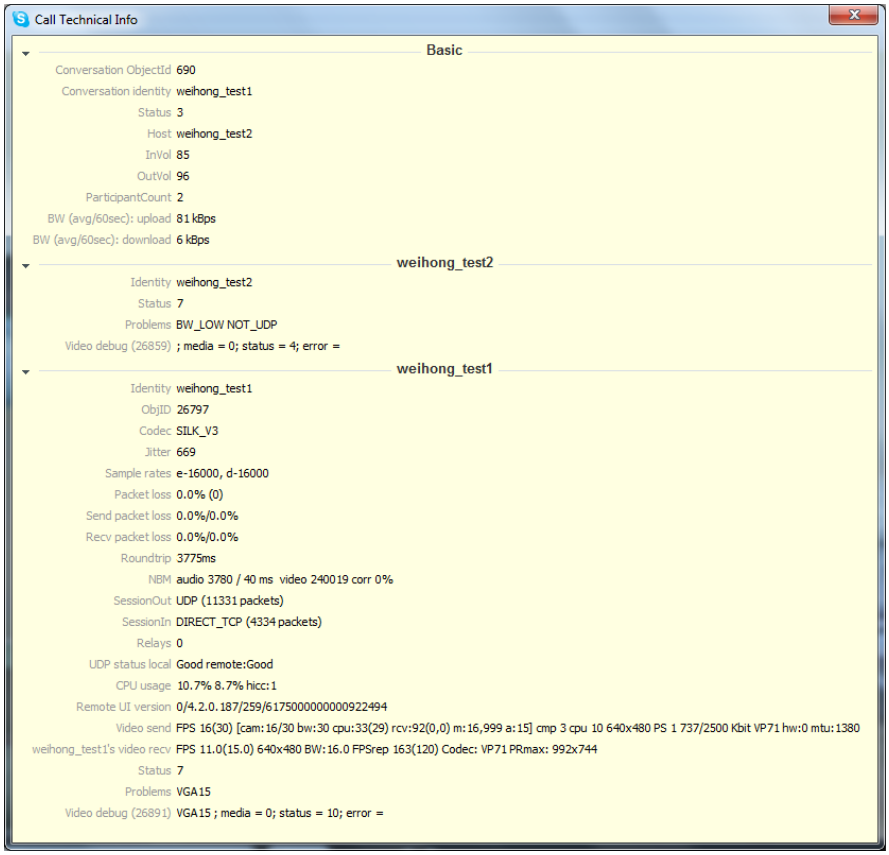
(a)



(b)

Figure 3.7: Screenshots showing Skype Video (a) and Call Technical Info (b) for 10% packet drop. Packet drop percentage is not accurately reflected by packet loss statistics. Video is received at 29 FPS and VGA size. SessionIn protocol changed to TCP. Jitter value is 668 ms. Inbound bandwidth is 58 kBps.

(a)



(b)

Figure 3.8: Screenshots showing Skype Video (a) and Call Technical Info (b) for 15% packet drop. Packet drop percentage is not accurately reflected by packet loss statistics. Video is received at 11 FPS and VGA size. SessionIn protocol changed to TCP. Jitter value is 669 ms. Inbound bandwidth is 6 kBps.

Reactions to Packet Errors. To investigate the reaction of Skype to packet errors, we similarly vary the packet error percentage from 5% to 10%, 15% and 20%. The screenshots of the results and their Call Technical Info are shown in Figures 3.9(a) - 3.12(a).

In the event of packet errors, Skype does not switch over TCP unlike the case for packet drops. This is seen in the Call Technical Info shown in Figures 3.9(b) - 3.12(b), where the SessionIn shows UDP protocol is used.

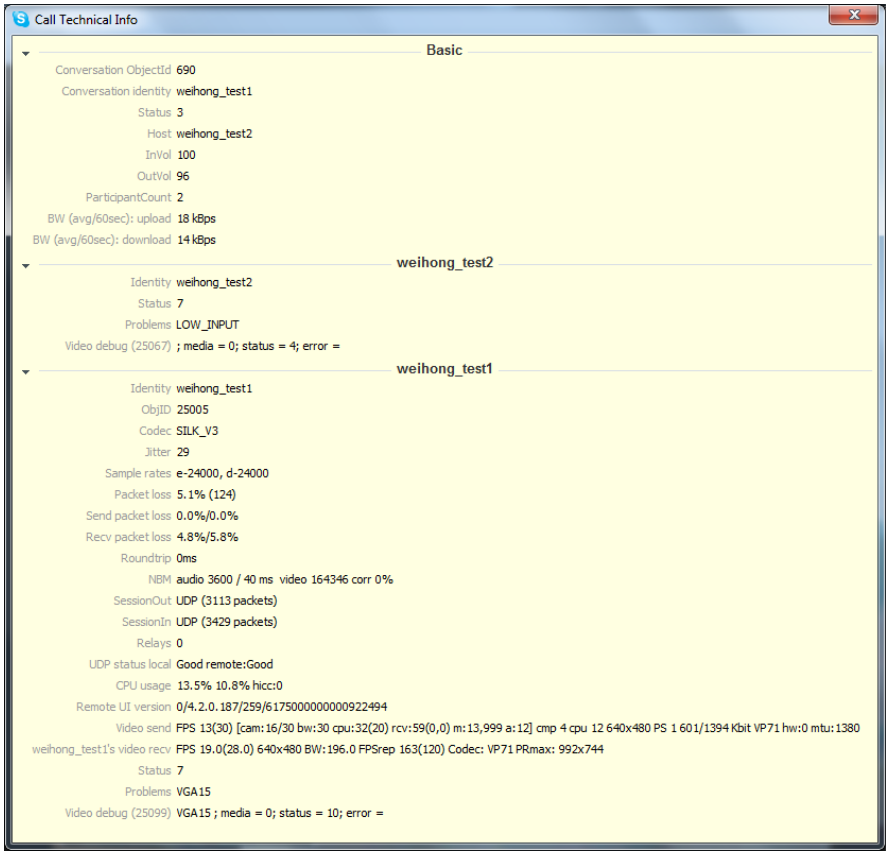
Skype is able to accurately show the error percentage in its Call Technical Info under its packet loss measurement. In Figures 3.9(b) - 3.12(b) we can see that the value of the packet loss measurement matches closely to the amount of error injected, which is unlike the case for packet drops. Hence, we believe that Skype is able to detect packets with errors and use it as their packet loss statistic. For packets that are lost during transmission over the network, Skype is not able to obtain that measurement.

It is observed that the frame rate of the video received falls from 19 FPS at 5% error rate to 1 FPS at 20% error rate. At 15% and 20% error rates, the frame size of the video received also falls from 640x480 (VGA) to 160x120 (QVGA), one fourth of the maximum size.

Jitter is seen to slightly increase from 30ms to 70ms, but this is much lower than the situation where packets are dropped and TCP is used. The bandwidth also falls from 14 kBps to 2 kBps, which may explain the decision made by Skype to scale down the size of the image.

Visually we can see in Figures 3.9(a) - 3.12(a) that the image experiences increase in levels of blurriness and blockiness with respect to the amount of error in the network. This is different compared to packet drops, where the image is still relatively clear up to 10% packet drops. For audio, with more packet errors, the level of noise increases. These observations are obtained subjectively, as they are not captured by the Call Technical Info of Skype.

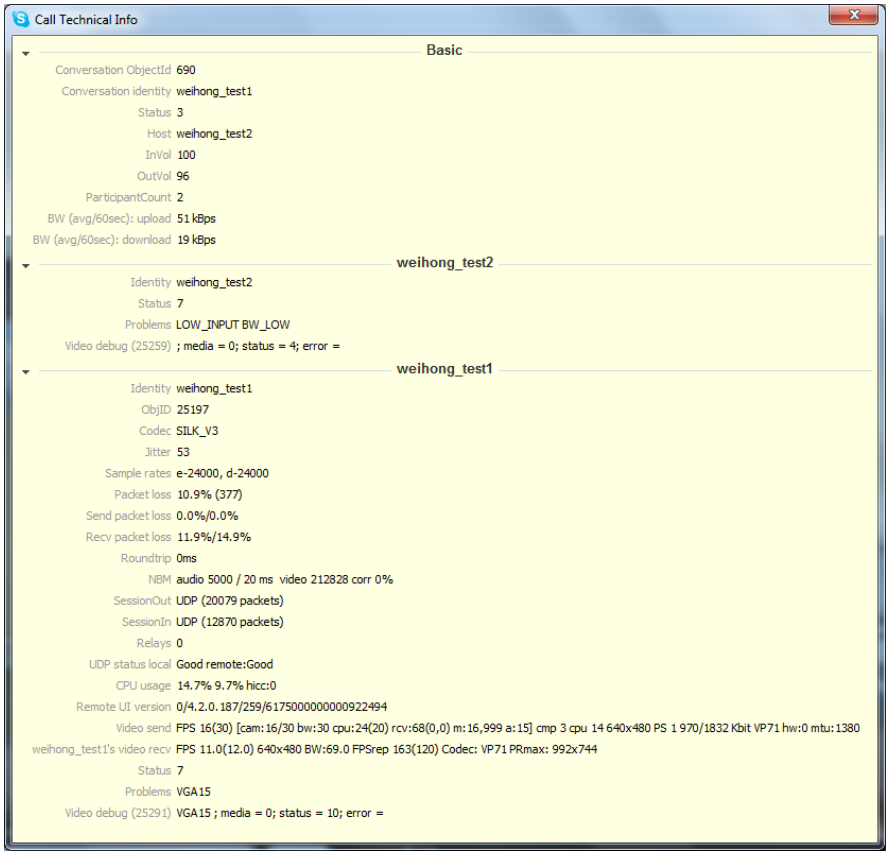
(a)



(b)

Figure 3.9: Screenshots showing Skype Video (a) and Call Technical Info (b) for 5% packet error. Packet loss percentage in figure(b) corresponds to error percentage. SessionIn protocol is UDP. Video received at 19 FPS and VGA size. Download bandwidth at 14 kBps. Jitter value at 29 ms.

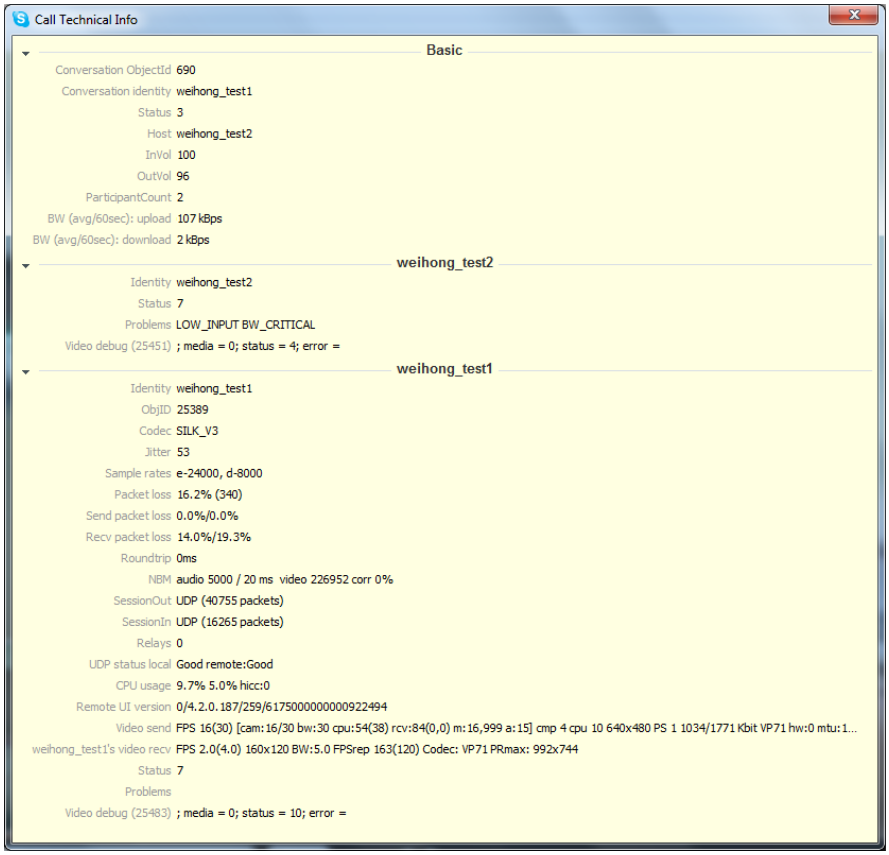
(a)



(b)

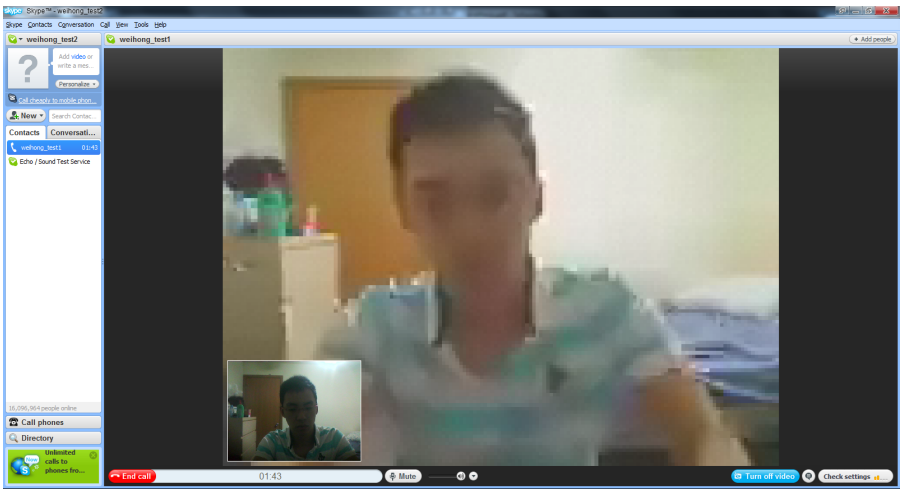
Figure 3.10: Screenshots showing Skype Video (a) and Call Technical Info (b) for 10% packet error. Packet loss percentage in figure (b) corresponds to error percentage. SessionIn protocol is UDP. Video received at 11 FPS and VGA size. Download bandwidth at 19 kBps. Jitter value at 53 ms.

(a)

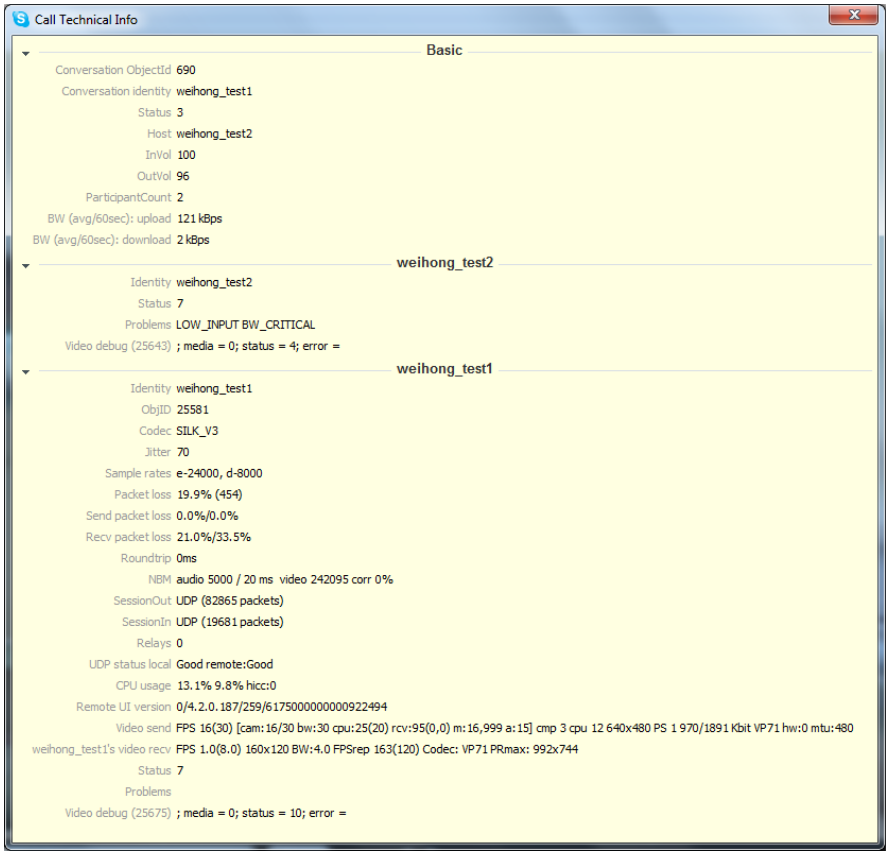


(b)

Figure 3.11: Screenshots showing Skype Video (a) and Call Technical Info (b) for 15% packet error. Packet loss percentage in figure (b) corresponds to error percentage. SessionIn protocol is UDP. Video received at 2 FPS and QQVGA size. Download bandwidth at 2 kBps. Jitter value at 53 ms.



(a)



(b)

Figure 3.12: Screenshots showing Skype Video (a) and Call Technical Info (b) for 20% packet error. Packet loss percentage in figure (b) corresponds to error percentage. SessionIn protocol is UDP. Video received at 1 FPS and QQVGA size. Download bandwidth at 2 kBps. Jitter value at 70 ms.

Reactions to Delays. To investigate the reaction of Skype to network delays, we vary the amount of delay in the network from 50 ms to 100 ms, 200 ms, 500 ms and 1000 ms (1 s). The screenshots of the results and their Call Technical Info are shown in Figures 3.13 - 3.14.

It is observed that the time between when the speech is spoken or the body movement is performed, and the time it is detected at the remote host, gets longer when the delay in the system is increased. The video and speech quality remains high; however, the interactivity falls. More detailed investigation of this effect can be seen in our other investigations in Chapter 6.

It is observed that by increasing the delay, all packets are still received correctly. The video and audio quality remains high. This is shown is Figures 3.13(a) - 3.17(a). The fact that all packets are still received correctly implies that the system is able to detect the increase and adjust its estimated MED accordingly to allow the frames sufficient time to arrive for the playout scheduler.

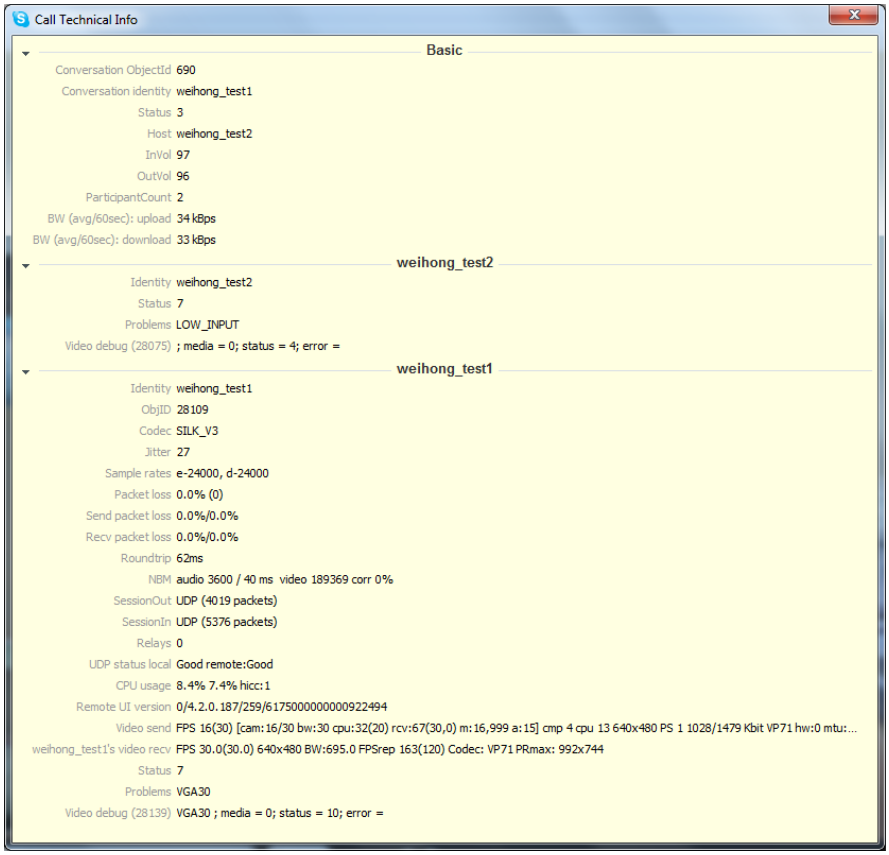
It is observed that any body movement or speech spoken is seen to be more clearly delayed with the increase in network delay.

All of these observations are obtained subjectively as Skype's Call Technical Info does not have any metric to capture these effects.

With reference to Figures 3.13(b) - 3.17(b), Skype is able to measure with high accuracy the delay in the network. The round trip values in the figures correspond closely to the delay injected.

Frame rate and frame size of video remains at 30 FPS and VGA size across all delay values. This indicates that Skype is able to adjust is playout schedule to accommodate the different amounts of network delays. Download bandwidth remains high and jitter in the system remains low at around $\sim$20 ms for all delay values.

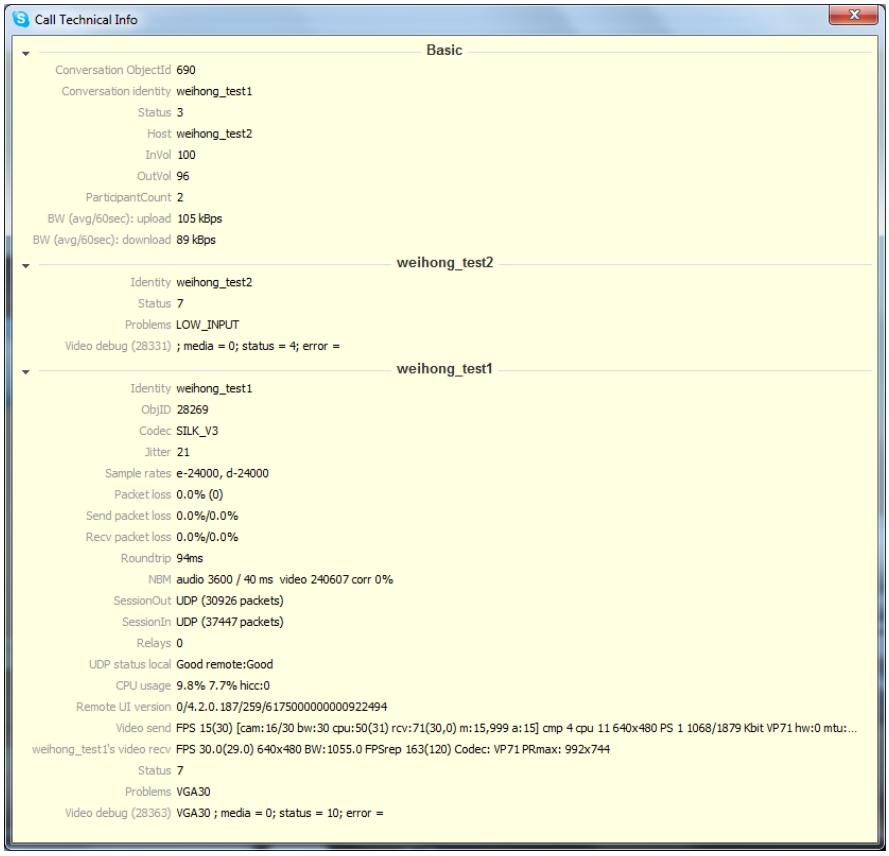
(a)



(b)

Figure 3.13: Screenshots showing Skype Video (a) and Call Technical Info (a) for 50 ms network delay. Roundtrip time measured is 62 ms. Frame rate and frame size of video received is 30 FPS and VGA size. Download bandwidth is 33 kBps. Jitter value at 27 ms.

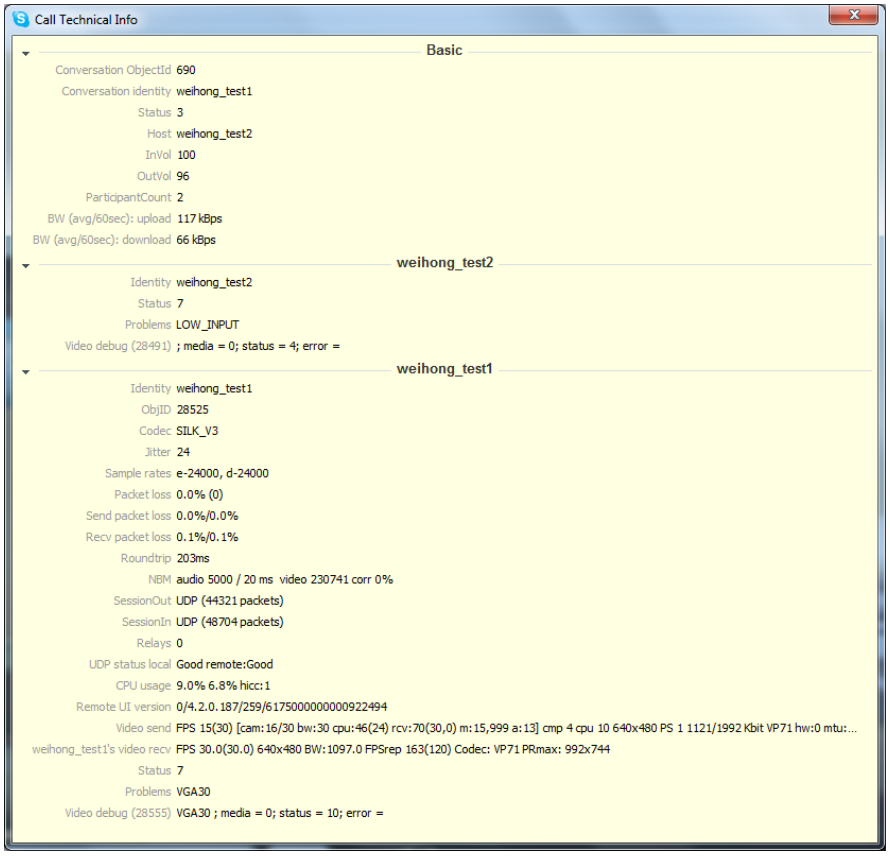
(a)



(b)

Figure 3.14: Screenshots showing Skype Video(a) and Call Technical Info(a) for 100 ms network delay. Roundtrip time measured is 94 ms. Frame rate and frame size of video received is 30 FPS and VGA size. Download bandwidth is 89 kBps. Jitter value at 21 ms.

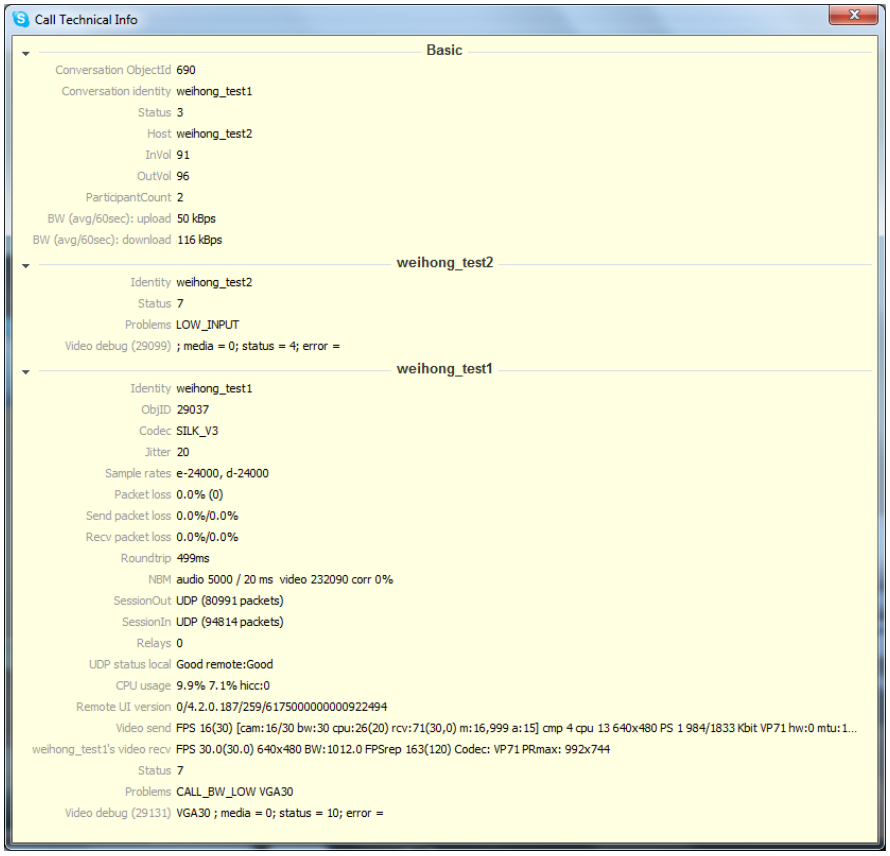
(a)



(b)

Figure 3.15: Screenshots showing Skype Video(a) and Call Technical Info(a) for 200 ms network delay. Roundtrip time measured is 203 ms. Frame rate and frame size of video received is 30 FPS and VGA size. Download bandwidth is 66 kBps. Jitter value at 24 ms.

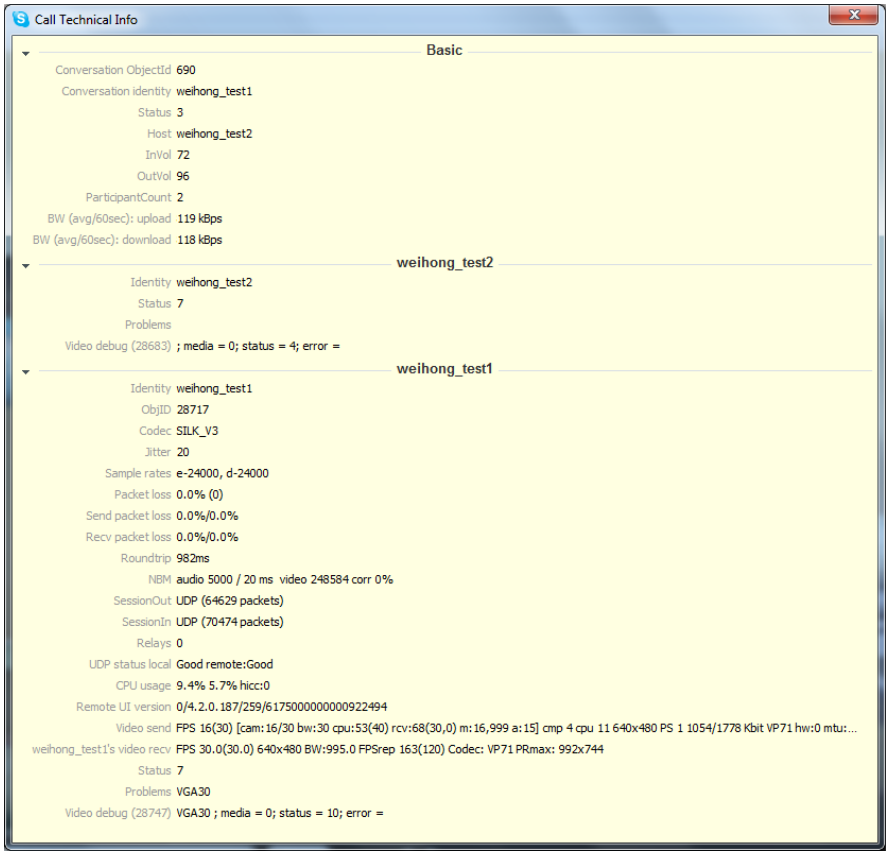
(a)



(b)

Figure 3.16: Screenshots showing Skype Video (a) and Call Technical Info (a) for 500 ms network delay. Roundtrip time measured is 499 ms. Frame rate and frame size of video received is 30 FPS and VGA size. Download bandwidth is 116 kBps. Jitter value at 20 ms.

(a)



(b)

Figure 3.17: Screenshots showing Skype Video (a) and Call Technical Info (a) for 1000 ms network delay. Roundtrip time measured is 982 ms. Frame rate and frame size of video received is 30 FPS and VGA size. Download bandwidth is 118 kBps. Jitter value at 20 ms.

Reactions to Jitter. To investigate the reaction of Skype to network jitter, we vary the amount of jitter in the network from 25 ms to 50 ms, 100 ms, 250 ms and 500 ms. Jitter is a measure of variation of network delays from the average value. For our experiment, a jitter of 50ms means that the delay experienced by a packet lies in the range of [0 ms, 100 ms]. The screenshots of the results and their Call Technical Info are shown in Figures 3.18 - 3.22.

An obvious effect that can be seen by introducing jitter is the increase in blockiness of the image, shown in Figure 3.18(a) and 3.19(a) where a jitter of 50 ms and 100 ms in introduced. When the level of jitter is increased, blockiness in the image increases, as reflected by Figures 3.20(a) - 3.22(a). This degradation in the image is observed to be different from the scenario when there are packet errors. When there are packet errors, the image gets blurred instead of blocky.

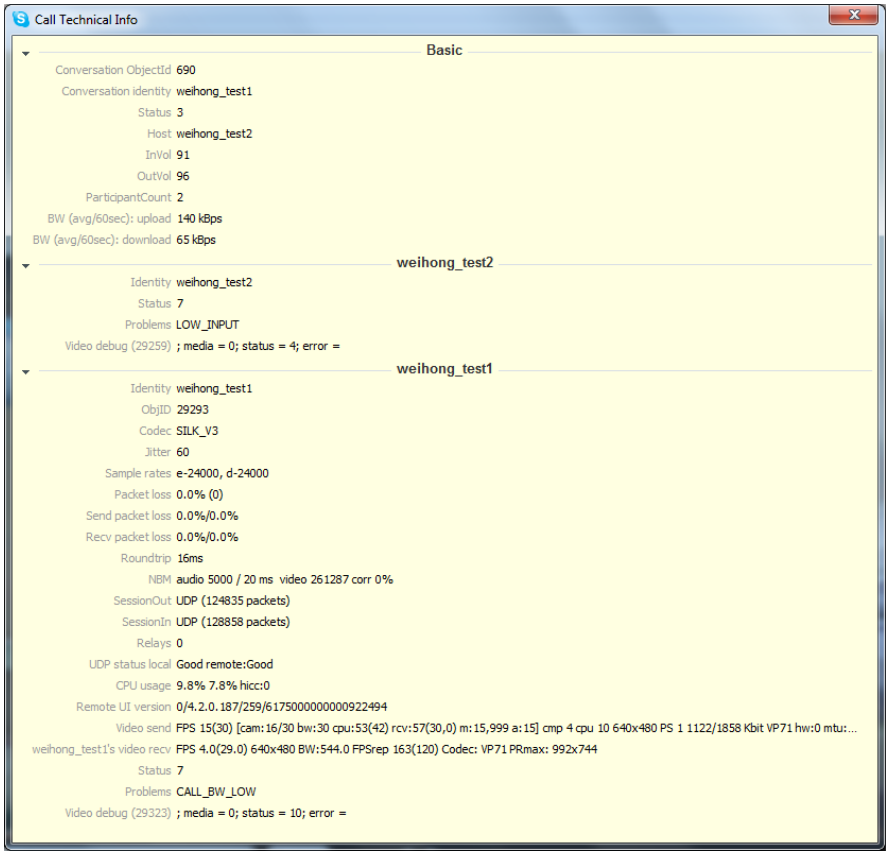
By observing the jitter entry in Skype's Call Technical Info, shown in Figures 3.18(b) to 3.22(b), we can see that Skype is able to measure the amount of jitter in the system. However, the measurements are not always accurate because jitter fluctuates.

Note that only when the jitter hits 500, some packets are lost, as reflected in Figure 3.21(b) where the received packet loss percentage is 11.5%. For jitter below 500 ms, all packets are still received, even though a jitter of more than 5% is considered high in [10]. This means that even though the packets are received, they may be late due to the jitter and considered to be lost. The system can only estimate the average jitter and set the estimated MED value based on this estimate. However, with a high level of jitter, there is a high probability that packets take longer than estimated MED to arrive. This means that the frames in the packet are not received in time to be played and hence will be considered lost. The loss of frames could be the reason for the increase in level of blockiness in the image.

In Figures 3.18(b) - 3.22(b), it is also observed that under high levels of jitter, frame rate falls drastically. The high levels of jitter seem to affect the frame rate measurement, as shown in Figures 3.19(b) - 3.22(b), the frame rate measured is 0 FPS. However in reality, it cannot be 0 FPS, as video is being played.

The frame size also falls to QQVGA once the jitter passes 250 ms. Download bandwidth also falls from 65 kBps to 2 kBps. With higher levels of jitter, the round trip delay is seen to fluctuate more.

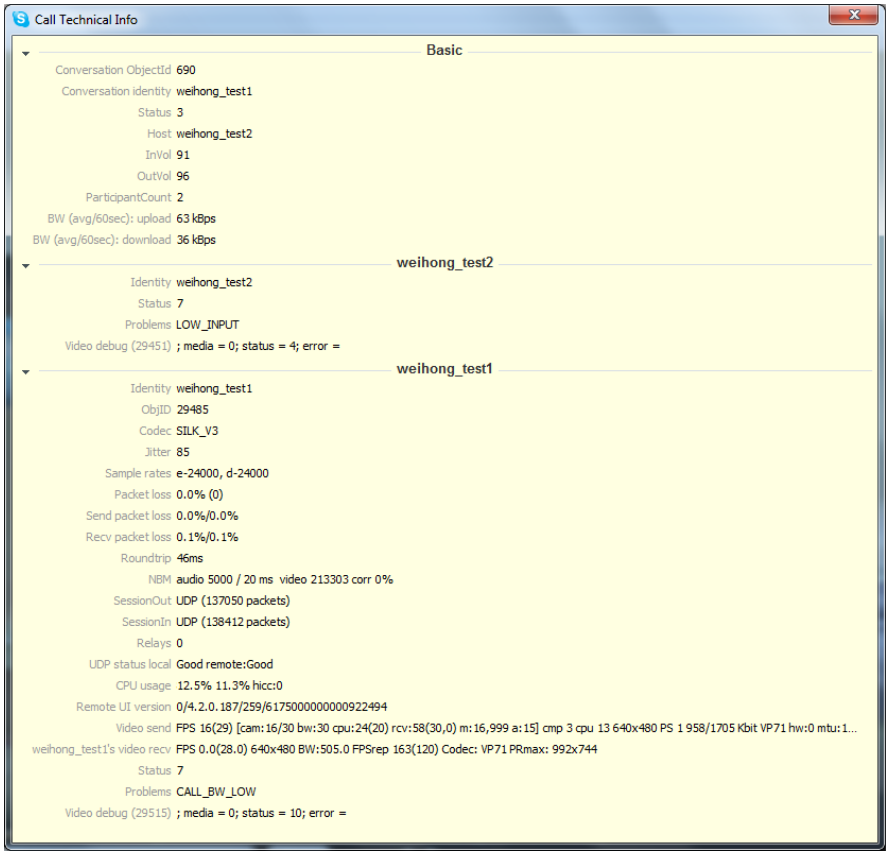
(a)



(b)

Figure 3.18: Screenshots showing Skype Video (a) and Call Technical Info (b) for 25 ms network jitter. Jitter estimated to be 60 ms. Video received at 4 FPS at VGA size. Download bandwidth is 65 kBps.

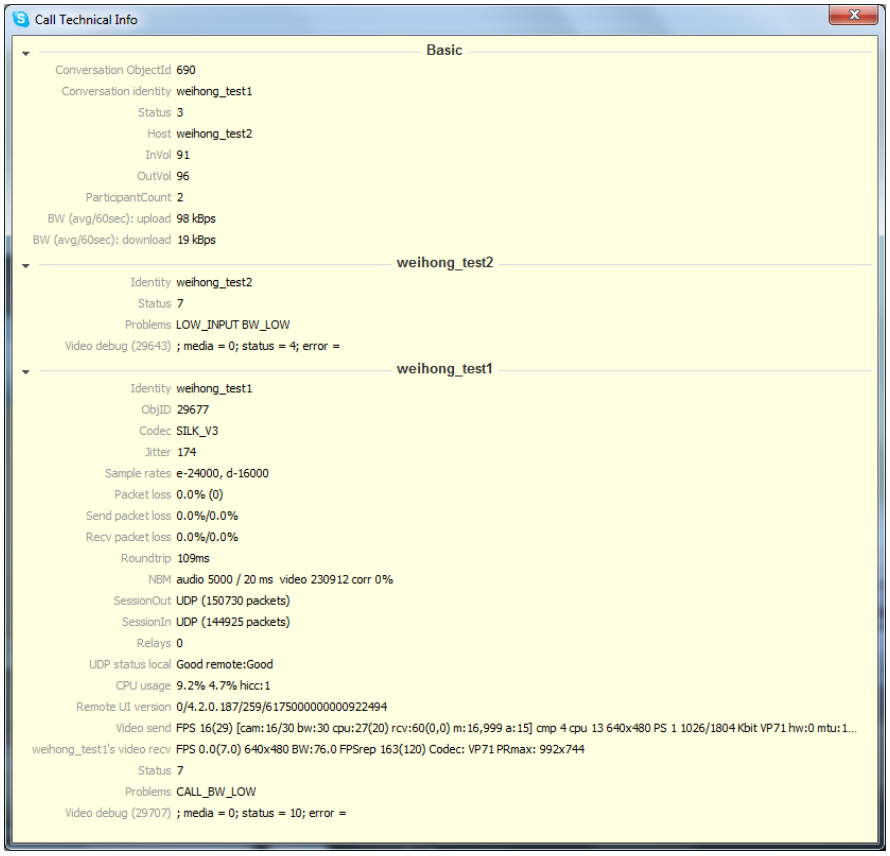
(a)



(b)

Figure 3.19: Screenshots showing Skype Video (a) and Call Technical Info (b) for 50 ms network jitter. Jitter estimated to be 85 ms. Video received at 0 FPS at VGA size. Download bandwidth is 36 kBps.

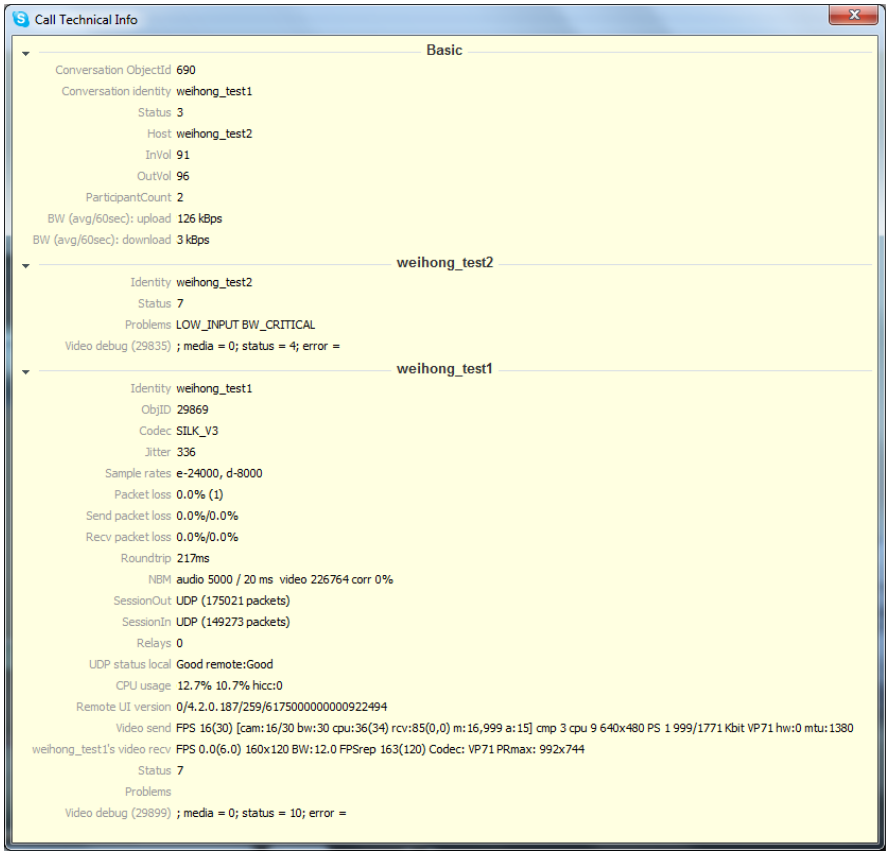
(a)



(b)

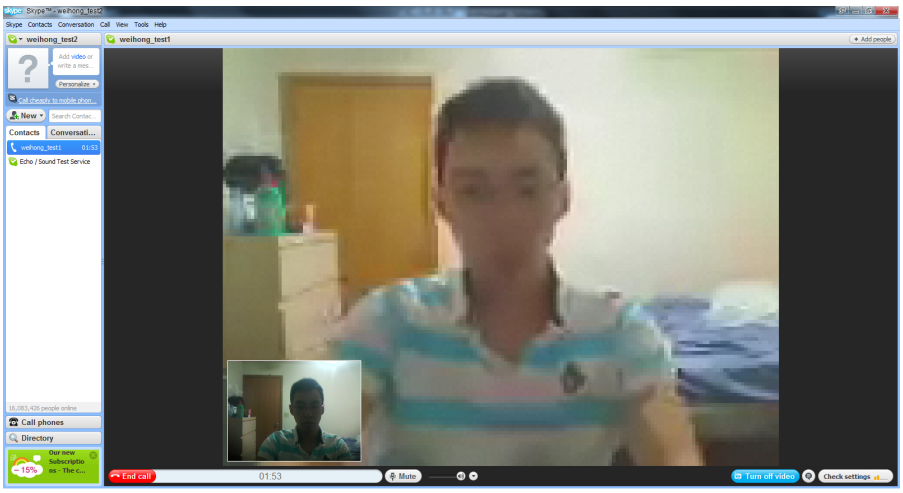
Figure 3.20: Screenshots showing Skype Video (a) and Call Technical Info (b) for 100 ms network jitter. Jitter estimated to be 174 ms. Video received at 0 FPS at VGA size. Download bandwidth is 19 kBps.

(a)

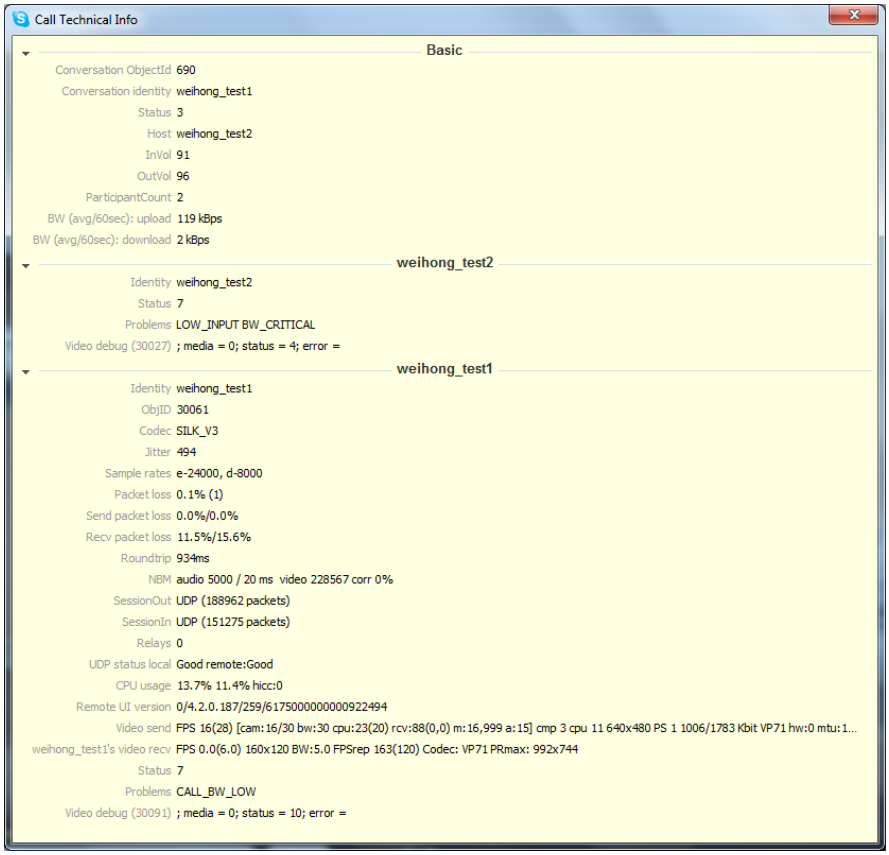


(b)

Figure 3.21: Screenshots showing Skype Video (a) and Call Technical Info (b) for 250 ms network jitter. Jitter estimated to be 336 ms. Video received at 0 FPS at QQVGA size. Download bandwidth is 3 kBps.



(a)



(b)

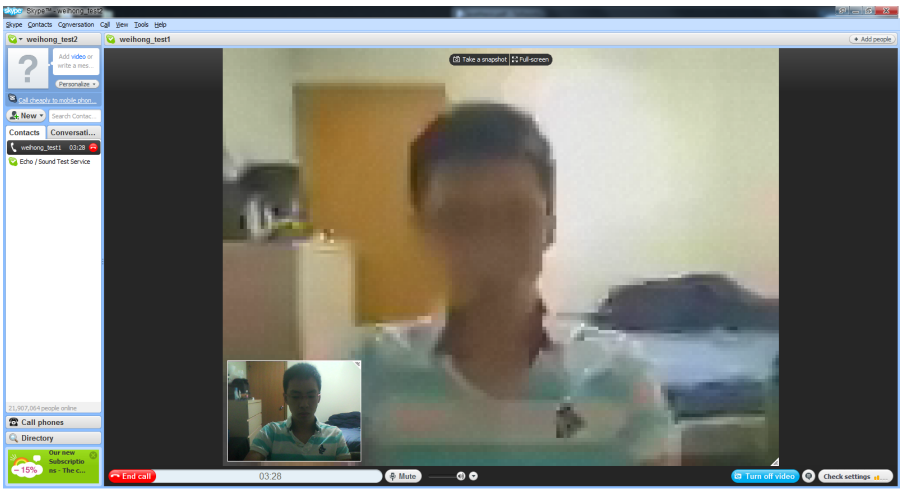
Figure 3.22: Screenshots showing Skype Video (a) and Call Technical Info (b) for 500 ms network jitter. Jitter estimated to be 494 ms. Video received at 0 FPS at QQVGA size. Download bandwidth is 2 kBps.

Reaction to Improvements in Network Conditions. When the network is returned to an ideal state of low delay and no losses, it is observed that the system does not recover immediately. In order to demonstrate this effect, 30% packet error rate is set as the network condition of the system. The result of that is shown in Figure 3.23. The video is severely blurred and jerky. Frame rate is down to 0 FPS and video size is the smallest (QQVGA). Jitter value is high at a value of 89 ms. Download bandwidth is minimal at 1 kBps.

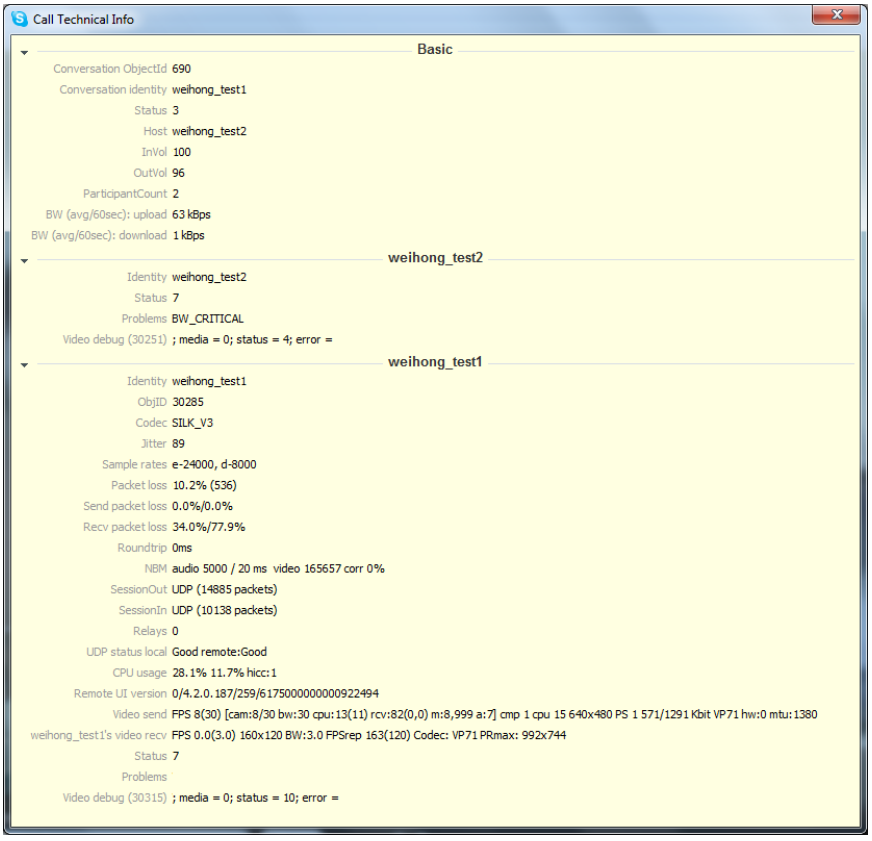
The 30% packet error rate is subsequently removed, and it is observed that often about ∼8 mins Skype recovers back to perfect quality. It is observed in Figures 3.24 - 3.25 that Skype recovers slowly in the first 6 mins by increasing its frame rate, while keeping the image size at QQVGA. It is noticed that the measurement of packet loss percentage drop to 0% within the first minute; however, the recovery of quality takes a much longer period of time

At the 6 min mark when the frame rate hits a maximum of 30 FPS, Skype drops the frame rate back to zero and increase the frame size by a level to QVGA. Over the next minute, it slowly increases the frame rate again, while keeping frame size constant. Once the frame rate hits a maximum of 30 FPS, Skype drops the frame rate back down to zero and brings the frame size up again to the next and final level (VGA), after which it spends about another minute to bring the frame rate back up to 30 FPS. This process is shown in Figures 3.26 - 3.29.

The download bandwidth is seen to increase steadily across Figures 3.23 - 3.29, from 1 kBps to 101 kBps.



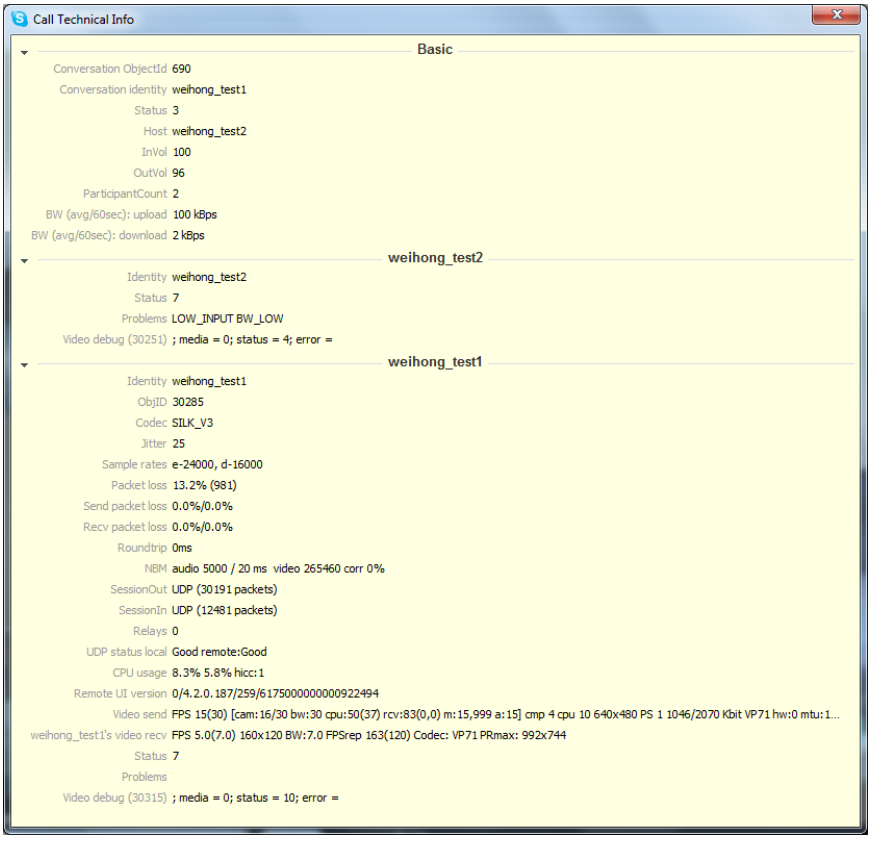
(a)



(b)

Figure 3.23: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting Skype's initial state with 30% packet errors at 0 min. Video shows a severe degree of bluriness. Packet loss percentage is 34%. Video received at 0 FPS and QQVGA size. Download bandwidth at 1 kBps. Jitter value at 89 ms.

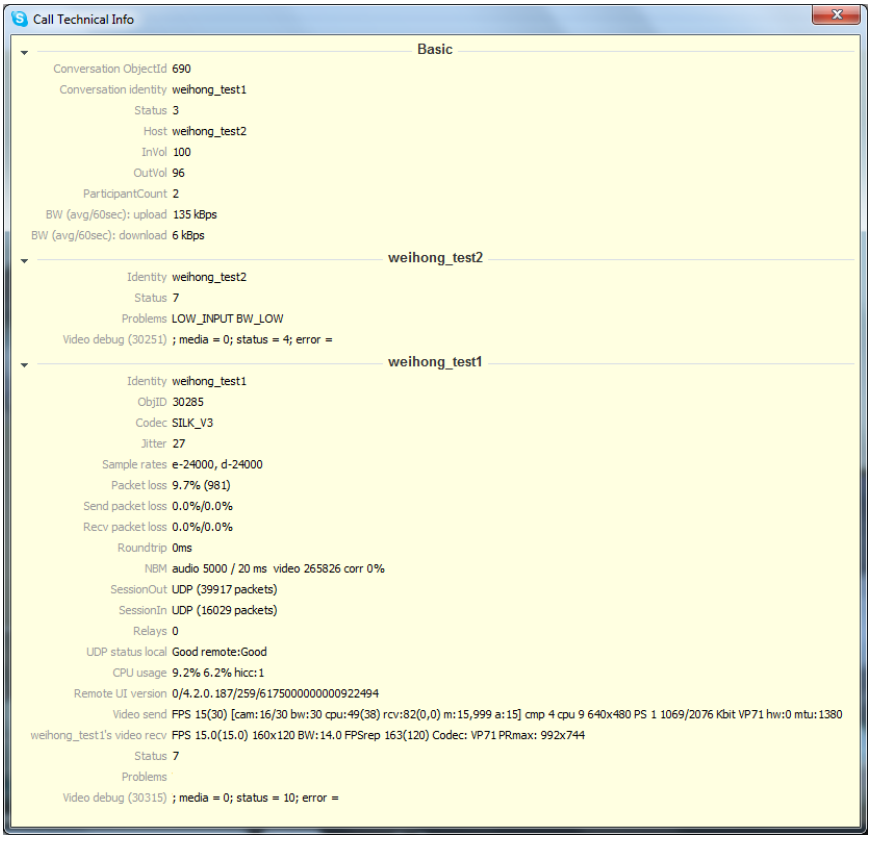
(a)



(b)

Figure 3.24: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (1 min) after the network condition is reverted to zero errors. Video shows bluriness reduced from Figure 3.23(a). Packet loss percentage is 0%. Video received at 5 FPS and QQVGA size. Download bandwidth at 2 kBps. Jitter value at 25 ms.

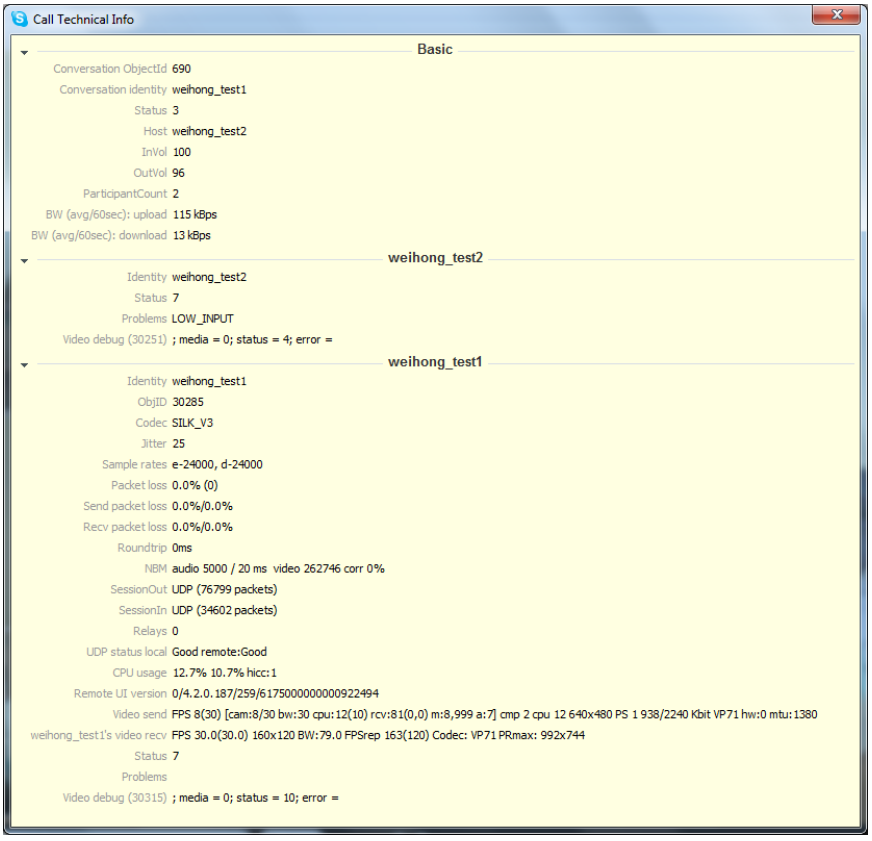
(a)



(b)

Figure 3.25: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (3 mins) after the network is reverted to zero errors. Bluriness in video about the same as Figure 3.24(a). Video frame rate increased to 15 FPS. Video size remains at QQVGA. Download bandwidth increased to 6 kBps.

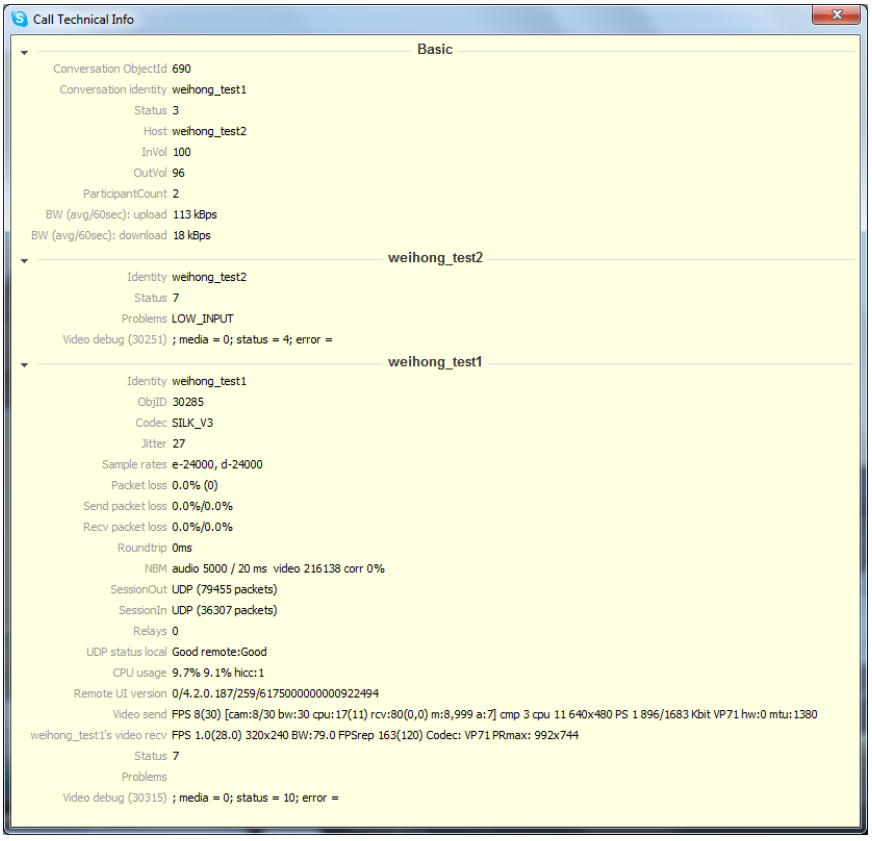
(a)



(b)

Figure 3.26: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (6 mins) after the network condition is reverted to zero errors. Bluriness in video about the same as Figure 3.24(a). Video frame rate increased to 30 FPS. Video size remains at QQVGA. Download bandwidth increased to 13 kBps.

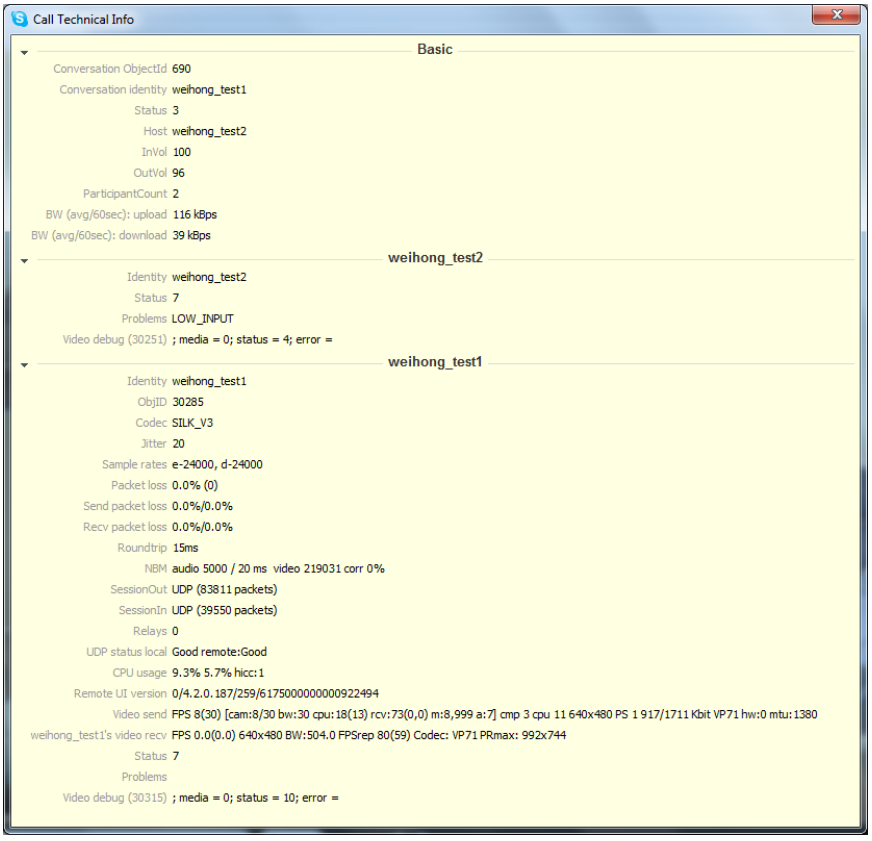
(a)



(b)

Figure 3.27: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (6+ mins) after the network conditions is reverted to zero errors. Image quality improved from Figure 3.24(a). Frame rate dropped to 1 FPS but frame size increased to QVGA. Frame rate slowly increase till 30 FPS. Download bandwidth increased to 18 kBps.

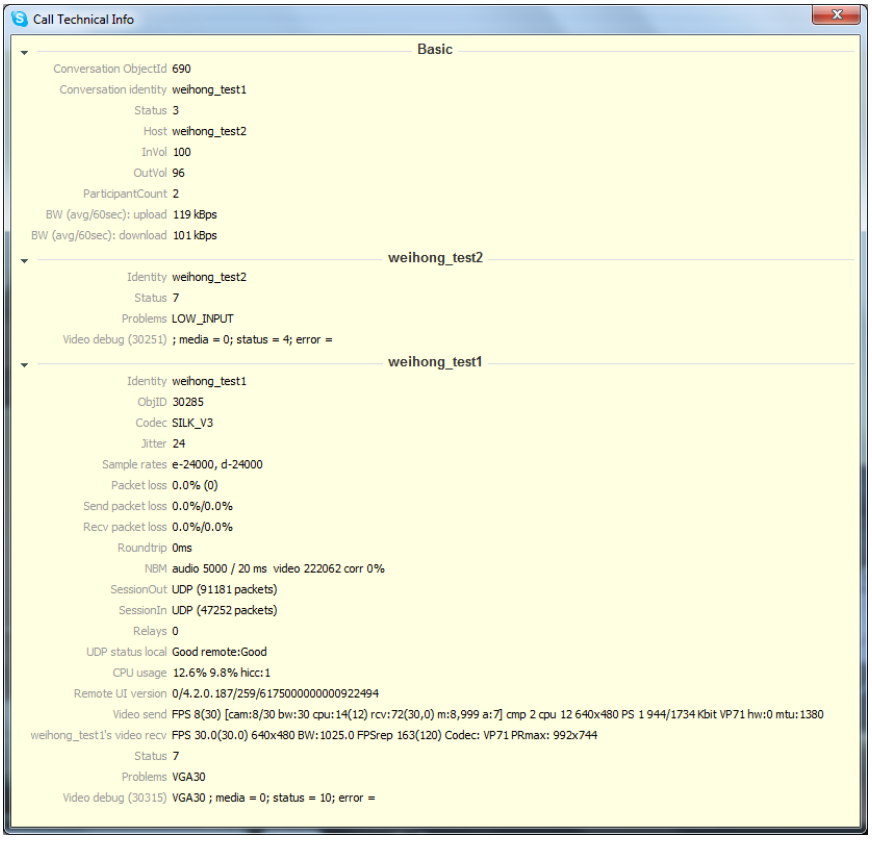
(a)



(b)

Figure 3.28: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (7 mins) after the network condition is reverted to zero errors. Perfect image quality is observed but video is not smooth. Frame rate dropped to 1 FPS and frame size increased to VGA. Download bandwidth increased to 39 kBps.

(a)



(b)

Figure 3.29: Screenshots showing Skype Video (a) and Call Technical Info (b), reflecting state of Skype (∼8 mins) after the network condition is reverted to zero errors. Perfect quality is observed. Video received at 30 FPS and VGA size. Download bandwidth increased to 101 kBps.

3.3 Summary

In this chapter, we have presented a setup that allows us to perform tests on existing systems and evaluate their reactions to network changes. We have shown the results of using the setup to evaluate the reactions of Skype to various changes in network conditions. As we have seen, Skype adjusts the frame rate, frame size and MED of the system in accordance to changes in the network. However with this testbed, we are not able to verify if those changes made by Skype will always give the optimal perceptual experience.

When determining the best operating configuration (frame rate, frame size, MED) for a system, we need to have a system in which we can adjust those parameters. In the next chapter, we present a new testbed setup. In this new setup, we have total control over various components of the video-conferencing system. This will allow us to perform the investigations in order to determine the perceptually optimal operating point of the system for a specific network and conversational condition.

CHAPTER 4

DESIGN OF A NEW TESTBED FOR THE EVALUATION OF VIDEO CONVERSATION QUALITY

Structure of Testbed The testbed consists of a total of 7 stages as shown in Figure 4.1. Its input parameters are MED, frame rate and frame size. The output video will be generated with the indicated MED, frame rate and frame size. The output video will consist of the video conversation of Speakers A and B pieced together side by side, with A on the left and B on the right. It will reflect the point of view of a third party looking at the video conversation standing next to Speaker A. Hence, the video of Speaker A has no network degradations, and his response time will be just his HRD. Speaker B is emulated to be the client across the network. Thus it will be observed that the video and speech suffers from the increase in MED and degradations due to network conditions. Therefore, Stages 4 and 5 only apply to Speaker B.

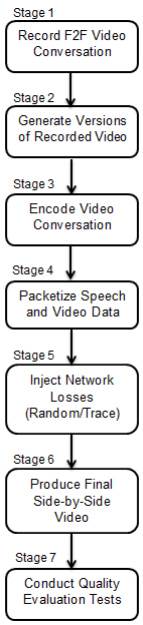


Figure 4.1: Flow chart showing the structure of the testbed.

Table 4.1: Recordings of 4 face-to-face conversations.

Rec. No.	Conv. No.	Background type	Avg. single-talk duration	Avg. HRD duration	# of switches	Total Time	Switching Frequency
1	1	simple static	1011ms	1076ms	11	26.1s	25.3/min
2	1	outdoors	1185ms	956ms	11	27.0s	24.4/min
3	2	simple static	5496ms	552ms	6	52.4s	6.8/min
4	2	complex static	5461ms	1106ms	6	47.0s	7.6/min

4.1 Stage 1: Recording of a F2F Video Conversation

The first stage in the testbed requires the recording of a face-to-face (F2F) conversation between the two parties involved. The ideal case for a video-conferencing system is to provide a video conversation experience that is similar to that of a face-to-face conversation. The video is recorded with the two participants speaking face-to-face with each other at the highest possible frame rate (30 FPS) and frame size (VGA). The recorded video represents the best result that the output of the system can achieve.

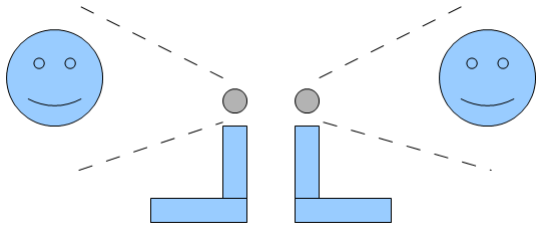


Figure 4.2: Recording setting.

To capture the F2F video conversation at the highest quality, we used two laptops equipped with Logitech webcams that can record videos at 30 FPS and VGA size. The two speakers will face each other with the laptops in between them to capture the conversation. Hence the MED for the videos recorded will be zero, as there is no network delay. A pictorial description can be seen in Figure 4.2.

Table 4.1 shows the statistics of the conversations recorded using the setup. We have two conversations, and each conversation is recorded in two different setting. Conversation 1 is recorded once in an office room and another time in an outdoor environment. Conversation 2 is record once in the same office room and another time in a conference room, with many objects in the

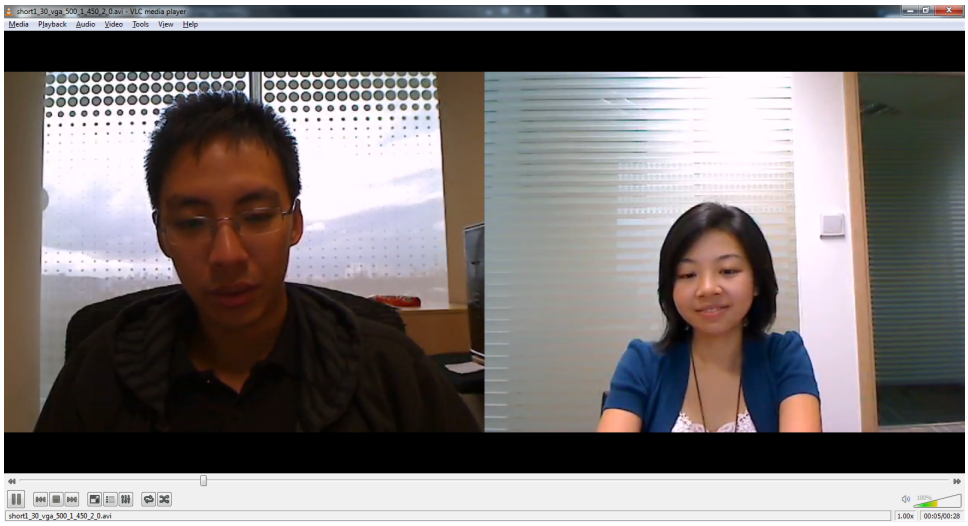


Figure 4.3: Simple static background. Videos of client A and B pieced together side-by-side.

background rather than a simple wall. Note that the recordings satisfy the condition of no rapid motion in the background. It can be observed from Table 4.1 that even for the same conversation, different recordings will have different durations, average HRDs and average STs. Hence it is necessary for the video generation algorithm developed in the testbed if the conversation conditions are to be kept the same. Figures 4.3 to 4.5 show the resulting side-by-side videos that can be played by a VLC media player for subjective evaluations. Figure 4.3 shows a typical screenshot for recording 1 and 3. As shown in the figure, the background is a typical office setup consisting mostly of simple shapes. Figure 4.4 shows what recording 2 looks like. It is conducted in a rooftop garden. Hence there are more moving elements in the scene and background noise in the recording. However the additional moving elements do not violate Assumption 3, which is defined in the next section, as they are random motion (e.g. rustling of hair and grass by the wind). Recording 4 is shown in Figure 4.5. The scene is that of a conference room with many random objects comprising complex shapes and sizes in the background.

The motivations behind the recordings are such that we can compare different conversation and background conditions for different network conditions.

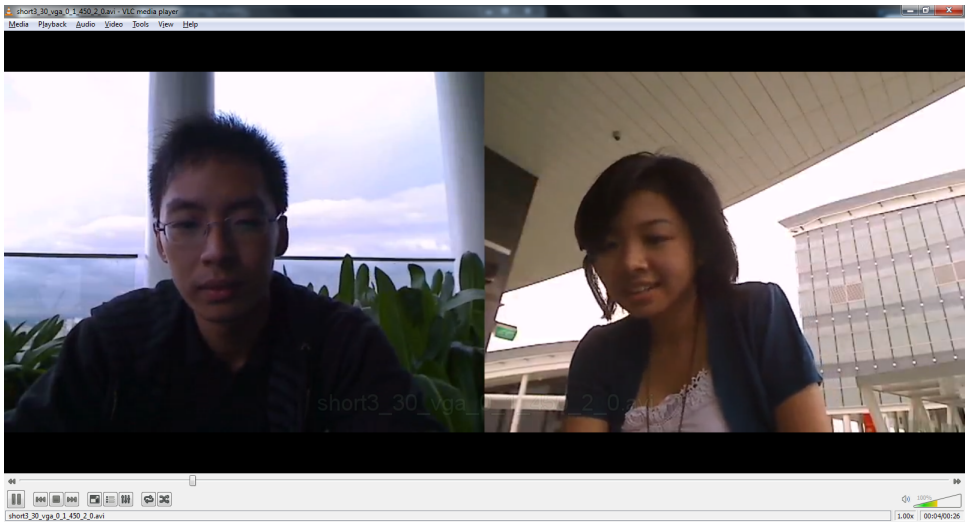


Figure 4.4: Outdoor environment with some background movements, e.g rustling of hair and grass by wind

4.2 Stage 2: Generating Multiple Versions of the Recorded Video

As it is not possible to perform an identical recording for every different set of system parameters (frame rate, frame size, MED), it is necessary to be able to generate videos that can be collected under different configurations from this one recording. Hence we have created a video generation algorithm that takes as input the desired system parameters and the recorded video. It will then output the corresponding video for that set of parameters. It is assumed that all videos conversations are scenes of a person talking with minimal rapid motion in the scene

Figure 4.6 shows the timeline of the frames in a recorded face-to-face conversation. The frames are labeled with respect to their speakers and frame numbers. The frames which show that a participant is speaking are highlighted in bold. There are a couple of issues that arise when we change the video parameters. Speech does not face the complications that video faces. When there are changes in the frame rate, or when delays need to be inserted, we can simply insert period of silences to get the timing right. However, it is not that trivial for the case of videos as there is no such thing as a "silence frame" that can be inserted. New frames need to be created according to

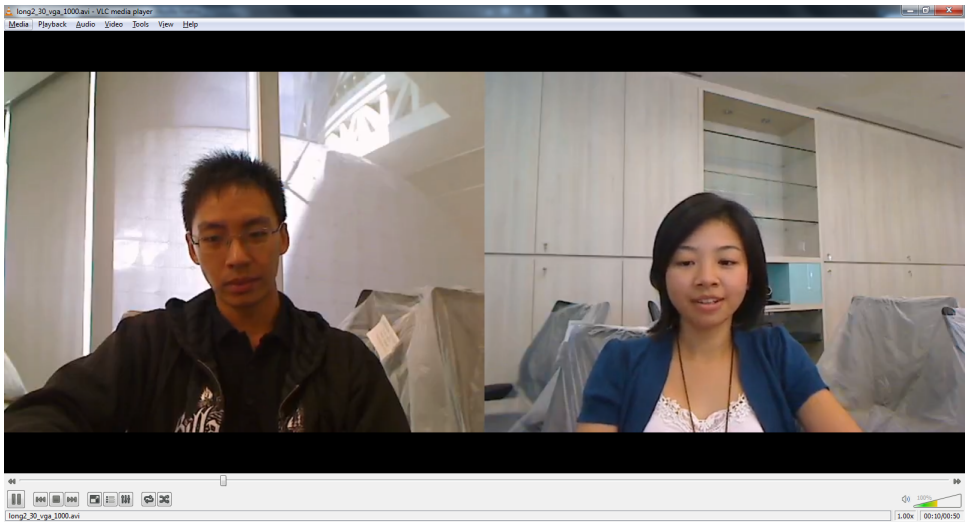


Figure 4.5: Complex static background

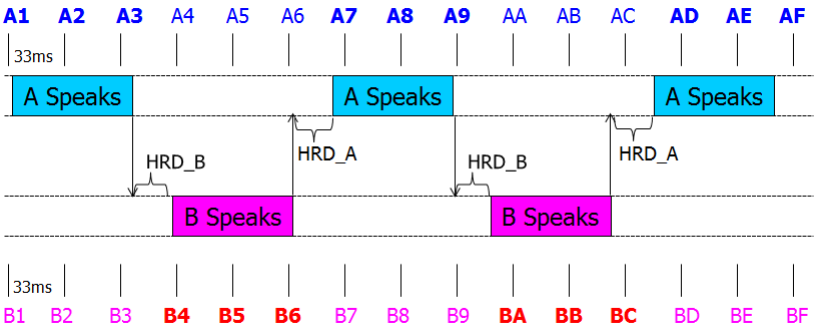


Figure 4.6: Timeline showing frames in a recorded face-to-face conversation. Frames where the speaker is talking are highlighted in bold.

the new parameters from the recorded frames. We have a couple of figures to illustrate the different scenarios and the problem faced in each of them. In the figures, the topmost and bottommost rows show the location of the frames recorded after delay has been inserted. The inner two rows show the location of the new frames. New frames that do not coincide with any existing frames have to be generated and are labeled by question marks in the figures. The frame period of the output file is shown in each of the figures (Figures 4.6 - 4.12).

Issues Arising from Changing Parameters Firstly, when network delay that is a multiple of the frame period is inserted, new frames need to

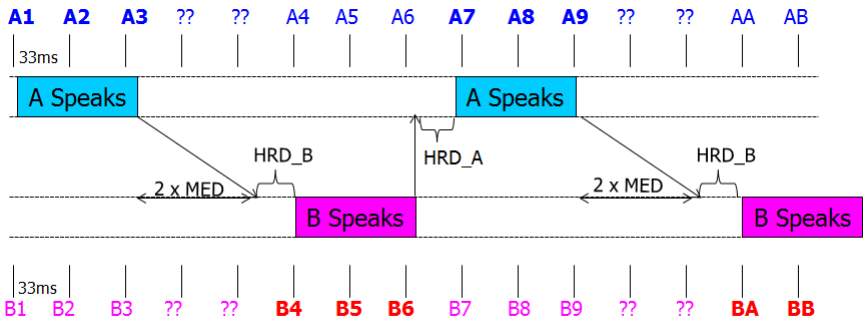


Figure 4.7: Scenario 1: Network delay of multiple frame periods inserted.

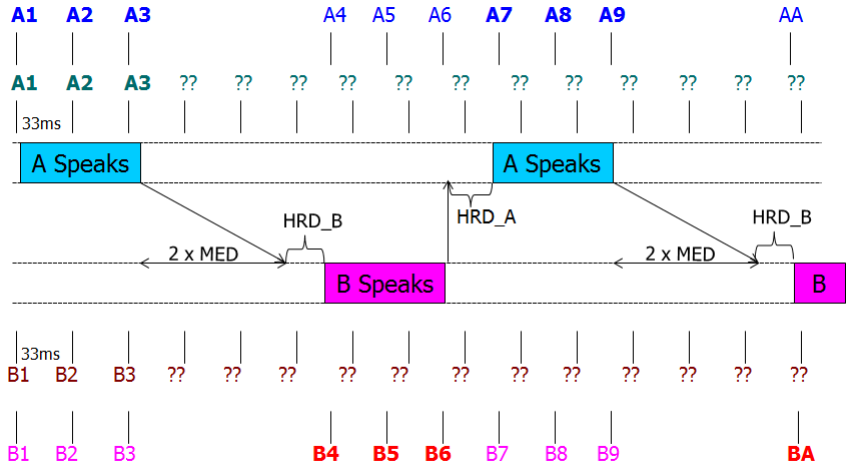


Figure 4.8: Scenario 2: Network delay of fractional frame period inserted.

be created to fill the gap where one speaker is waiting for the other speaker to respond. This scenario is depicted in Figure 4.7. Secondly, there is the problem when the new frame period is not a multiple of the current frame period. As a result, many of the new frames do not coincide with the recorded frames, as shown in Figure 4.8. Lastly, when the frame period of the new video conversation is changed, combined with the first two scenarios, we will have the situation shown in Figure 4.9. To the end, we have devised a video generation algorithm that address all these issues.

The algorithm is based on the following assumptions and substantiated by corresponding references:

Assumption 1 *Shifting of 1 frame within ±1/2 of the frame period (15 ms) is not perceptible.*

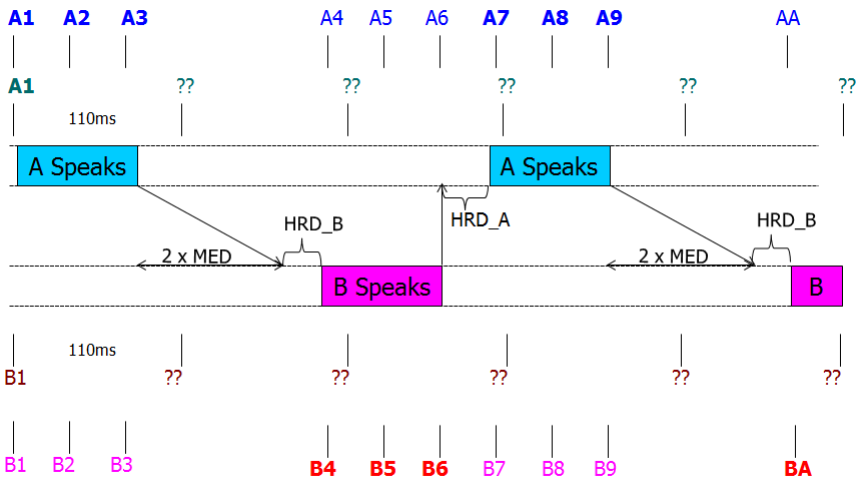


Figure 4.9: Scenario 3: Different frame period and fractional delay.

In [39], results show that increasing the frame period from 33 ms to 50 ms has no significant drop in perceptual quality over a myraid of video sequences. In the 50 ms frame period version of the video, all the frames will be similar to the original shifted around by a maximum of ± 17 ms. Hence we can make the assumption that if we shift a frame to within ± 15 ms, it is not perceptible

Assumption 2 *The duplication of a frame once is not perceptible.*

In [40], it is stated that for start-end transitions in videos, people are able to notice if the transition duration gets extended by two frames or reduced by three frames. Hence, this result means that for the transition frame, if it is duplicated once, it will not be perceptible. The transition sequence, however, has to be smooth and have no sudden motion, which the recorded video conversations satisfies.

Assumption 3 *The region of frames where delay is inserted does not contain rapid movement.*

According to [41], "However for videotelephony applications, it was found that on average upwards of 90% of the pixels can be considered low-energy

pixels and left unreplenished without causing visual discrepancy (p. 439)." This means that for a video conversation, most of the frames are similar. This will be even more so during the time where the participant is sitting and waiting for the other party to respond.

Algorithm 1 Pseudocode for Video Generation Algorithm

1: **for** each new video frame N_x **do**
2: **if** N_x lies within $\pm 1/2$ frame period of recorded frame R_y **then**
3: $N_x = R_y$
4: **else**
5: N_x remains undefined
6: **end if**
7: **end for**
 {at this point, all remaining unallocated frames are located at the gaps due to delay insertion}
8: **for** each remaining undefined new frame **do**
9: apply gradient filling using nearby recorded idle frames
10: **end for**

The pseudocode of the video generation algorithm is shown in Algorithm 1. Figure 4.10 shows how Steps 1 to 7 of Algorithm 1 are applied to the situation depicted in Figure 4.8. By comparing Figure 4.8 with Figure 4.10, some of the undefined frames have been defined with existing frames. However, some of the undefined frames are still undefined, which will be resolved in the later steps.

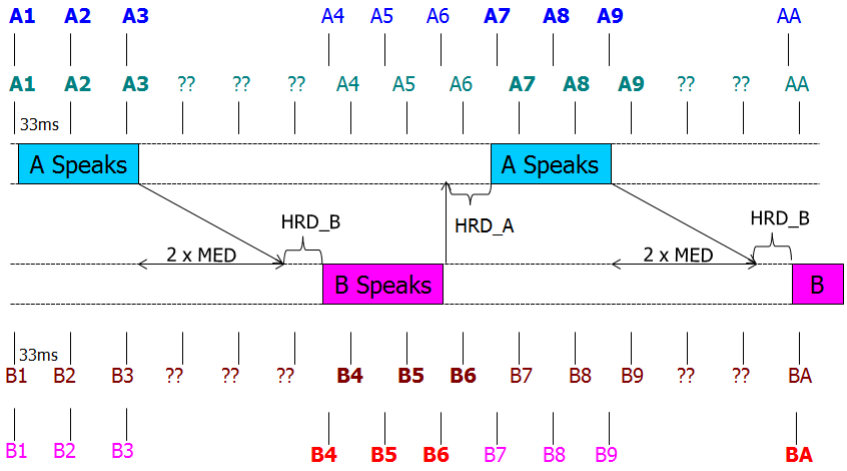


Figure 4.10: Catching up and pulling back. Algorithm 1: Steps 1 to 7.

For each of the new undefined frames in Figure 4.8, we look within $\pm 1/2$ of the frame period (15 ms) for any recorded frames. If a recorded frame is found, that recorded frame is used to define the undefined frame. Referring to those frames for Speaker A in Figure 4.10, we can see that for the first three undefined new frames, we are not able to find any recorded frames in the vicinity. Thus they cannot be defined. The next six undefined new frames, however, fall within the vicinity of A4 to A9. Hence A4 to A9 are used to define using those frames. The same procedure is performed for the frames for Speaker B. Steps 1 to 7 of Algorithm 1 are called *catching up and pulling back*. This part of the algorithm is based on Assumption 2.

As shown in Figure 4.10, some of the frames in the new video sequence are still undefined, as they are too far away from these existing frames. The existence of these frames is due to the insertion of MED that was not present in the recording. Steps 8 to 10 for Algorithm 1 address this issue. This next stage of Algorithm 1 is called *gradient filling*.

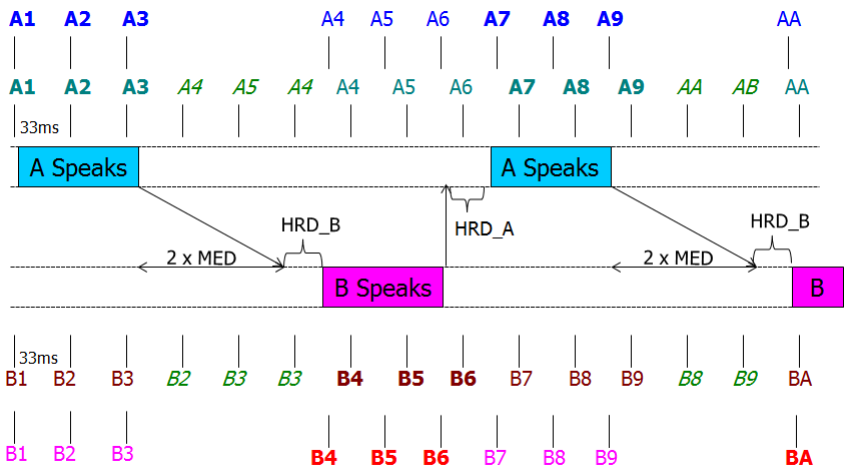


Figure 4.11: Gradient filling. Algorithm 1: Steps 8 to 10.

To perform gradient filling, we first identify where the remaining undefined frames are and the number of consecutive undefined frames. Referring to Figure 4.10, we can easily recognize that the first series of undefined frames is between A3 and A4. In that section, we have 3 consecutive undefined frames.

Now we can use the neighboring frames to fill up this gap. The neighboring frames have to be chosen from periods where the speaker is idle (not

speaking). Otherwise, there will be rapid motion of the lips and Assumption 3 will be void. The generated video will then look unnatural. The idea is to use frames to show the speaker being idle. A sequence of recorded idle frames will be used, rather than to just duplicate a single idle frame. As under Assumption 2, any consecutive sequence of more than two similar frames will be detectable.

With reference to Figure 4.11, the first frame in the gap of undefined frames is defined using A4, which was originally the frame after A3. The second undefined frame is then filled with A5, which is the frame following A4. This ensures the video looks smooth and natural. Once we hit the middle of the series of undefined frames, we reverse the order. Hence the third undefined frame is filled with A4. This ensures that the transition at the end of the series of undefined frames is smooth. Note that A4 and A5 are all recorded frames during the period when A is idle. This procedure only works if the sequence of frames used satisfies Assumption 3. Note that for Speaker B, the frames used are B2, B3, and B3 and not B4, B5, and B4 as B4 and B5 are not idle frames.

At this point, all new frames are now defined using existing recorded frames. Note that in the new sequence, there are at most two consecutive frames that are the same, satisfying Assumption 2.

Figure 4.12 shows how the same algorithm is applied to the scenario shown in Figure 4.9.

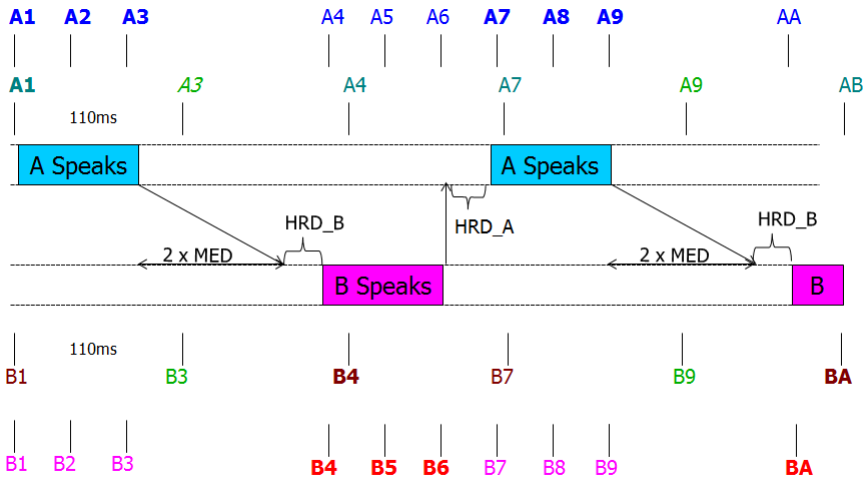


Figure 4.12: Video generation algorithm applied to Scenario 3.

4.3 Stage 3: Encoding the Video Conversation

The video conversations at this point are in a raw form encapsulated within AVI file containers [42]. During this part of the testbed, the video conversations are encoded using the desired codecs. For this research, we have chosen H.264 as the video codec, as at this point in time, it is the most recent well deployed video codec used in video-conferencing systems, where its source code is accessible. For similar reasons, we have chosen G.726 as the speech codec. Both codecs are ITU-T recommendations [12, 16]. In the testbed implemented, we used the version of the codecs in the FFmpeg package v.0.6.1.

The FFmpeg package is an open source software that has libraries of audio and video codecs. It uses libx264, version 1745, a freely available open source H.264 codec that can encode and decode H.264 streams efficiently in real time (30 frames per second). They have been released under the GNU GPL [43].

As video frames can usually be larger than the maximum transfer unit (MTU) of *1500 bytes*, the H.264 codec breaks the video frames into small independently decodable chunks called NALUs (Network Abstraction Layer Units) [16]. The H.264 codec source code from the libx264 library of the FFmpeg package has been modified for the testbed such that each NALU is less than 1500 bytes, so that each of them can fit in one packet. It is found that if the full 1500 bytes is used for a NALU, some packets will be packed full with a single NALU, while many other packets only contain 100 bytes of audio data, as we will not be able to fit the audio and video data chunks together into one packet. As a result, many packets are underutilized and will result in an excessively high packet rate. Through a series of trials, we have concluded that *250 bytes* is a good size for the NALU to allow good packet utilization when packed together with the small audio chunks ($\sim$*90 bytes* each).

4.4 Stage 4: Packetizing Speech and Video Data

This stage only applies to Speaker B. In this stage, the encoded speech and video data streams in the AVI files are parsed. The speech and video chunks are extracted accordingly. Within the AVI file, the speech and video data chunks are interleaved. Data chunks that correspond to the earlier frames are

found in the beginning of the file. The data chunks are extracted serially out of the AVI file. They are then placed into packets based on a packetization strategy. Piggybacking redundancy is added in this stage. Packetization strategy is as shown in Algorithm 2.

Algorithm 2 Packetization Strategy

```
1: while not end of AVI file do
2:    Read data chunk
3:    Determine size of data chunk
4:    Remove data chunks in current packet that have been sent R times,
      where R is the piggybacking degree
5:    if packet has space for new data chunk then
6:       add data chunk to packet and send packet
7:    else
8:       send packet and goto step 4
9:    end if
10: end while
```

The entire AVI file is parsed to locate all the data chunks. Every AVI file starts with a header marked with the four-character code (FOURCC) "RIFF". From this header, we can extract information about the total size of the file. The structure of the AVI file consists of multiple lists and headers. To find the first data chunk, the file is parsed serially until the FOURCC "movi" is found. This marks the beginning of the list of data chunks.

The structure of the data chunk is marked with a starting FOURCC as the chunk id, followed by four bytes indicating the size of the chunk. The chunk id can be either "00dc", indicating it is a video data chunk on stream 0 or "01wb", indicating it is an audio data chunk on stream 1. The rest of the chunk is the encoded video/audio data.

The chunk is then placed into the current packet and tagged with a copy number, which reflects how many times the chunk has been sent. When the copy number of a chunk is equivalent to the piggybacking degree, it is removed from the next packet.

In addition to what is shown in Algorithm 2, it is ensured that for each packet, the new data added is less than MTU divided by the piggybacking degree. It is also enforced that within the new data for each packet, no data from more than two new frames can be added. This is to ensure that packets do not incur too much packetization delay . A packet can only be sent when all the data in the packet is ready. Furthermore, if too many frames are

placed in one packet, when that one packet is lost, information on too many frames will be lost.

The packets are sent based on an uneven packet transmission rate (UPTR) [44]. In UPTR, packets are not sent at fixed intervals. The packet rate varies during times when there are frames to be sent and when there are not. Essentially when frames are ready, the packets are sent at a higher packet rate to minimize the processing delay. The packet rate then falls to a low rate during the time where no new frames are to be sent. The average rate is still kept at a maximum packet rate of *50 packets per second*. This has been shown to be adequate based on past experiments.

4.5 Stage 5: Injecting Network Losses (Random/Trace)

This stage only applies to Speaker B. Taking the packet stream as input, this stage determines whether each packet is received or not. It will take the piggybacking degree into its calculation and determines whether individual data chunks (speech/video) are received or not. For random losses, a random number generator is used to determine if a packet is lost or not. For losses based on a packet trace, each packet is tagged with a received time based on a network trace. All data chunks in the packet will have the same received time. For each data chunk, its received time is compared with its scheduled playout time determined by the frame rate. If it is late, the chunk will be marked as lost. If all sent copies of the data chunk are lost, changes are made to the AVI file to mark that data chunk as lost.

4.6 Stage 6: Producing the Final Side-by-Side Video

The video recordings of Speakers A and B are pieced together side-by-side into one AVI file that is playable by the VLC player, using a combination of FFmpeg and AVISynth scripts [43, 45, 46]. For the codecs chosen, the VLC player has built-in decoders for those codecs and hence is the recommended player to use. Figures 4.3 - 4.5 show some screenshots of the output videos played in VLC. Speaker A is on the left and Speaker B is on the right. Any

effects of network degradation will be seen only for Speaker B.

4.7 Stage 7: Conducting Quality Evaluation Tests

Once the final video is generated, we can conduct a series of objective and subjective tests to evaluate the quality. For objective metrics, we calculate the VQM, PESQ, CS and CE of the video, comparing it with the original recorded conversation. VQM is used to represent the one-way video quality. PESQ represents the one-way LOSQ. CS and CE represent the interactivity of the conversation. Reference to VQM software can be found at [25].

The final video is verified to meet the frame rate, frame size and MED modifications using FFprobe. FFprobe is a software that is part of the FFmpeg package that allows us to measure the frame rate, frame size and duration of the generated video. Using the recorded videos described in Table 4.1, all generated videos have been verified to satisfy the input specifications. Subjective tests have been carried out to verify that all generated videos look perceptually natural.

4.8 Summary

In the chapter, we have presented the setup of a testbed that allows the recording of a face-to-face conversation at high quality and zero delay.

We have also presented the algorithm to generate different versions of the recorded video at different frame rates, frame sizes and MEDs.

Lastly, we have shown how we can extract the data from AVI files, packetize it and inject losses to obtain an output video that will show the effects of a simulated network condition. The processed videos of the two speakers are then pieced together to produce a side-by-side video conversation that can be used for subjective and objective tests.

Using this setup, we can now perform pair-wise comparison tests between two conversations of different network and conversational conditions to see which is better. We can also make observations on how changing one parameter affects the overall perceptual quality through subjective and objective tests.

CHAPTER 5

USING THE TESTBED TO STUDY THE EFFECTS OF VIDEO AND NETWORK PARAMETERS ON PERCEPTUAL QUALITY

This chapter highlights the results of some preliminary tests conducted. As the number of tests conducted is not extensive, the results here are to be taken as our observations and not assumed to be always true. This thesis does not propose a solution to finding the optimum input parameters for any network and conversational condition. However, the testbed can be utilized for conducting subjective tests to find the optimal set of parameters for a specific network and conversational condition.

All tests are conducted using random losses and recording 1 as described in Table 4.1. The default parameters used, unless otherwise stated, are 30 FPS, VGA size, piggybacking of degree 1 (no redundancy) and 2 new frames per packet. All subjective evaluations are done with one subject only due to resource limitations, and hence may not be generalized. However, it does provide us with a preliminary idea on the effects of these parameters on the perceptual quality of the video conversation.

A limitation of VQM is that it is not able to compare videos of different frame rates and frame sizes. Hence, the measurements are taken by comparing the test videos with the highest bit rate and no losses for that specific frame rate and frame size.

5.1 Impact of Changes in Frame Rate

In order to investigate how adjusting the frame rate will affect the overall quality under various bandwidth conditions, we kept all other parameters constant and varied the frame rate and the available bit rate. Using the testbed, we generate all the videos using the desired parameters. PESQ and VQM measurements are then taken. At the same time, the videos are

Table 5.1: VQM values of video conversations under different frame rates and bit rates.

Frame Rate (FPS)	Bitrate (kBps)			
	56	25	16	9
30	0.004	0.196	0.266	0.356
25	0.006	0.186	0.242	0.337
20	0.011	0.177	0.244	0.323
15	0.016	0.166	0.219	0.345
10	0.026	0.153	0.190	0.291
5	0.072	0.150	0.183	0.251

subjectively evaluated.

From our subjective evaluations, the following are observed:

- When the frame rate is reduced from 30 FPS down to 20 FPS, there is no perceptually noticeable degradation in quality. This matches the results found in [39].
- When the frame rate falls to 15 FPS and below, it is observed that the video becomes more jittery.
- When the frame rate falls below 5 FPS, the movement is very jerky and slow. The perceived quality is considered to be low. Lips may appear to be out of sync.

The VQM results are shown in Table 5.1. PESQ readings are consistently at 4.50 (perfect quality), indicating that changing the frame rate does not affect the speech quality under no-loss conditions. This is expected as speech requires very little bandwidth relative to video.

From Table 5.1, we can see that for high bit rate (56 kBps), the VQM values increase with declining frame rate, indicating that for high bit rate, a high frame rate is preferred. For low bit rate (9 kBps) the VQM values decrease with declining frame rate, indicating that a lower frame rate is preferred. The results are plausible because when bandwidth is limited, lowering the frame rate allows more bits to be used for the encoding of each frame. Hence each frame can be encoded at a lower quantization level, giving better quality.

However, by comparing the results of our subjective evaluations and VQM measurements, it is observed that VQM is not able to adequately capture the effects of frame rate. Any frame rate lower than 15 FPS does give us

Table 5.2: VQM values of video conversations with different frame size and bitrate

Frame Size	Bitrate (kBps)			
	56	25	16	9
VGA (640x480)	0.004	0.196	0.266	0.356
QVGA (320x240)	0.000	0.101	0.115	0.176
QQVGA (160x120)	0.031	0.063	0.060	0.095

better video image quality, but the video sequence is not smooth. Hence other metrics need to be considered apart from VQM, if the effects of frame rate are to be captured. An example of such a metric in development can be found in [47].

5.2 Impact of Changes in Frame Size

In this section, we present our results on varying the frame size under different bandwidth conditions while keeping all other parameters constant. As in the previous section, all videos are generated using the testbed. Subjective tests and objective measurements (PESQ and VQM) are carried out on the videos.

For all cases, the PESQ remains at 4.5, representing perfect quality. This means that changing the frame size and bandwidth does not affect the speech quality. This is expected, as speech requires very little bandwidth relative to video. The VQM measurements are shown in Table 5.2.

From the results in Table 5.2, we can see that when the bit rate is low, a smaller image gives better quality because as the frames are smaller, each frame under low bandwidth conditions still has sufficient bits to be encoded at good quality. As observed for VGA videos, the VQM value increases from 0.004 to 0.356 as the bit rate falls from 56 kBps to 9 kBps. However, for QQVGA, the VQM value only increases from 0.031 to 0.095.

5.3 Impact of Changes in Piggybacking Degree

Table 5.3 shows the results of experiments where we vary the piggybacking degree for different loss rates. Table 5.3(a) shows the VQM measurements and Table 5.3(b) shows the PESQ measurements.

Table 5.3: Effects of different piggybacking degrees under different random loss rates

(a) VQM

Piggybacking Degree	Random Loss Pct %				
	0	5	10	15	20
1	0.004	0.210	0.334	0.396	0.430
2	0.196	0.210	0.218	0.260	0.316
3	0.266	0.266	0.267	0.268	0.278
4	0.356	0.356	0.356	0.356	0.371

(b) PESQ

Piggybacking Degree	Random Loss Pct %				
	0	5	10	15	20
1	4.50	3.27	2.80	2.37	2.07
2	4.50	4.10	3.79	3.28	3.33
3	4.50	4.50	4.50	4.39	3.80
4	4.50	4.50	4.50	4.50	4.14

As seen in Table 5.3(a), when the loss percentages increase, both VQM and PESQ quality falls. This is shown by an increase in VQM values and a decrease in PESQ values. However, by increasing the level of redundancy we can achieve better output quality. For example, for under 10% packet losses, the VQM for piggybacking degree 1 (no redundancy) is 0.334. By increasing the piggybacking degree to 2, the VQM improves to 0.26. However, when the piggybacking degree is increased further, the VQM value increases again. This is due to the fact that by having more redundancy in the system, the amount of bandwidth available is reduced by the same factor. Hence, this shows that the degradation caused by the reduction in bandwidth outweighs the improvement achieved by the extra redundancy. Thus, it is not a safe assumption that using a high level of piggybacking degree and minimizing packet losses will always achieve better output. This implies that in order to achieve the best perceptual output quality, a suitable piggybacking degree dependent on network conditions needs to be found. There has been other research conducted to find the optimal piggybacking degree for other systems, such as [48].

Similar observations can be made in Table 5.3(b) for LOSQ measured by PESQ. By increasing packet losses, PESQ falls from 4.50 to 2.07, with no redundancy in the system. When the piggybacking degree is increased, the

PESQ value does not degrade as much. This scenario can be clearly seen for piggybacking degree 4 where with 15% packet losses, we can still achieve perfect audio quality.

One important difference between speech and video is that by increasing the piggybacking degree, speech does not suffer the degradation due to a reduction in available bandwidth, unlike the case for video. This can be clearly seen in the column for 0% loss. The PESQ value for all levels of piggybacking remains at 4.50, while for VQM the value increases indicating a decline in video quality.

5.4 Impact of Changes in Number of New Frames per Packet

Table 5.4: Effect of increasing number of new frames per packet on packet rate

Number of New Frames Per Packet	Packet Rate (Packets/sec)
1	59
2	47
3	46
4	46

In this section, we show the results of our experiments, in which we vary the number of new frames that can be placed in each packet. The packet rate is recorded for each case. Table 5.4 shows how the packet rate changes when different numbers of new frames are placed in each packet.

It can be seen that if only 1 frame is placed in each packet, the packet rate will be 59 packets per second. This exceeds the maximum size specified in Section 4.4. However, by simply increasing it to 2 new frames per packet, the packet rate falls to 47. Increasing the number of new frames further does not seem to reduce the packet rate, as most of the time no more than 2 new frames can fit in a packet at a time.

As with the piggybacking degree, it may not be advisable to put too many frames in one packet, as this means that each packet lost results in more frames being lost at the same time. Since increasing the new frames per

packet to more than 2 does not seem to reduce the packet rate further, for all experiments in this thesis, this parameter is fixed at 2 to avoid the situation of losing too many frames per packet.

5.5 Summary

In this chapter, we have presented our observations on the effects of each system parameter using the new testbed design presented in Chapter 4. We see from the results that by varying each of these parameters, under the same network and conversational condition, the output quality varies. Hence for an interactive video-conferencing system, these parameters must be selected carefully based on the operating conditions in order to achieve the best perceptual quality.

The trade-offs between the parameters, however, are not studied in this research. However, it is clear that being able to find the balance between the parameters is essential in locating the best operating point for the system.

In the next chapter, we utilize this testbed to investigate the effects of MED under different network and conversation conditions. At the same time, we apply our existing algorithm that is capable of finding a good estimate of the perceptually optimal MED [49] with relatively few subjective tests.

CHAPTER 6

APPLICATION OF EFFICIENT SEARCH ALGORITHM TO ESTIMATE PERCEPTUALLY OPTIMAL MOUTH-TO-EAR DELAY

6.1 Overview of the Efficient Search Algorithm

In [49], an efficient search algorithm to locate the perceptually optimal MED of a VoIP conversation is presented. Here we present the application of the algorithm to an interactive video conversation. An overview of the algorithm is shown in Algorithm 3.

Algorithm 3 Efficient Search Algorithm to Estimate Perceptually Optimal MED

1: make an initial estimate for A* ($\widehat{A^*}$) and CND ($\widehat{CND}$)
2: choose two starting points (A, B) where B > A based on ($\widehat{A^*}$, $\widehat{CND}$)
3: use testbed to generate video conversations for those two points
4: conduct subjective tests on pair
5: update ($\widehat{A^*}$, $\widehat{CND}$)
6: choose next pair to compare based on new ($\widehat{A^*}$, $\widehat{CND}$)
7: repeat steps 2 - 6 to get a better estimate

The algorithm starts by making an initial estimate of the perceptually optimal MED($\widehat{A^*}$) and the Complete Noticeable Difference ($\widehat{CND}$) of $\widehat{A^*}$.

The CND is defined as follow: *For a fixed A and a variable B, the CND(A) is the minimum $|B - A|$ such that the probability of not being able to distinguish between A and B is zero* [10].

Two starting points (A_n, B_n) within the range of $[MED_{min}, MED_{max}]$ is then chosen based on the current $\widehat{A^*}$ and $\widehat{CND}$. They are calculated using the relationship as follows [10]:

$$(A_n, B_n) = \begin{cases} (\widehat{A^*} - \widehat{CND}, \widehat{A^*}) & \text{if } n \text{ is odd} \\ (\widehat{A^*}, \widehat{A^*} + \widehat{CND}) & \text{if } n \text{ is even.} \end{cases} \tag{6.1}$$

Table 6.1: Internet traces collected on the PlanetLab in July and August, 2007.

Set	DL JT LR (L/H/M)	Hour (CST)	Source		Dest.	Mean DL (ms)		JT60 (%)		LR (%)	
			Location	IP Address	(S,A,U)	Min	Max	Min	Max	Min	Max
1*	L L L	20:00	CA,USA	169.229.50.14	(1,2,4)	42.2	94.6	0.00	0.15	0.00	0.00
2	H L L	18:00	China	219.243.201.77	(0,3,4)	107.3	190.4	0.00	3.5	0.00	0.01
3*	H L H	23:00	Hong Kong	137.189.97.18	(0,3,4)	101.2	204.3	0.00	1.64	14.7	22.7
4*	H H L	22:00	Taiwan	140.112.107.80	(1,3,3)	198.0	280.4	68.3	72.2	0.14	0.22
5	M L L	20:00	Czech	195.113.161.82	(2,3,2)	56.0	158.4	0.45	0.97	0.00	3.39
6*	M H L	17:00	CA,USA	171.66.3.181	(2,2,3)	74.9	170.9	5.2	6.2	0.00	4.33
7	M L H	1:00	Hong Kong	137.189.97.18	(1,3,3)	85.4	195.9	0.00	1.6	15.3	22.8
8*	M L M	11:00	Canada	198.163.152.229	(2,2,3)	52.4	147.3	0.00	0.83	0.00	16.9
9*	M M L	5:00	UK	128.232.103.203	(2,3,2)	26.5	139.9	0.00	8.10	0.00	3.2
10	H M M	1:00	China	211.94.143.61	(0,4,3)	103.7	198.9	1.2	6.6	1.9	8.6
11	M M M	8:00	Hungary	152.66.244.49	(3,2,2)	22.6	190.6	0.00	79.0	0.00	25.1

Keys: Each set is based on a broadcast connection from one source to 7 destinations (duration 10 min; packet period 30 ms; DL: delay; JT: jitter; JT60: jitters larger than 60 ms with respect to mean delay; and LR: loss rate). Delays are classified into low (< 100 ms), high (≥ 100 ms), and mixed (a combination of both). Similarly, jitters are classified into low ($< 5\%$ in JT60), high ($\geq 5\%$ in JT60), and mixed; and losses into low ($< 5\%$), high ($\geq 5\%$) and mixed. Each destination is listed by a triplet of three numbers (# in aSia, # in America, # in eUrope). '*' indicates a connection used in subjective tests.

The testbed in Chpater 4 is then used to generate the video for each operating point A and B. The two videos for A and B are then evaluated and the subject decides whether A is subjectively worse than B ($A <_s B$), A is approximately equal to B($A \approx B$), A is subjectively better than B ($A >_s B$) or A is indistinguishable from B ($A?B$).

From the subjective evaluation results, we then update ($\widehat{A^*}$, $\widehat{CND}$) using the equations developed in [49] and repeat Steps 2 to 6.

6.2 Experimental Setup

The experiment is set up using 6 traces collected via PlanetLab [50]. The details of the traces can be found in Table 6.1 reproduced from [10]. They vary in terms of delay, jitter and loss rate; for example 'HLH' means the trace has high delay (>100 ms), low jitter and high loss rate (>5%). This will allow us to see the difference in results for the same conversational conditions but different network conditions. For every test, the only variable that is adjusted

is MED. All the other variables are fixed as follows: 30 FPS, VGA and 2 new frames per packet. The piggybacking degree is chosen to be the one that gives us the highest PESQ.

For each of the 6 representative network conditions, we attempt to find the perceptually optimal MED for the four recorded video conversations listed in Table 4.1. As there are infinitely many possible operating points, the videos are generated in batches after each subjective evaluation. The videos are generated using the setup described in Chapter 4.

For our purposes, MED_{max} is fixed at 1000ms and MED_{min} is fixed at the point where lowering the MED any further will result in a PESQ of less than 1.0, which is intolerable. The two starting points in our experiments are chosen to be 250ms and 500ms, respectively, for each combination of network trace and recorded conversation.

For each MED, the piggybacking degree that maximizes the PESQ value is chosen. The reason is that losses in speech are much more easily detectable than video. Furthermore, as speech uses little bandwidth, maximizing speech quality will not have a huge impact on video quality.

When subjective tests are performed, objective metrics VQM, PESQ, CS and CE are calculated accordingly. For VQM, the lower the value, the better the quality. A VQM value of 0 indicates perfect quality. For PESQ, the lower the value the lower the quality. A PESQ value of 4.5 represents perfect quality and anything below 1.0 is intolerable. For CS, the greater the asymmetry, the larger the value. The lowest ideal value to attain is 1. A CE of 1 represents the highest efficiency. Anything lower indicates a less effective conversation.

Due to resource limitations, we were not able to find multiple subjects for the subjective tests. As only one subject is available for the subjective tests, the results are not statistically valid. However, we are still able to make various observations from the results collected.

6.3 Results and Observations

The results of our experiments are presented in Tables 6.2 to 6.7. The column for the subjective preference *A is indistinguishable from B (A?B)* has been omitted because it was never preferred throughout all of the subjective tests. If the results of the subjective tests are inconsistent, it may indicate

Table 6.2: Pair-wise subjective preferences under LLL network and four conversational conditions.

Netw.	Rec.	A					B					Subj. Preference		
Cond.	No.	MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
LLL	1	250	4.50	0.00	2.38	0.45	500	4.50	0.01	1.56	0.40	0	0	1
		60	1.06	0.55	4.89	0.51	156	4.50	0.00	3.20	0.48	1	0	0
		203	4.50	0.00	2.73	0.47	312	4.50	0.01	2.04	0.44	0	0	1
		107	4.50	0.00	3.88	0.49	179	4.50	0.01	2.95	0.47	1	0	0
	2	250	4.50	0.05	1.61	0.51	500	4.50	0.05	1.79	0.45	0	0	1
		60	1.19	0.92	3.16	0.56	155	4.50	0.06	2.13	0.53	1	0	0
		203	4.50	0.06	1.83	0.52	312	4.50	0.06	1.49	0.49	0	0	1
		107	4.50	0.06	2.55	0.55	179	4.50	0.06	1.97	0.53	0	1	0
	3	250	4.50	0.00	2.74	0.73	500	4.50	0.00	3.12	0.71	0	0	1
		60	1.53	0.46	2.99	0.75	155	4.50	0.00	2.69	0.74	1	0	0
		203	4.50	0.00	2.66	0.73	312	4.50	0.00	2.83	0.72	0	1	0
		60	1.53	0.46	2.99	0.75	203	4.50	0.00	2.66	0.73	1	0	0
	4	250	4.50	0.01	2.18	0.79	500	4.50	0.01	2.31	0.76	0	0	1
		60	1.50	0.48	2.66	0.81	155	4.50	0.01	2.19	0.80	1	0	0
		203	4.50	0.01	2.18	0.79	312	4.50	0.01	2.18	0.78	0	0	1
		107	4.50	0.19	2.41	0.80	179	4.50	0.01	2.18	0.80	1	0	0

that there are multiple locally optimal MEDs [10]. However, all subjective tests conducted are consistent; and hence it concluded that there is only one optimal MED for each operating curve.

After 4 batches, the process is halted as we are able to obtain a good estimate of the perceptually optimal operating point, similar to the results in [10].

Observations on the effects of different conversation conditions. In Tables 6.2, 6.5 and 6.6, we can see that under LLL, HLL and HLH conditions, changing the conversational conditions does not seem to have much impact on the perceptually optimal MED. All four recordings for each network condition have similar $\widehat{A^*}$ of about 179 ms for LLL, 209 ms for HLL and 522 ms for HLH.

However, in Tables 6.3, 6.4 and 6.7, we can see that for each network condition, having different conversation conditions results in different perceptually optimal MEDs, indicating that the perceptually optimal operating point does not solely depend on network conditions, but both network and conversational conditions. For example, in Table 6.3 (LLH) recording 1, the

Table 6.3: Pair-wise subjective preferences under LLH network and four conversational conditions.

Netw. Cond.	Rec. No.	A					B					Subj. Preference		
		MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
LLH	1	250	4.00	0.36	2.38	0.45	500	4.32	0.34	1.56	0.40	1	0	0
		585	4.20	0.34	1.64	0.38	750	4.37	0.33	1.93	0.35	1	0	0
		875	3.98	0.32	2.15	0.33	984	3.84	0.31	2.35	0.31	0	0	1
		741	4.21	0.34	1.92	0.35	813	4.05	0.33	2.04	0.34	0	0	1
	2	250	3.73	0.49	1.61	0.51	500	4.50	0.51	1.79	0.45	0	0	1
		83	1.17	1.00	2.83	0.55	167	2.98	0.48	2.05	0.53	1	0	0
		209	3.58	0.61	1.80	0.52	318	3.90	0.47	1.49	0.49	1	0	0
		246	3.82	0.49	1.62	0.51	318	3.90	0.47	1.49	0.49	1	0	0
	3	250	3.86	0.28	2.74	0.73	500	4.18	0.28	3.12	0.71	1	0	0
		585	4.25	0.29	3.26	0.70	750	4.09	0.27	3.51	0.68	1	0	0
		875	4.12	0.28	3.70	0.67	984	3.99	0.27	3.87	0.66	0	0	1
		741	4.33	0.27	3.50	0.69	813	4.23	0.27	3.61	0.68	0	1	0
	4	250	3.80	0.32	2.18	0.79	500	4.02	0.33	2.31	0.76	0	0	1
		83	1.41	0.53	2.53	0.81	167	3.37	0.34	2.18	0.80	1	0	0
		209	3.95	0.33	2.18	0.79	318	3.89	0.33	2.18	0.78	0	0	1
		116	1.50	0.44	2.36	0.80	188	3.71	0.37	2.18	0.80	1	0	0

optimal MED is estimated to be around 740 ms. However, for recording 2 under the same network condition, the optimal MED is estimated to be around 318 ms

Observations of the effects of losses. Comparing Table 6.2 (LLL) with Table 6.3 (LLH) and Table 6.5 (HLL) with Table 6.6 (HLH), we can see that for every conversational condition, when the loss rate is increased, the perceptually optimal MED value is increased.

Observations of the effects of jitter. Similarly by comparing Table 6.2 (LLL) with Table 6.4 (LHL) and Table 6.5 (HLL) with Table 6.7 (HHL), we can see that for every conversational condition, when the jitter is increased, the perceptually optimal MED value is increased.

Observations of the effects of different backgrounds for the same conversation. The same interactive video conversation with different video backgrounds results in different perceptually optimal MEDs. An instance where this is reflected can be seen in Table 6.3 (LLH) by comparing record-

Table 6.4: Pair-wise subjective preferences under LHL network and four conversational conditions.

Netw. Cond.	Rec. No.	A					B					Subj. Preference		
		MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
LHL	1	250	4.50	0.03	2.38	0.45	500	4.50	0.19	1.56	0.40	1	0	0
		585	4.50	0.01	1.64	0.38	750	4.50	0.18	1.93	0.35	0	0	1
		543	4.50	0.01	1.57	0.39	652	4.50	0.00	1.76	0.37	0	0	1
		450	4.50	0.24	1.56	0.41	522	4.50	0.24	1.57	0.39	0	1	0
	2	250	4.50	0.29	1.61	0.51	500	4.50	0.07	1.79	0.45	0	0	1
		110	3.11	0.58	2.52	0.55	180	4.50	0.37	1.96	0.53	1	0	0
		215	4.18	0.30	1.77	0.52	324	4.50	0.10	1.49	0.49	1	0	0
		252	4.50	0.29	1.60	0.51	324	4.50	0.10	1.49	0.49	1	0	0
	3	250	3.91	0.38	2.74	0.73	500	4.27	0.20	3.12	0.71	1	0	0
		585	4.50	0.16	3.26	0.70	750	4.50	0.18	3.51	0.68	0	0	1
		543	4.27	0.20	3.19	0.70	652	4.50	0.21	3.36	0.69	1	0	0
		575	4.50	0.27	3.24	0.70	647	4.50	0.21	3.35	0.69	1	0	0
	4	250	4.05	0.27	2.18	0.79	500	4.24	0.31	2.31	0.76	1	0	0
		585	4.50	0.24	2.47	0.75	750	4.50	0.30	2.77	0.74	0	0	1
		543	4.50	0.30	2.39	0.76	652	4.50	0.23	2.59	0.75	1	0	0
		575	4.26	0.23	2.45	0.75	647	4.50	0.23	2.58	0.75	0	1	0

ings 1 with 2 and recordings 3 with 4. Recordings 1 and 2 have the same conversation, but recording 1 has a simple background while recording 2 is conducted outdoors. It can be seen that for recording 1 the perceptually optimal MED is around 741 ms, while for recording 2 the value is around 318 ms. Similarly, recordings 3 and 4 have the same conversation with a different background and the final perceptually optimal MEDs found differs. This phenomenon can also be seen for network condition LHL in Table 6.4 and HHL in Table 6.7.

Observations of the effects of different conversations for the same background. Next we compare recordings 1 and 3 for all network conditions (Tables 6.2 - 6.7), where recording 3 is longer than recording 1 with speakers taking longer turns talking (lower turn taking frequency). The two recordings have the same background. Interestingly, the perceptually optimal MED seems to be similar for recordings 1 and 3 across all tested network conditions. This could mean that for an interactive video conversation, the background plays a more important role in affecting the choice of the perceptually optimal MED than does the conversation itself. This may be plausible

Table 6.5: Pair-wise subjective preferences under HLL network and four conversational conditions.

Netw.	Rec.	A					B					Subj. Preference		
Cond.	No.	MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
HLL	1	250	4.50	0.00	2.38	0.45	500	4.50	0.00	1.56	0.40	0	0	1
		139	1.21	0.86	3.40	0.48	195	4.50	0.00	2.80	0.47	1	0	0
		223	4.50	0.00	2.57	0.46	332	4.50	0.00	1.95	0.43	0	0	1
		139	1.21	0.86	3.40	0.48	209	4.50	0.00	2.67	0.46	1	0	0
	2	250	4.50	0.05	1.61	0.51	500	4.50	0.05	1.79	0.45	0	0	1
		139	1.37	1.03	2.25	0.54	195	4.50	0.06	1.87	0.52	1	0	0
		223	4.50	0.06	1.73	0.51	332	4.50	0.06	1.49	0.49	0	0	1
		139	1.37	1.03	2.25	0.54	209	4.50	0.05	1.80	0.52	1	0	0
	3	250	4.50	0.00	2.74	0.73	500	4.50	0.00	3.12	0.71	0	0	1
		139	1.56	0.56	2.73	0.74	195	4.50	0.00	2.65	0.74	1	0	0
		223	4.50	0.00	2.70	0.73	332	4.50	0.00	2.86	0.72	0	1	0
		139	1.56	0.56	2.73	0.74	223	4.50	0.00	2.70	0.73	1	0	0
	4	250	4.50	0.01	2.18	0.79	500	4.50	0.01	2.31	0.76	0	0	1
		139	1.44	0.97	2.26	0.80	195	4.50	0.01	2.18	0.79	1	0	0
		223	4.50	0.01	2.18	0.79	332	4.50	0.01	2.18	0.78	0	0	1
		139	1.44	0.97	2.26	0.80	209	4.50	0.01	2.18	0.79	1	0	0

as the same errors in different scenes will appear differently. A reason for that may be that if the background is complex, video artifacts may not be as obvious as those in a simple background scene.

Observations of trends in CS. As discussed in Chapter 2, CS is dependent on HRD of the participants recorded. It is also stated that the ideal CS is 1, based on the assumption that the participants have similar HRDs in a face-to-face conversation. However, if the HRDs are not similar, then the CS will not be close to 1 in a face-to-face condition. This can be seen in Table 6.2 (LLL) under recording 1, batch 1. A lower MED of 250 ms gives a CS of 2.38, while a higher MED of 500 ms gives a CS of 1.56.

It is also noted that the subjective preference does not always tend to that of a lower CS. An example of this is also seen in Table 6.2 (LLL) for recording 1. The final choice of 179 ms after batch 4 gives a CS of 2.95, which is higher than the initial estimate 250 ms, which gives a CS of 1.56.

Observations of trends in CE. For the case of CE, it is clear that for all recordings and network conditions (Tables 6.2 - 6.7), increasing the MED

Table 6.6: Pair-wise subjective preferences under HLH network and four conversational conditions.

Netw.	Rec.	A					B					Subj. Preference		
Cond.	No.	MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
HLH	1	250	3.90	0.36,	2.38	0.45	500	4.50	0.33	1.56	0.40	1	0	0
		585	4.50	0.33	1.64	0.38	750	4.50	0.33	1.93	0.35	0	0	1
		543	4.50	0.26	1.57	0.39	652	4.50	0.25	1.76	0.37	0	0	1
		450	4.00	0.24	1.56	0.41	522	4.50	0.25	1.57	0.39	0	0	1
	2	250	4.50	0.53	1.61	0.51	500	4.50	0.37	1.79	0.45	1	0	0
		585	4.50	0.45	2.00	0.43	750	4.50	0.44	2.39	0.40	0	0	1
		543	4.50	0.44	1.90	0.44	652	4.50	0.45	2.16	0.42	0	0	1
		450	4.50	0.46	1.67	0.46	522	4.50	0.46	1.85	0.44	1	0	0
	3	250	3.79	0.35	2.74	0.73	500	4.50	0.27	3.12	0.71	1	0	0
		585	4.50	0.27	3.26	0.70	850	4.50	0.26	3.67	0.68	0	0	1
		543	4.50	0.27	3.19	0.70	652	4.50	0.27	3.36	0.69	0	0	1
		450	4.33	0.27	3.05	0.71	522	4.50	0.27	3.16	0.70	1	0	0
	4	250	3.88	0.32	2.18	0.79	500	4.36	0.31	2.31	0.76	1	0	0
		585	4.50	0.30	2.47	0.75	750	4.50	0.30	2.77	0.74	0	0	1
		543	4.50	0.30	2.39	0.76	652	4.50	0.30	2.59	0.75	0	1	0
		130	1.41	0.50	2.30	0.80	543	4.50	0.30	2.39	0.76	1	0	0

results in lower efficiency. However, sometimes a lower efficiency has to be chosen such that we can achieve higher video and speech quality. For example in Table 6.2 (LLL) under recording 1, batch 2, a CE of 4.8 is chosen over a CE of 5.1, because by sacrificing a little on conversation efficiency, we have PESQ increasing from 1.06 to 4.5 and VQM decreasing from 0.55 to 0.00.

6.4 Summary

In this chapter, we have presented the experimental setup, in which we have shown the successful application of an efficient search algorithm for the perceptually optimal MED. Tables 6.2 - 6.5 show that after 4 batches we are able to find a good estimate of the perceptually optimal MED, similar to the results in [10]. Hence, we have validated that even though the algorithm was designed for a VoIP system, it has shown to be able to work for a videoconferencing system. We have also presented various observations that we made from the results of the experiments.

From the experimental results, we can see that there is no simple rela-

Table 6.7: Pair-wise subjective preferences under HHL network and four conversational conditions.

Netw.	Rec.	A					B					Subj. Preference		
Cond.	No.	MED	PESQ	VQM	CS	CE	MED	PESQ	VQM	CS	CE	A$<_s$B	A$\approx$B	A$>_s$B
HHL	1	250	2.67	0.26	2.38	0.45	500	3.32	0.28	1.56	0.40	0	0	1
		130	1.24	0.63	3.52	0.49	190	2.34	0.21	2.84	0.47	1	0	0
		220	2.51	0.17	2.59	0.462	329	2.79	0.25	1.96	0.43	1	0	0
		257	2.73	0.26	2.34	0.45	329	2.79	0.25	1.96	0.43	1	0	0
	2	250	2.78	0.47	1.61	0.51	500	2.92	0.48	1.79	0.45	1	0	0
		585	2.83	0.40	2.00	0.43	750	3.46	0.48	2.39	0.40	1	0	0
		875	3.48	0.47	2.69	0.37	984	3.49	0.50	2.96	0.36	0	0	1
		741	3.46	0.35	2.37	0.40	813	3.46	0.29	2.55	0.39	1	0	0
	3	250	3.93	0.11	2.74	0.73	500	4.02	0.09	3.12	0.71	0	0	1
		130	1.54	0.45	2.76	0.74	190	3.07	0.26	2.64	0.74	1	0	0
		220	3.63	0.15	2.69	0.73	329	3.87	0.13	2.86	0.72	1	0	0
		257	3.80	0.14	2.75	0.73	329	3.87	0.13	2.86	0.72	1	0	0
	4	250	3.25	0.35	2.18	0.79	500	3.36	0.38	2.31	0.76	1	0	0
		585	3.50	0.17	2.47	0.75	750	3.74	0.14	2.77	0.74	1	0	0
		875	3.97	0.15	3.00	0.72	984	4.03	0.25	3.20	0.71	1	0	0
		920	4.07	0.33	3.08	0.72	992	3.99	0.39	3.22	0.71	1	0	0

Note: Final PESQ reading for recording 1 is low because under high losses there is an extended period of silence due to part of the conversation being lost. However for the parts of the conversation that are not lost, the quality is good. An MED of 329 ms is still preferred over 500 ms because it gives better VQM and CE measurements. This illustrates that maximization on one quality (e.g. PESQ) will not always lead to the perceptually optimal operating point.

tionship to describe the trade-offs between PESQ, VQM, CS and CE. Each metric captures some aspects of the overall perceptual quality, and hence it is essential to take multiple metrics into consideration in order to determine if one video conversation has better perceptual quality than another.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Summary of Accomplished Research

The following is a summary of the accomplishments of this thesis:

- Firstly, we are able to conclude that the measurement of the perceptual quality of a video-conferencing system cannot be simply captured by a single metric. Optimizing along one quality metric will result in some degree of trade-off in another metric, as no one metric is able to capture the dynamics of the entire system. Hence a set of objective metrics is necessary in evaluating the perceptual quality.

- Secondly, we put together a platform that is capable of performing black box testing on existing video-conferencing systems. We have shown its effectiveness by using it to perform evaluations of how Skype reacts to changes in the network conditions.

- Thirdly, we have designed an algorithm that is able to take one recorded video conversation and generate versions of the same conversation with different frame rate, frame size and MED. This overcomes the problem of needing to record a new conversation for each possible set of parameters.

- Fourthly, by utilizing the algorithm we have created, we have successfully built a new testbed that allows us to compare the the same conversation under different system parameters and network conditions.

- Fifth, by utilizing the testbed we created, we have performed preliminary investigations into how frame rate, frame size, piggybacking degree and the number of new frames per packet will affect the perceptual output quality of an interactive video conversation.

- Lastly, we conclude that the efficient search algorithm to locate the perceptually optimal MED for a VoIP system can be applied effectively to that of a video-conferencing system.

7.2 Future Work

- By collecting more subjective evaluation results over different network and conversational conditions and their respective objective measurements, we can train a learning algorithm to identify for seen and unseen conditions their perceptual configuration [51].

- Within this thesis, we have applied an algorithm to locate only the perceptually optimal MED while keeping other parameters in the system constant. Using the current testbed, we can experiment by varying other parameters and possibly extend the algorithm to find not just the perceptually optimal MED, but also its perceptually optimal frame rate, frame size and piggybacking degree to use.

REFERENCES

[1] B. Sat and B. W. Wah, "Statistical testing of off-line comparative subjective evaluations for optimizing perceptual conversational quality in VoIP," in *Proc. IEEE Int'l Symposium on Multimedia*, Dec. 2008, pp. 424–431.

[2] S. S*ü*sstrunk and S. Winkler, "Color image quality on the Internet," in *Proc. SPIE*, 2004, pp. 118–131.

[3] ITU-P.862, "Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," Feb. 2001. [Online]. Available: http://www.itu.int/rec/T-REC-P.862/en

[4] ITU-J.144, "Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference," Mar. 2004. [Online]. Available: http://www.itu.int/rec/T-REC-J.144/en

[5] P. M. Frederic, F. Dufaux, S. Winkler, T. Ebrahimi, and G. Sa, "A no-reference perceptual blur metric," in *IEEE 2002 International Conference on Image Processing*, 2002, pp. 57–60.

[6] ITU-P.563, "Single-ended method for objective speech quality assessment in narrow-band telephony applications," May 2004. [Online]. Available: http://www.itu.int/rec/T-REC-P.563/en

[7] S. Winkler, A. Sharma, and D. McNally, "Perceptual video quality and blockiness metrics for multimedia streaming applications," in *Proceedings of the International Symposium on Wireless Personal Multimedia Communications*, 2001, pp. 547–552.

[8] ITU-P.800, "Methods for subjective determination of transmission quality," Aug. 1996. [Online]. Available: http://www.itu.int/rec/T-REC-P.800/en

[9] International Telecommunication Union, "ITU-T P-Series recommendations," Mar. 2011. [Online]. Available: http://www.itu.int/rec/T-REC-P/en

[10] B. Sat, "Design and evaluation of VoIP systems with high perceptual conversational quality," Ph.D. dissertation, Dept. of Electrical and Computer Engineering, Univ. of Illinois, Urbana, IL, Sep. 2010.

[11] "Polycom videoconferencing systems," 2011. [Online]. Available: http://www.polycom.com/products/telepresence_video/video_conference_systems/

[12] ITU-G.726, "40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)," Dec. 1990. [Online]. Available: http://www.itu.int/rec/T-REC-G.726/en

[13] ITU-G.729, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP)," Jan. 2007. [Online]. Available: http://www.itu.int/rec/T-REC-G.729/en

[14] ITU-G.722.2, "Wideband coding of speech at around 16 kbit/s using adaptive multi-rate wideband (AMR-WB)," Jul 2003. [Online]. Available: http://www.itu.int/rec/T-REC-G.722.2/en

[15] ITU-H.263, "Video coding for low bit rate communication," Jan. 2005. [Online]. Available: http://www.itu.int/rec/T-REC-H.263/e

[16] ITU-H.264, "Advanced video coding for generic audiovisual services," Mar. 2010. [Online]. Available: http://www.itu.int/rec/T-REC-H.264/en

[17] "VP7," 2011. [Online]. Available: http://en.wikipedia.org/wiki/VP7

[18] IETF-RFC1889, "RTP: A transport for real-time applications," Jan. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1889.txt

[19] IETF-RFC768, "User datagram protocol," Aug. 1980. [Online]. Available: http://www.ietf.org/rfc/rfc768.txt

[20] IETF-RFC793, "Transmission control protocol," 1981. [Online]. Available: http://www.ietf.org/rfc/rfc793.txt

[21] ITU-G.114, "One-way transmission time," May 2003. [Online]. Available: http://www.itu.int/rec/T-REC-G.114/en

[22] ITU-G.107, "The E-model, a computational model for use in transmission planning," Apr. 2009. [Online]. Available: http://www.itu.int/rec/T-REC-G.107/en

[23] ITU-P.562, "Analysis and interpretation of INMD voice-service measurements," May 2004. [Online]. Available: http://www.itu.int/rec/T-REC-P.562/en

[24] Y. Wang, "Survey of objective video quality measurements," Worcester Polytechnic Institute, Tech. Rep., June 2006.

[25] "ITS video quality research," 2011. [Online]. Available: http://www.its.bldrdoc.gov/n3/video/

[26] B. Sat and B. W. Wah, "Analyzing voice quality in popular VoIP applications," *IEEE Multimedia*, vol. 16, pp. 46–58, Jan-Mar 2009.

[27] B. Sat and B. W. Wah, "Playout scheduling and loss-concealments in VoIP for optimizing conversational voice communication quality," in *Proc. ACM Multimedia*, Augsburg, Germany, Sep. 2007, pp. 137–146.

[28] B. W. Wah and B. Sat, "The design of VoIP systems with high perceptual conversational quality," *Journal of Multimedia, Special Issue on Advances in Interactive Digital Entertainment Technologies*, 2009.

[29] N. Kiatawaki and K. Itoh, "Pure delay effect on speech quality in telecommunications," *IEEE Journal on Selected Areas of Communication*, vol. 9, no. 4, pp. 586–593, May 1991.

[30] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for Internet telephony," in *Proc. IEEE INFOCOM*, vol. 3, 1999, pp. 1453–1460.

[31] R. Steinmetz, "Human perception of jitter and media synchronization," *Selected Areas in Communications, IEEE Journal on*, vol. 14, no. 1, pp. 61 –72, Jan. 1996.

[32] T.-S. Yum, M.-S. Chen, and Y.-W. Leung, "Video bandwidth allocation for multimedia teleconferences," *Communications, IEEE Transactions on*, vol. 43, no. 234, pp. 457 –465, 1995.

[33] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication: a review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974 –997, may 1998.

[34] L. D. Cicco, S. Mascolo, and V. Palmisano, "Skype video responsiveness to bandwidth variations," in *NOSSDAV*, 2008.

[35] "Libipq by example," 2006. [Online]. Available: https://www.linuxquestions.org/linux/answers/Programming/Libipq_by_example

[36] "Wireshark. Go deep," 2011. [Online]. Available: http://www.wireshark.org

[37] "IMCapture," 2011. [Online]. Available: http://www.imcapture.com/

[38] "Skype," 2011. [Online]. Available: http://www.skype.com/

[39] Y.-F. Ou, T. Liu, Z. Zhao, Z. Ma, and Y. Wang, "Modeling the impact of frame rate on perceptual quality of video," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, 2008, pp. 689 –692.

[40] J. Wang and B. Bodenheimer, "The just noticeable difference of transition durations," in *ACM SIGGRAPH 2005 Posters*, ser. SIGGRAPH '05. New York, NY, USA: ACM, 2005. [Online]. Available: http://doi.acm.org/10.1145/1186954.1187072

[41] Y.-J. Chin and T. Berger, "A software-only videocodec using pixelwise conditional differential replenishment and perceptual enhancements," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 9, no. 3, pp. 438 –450, Apr. 1999.

[42] "AVI RIFF file reference," 2011. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms779636%28VS.85%29.aspx

[43] "FFmpeg, a complete, cross-platform solution to record, convert and stream audio and video," 2011. [Online]. Available: http://www.ffmpeg.org/

[44] J. Lu and B. W. Wah, "Scheduling transmissions of real-time video coded frames in video conferencing applications over the internet," in *Proc. Int'l Conf. on Multimedia and Expo.* IEEE, 2010.

[45] "AviSynth, tool for video post-production," 2011. [Online]. Available: http://avisynth.org/mediawiki/Main_Page

[46] "VideoLan - officialpage for VLC media player, the open source video framework!" 2011. [Online]. Available: http://www.videolan.org/vlc/

[47] Y. W. Yen-Fu Ou, Zhan Ma, "A novel quality metric for compressed video considering both frame rate and quantization artifacts," in *Proc. if Intl. Workshop Video Processing and Quality Metrics for Consumer (VPQM)*, Jan. 2009.

[48] W. H. Yeo, B. Sat, and B. W. Wah, "New piggybacking algorithm in VoIP using enhanced G.722.2 codec with larger frames," in *IEEE Int'l Workshop on Multimedia Signal Processing*, Oct. 2009.

[49] B. Sat and B. W. Wah, "Statistical scheduling of offline comparative subjective evaluations for real-time multimedia," *IEEE Trans. on Multimedia*, vol. 11, no. 6, pp. 1114–1130, Oct. 2009.

[50] "Planetlab: An open platform for developing, deploying and accessing planetary-scale services," 2007. [Online]. Available: http://www.planet-lab.org/

[51] Z. X. Huang, B. Sat, and B. W. Wah, "Automated learning of play-out scheduling algorithms for improving the perceptual conversational quality in multi-party VoIP," in *Proc. IEEE Int'l Conf. on Multimedia and Expo*, July 2008, pp. 493–496.