TRANSCODING OF JPEG IMAGES FOR WIRELESS PROXY COMMUNICATIONS

By

Mohamed El Shentenawy

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario

May 2001

The undersigned recommend to the Faculty of Graduate Studies and Research acceptance of the thesis

TRANSCODING OF JPEG IMAGES FOR WIRELESS PROXY COMMUNICATIONS

Submitted by **Mohamed El Shentenawy**, **B.Sc.**, in Electrical Engineering, Professional Certificate of Computer Systems and Applications

---

Prof. Thomas Kunz, Thesis Co-Supervisor   Prof. Roshdy Hafez Co-Supervisor

---

Prof. Frank Dehne, Director of School of Computer Science

Carleton University

May 2001

Abstract

The explosive growth of the Internet and handheld wireless devices is widening a large gap between mobile clients. Mobile clients vary in their hardware and software resources such as screen size, color depth, effective bandwidth, processing power, and the ability to handle different data formats. This makes it difficult for servers to support a wide range of client variations. Application-level adaptation is required to provide a meaningful Internet experience across the range of client capabilities.

In this research we introduce an image transcoding proxy server placed between the generic WWW servers and the heterogeneous mobile hosts. The main function of this proxy is to reprocess, on the fly, the most common image formats used on the web for quick transmission of reasonable quality. The transcoding proxy uses the mobile device characteristics as input parameters and can adapt to dynamically changing bandwidth on the proxy-client link. The policy decisions are used to select the most appropriate operations in order to achieve the smallest file size and acceptable image quality.

Moreover, our proxy can perform on-demand specific image compression operations, tailoring content to the specific constraints of the user.

Acknowledgment

I would like to take this opportunity to thank Prof. Thomas Kunz and Prof. Roshdy Hafez for supporting my research and providing all the facilities to conduct this work.

I wish also to thank Nortel Networks and CITO, the Communication and Information Technology Ontario for funding the project.

I would also like to acknowledge the support received from the Computer Systems and Engineering Department and School of Computer Science in giving access and support to their facilities.

Lastly, I would like to thank my wife, Rehab, and my son, Tarek, who allowed me time off from family duties to write this thesis.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction and Overview

1.1 Introduction

Communication over wireless links is characterized by limited bandwidth and high latencies. In addition, mobile hosts have relatively less resources than their static counterparts. Therefore, user might face some difficulties in accessing and downloading the data over the World Wide Web in this environment.

In this research we describe the design and the implementation of an image transcoding proxy server that can be used on behalf of the Web user. Then, we present the results of several experiments using this proxy.

Our aim is to improve the performance of accessing the WWW over low bandwidth networks without affecting the important information. We achieve this by a set of transcoding operations to the images at the proxy that are mainly selected and parameterized based on the device characteristics and network bandwidth available.

1.2 The Use of Proxies in the Low Bandwidth Access Networks

When using a web browser to view web pages from the Internet, the browser uses HTTP (Hyper-Text Transfer Protocol) to request the text and images from the remote server. These requests are made directly from the browser to the remote machine.

HTTP allows using an intermediary to make the requests on the users behalf. Such an intermediary is known as a proxy server.

A proxy server (sometimes referred to as an application gateway or forwarder) is an application that mediates traffic between a web browser and the remote server.

When configured to use a proxy, a browser makes HTTP requests to the proxy instead of going directly to the remote host. In turn, the proxy makes the same request on the user's behalf.

A number of advantages derive from filtering data at the proxy. First, since the communication link between proxy and mobile host may be slow, reducing the amount of traffic can improve performance. Second, even if data filtering reduces performance it may reduce cost. Depending on the relative degrees of reduction, reduced cost in return for reduced performance might be desirable.

Most of the work in this area applies filtration on a static basis [1], [2],[3]. This means that the proxy does such filtration using static parameters defined by user or specified in the proxy software as a constant. Unfortunately, this type of filtering is rather coarse; all image data will be filtered even if the original image may be supported by the device and the available network bandwidth in some cases.

Our proxy differs from other applications in that it uses the mobile device characteristics and the network bandwidth currently available, to perform a dynamic adaptation based on these two factors. Unless a client uses his own parameters, which could be entered using user preferences, our proxy will use these dynamic parameters in performing its transcoding operation, taking into account the user satisfaction for the final quality of the transcoding image. Figure 1.1 illustrates this operation.

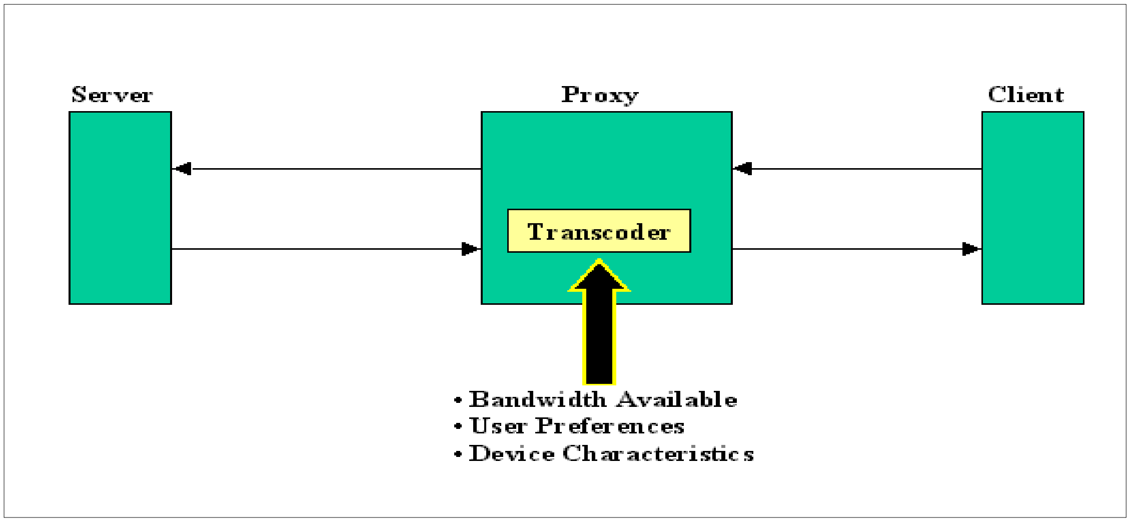


Figure 1-1: Transcoding Proxy Server

1.3 Web Traffic Statistics and Characterization

Clicking on hyperlinks that are part of HTML pages generates traffic and, as a result, a new HTML page or an image is displayed. HTML pages contain formatted text and graphics. Sometimes links in HTML pages lead to other types of media, such as video or audio. With respect to users, the low level of expertise required to navigate with a Web browser has resulted in a large and diverse user population.

In the past, several studies have attempted to characterize Web traffic. Most of these studies are represented from the server or proxy server perspective [4], [5], which may not accurately reflect the data stream requested and experienced by users. This is also because of the complexity of collecting this specific information using the Web server. A user may purposely or accidentally open multiple browsers and generate requests from these browsers at the same time. A user may abandon the on-going page in the middle by moving to another page or clicking the Back or Stop buttons. Current HTML versions enable a browser to request multiple pages at once (e.g. frames and Java-scripts). The frame allows authors to present independently designed pages inside of sub-windows as if they were a single page. Java-script cooperates with *HyperText Markup Language* (HTML) code and enables authors to present multiple pages in an independent window. Using the proxy to characterize workloads is also not accurate. This is due to the difficulty of collecting proxy log files from different sources and the privacy issues in information contained in such logs.

We therefore collected the information at the client side. We applied a simple batch program on selected user machines to count this kind of information when using the Internet. We applied this batch on some users. Users had a widely different range of interests and browsed a very heterogeneous collection of websites.

We verified that GIF and JPEG represent the most frequently used file types in the Web accounting for the most majority of data traffic [6], [7], [8]. We therefore focused on these two web elements in our research. We also spent some time to collect a large number of these images, taking into consideration the typical web page components and ratios we came up with. We classified the collected images into several categories to find a relationship between the image high-level information and the most effective compression operation we can apply.

1.4 Image Quality Measurements

The measurement of image quality is very difficult and, to the best of my knowledge, no standardized method of accomplishing this has been developed. This is because so many different attributes of an image make contributions to its overall quality. Some of these attributes are objective and are subject to rigorous measurement. Others are subjective and may be highly dependent upon the intended end use of the image.

In our proxy, images might be processed and filtered, which include material that might be of interest to the user. The goal of doing these measurements is to evaluate whether image information can be effectively filtered.

Originally, we set out to search for a specific formula we can apply for each processed image to make sure it meets the user satisfaction. We could not find such a formula that takes the high level information we are currently using in our proxy into consideration. We therefore applied some basic traces on some web users with different interests. Evaluating the results of these traces, we provided some useful limits for each operation applied by the proxy server.

It is highly recommended in the future to use a more accurate method for measuring the image quality based on mathematical formula and to merge this factor into the dynamic parameter used in our proxy.

1.5 Thesis Contributions

We present a proxy server for transcoding images in order to adapt, on the fly, these images to the current network bandwidth available and client device display capabilities taking into account the user satisfaction with the transcoded image.

We published a conference paper entitled: "Image transcoding for proxy Internet wireless access" [9], and in the proceeding of the ninth ACM Multimedia Conference [10].

1.6 Thesis Outline

In Chapter 2, we discuss some definitions and concepts related to the low bandwidth environment and GIF/JPEG image transcoding. In Chapter 3, we review related work and contrast our approach to the predominant approaches found in the literature. Chapter 4 discusses the experimental setup we created, analyzes some Web content characteristics and user behavior. In Chapter 5 we define our proxy architecture and the software design. In Chapter 6, we evaluate our work by performing case studies. Chapter 7 concludes the thesis with a summary and outlook on future work.

Chapter 2: Background

2.1 Introduction

Reducing file size is an advantage for transmitting files across a low speed (wireless) network. Surveys done on web traffic within the last two years have consistently confirmed that *JPEG* and *GIF* images are still the most commonly used over the Web [6], [7], [8]. In our research, we present an images' transcoding process that improves image delivery to a variety of mobile computers. The proposed technique takes into account the fact that a transcoded image will be transmitted over a low-bandwidth connection and to mobile devices with different capabilities.

The transcoding proxy's job is to retrieve images on the client's behalf, transcode the images in a format that can be transmitted quickly on the slow link, and finally transmit them to the client. Moreover, such a trancoding proxy can adaptively detect, on the fly, the mobile device characteristics, the network bandwidth available, and the nature of the incoming images.

In this chapter, we discuss some definitions and concepts related to the low bandwidth environment and GIF/JPEG image transcoding. We will also give an overview of ImageMagick [11], the image processing library we use in our proxy, and we will list some of the functions we utilize in our implementation.

2.2 Proxy Server Advantages

Since the communication link between the proxy server and a mobile host may be slow, reducing the amount of traffic can improve performance. Even if data filtering reduces performance it may reduce cost because the charges may be based on traffic volume and download time. Depending on the relative degrees of reduction, reduced cost in return for reduced performance might be desirable.

If properly designed, the proxy server can act not only as a filter, but also as an ‘‘agent;” i.e., as a permanent representative for a mobile host that might not always be powered up or connected to the Internet.

2.3 Constraints of Mobility

Mobile computing is characterized by some constraints:

- **Mobile elements are resource-poor relative to static elements**

  For a given cost and level of technology, considerations of weight, power, size and ergonomics will exact a penalty in computational resources such as processor speed, memory size, bandwidth, and disk capacity. While mobile elements will improve in absolute ability, they will always be resource-poor relative to static elements.

  Table 2.1 illustrates some of the variability in device bandwidth, display size, display color and storage among devices.

Client Device	Bandwidth (bps)	Display size (pixel)	Display color	Device Storage
PDA (Personal Digital Assistant)	14.4K	320 x 200	b/w	1MB
HHC (hand-held computer)	28.8K	640 x 480	Gray	4MB
TV browser	56K	544 x 384	NTSC	1GB
Color PC	56K	1024 x 768	RGB	2-4GB
Workstation	10M	1280 x 1024	RGB	> 4GB

Table 2.1: Summary of Client Device Capabilities

Since many devices are constrained in their capabilities, they cannot simply access image content as-is on the Internet. For example, many Personal Digital Assistants (PDAs) cannot handle JPEG images, regardless of size. The hand-held computers (HHCs) cannot easily display Web pages loaded with images because of screen size.

- **Mobile connectivity is highly variable in performance and reliability.**

  Some buildings may offer reliable, high-bandwidth wireless connectivity while others may only offer low-bandwidth connectivity. Outdoors, a mobile client may have to rely on a low-bandwidth wireless network with gaps in coverage.

- **Mobile elements rely on a finite energy source.**

  While battery technology will undoubtedly improve over time, the need to be sensitive to power consumption will not diminish. Concern for power consumption must span many levels of hardware and software to be fully effective.

2.4 Proxy Servers

The primary use of proxies is to allow access to the Web from within a firewall. A proxy is a special HTTP server that typically runs on a firewall machine. The proxy waits for a request from inside the firewall, forwards the request to the remote server outside the firewall, reads the response and then sends it back to the client. In the usual case, all the clients within a given subnet use the same proxy. This makes it possible for the proxy to do efficient caching of documents that is requested by a number of clients. Setting up a proxy server is easy, and the most popular Web client programs already have proxy support built in. So, it is simple to configure an entire work group to use a caching proxy server. This cuts down on network traffic since many of the documents are retrieved from a local cache once the initial request has been made.

2.5 Typical Home Page

The typical web page consists of a Hypertext document with links to other objects that make up the whole page. An object is an entity stored on a server as a file. There are two kinds of objects, a main object and inline objects. The file containing an HTML document is referred to as a main object and the objects linked from the Hypertext document are referred to as in-line objects.

The in-line objects could be an image file or sound or java applet. The main goal of defining the typical home page is to find an accurate percentage for each of these components in the typical web page.

We found that measuring the data traffic at the browser side is the best way to get accurate results.

2.6 JPEG and GIF Image Formats

JPEG (pronounced "jay-peg") is a standardized image compression mechanism. JPEG stands for Joint Photographic Experts Group, the original name of the committee that wrote the standard in the late 80's.

There are four key steps in the JPEG compression algorithm. The first step is to extract an 8x8 pixel block from the image. The second step is to calculate the discrete cosine transform for each element in the block. Third, a quantizer rounds off the discrete cosine transform (DCT) coefficients according to the specified image quality (this phase is where most of the original image information is lost, thus it is dubbed the *lossy* phase of the JPEG algorithm). Fourth, the coefficients are compressed using an encoding scheme. The final compressed code is then written to the output file [12].

JPEG is "lossy," meaning that the decompressed image is not quite the same as the one we started with. JPEG is designed to exploit known limitations of the human eye [13], notably the fact that small color changes are perceived less accurately than small changes in brightness. Thus, JPEG is intended for compressing images that will be looked at by humans. If we plan to machine-analyze our images, the small errors introduced by JPEG may be a problem for us, even if they are invisible to the eye.

A useful property of JPEG is that the degree of lossiness can be varied by adjusting compression parameters. This means that the image-maker can trade off file size against output image quality. We can make extremely small files if we do not mind poor quality; this is useful for applications such as indexing image archives. Conversely, if we are not happy with the output quality at the default compression setting, we can increase the quality to get close to 100% until we are satisfied, and accept lesser compression, keeping in mind that every time we apply the quality we use the original image.

Another important aspect of JPEG is that decoders can trade off decoding speed against image quality, by using fast but inaccurate approximations to the required calculations. Some viewers obtain remarkable speedups in this way.

GIF, 'Graphics Interchange Format' was introduced in the 1980s on CompuServe for displaying high-resolution graphics on a variety of graphics hardware and was intended as an exchange and display mechanism for graphics images [14].

Since the color depth of GIF is 8 bits, saving a photographic image as GIF firstly achieves a 3:1 compression by a reduction in color depth from 16 million colors to 256. A GIF graphic is stored as a sequence of pixels with 256 color values from an image specific color palette. Dithering reduces the visual impact of the reduction in number of colors. This data is then compressed according to the LZW (Lempel-Ziv-Welch) algorithm, a way of compressing data that takes advantage of repetition of strings in the data [15].

GIF is potentially useful for images having large areas of pixels that are all exactly the same color, icons that use only a few colors, very sharp edges such as pure-black pixels adjacent to a row of pure-white pixels, small text that's only a few pixels high, and plain black-and-white (two level) images.

GIF does significantly better than JPEG on images with only a few distinct colors, such as line drawings. Not only is GIF lossless for such images, but it often compresses them more than JPEG can. For example, large areas of pixels that are all exactly the same color are compressed very efficiently indeed by GIF. JPEG can't squeeze such data as much as GIF does without introducing visible defects.

2.7 Quality Settings for JPEG

Most JPEG compressors pick a file size vs. image quality tradeoff by selecting a quality setting. There seems to be widespread confusion about the meaning of these settings. "Quality 95" does NOT mean, "keep 95% of the information", as some have claimed. The quality scale is purely arbitrary; it is not a percentage of anything. In fact, quality scales are not even standardized across JPEG programs. Different JPEG implementations use completely different quality scales. For example: Apple used to use a scale running from 0 to 4, not 0 to 100. Recent Apple software uses a 0-100 scale that has nothing to do with the IJG scale (their Q 50 is about the same as Q 80 on the

IJG scale[*]). PSP (Paint Shop Pro) scale is the exact opposite of the IJG scale, PSP setting N = IJG 100-N; thus lower numbers are higher quality in PSP. Adobe Photoshop does not use a numeric scale at all, it just gives "high"/"medium"/"low" choices.

Fortunately, this confusion does not prevent different implementations from exchanging JPEG files. But we should keep in mind that quality scales vary considerably from one JPEG-processing program to another, and that just saving a JPEG image at “Q=75” does not mean much unless it is combined with which program was used. In this research, we use IJG scale for measuring the JPEG Quality. In most cases the goal is to pick the lowest quality setting, or smallest file size, that decompresses into an image indistinguishable from the original. This setting will vary from one image to another and from one observer to another, but here are some rules of thumb.

If an image contains sharp colored edges, slight fuzziness or jagginess around such edges may be noticed no matter how high the quality setting was. This can be suppressed, at a price in file size. Quality settings around 50 (based on IJG scale) are often recommended as perfectly acceptable on the Web. However, using the results of our research, we found that this number is lower. In fact, a user viewing such an image on a browser with a 256-color display is unlikely to be able to see any difference to a higher quality setting, because the browser's color quantization artifacts will swamp any imperfections in the JPEG image itself.

---

[*] IJG (Independent JPEG Group) scale is a scale for measuring the JPEG quality

2.8 ImageMagick Library

ImageMagick [11] is a robust collection of tools and libraries to read, write, and manipulate an image in many image formats including popular formats like TIFF, JPEG, PNG, PDF, Photo CD, and GIF. With ImageMagick, images can be dynamically created, made suitable for Web applications. They can also be resized, rotated, sharpened, color reduced, or added special effects and saved in the same or a differing image format. Image processing operations are available from the command line, as well as through C, C++, and PERL-based programming interfaces.

The most common function we used in our proxy is the Convert function. Convert converts an input file using one image format to an output file with a differing image format. In addition, various types of image processing can be performed on the converted image during the conversion process.

CONVERT takes many different parameters for this function. Here are some of the parameters we used in our experiments and in our proxy:

- Quality Value**: JPEG/MIFF/PNG compression level.** For the JPEG image format, quality is 0 (worst) to 100 (best). The default quality is 75.
- **Colors Value:** preferred number of colors in the image.

  The actual number of colors in the image may be less than the request, but never more. This is a color reduction option. Images with less unique colors than specified with this option will have any duplicate or unused colors removed.
- **Equalize:** perform histogram equalization to the image.

- **Colorspace Value: the type of colorspace: GRAY, RGB, Transparent, XYZ, YCbCr, YIQ, YPbPr, YUV, or CMYK.**

  Color reduction, by default, takes place in the RGB color space. Empirical evidence suggests that distances in color spaces such as YUV or YIQ correspond to perceptual color differences more closely than do distances in RGB space. These color spaces may give better results when color reducing an image.

- **Contrast: enhance or reduce the image contrast**

  This option enhances the intensity differences between the lighter and darker elements of the image.

- **Gamma Value: level of gamma correction.**

  The same color image displayed on two different workstations may look different due to differences in the display monitor. We use gamma correction to adjust for this color difference. Reasonable values extend from 0.8 to 2.3.

- **Depth Value: depth of the image.**

  This is the number of bits in a pixel. The only acceptable values are 8 or 16

- **Geometry <width>x<height>{+-}<x offset>{+-}<y offset>{%}{!}{<}{>}:** preferred size and location of the Image window. By default, the window size is the image size and the location is chosen by the user when it is mapped to the display. By default, the width and height are maximum values. That is, the image is expanded or contracted to fit the width and height value while maintaining the aspect ratio of the image.

Chapter 3: Related Work

3.1 Introduction

The need to reduce the data transmission and to speed up delivery for the Internet user especially in the wireless environment causes a lot of researchers to focus on processing web content in the Internet. It also encourages some researches to analyze the Internet content characteristics and user behavior to keep such users satisfied with the processed information. In this chapter we will discuss these efforts that are more relevant to our work.

3.2 Related Work

Smith, Mohan, and Li [2] developed a content-image transcoding system that analyzes the images, the related text and Web document context in order to select suitable policies to adapt the images to the client devices. The content-based image transcoder analyzes the images and classifies them into two classes: image type and image purpose. The system then applies transcoding policies based on the content classes to manipulate, transcode, and adapt the images.

For image type they defined 7 classes:

1. BWG (b/w graphic)
2. BWP (b/w photo)
3. GRG (gray graphic)
4. GRP (gray photo)

5. SCG (simple color graphic)
6. CCG (complex color graphic)
7. CP (color photo)

For an image purpose they defined 8 classes:

1. ADV (advertisement, i.e., banner ads)
2. DEC (decoration, i.e., background textures)
3. BUL (bullets, points, balls, dots)
4. RUL (rules, lines, separators)
5. MAP (maps, i.e., images with click focus)
6. INF (information, i.e., icons, logos, mastheads)
7. NAV (navigation, i.e., arrows)
8. CON (content related, i.e., news photos)

They map the images into subject classes using related text that is usually mentioned as alternative text in the HTML tag for the image. The semantic information potentially provides substitute text for the images for client devices that cannot handle images.

As transcoding operations, the system provides a set of functions such as:

1. Size: minify, crop, and sub-sample.
2. Fidelity: JPEG compress, GIF compress, quantize, reduce resolution, enhance edges, contrast stretch, histogram equalize, gamma correct, smooth, sharpen, and denoise.
3. Color content: reduce color, map to color table, convert to gray, convert to b/w, threshold, and dither.

Unlike our transcoding proxy, the system does not use the network bandwidth available as one of its transcoding parameters. As a result any image will be transcoded using fixed parameters regardless of the available bandwidth. This results in the application of some transcoding functions such as quality reduction even if the current network bandwidth is high enough to transmit the image in full quality.

From a practical point of view we find that distinguishing between classes of images, (type classifications) is very hard unless it is defined in the image header. For example, taking the decision that a certain image is natural or synthetic based on mathematical formulas is often not very accurate.

They also obtained the classification parameters from a training set of 1282 images retrieved from the web, which is a very small number and cannot be used as a classification reference.

For the image purpose classification and mapping the images into subject classes using related text, the system uses a dictionary of terms extracted from the text related to the images. The terms are extracted from the "alt" tag text, which is not available in many cases.

We also find that some of the transcoding function such as convert to gray and dither are not very effective in reducing the image file size. Others like contrast stretch, smooth, sharpen are usually used as image effects and enhancements, not for the purpose of the file size reduction. In our transcoding proxy we avoid applying such operations.

Han, Bhagwat, LaMaire, Mummert, Perret, and Rubas [1] presented a transcoding proxy server built by integrating a transcoding subsystem into an HTTP proxy. The transcoding subsystem can be separated into two components: the policy module and the transformation modules. The transformation modules modify the downstream data (i.e., HTML pages and GIF and JPEG images) that are being returned to the client Web browser. The decision concerning what transcoding policy to use is made by the policy module based on the characteristics of the data such as the image file size, the current estimate of the network bandwidths on the client-to-proxy and proxy-to-server links, the characteristics of the client, particularly the client display capabilities, and the user preferences concerning the preferred rendering of the data.

They also presented an analytical framework for taking the decision to transcode or not to transcode for streaming and store-and-forward proxies under certain idealized assumptions. Transcoding is performed only when response time is reduced. The response time is calculated based on predicted image transcoding times and image output sizes, as well as the estimated network bandwidth of proxy-client and server-proxy links.

This prediction should be performed in advance of transcoding and is obtained from the image header and prior image statistics and analysis. These analysis were performed based on a collection on 2616 GIF images and 108 JPEG images to derive correlation patterns between output file size and transcoding parameters.

We believe that this number of collected images is not large enough to represent the typical image behavior and to generalize the results of this analysis and use it for prediction of the output image file size. For example, we could not find such correlations in our samples set of images.

Bharadvaj, Joshi, and Auephanwiriyakul [16] presented a proxy HTTP server agent (written in Perl) called “MOWSER”. Unlike most of the traditional transcoding proxies, which are unidirectional processes, i.e. the request from the client is passed as is to the target server, MOWSER performs an active transcoding of HTTP/1.1 requests from the client while sending it to the server, according to the preferences specified by the mobile host user, so that the document in the most suitable format is retrieved. It also processes the received HTTP data before sending it to the mobile host if necessary.

A Mowser user has to fill out a CGI form maintained by the proxy server assigning his/her device color capability, video resolution, sound capability, maximum allowed size for text, image, video, audio files and files of unknown type, and the size reduction technique for image files.

When the proxy receives a request from the mobile host, it looks up the preferences stored with the IP address of the mobile host and processes the request accordingly. Default preferences are used if no preferences had been specified. The proxy processes requests to set preferences and the GET requests. All other requests are forwarded to the target WWW server.

On any change to the network connection or the available resources, the user has to update the preferences manually through the CGI form, which is very hard in environments where the network bandwidth is changing very often.

As for image processing, Mowser converts all images to be reduced to portable pixmap format for processing and then converts them back to GIF format for displaying. Then the original URL in the image tag is replaced with the URL of the modified image stored locally by the proxy and sent to the mobile host.

The idea of active transcoding is a good idea. However, using Perl as a programming language introduces a high overhead and slows down the transcoding process. Also, not all of the user preferences set by the user are utilized, making the proxy very static in some cases.

Ma, Bedner, Chang, Kuchinsky, and Zhang [17] presented a framework for adaptive content delivery in heterogeneous network environments. They introduced different aspects of system technologies including the discovery modules for detecting client capabilities, and network characteristics, various content adaptation techniques to improve web accessibility and information delivery, and a general decision-making framework for optimizing adaptive content delivery over the Internet. They provided a list of content adaptation techniques that can be used as a guideline to motivate research in this area.

We believe that the mechanism they introduced for reliably detecting the software and hardware capabilities of a client device is very useful in many cases. As observed, most of the transcoding proxy servers do not have the ability to detect the client device characteristics. Instead they get this information from the user preferences that reduce the flexibility of using such proxies.

Acharya and Smith [18] presented MiddleMan, a cooperative caching video server. MiddleMan is a collection of proxy servers that, as an aggregate, cache video files within a well-connected network (e.g., a campus network or LAN). By cooperatively caching video files, MiddleMan can have a large aggregate cache while placing minimal load on each participating client.

The architecture they proposed, for MiddleMan, consists of a collection of caching video proxy servers that are organized by *coordinator*s. A coordinator is a process that keeps track of the files hosted by each proxy and redirects requests accordingly. The coordinator also uses the proxies to manage the copied video files stored at each machine. If there is no free space left in the system, it is the coordinator that decides which files to eliminate in order to make room.

Although MiddleMan focuses on video files while we focus on images, the idea of introducing a set of proxies may be a promising avenue for future work.

Hori, Kondoh, Ono, Hirose, and Singhal [19] introduced an annotation-based transcoding system developed on top of a programmable proxy server. They used an annotation vocabulary that is used for adaptation hints on rendering HTML documents for personal computing devices. The vocabulary includes three types of annotation: *alternatives, splitting hints* and *selection criteria*.

Alternative representations of a document or any set of its elements can be provided. For example, a color image may have a grayscale image as an alternative for clients with monochrome displays. A transcoding proxy selects the one alternative that is suitable for the capabilities of the requested client device. Elements in the annotated document can then be altered either by replacement or by on-demand conversion. An HTML tag that specifies a list of alternative representations for an annotated element is introduced.

Splitting hints are used to divide an HTML file, which can be shown as a single page on a normal desktop computer, into multiple pages on clients with smaller display screens. Another use for the element is to provide hints for determining appropriate page break points. Alternatives may be provided for the group as a whole. For example, a news headline may be associated with an alternative for a news story that consists of paragraphs of text and some images.

Selection Criteria: An annotation may contain information to help a transcoding proxy select from several alternative representations the one that best suits the client device. The *selection criteria* include information about the client device capability expected for an alternative resource, resource requirements of an alternative representation, fidelity of an alternative representation in relation to an original item, role of an annotated element and importance of an annotated element.

As we focus on image transcoding, Annotation-Based Web Content Transcoding is a good alternative for devices that have limited capabilities of viewing an image. However, replacing the image with an alternative text dose not mean that the image was not downloaded on the proxy side. Since HTML dose not contain any information about the image, it has to be downloaded first at the proxy and then obtain such information used in the annotation algorithm, which seems to be one of the drawbacks of the system. On the other hand, replacing an image with an alternative text seems not acceptable in some cases where a user is interested in seeing such image. The text-only mode, which is equivalent to replacing image with text, is already one of the options that can be set at the browser side for most mobile devices. In our research we deal with such a case using thumbnail option, which is a more practical way for viewing a smaller image. Finally, this system relies on content authors to provide all alternatives.

Deng and Lim [20] investigated web data compression as the mechanism to solve the Internet bandwidth limitation. They proposed a new classification for web data compression: file, block, and stream.

1. *Whole File Compression:* Under this class, the compression is applied on the entire file of a web object. There is no compression/decompression on the partial object data.

2. *Data Block Compression:* This class of compression processes the information that is stored in the HTTP proxy buffer. Since a web object is made up of one or more data blocks, compression on a web object implies individual compression and decompression processes on data blocks that appear in the proxy buffer.

3. *Data Stream Compression:* As its name implies, this data stream compression treats data as a continuing stream. Whenever a proxy server receives some data, it will compress/decompress the data immediately and the result will then be passed to the next level of the network. There is no need to buffer data for future compression/decompression.

Based on the different stages of triggering these compression processes, each of these three approaches has different implications on the complexity of the proxy system and the compressor. The authors recommended that data block compression is the most suitable compression approach for web multimedia data in the HTTP proxy.

The portion we are must interested in is the GIF and JPEG transcoding. For GIF image transcoding they recommend to use the GIF-to-JPEG conversion and perform quality reduction of 25% of the converted image. They apply the quality reduction in this case because most GIF objects are for decoration purposes and can tolerate some degree of loss. Furthermore, the loss in the image quality due to compression is too small to be noticed by web surfers. Due to the multi-pass nature of the GIF-to-JPEG transformer, it cannot be used in data stream compression. Furthermore, for data block compression, there are two additional requirements:

- If the GIF object size is greater than the proxy buffer size, no GIF-to-JPEG compression will be performed. This is because the transformer cannot work on partial images.

- If the GIF object size is less than 4 Kbytes, no GIF-to-JPEG compression will be performed.

For JPEG objects, no transcoding compression is done for all three types of compression. They listed some reasons for this. First, JPEG objects are already in compressed format. So applying more compression to JPEG object will cause a loss in the image quality. Secondly, the JPEG objects on the web are used much less often for decoration than GIF objects are; hence losing image quality for further JPEG transcoding might result in user dissatisfaction.

We believe that JPEG transcoding is a very effective way for speeding up the download time and should be considered in any transcoding system. Otherwise the system will not be efficient. The concept of losing image quality for further transcoding should not be generalized. Based on many experiments and surveys, a user can still be satisfied with a reasonable degree of lossy JPEG compression.

New business ProxiNet [21] is a company focusing on improving Internet access and personalized information delivery to mobile clients, such as handheld devices. The engine driving ProxiNet's products is TranSend [22], a Web accelerator proxy technology that reduces the Web traffic by a factor of three to seven. TranSend speeds up Web access and reduces loading times by reducing the size of images and video on the Web. TranSend transforms data to a more reasonable form by throwing away image resolution or color information and by transcoding data (images and text) into formats the client device can best understand.

Based on their experiments, the TranSend proxy translates an image into a poorer quality image with a one-second download time over a 28.8 kbps modem (two seconds over a 14.4 kbps modem) where it takes 11 seconds over a 28.8 kbps modem (22 seconds over a 14.4 kbps modem) without the proxy. This equals a speed improvement factor of 10.

The TranSend proxy is browser-independent and runs inside the network. TranSend sits right next to the bank of modems between the browser and the server, and acts as an intermediary. To the server, TranSend looks like a client. TranSend intercepts the client request and then interacts with the Web server on the client's behalf.

TranSend is built on three parameter-based distillers:

- Geometry resizing and low-pass filtering of JPEG images using the off-the-shelf *jpeg-6a* library.
- GIF-to-JPEG conversion followed by JPEG quality reduction.
- A Perl HTML "munger" that marks up inline image references with distillation preferences and adds extra links to distilled images so that users can retrieve the original content

In the UC-Berkeley service, TranSend is deployed across a cluster of PCs, and, thus, allows a large number of distillation requests to be processed simultaneously. UC-Berkeley's TranSend is supporting 250,000 requests per second using 10 PCs.

There are some major problems that prevent TranSend to be in the commercial world. The least of which is that transparent GIFs are no longer transparent after distillation. This problem actually is a common problem in all transcoding proxy servers as well as the ability to support Java applets. Most of those proxies discard these applets as a first step of filtration. Unfortunately, there are many websites based on JavaScript menus and forms and others use Java applets for real time applications.

The biggest problem to user acceptance of TranSend is that it cannot be accepted through firewalls and it conflicts with end-to-end user-authentication protocols such as Secure Sockets Layer (SSL), the most popular form of Web security. Servers that authenticate accesses partially based on the IP address from which the request originates will break.

This SSL problem which is also one of the common problems in all transcoding proxy servers, could prove to be a major setback to TranSend's success if not fixed before it goes commercial, since the bulk of the websites are beginning to implement user-authentication systems, most of which rely on the IP addresses for authentication.

3.3 Conclusion

There is a growing need for using transcoding proxy servers in the wireless environment in order to reserve radio resources and not to overload mobile hosts with unnecessary information and wasteful time to download this information.

As we discussed, some researches focused on compressing and caching the contents such as images. Others focused on annotation-based Web content transcoding.

There are two common problems that make these proxies inflexible:

1. Most of them cannot efficiently support various kinds of mobile computers of different capabilities. They also cannot detect the network bandwidth available at the moment of downloading. In other words, the user has to submit this information to the proxy indicating his/her device characteristics and network bandwidth, which is very hard in most of the cases.
2. User satisfaction is not considered in many of these proxies when they apply certain transcoding operations on the image.

We develop a more powerful and flexible image transcoding proxy that uses all the available information about the device and network bandwidth. We use this information for selecting the most appropriate transcoding operation in order to improve adaptation to the client device. We also use our own surveys and analysis to determine the limits of user satisfaction and consider this during transcoding process. As a result, the delivered image matches the user satisfaction and also can be downloaded very fast due to the significant file size reduction.

As with all proxies, we do not support Java, JavaScript, PING or MPEG formats. We focus only on images because they represent the bulk of data on the net. However this is one potential area of future work.

Chapter 4: Data Collections and Surveys

4.1 Introduction

In this chapter, we analyze some characteristics of the web content and show our results of one main survey on the Internet. Our target is to come up with the appropriate transcoding techniques for the most important classes of Internet images such as JPEG and GIF.

4.2 Data Collection Environment

Many statistics and surveys have been done to give the proxy server developers more understanding of the nature of typical Internet images and their transcoding characteristics. Unlike most of the methods used for measuring Web traffic near proxies or near Web servers [6], [7] using methods such as routing requests, we collected and measured our data by applying a simple batch program at the browser. This batch was applied on 7 different users with a widely different range of interests. For a period of 3 weeks, they browsed a very heterogeneous collection of websites. After classifying these traces for each user we confirmed that the most frequently downloaded file types are GIF and JPEG images.

Referring to Table 4.1, GIF images represent 62% of the total number of accessed files. They are also representing 50% of the bytes downloaded by the user. By browsing these images we also verified that the GIF format is usually used for representing buttons, banners, text in graphic form, lines, animated advertisements and simplistic images.

JPEG comes after GIF, representing 29% of the total number of accessed files and 32.5% of the downloaded bytes. The average file size for JPEG is 5.8 kB, 1.6 kB more than the average for a GIF file, which is 4.2 kB.

	Number of Files	Sum of File Size	% File Type	% File size
GIF	181,114	760,034,763	62.24%	50.43%
JPEG	85,077	490,751,947	29.24%	32.56%
HTM	18,906	153,106,542	6.50%	10.16%
TXT	1,370	70,117,148	0.47%	4.65%
JS	1,233	4,232,615	0.42%	0.28%
CSS	822	2,691,776	0.28%	0.18%
CLA	685	3,763,664	0.24%	0.25%
JPE	548	4,077,120	0.19%	0.27%
AU	274	11,252,906	0.09%	0.75%
MAI	137	1,042,570	0.05%	0.07%
MPG	137	2,356,537	0.05%	0.16%
YOD	137	1,548,374	0.05%	0.10%
EXE	137	1,419,320	0.05%	0.09%
Total	**290,577**	**1,506,395,282**		

Table 4.1: File Type and File Size Distributions for Selected WWW Users

After collecting 1500 GIF and JPEG images using Gozilla [23], we found that the average file size in our initial sample set is bigger than the typical file size for both formats. We downloaded 2000 more small images to reach this typical number. Finally we got 2000 JPEG and 1500 GIF images.

We initially applied the following optimization operations on these sets of images:

1. Reduce the image size using different resizing factors
2. Convert colored images to gray scale
3. Reduce the number of colors for colored GIF images
4. Reduce the JPEG image quality using different quality factors
5. Apply gamma correction

6. Smoothing
7. Change the image contrast
8. Converting image format (GIF to JPEG or visa versa).

The objective of applying such operations is to find a direct relation between the nature of the image and the most effective operation. We classified this huge number of images into categories based on the image parameters such as number of colors, histogram*, image size, and color depth. We derived tables that correlate the operation type, reduction percentage and image characteristics. We used different ways like horizontal and vertical methods for analyzing this data to derive a direct, linear or non-linear relationship between the compression ratio and the image parameters. Unfortunately we were unable to find such relationships. However, we noticed that images coming from the same website have similar compression ratios. One possible explanation could be that images from the same website are created or scanned by the same software. Subsequently, when we applied the same operation on these images we got similar compression ratios. This also means that the way in which an image has been originally created or scanned is an important factor of the image file size, i.e. the software used for

* A Histogram is a graph of the distribution of red, green, blue, and/or luminance values in an image. The horizontal axis represents the lightness of the image from black to white (0 to 255). The vertical axis indicates the number of pixels. At a lightness level where there are many pixels of one of the values, the corresponding line spikes; where there are few pixels, it appears close to the horizontal axis. In dark images, most of the pixels are grouped at the left side. If an image is very light, the pixels are grouped at the right.

creating or scanning the image could affect its file size. To verify this, we performed a simple test on 4 different images, saving the same image using different image processing software without changing the default settings for any of them.

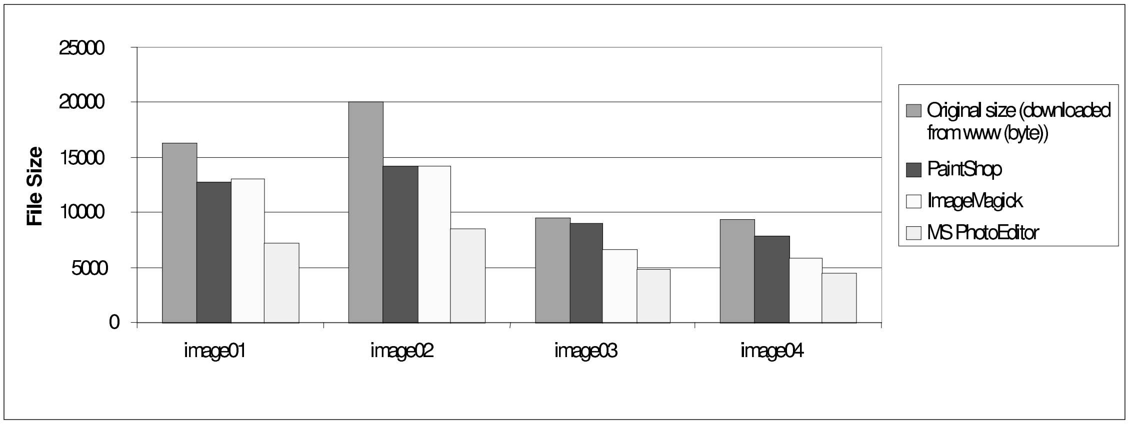


Figure 4-1: Saving Identical Images with Different Image Processing Software

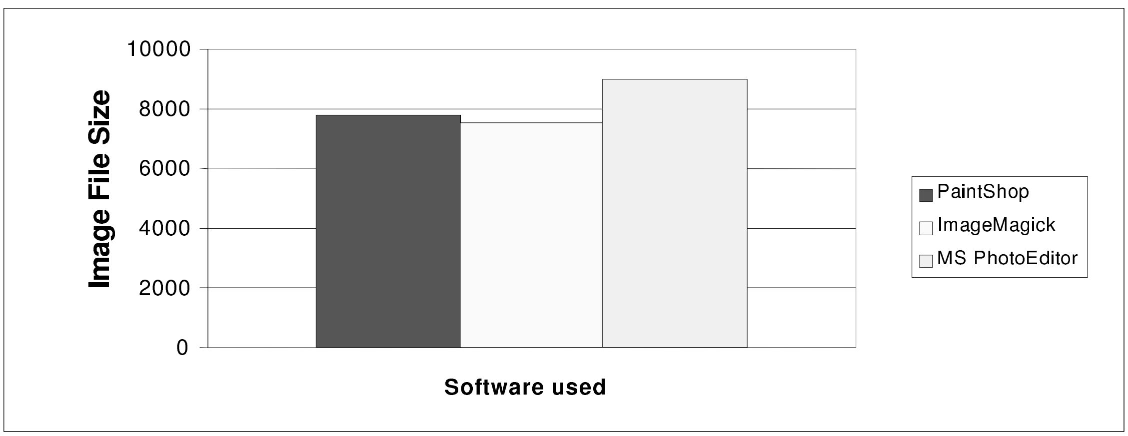


Figure 4-2: Creating Identical Image with Different Image Processing Software

As shown in Figure 4.1, the output file size depends on the software used. We also performed a test (Figure 4.2) to create an image using different software and another test scanning the same picture using different scanner software. They also resulted in different file sizes. These tests and their results can be reviewed in Appendix A.

To eliminate the software factor and obtain a more predictable Compression Ratio, all experimental images should be created/scanned by the same software. While this will not reflect the reality on the Internet, we therefore opened and resaved all our images with ImageMagick. By doing this, we eliminate software factor and the resulting compression ratio will be more predictable. Even after this, no direct relationships were identifiable.

4.3 Most Effective Operations for Reducing File Size

As we did not come up with a relationship between image parameters and the best operation to be applied, we tried to determine which operations are most effective for all images in terms of file size reduction. Returning to the results of the 9 ImageMagic operations we applied, taking into account the effect operations that change image appearance, we were easily able to determine the most effective operations for both formats.

As for GIF images, we found that the most effective operations are as follows: resizing, reducing number of color, converting from GIF to JPEG, and sometimes gray scale. When we applied these operations, it is difficult to predict which one gives the best result. The size reduction depends on the operation factor. For example, applying 50% resizing factor results in higher reduction than applying 30% using the same operation. In some cases, resizing the image is more efficient than color reduction. For other cases we observe reverse results. As for the rest of the operations such as *blur, contrast,* or *Gamma*, they provided very poor file size reduction with most GIF images.

For JPEG images, we found that the most effective operations are as follows: reducing image quality, resizing, gray scale and Despeckle. Table 4.2 shows the number of JPEG images among 1989 collected JPEG's, which were reduced as a file size by applying each operation. The rest of the images in each row

Order	Effective Operation	Number of Reduced Images (as a File Size)	Percentage of Reduced Images	Number of Images Not-Reduced (as a file size)	Percentage of Not-Reduced Images
1	Quality = 10%	1915	96.28%	74	3.72%
2	Geometry resize by 50%	1608	80.84%	381	19.16%
3	Quality = 20%	1968	98.94%	21	1.06%
4	Quality = 30%	1956	98.34%	33	1.66%
5	Quality = 40%	1939	97.49%	50	2.51%
7	Quality = 50%	1599	80.39%	390	19.61%
8	Convert to Gray Scale	1130	56.81%	859	43.19%
9	Despeckle	1569	78.88%	420	21.12%

Table 4.2: Effective JPEG Operations

After determining the most effective operations for GIF and JPEG images, we performed them in different orders to see whether the reduction in file size would be the same. By doing so, we can decide in which order we should perform these operations in order to get the best reduction. Therefore, we applied these operations in different orders on some selected images. As a result we found that the resulting file size reductions are almost the same. However, we noticed that the output images are more acceptable with some sequences. Unfortunately, we could not come up with a general conclusion so far.

4.4 Translating between Image Formats

Converting images to another format often reduces file size without overly effecting image quality, as demonstrated by Mowgli [24]. However, the amount of reduction usually depends on the image contents and the type of the selected new format. Moreover, there is no guarantee that this operation will be always successful with all images all the time, especially if the image is converted to an unsuitable format. Therefore, some experiments were performed with GIF and JPEG to explore suitable conversion patterns. After applying conversion operations on all our images, the following are the observed results.

4.4.1 Converting from GIF to JPEG

1. Excellent reduction in file size was achieved when GIF images that contain natural scenes were converted to JPEG. Moreover, the quality of images after converting is still close to the original.
2. Converting small GIF images to JPEG will increase the file size. It seems that GIF images with file sizes below 1 kB are not worth to convert.
3. It is not recommended to convert a GIF image to JPEG format when its **p**ixel **p**er **b**it (PPB) is less than 0.1. We obtain PPB by dividing the file size by the multiplication of width and height of an image: PPB = File Size / (width x height).

4.4.2 Converting from JPEG to GIF:

About 90% of the experimental images had a bigger file size after applying this type of conversion. This implies that converting JPEG images to GIF does not appear promising.

4.5 Measuring the Client Interactions Using Different Scenarios

We conducted two different surveys for measuring the client interactions:

- Image Quality Satisfaction.
- Accessing and Downloading Cached Files (Appendix B).

For more accurate results, both surveys were done close to the browser side. Here, we will present and analyze the Image Quality Satisfaction Survey utilized the conclusion of this survey in our proxy algorithm.

Image Quality Satisfaction

The objective of the image quality satisfaction survey is to measure the lowest image quality that can be accepted from the user perspective and to come up with an optimized quality reduction factor, color reduction factor and resizing factor. These values can be used as a limit when applying any of the image processing operations through our proxy. As a starting point, we built 4 CGI forms showing different categories of images taking into consideration the typical web page (Appendix C).

- The first form has 10 JPEG original images collected from the web. A quality reduction operation was performed using 10 different percentages.
- The second form contains 4 different original GIF images. A color reduction operation was performed using 10 different percentages.
- The third form displays 4 different JPEG/GIF images. Resizing operations were performed using 9 different percentages.

- The last form contains samples of 4 GIF images. "GIF to JPEG" file format conversion operations were performed.

For each of these forms, a user checks which of those images are of acceptable quality from his/her perspective. After submitting the forms, an email will be automatically submitted with their contents. We received more than 220 replies. 166 of these replies were used (complete and consistent).

Tables 4.3, 4.4 and 4.5 show samples of results for one user for measuring the quality acceptance. "1" means that the user accepts the image quality, "0" means that he/she does not accept it. For more details see Appendices C, D, E and F.

Resolution 1280 x 1024
Colors 32 bit

% Quality	100	80	60	40	30	20	10	5
Image 1	1	1	1	1	1	1	0	0
Image 2	1	1	1	1	1	0	0	0
Image 3	1	1	1	1	0	0	0	0
Image 4	1	1	1	1	1	1	1	0
Image 5	1	1	1	1	1	1	0	0
Image 6	1	1	1	1	1	1	1	0
Image 7	1	1	1	1	1	1	0	0
Image 8	1	1	1	1	1	1	0	0
Image 9	1	1	1	1	1	1	0	0
Image 10	1	1	1	1	1	1	0	0
Accept. Avg.	**10**	**10**	**10**	**10**	**9**	**8**	**2**	**0**

Table 4.3: Sample of Results for Image Quality Acceptance

% Color Reduction	100	80	60	40	30	20	10	5
Image 1	1	1	1	1	1	1	1	0
Image 2	1	1	1	1	1	1	1	0
Image 3	1	1	1	1	1	1	1	0
Image 4	1	1	1	1	1	1	0	0
Accept. Avg.	4	4	4	4	4	4	3	0

Table 4.4: Sample of Results for Image Colors Acceptance

% Resizing	100	80	60	40	30	20	10	5
Image 1	1	1	1	1	1	1	1	0
Image 2	1	1	1	1	1	1	1	0
Image 3	1	1	1	1	1	1	1	0
Image 4	1	1	1	1	1	1	0	0
Accept. Avg.	4	4	4	4	4	4	3	0

Table 4.5: Sample of Results for Image Resizing Acceptance

Summing up these numbers for a set of different images for each processed image gives an average number that indicates how much the user is satisfied with this transcoding operation.

For example, in Table 4.3, "8" means that user satisfaction for the "20% quality factor" operation equals to 8 images among 10 (80%). "2" means that user satisfaction for the "10% quality factor" equals to 2 images among 10 (20%) etc. We classified the 166 users into three categories:

1. Users with High Quality Displays
2. Users with Medium Quality Displays
3. Users with Low Quality Displays

For each of these categories, we calculated the 95% confidence interval for our measure of user satisfaction with a given transcoding operations. As a definition, the confidence interval indicates the range within that the true value of a statistical measure lies with a certain probability. A 95% confidence interval in our case means that the true value of the minimum accepted quality factor, colors reduction factor or geometry resizing factor lies with a probability of 95% within this interval.

Usually, the confidence interval is expressed as the range within which only 5 percent of samples would generate estimates lying outside the range (a 95 percent confidence interval). For more precision, one might describe a wider confidence interval, that suggests the range outside of which only 1 per cent of random samples would generate estimates. Clearly, the bigger the sample, the more certain one can be of the estimate, and the smaller the confidence interval [29].

We calculated the Lower Limit (L) and Upper Limit (U) follows:

$$L = (X + C^2/2 - C * SQRT(X*(N-X)/N + C^2/4)) / (N+C*C)$$

$$U = (X + C^2/2 + C * SQRT(X*(N-X)/N + C^2/4)) / (N+C*C)$$

{Distribution: Bi-Model Random Variable}

Where: N: the number of observations

X: number of respondents who said "acceptable"

C: constant = 1.96

We evaluated these two equations for each quality reduction operation, color reduction operation, and resizing operation.

Category 1: Users with High Quality Displays

The High Quality Display category contains 39 users who have devices with 1280x1024 or 1024x768 pixel resolution and support true colors (32 bit). Figure 4.3 shows the quality reduction factor as x-axis against the average number of respondents who say "acceptable" as y-axis (applied to JPEG images). Figure 4.4 shows the colors reduction factor as x-axis against the average number of respondents who say "acceptable" as y-axis (applied to GIF images). Figure 4.5 shows the resizing factor as x-axis against the average number of respondents who say "acceptable" as y-axis (applied to GIF and JPEG images).

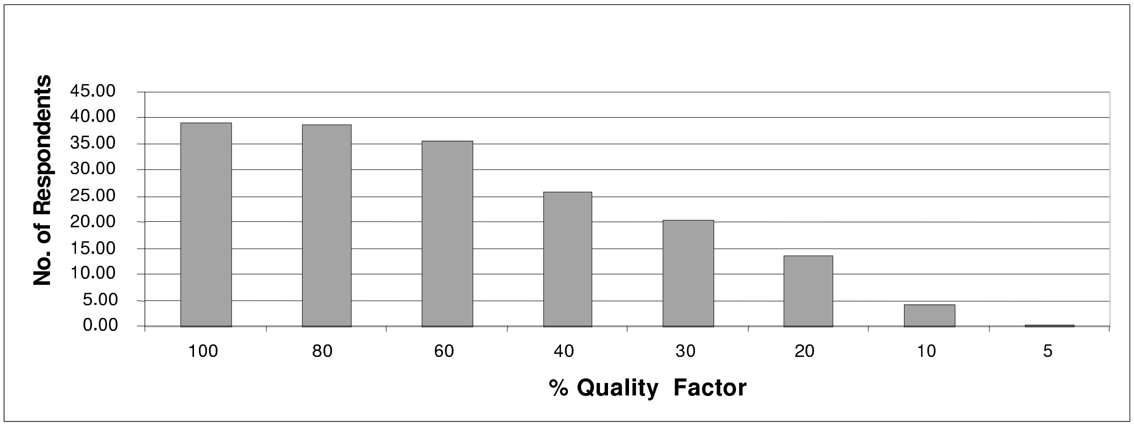


Figure 4-3: Quality Reduction Test - Users with High Quality Displays

Quality	100	80	60	40	30	20	10	5
No. of respondents	39.00	38.90	35.40	25.80	20.30	13.50	4.20	0.30
Percentage of respondents	100.00%	99.74%	90.77%	66.15%	52.05%	34.62%	10.77%	0.77%
Lower Limit	0.9103	0.9057	0.7771	0.5046	0.3691	0.2168	0.0436	0.0005
Upper Limit	1.0000	0.9999	0.9652	0.7895	0.6683	0.5031	0.2421	0.1032

Table 4.6: Quality Reduction Test Results for Users with High Quality Displays

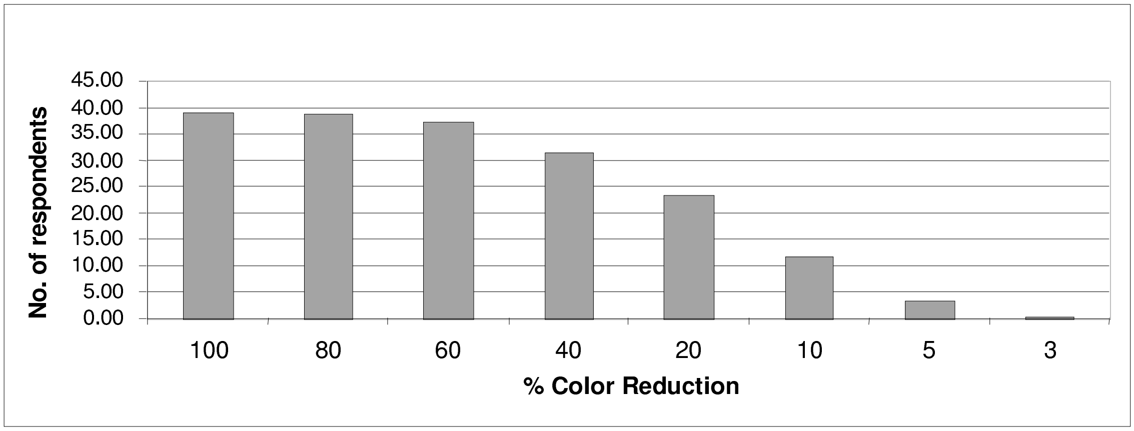


Figure 4-4: Colors Reduction Test - Category 1

Colors reduction %	100	80	60	40	20	10	5	3
No. of respondents	39.00	38.75	37.00	31.50	23.25	11.75	3.25	0.25
Percentage of respondents	100.00%	99.36%	94.87%	80.77%	59.62%	30.13%	8.33%	0.64%
Lower Limit	0.9103	0.8990	0.8311	0.6589	0.4403	0.1806	0.0299	0.0004
Upper Limit	1.0000	0.9996	0.9858	0.9013	0.7347	0.4576	0.2115	0.1010

Table 4.7: Colors Reduction Test Results for Users with High Quality Displays

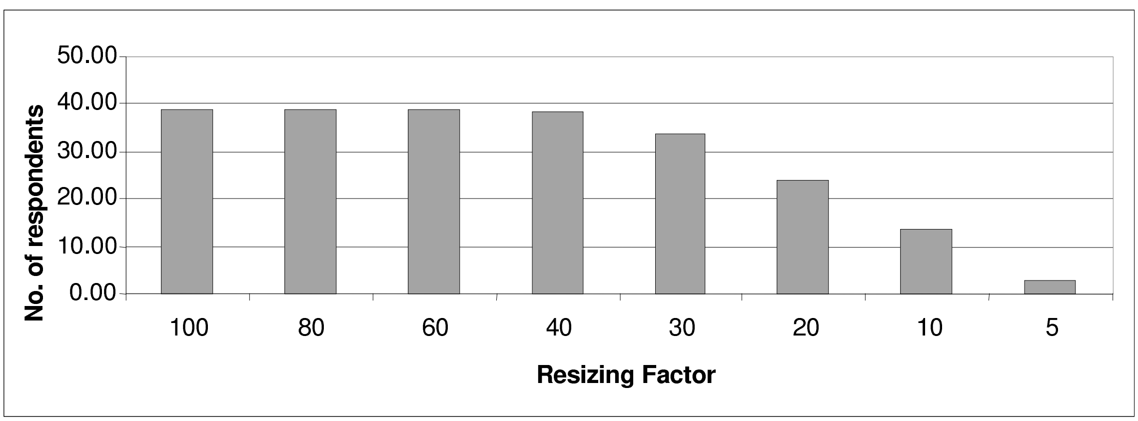


Figure 4-5: Resizing Test - Users with High Quality Displays

Resizing %	100	80	60	40	30	20	10	5
No. of respondents	39.00	39.00	39.00	38.25	33.50	24.00	13.50	2.75
Percentage of respondents	100.00%	100.00%	100.00%	98.08%	85.90%	61.54%	34.62%	7.05%
Lower Limit	0.9103	0.9103	0.9103	0.8781	0.7177	0.4590	0.2168	0.0232
Upper Limit	1.0000	1.0000	1.0000	0.9972	0.9359	0.7511	0.5031	0.1948

Table 4.8: Resizing Test Results for Users with High Quality Displays

Category 2: Users with Medium Quality Displays

The Medium Quality Display category contains 86 users who have devices with 800x600 pixel resolution and support 16-bit color depth. Figure 4.6 shows the quality reduction factor as x-axis against the number of respondents who say “acceptable” as y-axis (applied to JPEG images). Figure 4.7 shows the colors reduction factor as x-axis against the number of respondents who say “acceptable” as y-axis (Applied to GIF images). Figure 4.8 shows the resizing factor as x-axis against the number of respondents who say “acceptable” as y-axis (applied to GIF and JPEG images).

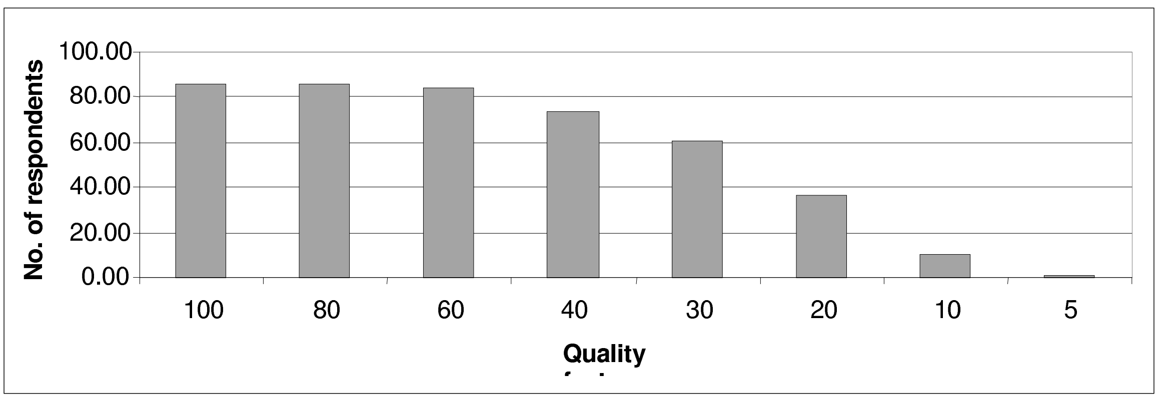


Figure 4-6: Quality Reduction Test - Users with Medium Quality Displays

Quality	100	80	60	40	30	20	10	5
No. of respondents	86.00	86.00	83.90	73.40	61.00	36.40	10.00	1.30
Percentage of respondents	100.00%	100.00%	97.56%	85.35%	70.93%	42.33%	11.63%	1.51%
Lower Limit	0.9572	0.9572	0.9174	0.7637	0.6060	0.3243	0.0644	0.0032
Upper Limit	1.0000	1.0000	0.9931	0.9130	0.7947	0.5288	0.2010	0.0685

Table 4.9: Quality Reduction Test Results for Users with Medium Quality Displays

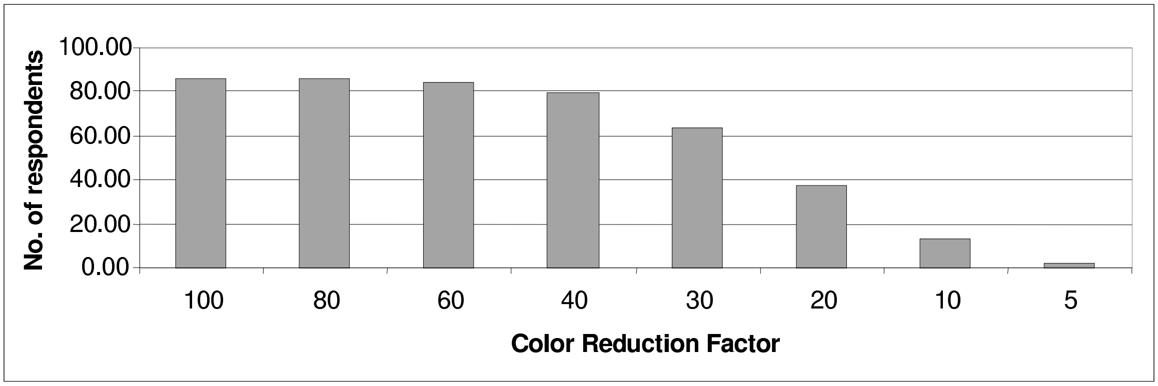


Figure 4-7: Colors Reduction Test – Users with Medium Quality Displays

Colors Reduction	100	80	60	40	30	20	10	5
No. of respondents	86.00	86.00	84.25	79.25	64.00	37.00	13.25	1.75
Percentage of respondents	100.00%	100.00%	97.97%	92.15%	74.42%	43.02%	15.41%	2.03%
Lower Limit	0.9572	0.9572	0.9235	0.8450	0.6429	0.3308	0.0928	0.0052
Upper Limit	1.0000	1.0000	0.9948	0.9619	0.8246	0.5356	0.2450	0.0765

Table 4.10: Colors Reduction Test Results for Users with Medium Quality Displays

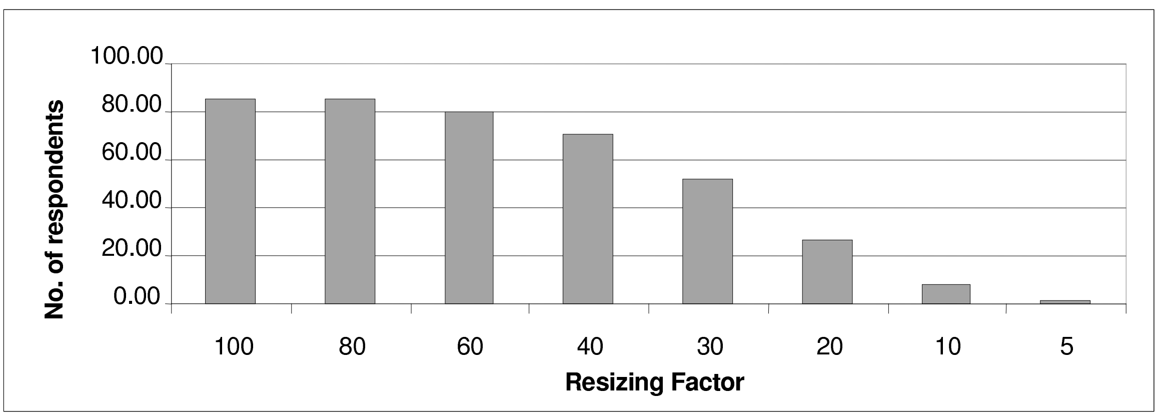


Figure 4-8: Geometry Resizing Test – Users with Medium Quality Displays

Resizing %	100	80	60	40	30	20	10	5
No. of respondents	86.00	85.50	80.00	70.75	51.75	26.25	7.50	1.25
Percentage of respondents	100.00%	99.42%	93.02%	82.27%	60.17%	30.52%	8.72%	1.45%
Lower Limit	0.9572	0.9467	0.8560	0.7287	0.4961	0.2180	0.0439	0.0030
Upper Limit	1.0000	0.9994	0.9676	0.8891	0.6987	0.4091	0.1658	0.0676

Table 4.11: Resizing Test Results for Users with Medium Quality Displays

Category 3: Users with Low Quality Displays

The Low Quality Display category contains 41 users who have devices with 640x480 pixel or lower resolution, supporting 256 or fewer colors. Figure 4.9 shows the quality reduction factor as x-axis against the number of respondents who say "acceptable" as y-axis (applied to JPEG images). Figure 4.10 shows the colors reduction factor as x-axis against the number of respondents as y-axis (applied to GIF images). Figure 4.11 shows the resizing factor as x-axis against the number of respondents who say "acceptable" as y-axis (applied to GIF and JPEG images).

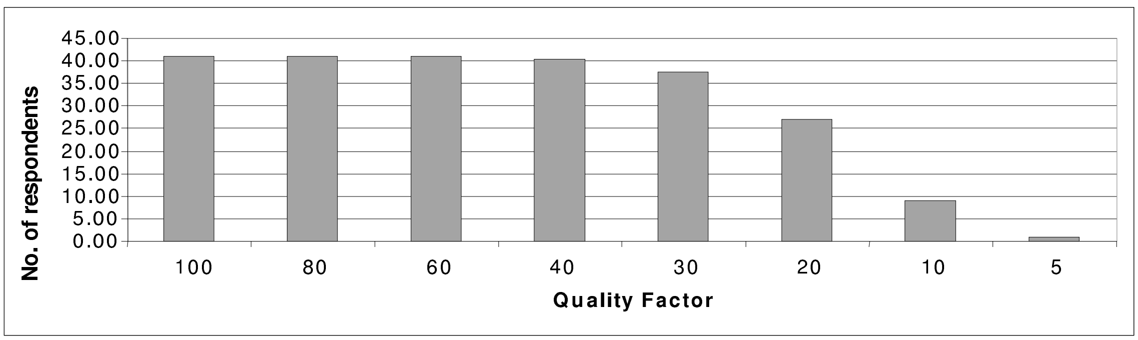


Figure 4-9: Quality Reduction Test – Users with Low Quality Displays

Quality	100	80	60	40	30	20	10	5
No. of respondents	41.00	41.00	41.00	40.50	37.40	27.10	9.10	1.00
Percentage of respondents	100.00%	100.00%	100.00%	98.78%	91.22%	66.10%	22.20%	2.44%
Lower Limit	0.9143	0.9143	0.9143	0.8933	0.7868	0.5079	0.1218	0.0043
Upper Limit	1.0000	1.0000	1.0000	0.9987	0.9669	0.7864	0.3697	0.1260

Table 4.12: Quality Reduction Test Results for Users with Low Quality Displays

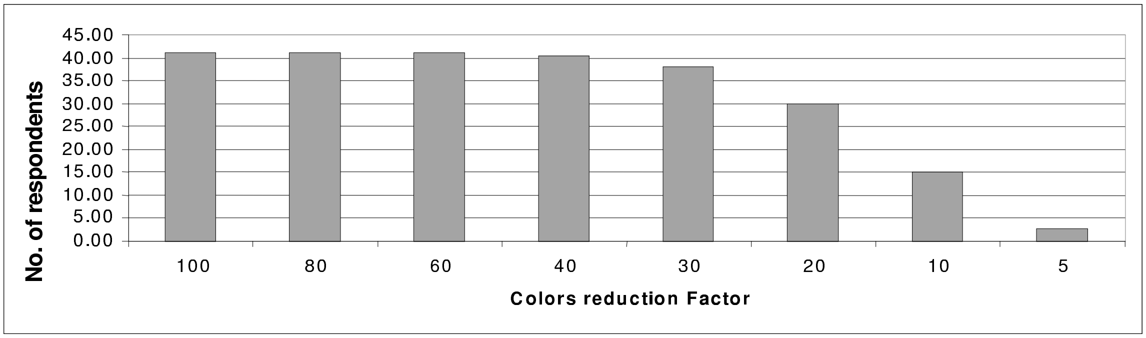


Figure 4-10: Colors Reduction Test – Users with Low Quality Displays

Colors Reduction %	100	80	60	40	30	20	10	5
No. of respondents	41.00	41.00	41.00	41.00	39.00	31.00	15.50	2.75
Percentage of respondents	100.00%	100.00%	100.00%	100.00%	95.12%	75.61%	37.80%	6.71%
Lower Limit	0.9143	0.9143	0.9143	0.9143	0.8386	0.6066	0.2462	0.0221
Upper Limit	1.0000	1.0000	1.0000	1.0000	0.9865	0.8618	0.5308	0.1862

Table 4.13: Colors Reduction Test Results for Users with Low Quality Displays

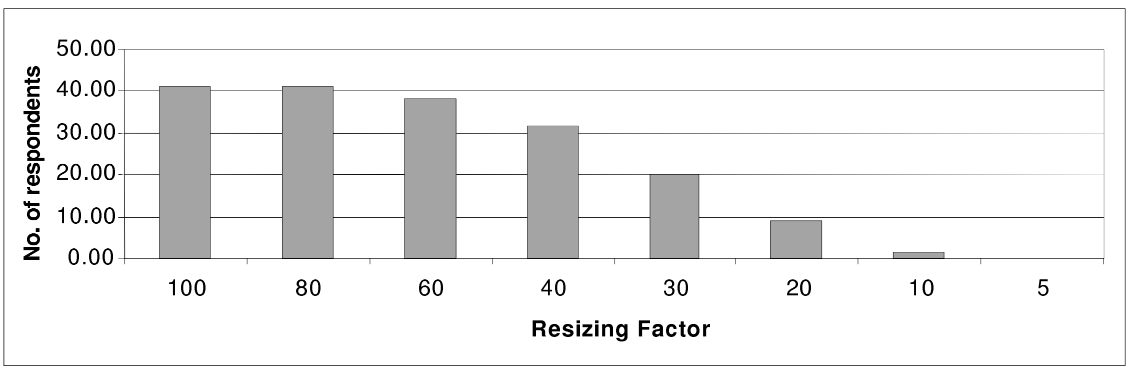


Figure 4-11: Geometry Resizing Test – Users with Low Quality Displays

Resizing %	100	80	60	40	30	20	10	5
No. of respondents	41.00	41.00	38.25	31.75	20.25	8.75	1.25	0.00
Percentage of respondents	100.00%	100.00%	93.29%	77.44%	49.39%	21.34%	3.05%	0.00%
Lower Limit	0.9143	0.9143	0.8138	0.6263	0.3481	0.1156	0.0063	0.0000
Upper Limit	1.0000	1.0000	0.9779	0.8755	0.6408	0.3604	0.1351	0.0857

Table 4.14: Geometry Resizing Test Results for Users with Low Quality Displays

Referring to these results we found that there is a relationship between the device display characteristics and the user image quality satisfaction level. The higher the device quality, the higher user sensitivity to the image quality. In other words, the lowest image quality that can be accepted by the user depends on the device resolution and number of colors supported by this device:

1. For image quality reduction, the highest quality factor reduction can be applied to an image rendered on a low device quality display. This tends to go down with higher quality displays. For a high quality display, a JPEG quality factor of 40% seems acceptable for 50% to 75% of images, while 88% - 99% of these images were judged acceptable by users with a low quality display. Based on the 95% confidence intervals, the differences are therefore statistically significant.
2. The same relationship holds true for the colors reduction factor.
3. The opposite relationship shows for the geometry resizing factor, i.e. the highest resizing factor can be applied to an image rendered by high quality displays, with increasingly lower geometry resizing factors acceptable for low quality displays.

4.6 Conclusion

After examining the 95% confidence intervals we settled on the minimum quality, color reduction, and geometry resizing factors that can be safely applied to an image. Table 4.15 summarizes this conclusion:

Device Quality Displays	Quality Factor (JPEG)	Colors Reduction Factor (GIF)	Resizing Factor
High	40%	40%	30%
Medium	30%	30%	40%
Low	20%	20%	60%

Table 4.15: Maximum Operation Factors for Different Device Quality Displays

Chapter 5: Implementation

5.1 Proxy Architecture:

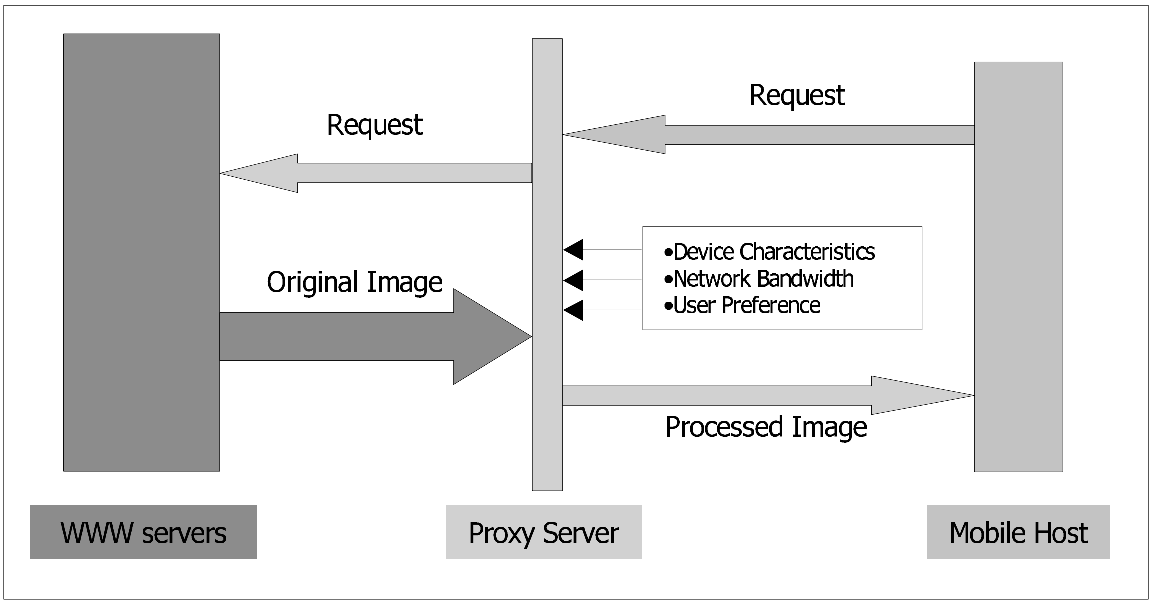


Figure 5-1: Proxy general Architecture

Our image transcoding proxy (Figure 5.1) handles the requests from the client devices for Web images. It checks if the same image already exists in its cache. If so, it compares the cached image with the image on the World Wide Web server, if they are similar, it uses the cached one. Otherwise, an original image is retrieved for the first time from a WWW server. The proxy also receives information about the device characteristics, network bandwidth and user preferences as input parameters, performs the most effective operations on the image using these parameters and produces a transcoded image. The proxy keeps the original image in its cache, writes the transcoded image and delivers it to the user.

1. **Device Characteristics Parameter:**

- Monitor resolution in pixels (DevXdim and DevYdim): These parameters will be used in the geometry resizing procedure for both JPEG and GIF

- Color depth (DevCol): This parameter will be used in the Color Reduction procedure applied on GIF images

Both characteristics are assumed to be detected by the proxy either through the request header, submitted manually by the user, or even by running a Java applet at the client browser. For the time being, we pass both characteristics as an external input file to our program.

2. **Network Bandwidth Parameter:**

The network bandwidth parameter is the current value of the bandwidth available on the network i.e. the measure of how many bits per second the connection between user and proxy can carry. This can be established by noting how long it took to send one of the data packets. As this measurement is not of our concern in this research, we passed this parameter as an input file to our program.

3. **User Preferences Parameter:**

Users can control the operation of the proxy server by entering some preferences at the browser side so the proxy takes them into consideration when performing the operation on the image:

I. Receive images: Yes/No. (Yes is the default) If user checks No, the program skips all processing steps and excludes images when transmitting a web page to the user.

II. Image quality: Users specify a number ranging from 10% to 100% (for JPEG only). In such a case, the proxy uses the user quality factor entered. Otherwise the program determines a suitable quality factor that should be applied, as will be mentioned in the software design later in this chapter.

III. Image color: color/gray. (Color is the default) If the user selects gray, then our proxy will skip the color reduction procedure and convert any colored image to gray scale.

IV. Freeze GIF animation Yes/No. (Yes is the default). If a user chooses yes, then the proxy will replace the animated GIF with the first frame of the animated GIF and remove the rest of the frames.

V. Thumbnail Yes/No. (No is the default). If a user is not interested in seeing the images in their original size to reduce the Download Time, he/she can choose yes, then the proxy will replace each image with an equivalent small image.

5.2 Proposed Proxy

We reviewed three software products, available in source, known as RabbIT [25], Apache [26], and DeleGate [27]. Apache and DeleGate are two widely used proxy servers. They also provide full source code written in C, but are more complicated to deal with.

Rabbit is intended to be HTTP/1.1 compliant. The whole package is written in Java and relatively easy to understand and modify. It does not, however, include a lot of functionality. It works by modifying the web pages so that a browser never sees advertising images, it only sees one fixed image tag (that image is cached in the browser the first time it is downloaded, so subsequent requests for it are served from the browser's cache). For GIF images RabbIT fetches the image and runs it through a processor resulting in a 10% JPEG instead of an (animated) GIF image. For JPEG images, RabbIT applies a fixed quality factor if the JPEG image is bigger than a certain size (can be adjusted in the JAVA code) and only sends the smaller image.

Image conversion is done using ImageMagick library, the same library we are using. The way the RabbIT proxy is performing image conversion is to just call the ImageMagick command line within the JAVA code.

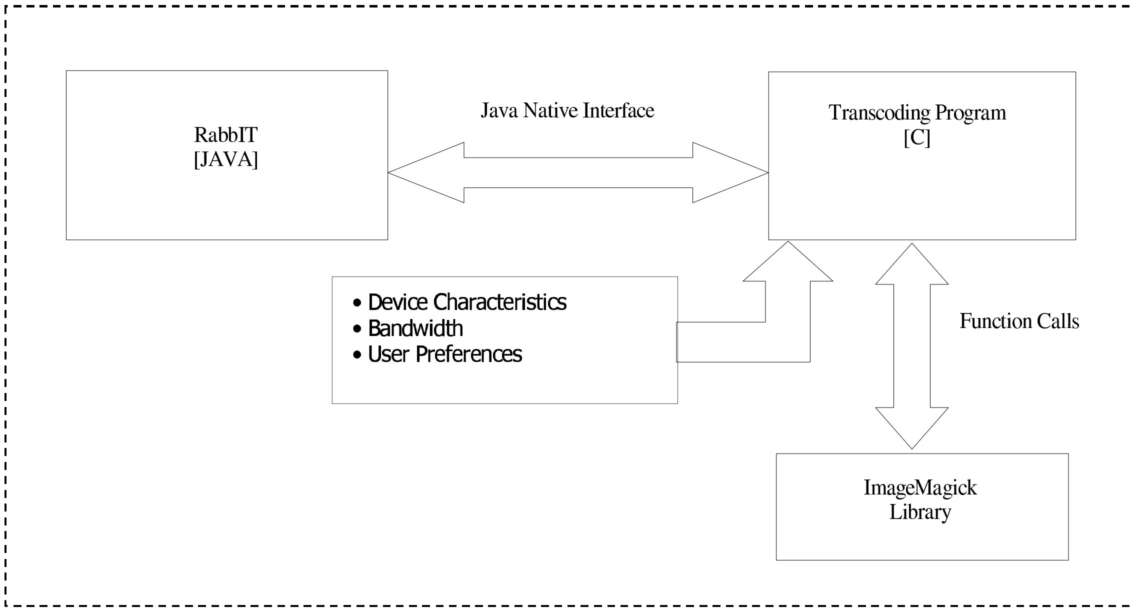


Figure 5-2: Proxy Links and Function Calls

After applying some modifications to the RabbIT source code to suit our logic, we linked it to a C program (Figure 5.2), which has all our new transcoding functionalities. Since RabbIT was written in Java, we used Java Native Interface (JNI) to call our C functions. Using C as a programming language give us the benefit of speeding up the processing which is essential for On-The-Fly image transcoding.

5.3 Software Design

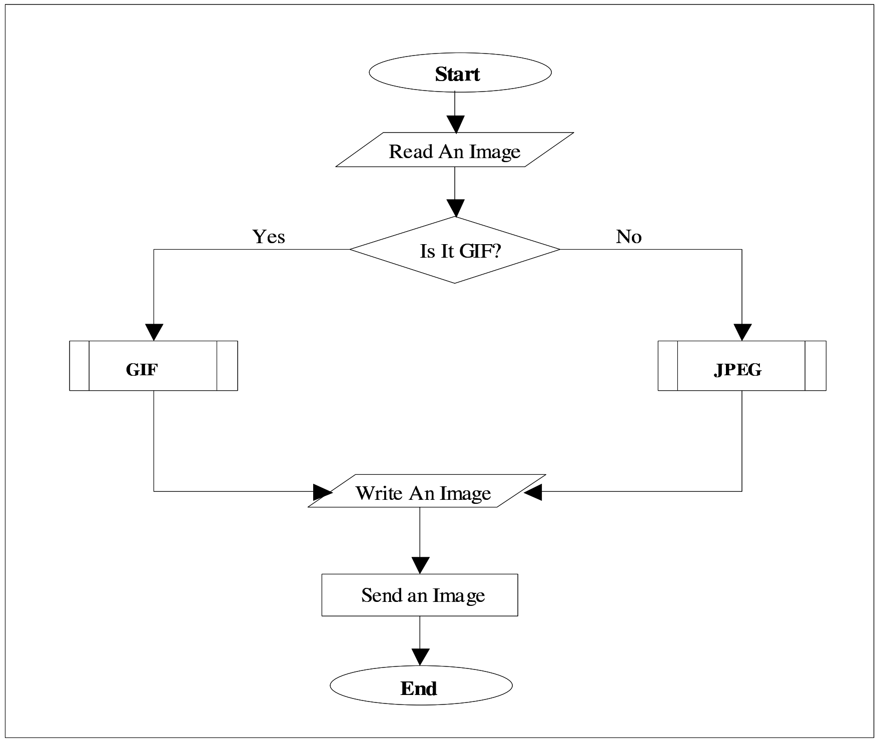
9. 

Figure 5-3: The Main Function

As mentioned, our proxy server does two main functions: image transcoding and caching. Since my main contribution focuses on the image transcoding portion, we discuss only this part in the following.

After downloading the requested image at the proxy, it reads the image information using the image header and checks the file format (Figure 5.3). If it is a GIF format the program calls the GIF procedure, if it is JPEG, the program calls the JPEG procedure. Otherwise, the program writes the image and sends it to RabbIT directly without processing. Since GIF image transcoding was discussed in Gaddah Thesis [28], I focus here on JPEG image transcoding.

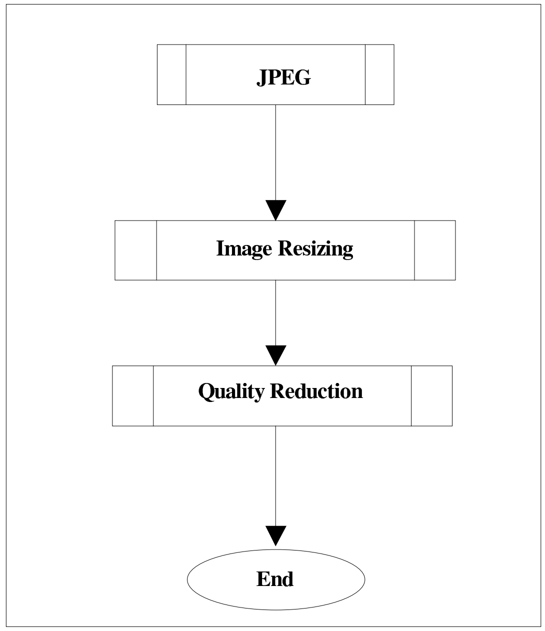


Figure 5-4: JPEG Transcoding

There are two operations that can be applied to JPEG images: quality reduction (Figure 5.4) and/or geometry resizing (Figure 5.5). Experimentally, we configured that ordering is not an effective factor so we can start with either the geometry resizing or the quality reduction procedure. Geometry resizing is a common procedure, which might be used for GIF or JPEG. However Quality Reduction is applied to JPEG images only.

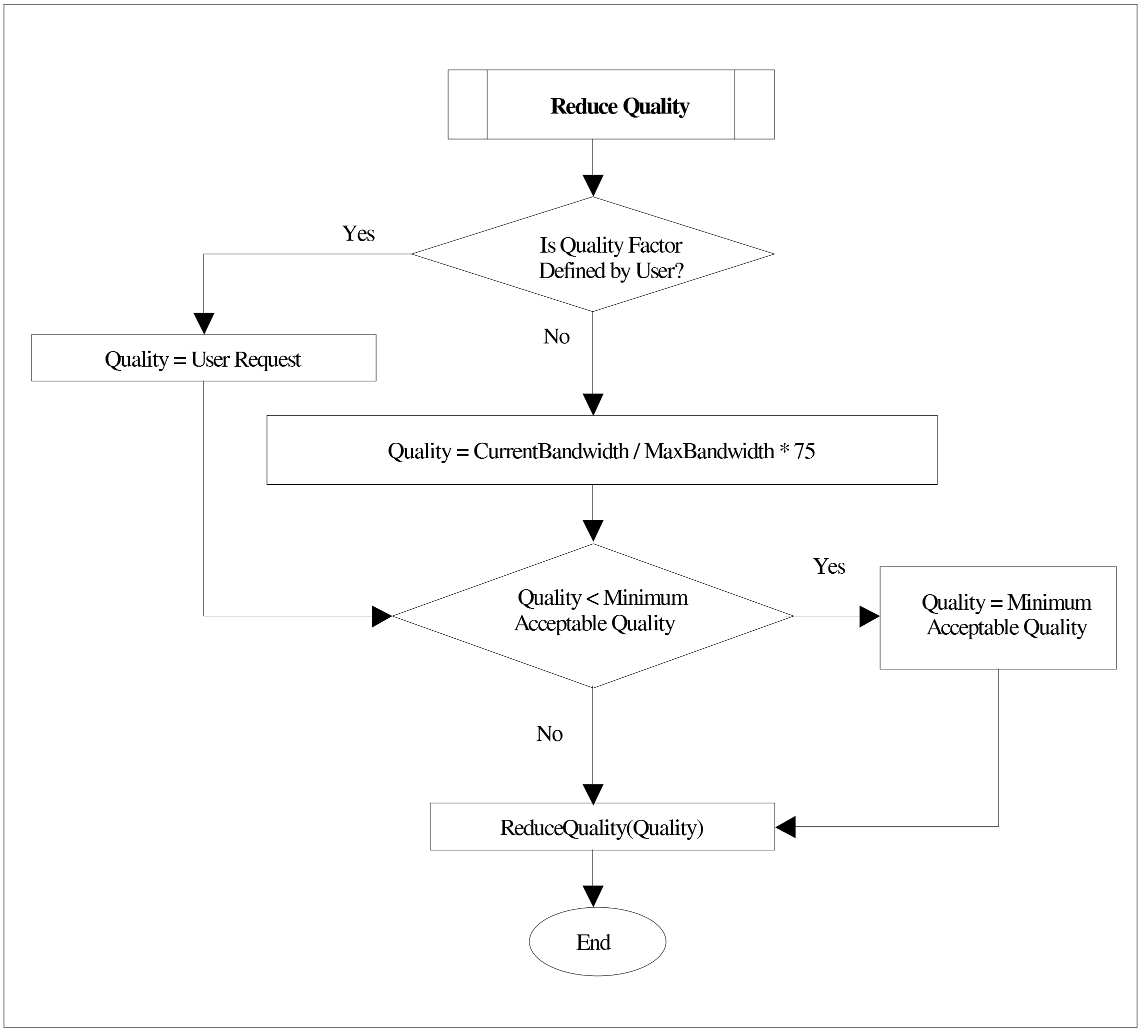


Figure 5-5: Quality Reduction Procedure

5.3.1 Quality Reduction Procedure (JPEG)

Using the user preferences, a user can initially define a fixed quality factor to be applied for all images. If this is the case, the program applies this quality factor without taking the available network bandwidth into consideration. Otherwise, we use the following formula to calculate Quality Factor:

$$\text{Quality Factor} = \text{CurrentBandwidth} / \text{MaxBandwidth} * C$$

Where: *CurrentBandWidth:* The bandwidth available at this time.

MaxBandwidth: A constant value represents the maximum network bandwidth that can be achieved within the network.

C: Constant, based on some experiments, we found that the value of this constant is 75.

Using this formula, any change in the network bandwidth will produce a change to the quality factor applied to JPEG images.

5.3.2 Geometry Resizing Procedure

Before applying the geometry resize operation (Figure 5.5), we check the original dimensions of the image. If it is less than 4x4 pixels, there is no need to call the resizing procedure. Otherwise we involve both the display size and network bandwidth to determine a geometry resizing factor as follows:

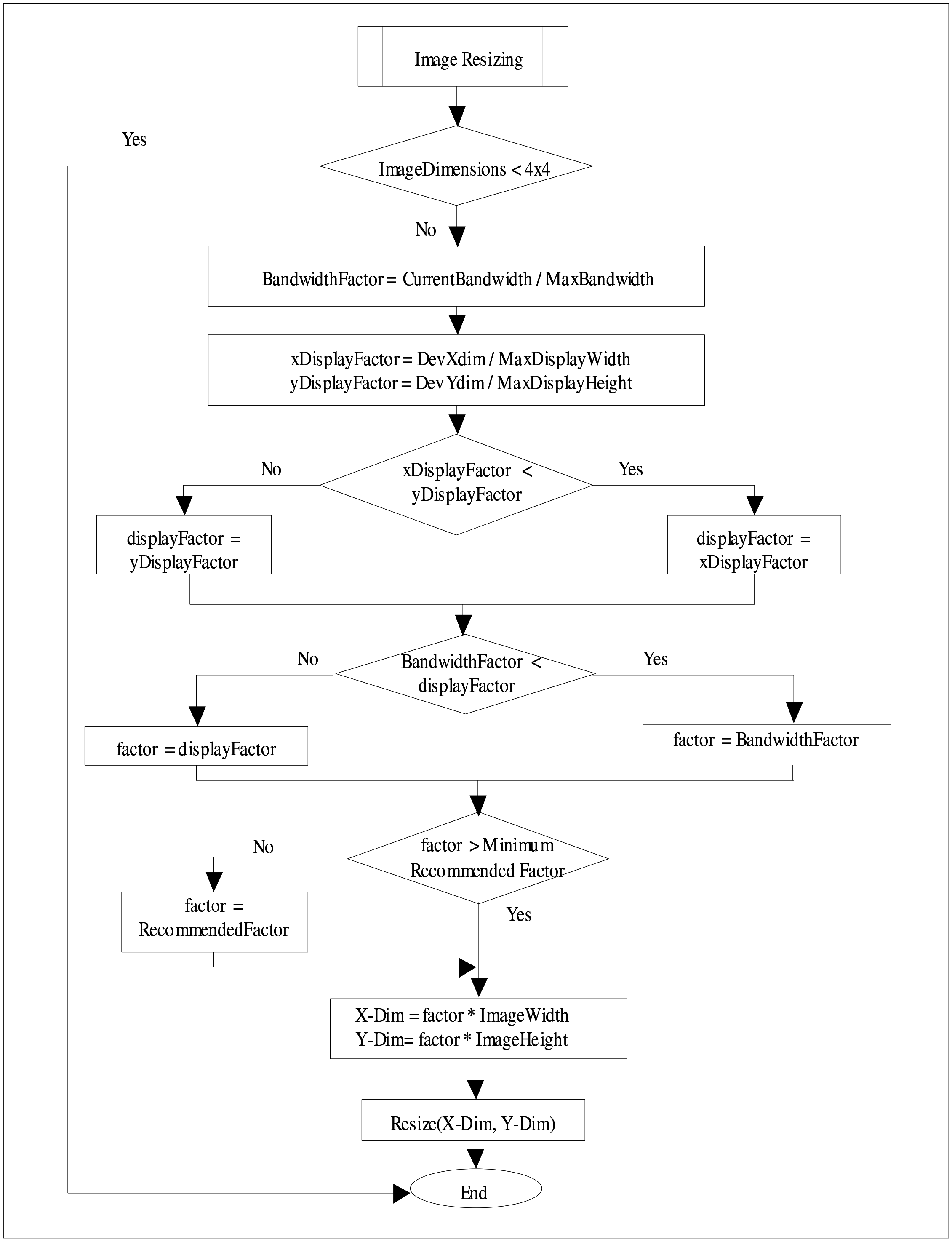


Figure 5-6: Geometry Resizing Factor

Display Factor:

xDisplayFactor = DevXdim / MaxDisplayWidth

Where:

- xDisplayFactor is the horizontal resizing factor
- DevXdim is the device width resolution in pixels
- MaxDisplayWidth is a constant representing the maximal display width resolution allowed for an image

yDisplayFactor = DevYdim / MaxDisplayHeight

Where:

- yDisplayFactor is the vertical resizing factor
- DevYdim is the device height resolution in pixels
- MaxDisplayHeight is a constant representing the maximum supported height for the display resolution.

Comparing xDisplayFactor and yDisplayFactor we select the lower value and use it as a display factor to ensures that the geometry resized image will never been larger than the display resolution, keeping the width : height ratio constant.

Bandwidth Factor:

BandwidthFactor = CurrentBandwidth / MaxBandwidth

Where:

CurrentBandWidth: The network bandwidth available at this time.

MaxBandwidth: A constant value representing the maximum network bandwidth that can be achieved within the network.

We use the lower value of the display factor and the network bandwidth factor as a geometry resizing factor, then we compare it with the minimum geometry resizing factor recommended for this device, pass the higher value to the ImageMagic geometry resizing function and write the transcoded image.

Chapter 6: Evaluation and Results

6.1 Introduction

In the wireless environment, the transcoding proxy is designed to have a relatively high network bandwidth at the http server side (wired link side). In most cases, the proxy has a relatively low bandwidth at the client side (wireless link side) and has to consider the limitations of the wireless devices. Reducing the file sizes of the images at the transcoding proxy via image geometry resizing, quality reduction and color reduction procedures can result in faster content delivery to the client.

The only restriction we introduced for applying such transcoding to the image is the user satisfaction. As a result of our surveys for measuring this term, our proxy treats the image based on the quality of the user display. This quality level is one of 3 classes:

- *High quality displays,* where the proxy performs a quality reduction ranging from 75% to 40% for JPEG, color reduction ranging from 100% to 40% for GIF image, and geometry resizing ranging from 100% to 30% for GIF and JPEG images.

- *Medium quality displays,* where the proxy performs a quality reduction ranging from 75% to 30% for JPEG, color reduction ranging from 100% to 30% for GIF image, and geometry resizing ranging from 100% to 40% for GIF and JPEG images.

- *Low quality displays,* where the proxy performs a quality reduction ranging from 75% to 20% for JPEG, color reduction ranging from 100% to 30% for GIF image, and geometry resizing ranging from 100% to 60% for GIF and JPEG images.

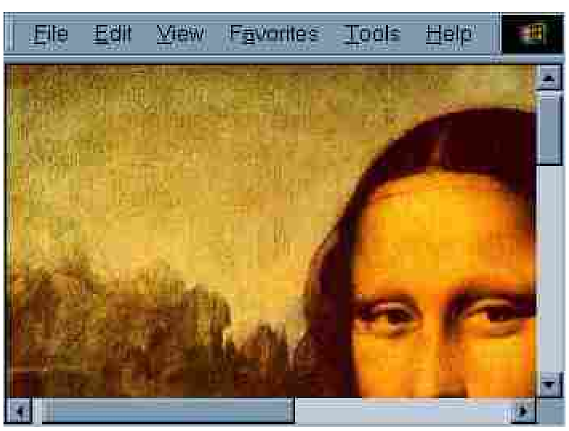
6.2 Case Studies

To evaluate our proxy, we performed several case studies for different image categories. For each case, we compare the original image parameters (i.e. image before transcoding) and the image transcoded by the proxy at certain device and/or bandwidth and/or user preference conditions. We also calculated the download time at the browser side taking into account the CPU time consumed to perform each transcoding session at the transcoding proxy, the transmission time at the http server side and the client side, and also the decoding time at the browser.

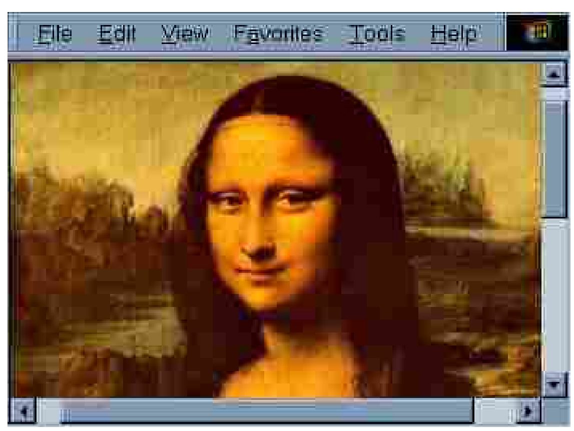
In general, for testing purposes, we installed our proxy on a 550 MHz PII computer running Linux as operating system. Using this machine as a proxy server, we conducted 6 case studies. For each case, we download the page under test at two computers. One of them is not connected to the transcoding proxy (Client 1), and the other is setup to be connected to the proxy (Client 2). Because we are focusing on transcoding images, we performed some of our case studies on a set of websites only containing images. We also performed some other case studies on general websites containing different elements such as images, text and tables.

6.2.1 Case Study 1: Resizing & Quality Factor (JPEG Images)

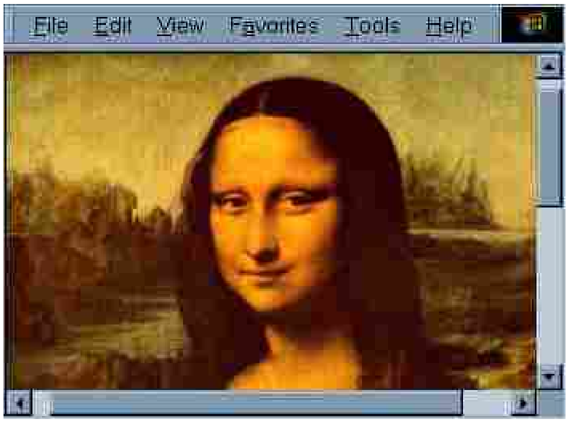
The purpose of the first case study is to test the proxy capability of performing "resizing" and "reducing image quality" for large JPEG images for different display devices. We selected MonaLisa as a test image. The image dimension is 605 x 790 pixel. The file size before transcoding is 98,938 bytes. We emulate 3 different devices with different capabilities and perform the test at 7 network bandwidth values. We recorded the characteristics of the received image with each bandwidth. The first test was performed with a 320x200 pixel display representing a PDA device. Referring to the 3 device classes mentioned in Chapter 4, this device is a low quality display device. We started the test at the maximum bandwidth, which is assumed to be 56 kb/s. Figure 6.1 shows a browser screen shot for the image at *Client 1* (not connected to the proxy). It also shows the transcoded image at each bandwidth at *Client 2* (connected to the proxy). Table 6.1 shows the change in the image dimensions, image file size and the quality reduction decided by the proxy for each change in the network bandwidth available. It also shows the download time at the client side.



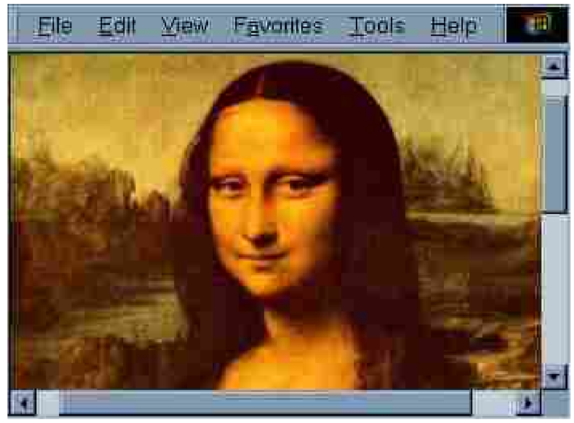
1. Image without transcoding



2. Transcoded Image, Network Bandwidth 56kb/s

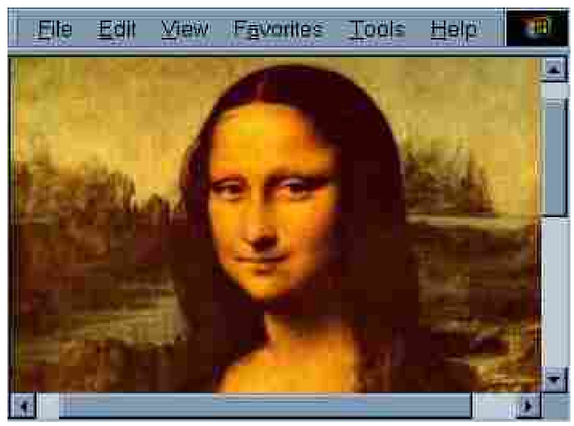


3. Transcoded Image, Network Bandwidth 50 kb/s

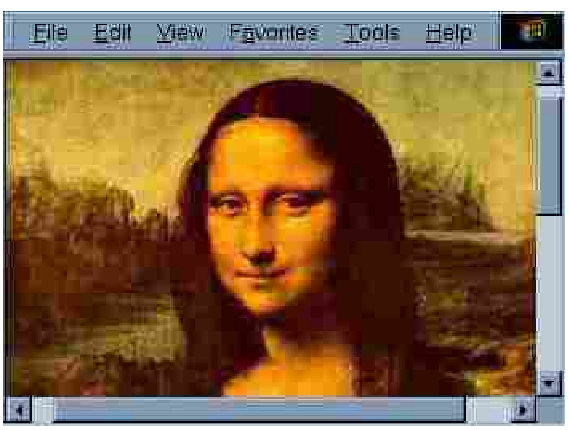


4. Transcoded Image, Network Bandwidth 40 kb/s

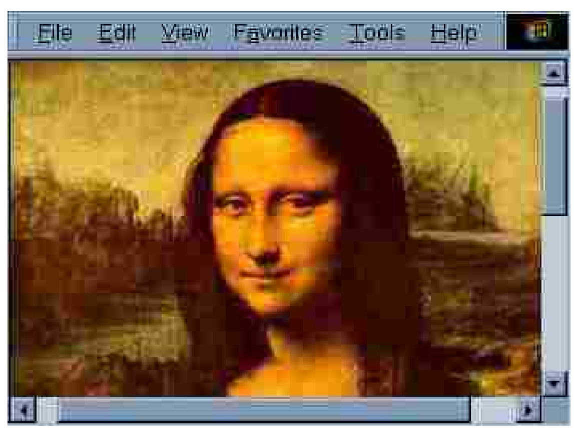
5. Transcoded Image, Network Bandwidth 30 kb/s



6. Transcoded Image, Network Bandwidth 20 kb/s



7. Transcoded Image, Network Bandwidth 10 kb/s



8. Transcoded Image, Network Bandwidth 5 kb/s

Figure 6-1: Screen Shots for Original and Transcoded JPEG Image at Different Network Bandwidths

Bandwidth (kbps)	Transcoded File Size (Bytes)	File Size Reduction (%)	Transcoded Image Width (Pixel)	Transcoded Image Height (Pixel)	Resizing Factor Calculated	Resizing Factor Applied	Quality Factor Calculated	Quality Factor Applied	Transcoding Time in seconds (Proxy)	Decoding Time in seconds (Client)	Download Time in seconds (without transcoding)	Download Time in seconds (after transcoding)
56	27096	72.61%	192	120	0.26	0.60	0.75	0.75	0.35	0.1424	2.5189	1.0784
50	22,285	77.48%	192	120	0.26	0.60	0.67	0.67	0.37	0.1367	2.7490	1.0527
40	17,562	82.25%	192	120	0.26	0.60	0.54	0.54	0.41	0.1357	3.2783	1.0796
30	13,988	85.86%	192	120	0.26	0.60	0.40	0.40	0.45	0.1398	4.1446	1.1527
20	10,711	89.17%	192	120	0.26	0.60	0.27	0.27	0.50	0.1502	5.8449	1.2837
10	8,380	91.53%	192	120	0.18	0.60	0.13	0.20	0.53	0.1540	10.8226	1.6208
5	8,380	91.53%	192	120	0.09	0.60	0.07	0.20	0.53	0.1540	20.7164	2.4588

Table 6.1: Output Image Characteristics and Operation Factors Calculated and Applied by the Proxy for Low Resolution Devices

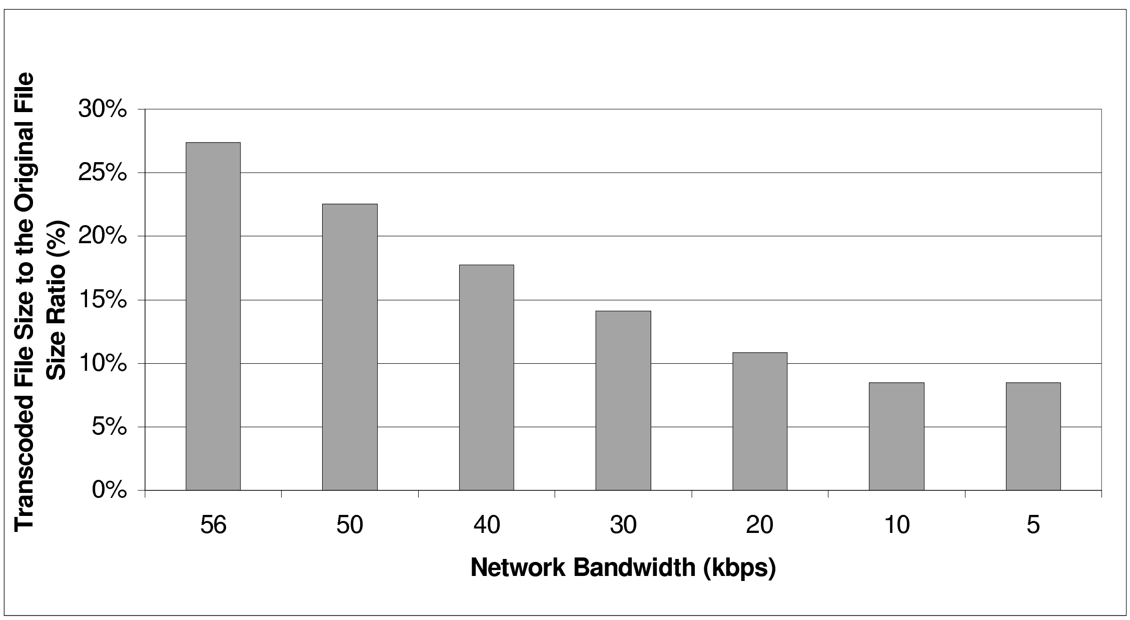


Figure 6-2: Transcoded File Size to the Original File Size Percentage at Different Bandwidths for Low Resolution Devices

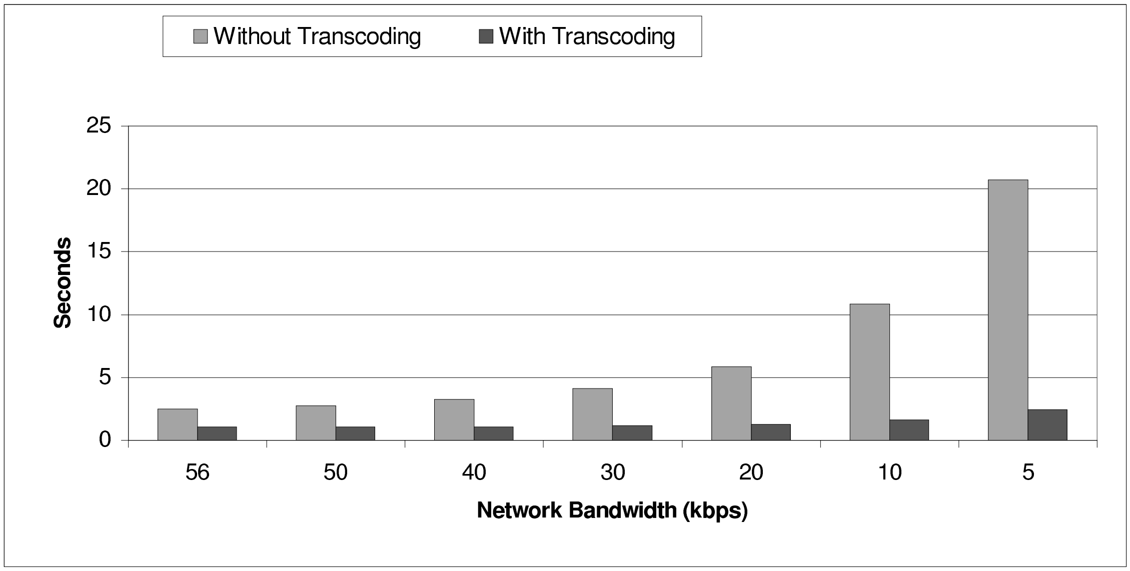


Figure 6-3: Download Time for Transcoded and Non-Transcoded Image for Low Resolution Devices

Figure 6.2 shows the percentage ratio between the transcoded image file size and the original image file size. For low network bandwidth, this ratio is very low, indicating a high reduction in the file size. Figure 6.3 illustrates the download time of the image at Client 1 (without proxy) and Client 2 (with proxy) at different network bandwidths.

Since the download time at the client browser is not accurately measurable, we calculated this time using the following formula:

Download time = transmission time from the web server to the proxy
+ transcoding time at the proxy server
+ transmission time from the proxy server to the client
+ decoding time at the client browser

Where:

- The transmission time from the web server to the proxy server = original image file size / network bandwidth at the web server side. Assuming the network bandwidth is very high at the WWW server side, this time is very short.
- The transcoding time at the proxy server is the time consumed by the CPU at the proxy to perform a transcoding operation. We measure this time by recording the time before we start transcoding and the time after completing the transcoding operation using a script file running under Unix.
- The transmission time from the proxy server to the client = the transcoded image file size / network bandwidth at the client side. Since the network bandwidth at the client side is low, this time will be a significant portion of the download time.
- The decoding time at the client browser is the time consumed by the browser to decode and render an image. This time is a function of the image quality and the file size. We consider the decoding time only with JPEG images. To calculate this time, we save a number of copies of the same image locally and open them together using the browser. We record the time just before and after the all copies were completely displayed. Then, we calculate the time for each copy of

the image. Using multiple copies of the same image reduces the error due to recording the time.

The same test was performed for a medium quality display (640 x 480 pixel) simulating a HHC device and a high quality display devices (1024 x 768) simulating a notebook.

Tables 6.2 and 6.3 show the output image characteristics, operation factors performed by the proxy and the download time in both cases. Figures 6.4 and 6.6 show the percentage ratio between the transcoded image file size and the original image file size. Figure 6.5 and 6.7 show the download time of the image at Client 1 (without proxy) and Client 2 (with proxy) at different network bandwidths.

Bandwidth (kbps)	Transcoded File Size (Bytes)	File Size Reduction (%)	Transcoded Image Width (Pixel)	Transcoded Image Height (Pixel)	Resizing Factor Calculated	Resizing Factor Applied	Quality Factor Calculated	Quality Factor Applied	Transcoding Time in seconds (Proxy)	Decoding Time in seconds (Client)	Download Time in seconds (without transcoding)	Download Time in seconds (after transcoding)
56	29460	27.39%	400	300	0.63	0.63	0.75	0.75	0.35	0.1488	2.5136	1.1217
50	24,352	22.52%	400	300	0.63	0.63	0.67	0.67	0.37	0.3500	2.7432	1.3014
40	19,171	17.75%	400	300	0.63	0.63	0.54	0.54	0.40	0.4200	3.2714	1.3972
30	11,495	14.14%	343	257	0.54	0.54	0.40	0.40	0.47	0.4400	4.1678	1.3930
20	6,084	10.83%	256	192	0.36	0.40	0.27	0.30	0.58	0.4700	5.9286	1.4559
10	6,084	8.47%	256	192	0.18	0.40	0.13	0.30	0.58	0.4900	10.8755	1.7801
5	6,084	8.47%	256	192	0.09	0.40	0.07	0.30	0.58	0.4900	20.7693	2.3885

Table 6.2: Output Image Characteristics and Operation Factors Calculated and Applied by the Proxy for Medium Resolution Devices

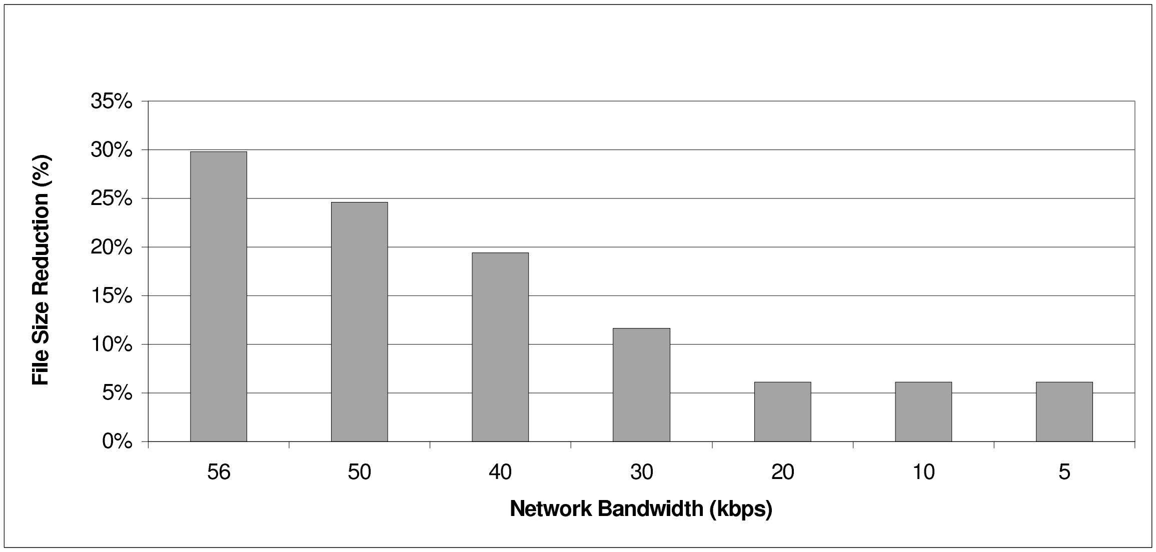


Figure 6-4: Transcoded File Size to Original File Size Ratio at Different Network Bandwidths for Medium Resolution Devices

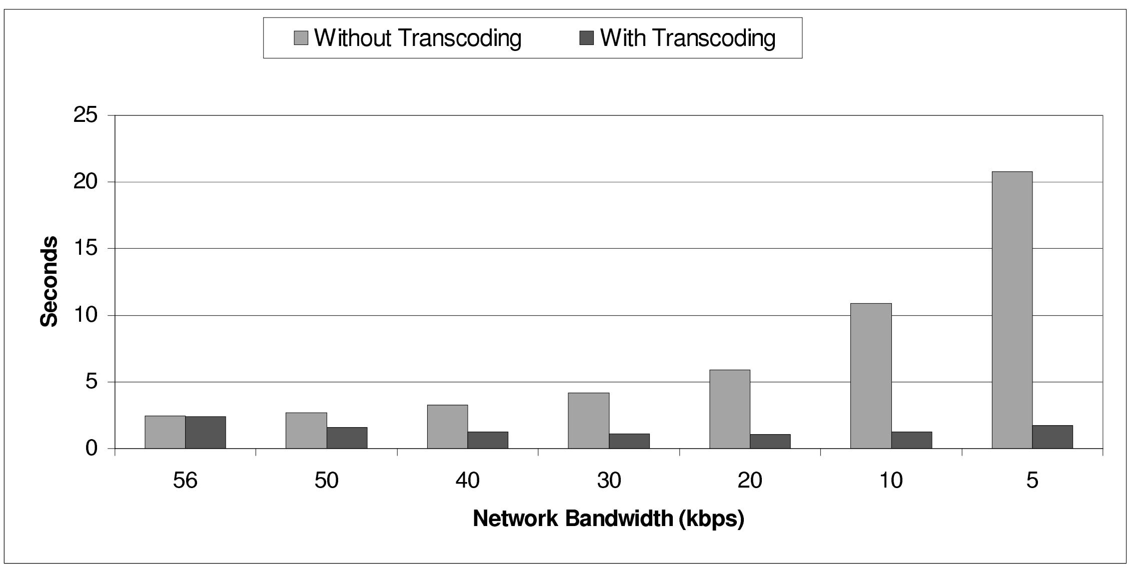


Figure 6-5: Download Time for Transcoded and Non-Transcoded Image for Medium Resolution Devices

Bandwidth (kbps)	Transcoded File Size (Bytes)	File Size Reduction (%)	Transcoded Image Width (Pixel)	Transcoded Image Height (Pixel)	Resizing Factor Calculated	Resizing Factor Applied	Quality Factor Calculated	Quality Factor Applied	Transcoding Time in seconds (Proxy)	Decoding Time in seconds (Client)	Download Time in seconds (without transcoding)	Download Time in seconds (after transcoding)
56	94,594	95.61%	1024	768	1.00	1.00	0.75	0.75	0.28	0.3229	2.4500	2.3953
50	47,574	48.09%	914	686	0.89	0.89	0.67	0.67	0.31	0.2124	2.6906	1.5758
40	24,018	24.28%	731	549	0.71	0.71	0.54	0.54	0.38	0.1599	3.2484	1.2353
30	11,495	11.62%	549	411	0.54	0.54	0.40	0.40	0.47	0.1274	4.1678	1.0804
20	6,083	6.15%	366	274	0.36	0.36	0.27	0.40	0.55	0.1005	5.8971	1.0549
10	4,658	4.71%	307	230	0.18	0.30	0.13	0.40	0.58	0.0933	10.8755	1.2409
5	4,658	4.71%	307	230	0.09	0.30	0.07	0.40	0.58	0.0933	20.7693	1.7067

Table 6.3: Output Image Characteristics and Operation Factors Calculated and Applied by the Proxy for High Resolution Devices

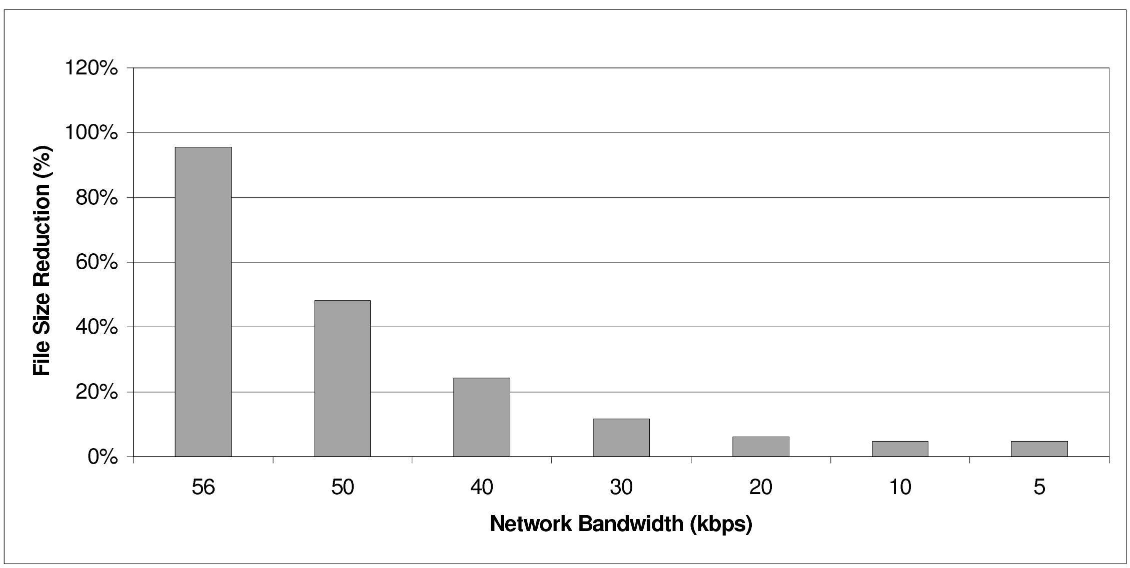


Figure 6-6: Transcoded File Size to Original File Size Ratio at Different Network Bandwidths for High Resolution Devices

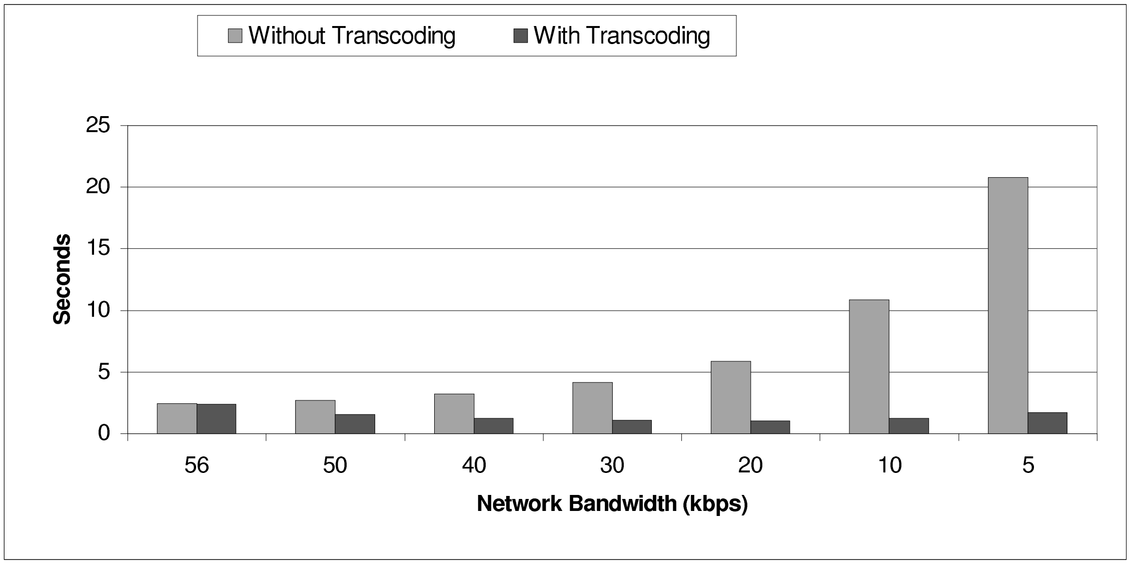


Figure 6-7: Download Time for Transcoded and Non-Transcoded Image for High Resolution Devices

Observations:

- The difference between the download time at Client 1 (with proxy) and Client 2 (without proxy) is small at the high bandwidths, when a file size reduction is not strongly needed. However, this difference is substantial at the low bandwidths, indicating high performance of the transcoding proxy at such conditions.

- Using the proxy, the download time varies in a small margins resulting in stable performance at different bandwidths. Referring to Figures 6.3, 6.5 and 6.7, we find that the download time ranges from 1 to 3 seconds. However, without using the proxy, the download time ranges from 2.5 to 20 seconds resulting in unstable conditions at different bandwidths.

- When the Geometry Resizing Factor calculated by the proxy is less than the recommended value, the proxy will apply the recommended value, which is 60%

(Table 6.1), 40% (Table 6.2) and 30% (Table 6.3) to keep the transcoded image acceptable by the user.

- The Quality Factor applied by the proxy decreases with each decrease of the network bandwidth. When the quality factor calculated by the proxy is less than 20% (Table 6.1), 30% (Table 6.2) and 40% (Table 6.3), the minimum acceptable value, the proxy will remain the quality as 20%, 30%, or 40% to ensure an acceptable transcoded image.
- High quality displays do better under low bandwidth since transcoded image is smaller.

6.2.2 Case Study 2: Applying Thumbnail

As discussed in Chapter 5, our proxy enables a user to enter his/her own preference such as preferred quality factor, thumbnail images, or freezing animated images. In a second case study, we assume that the user selected "Thumbnail" from the user preference panel. Table 6.4 shows the change in the image dimensions and the file size for a set of 5 images. It also shows the time consumed in seconds to download each image. Figure 6.8 shows the original images. Figure 6.9 shows a browser screen shot for this set of images without transcoding. As shown, the user has to scroll down and right to see all the images. Figure 6.10 shows a browser screen shot for these images after applying our proxy.

Device:	Resolution:	640 x 480 pixel
	Colors supported:	256
Bandwidth:	30 kbps	

		Original Image		Transcoded Image			
Image Name	Type	File Size (Bytes)	Dimension (Pixel)	File Size (Bytes)	Dimension (Pixels)	File Size Reduction%	Download Time (Seconds)
Image 1	GIF	4790	72 x 80	4059	72 x 90	15.26%	0.45
Image 2	GIF	38389	450 x 370	2914	80 x 65	92.41%	0.61
Image 3	JPEG	31892	495 x 329	1433	80 x 53	95.51%	0.57
Image 4	GIF	2590	67 x 52	2121	67 x 52	18.11%	0.42
Image 5	JPEG	92776	800 x 640	7130	80 x 64	92.31%	0.90

Table 6.4: Case Study 2: Thumbnail

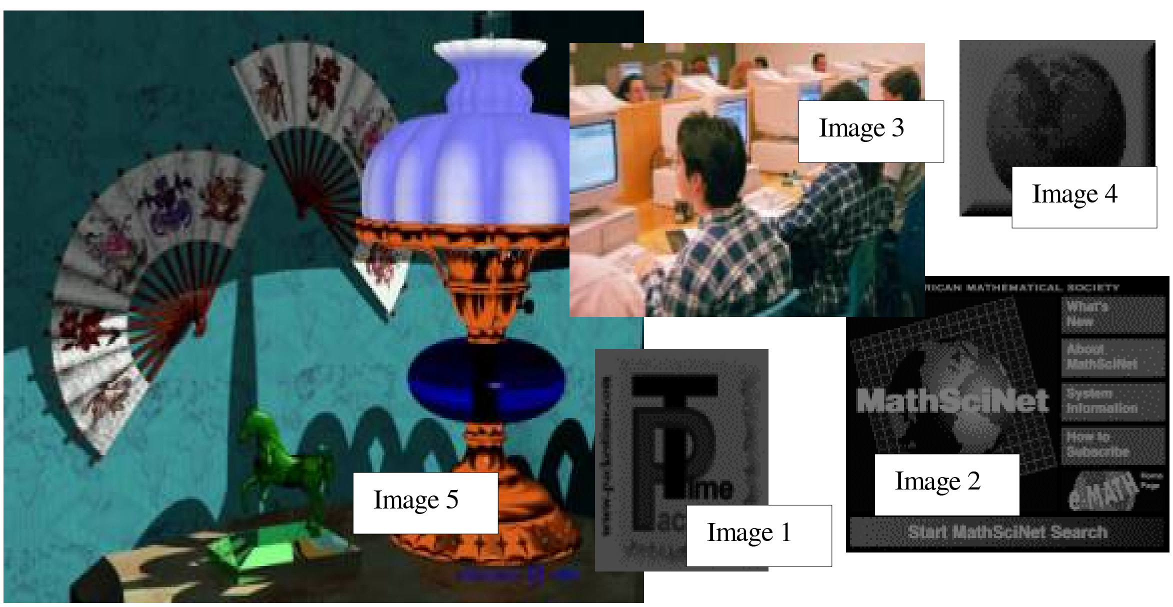
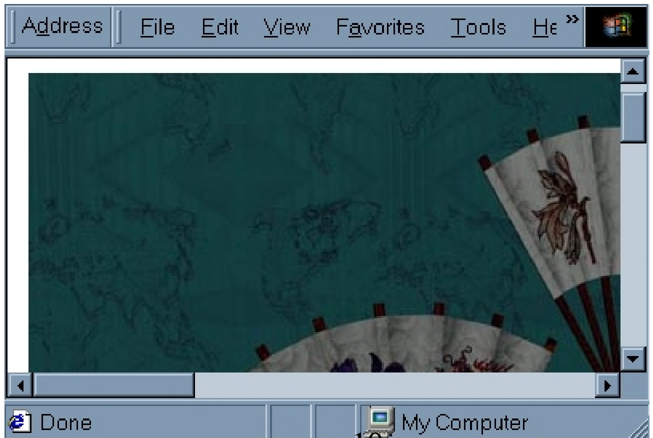


Figure 6-8: Images Before Performing a Thumbnail Transcoding



10.

Figure 6-9: Screen Shot for Displaying a Set of Images Without Transcoding

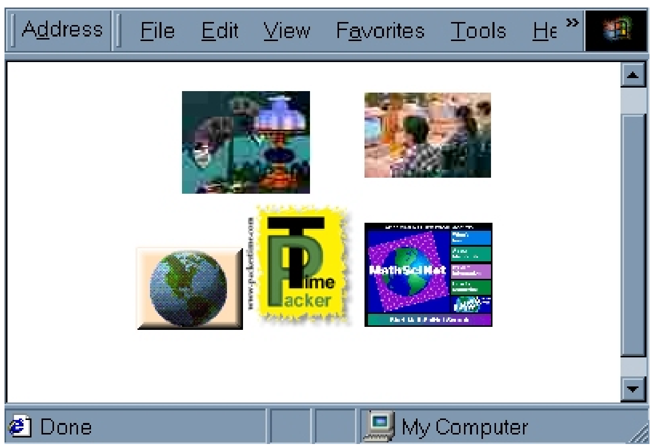


Figure 6-10: Screen Shot for Displaying a Set of Images after Thumbnail Transcoding

6.2.3 Case Study 3: Applying Freeze Animation

In a third case study, we assume that the user selects “Freeze Animated Images” from the user preference panel. The proxy will take the first frame of the animated image and perform resizing and color reduction on this frame. Table 6.5 shows the change in image dimensions and file size for a set of 3 animated GIF images. It also shows the time consumed to perform the operation for each image.

Device: Resolution: 640 x 480 pixel
Colors supported: 256
Bandwidth: 56 kbps.

Image Name	Original Image File Size (bytes)	Transcoded Image File Size (bytes)	File Size Reduction%	Download Time (Seconds)
Image 1	51,235	4,886	90.46%	0.81
Image 2	22,170	4,221	80.96%	0.62
Image 3	49,661	27,175	45.28%	0.70

Table 6.5: Case Study 3: Freeze Animation

6.2.4 Case study 4: GIF-JPEG Conversion

If a GIF image matches the conditions mentioned in Chapter 5, our proxy will perform "GIF to JPEG" conversion and then perform other operations to the converted image. We selected 3 GIF images that match these conditions. Table 6.6 shows the reduction in file size for each image due to such conversion.

Device: Resolution: 800 x 600 pixel
Colors supported: 256
Bandwidth: 30 kbps

Image Name	Original Image		Transcoded Image		Reduction due to Conversion (%)	All Operations Reduction (%)	Download Time (Seconds)
	File Size (Bytes)	Dimension (Pixels)	File Size (Bytes)	Dimension (Pixels)			
Image 1	95550	396 x 481	4335	212 x 257	78.44%	95.46%	0.92
Image 2	239398	487 x 760	26018	260 x 407	46.38%	89.13%	2.34
Image 3	130905	640 x 512	4933	342 x274	84.77%	96.23%	1.36

Table 6.6: Case Study 4: GIF-JPEG Conversion

6.2.5 Case Study 5: Complete Webpage

The purpose of the fifth case study is to test the general behavior of the transcoding proxy when downloading a normal page containing images, html, tables and text.

Although we are focusing on images, we also want to check how the transcoded web page looks when it contains different elements that the proxy dose not deal with, such as HTML text, and tables.

We requested the Computer Science website: www.scs.carleton.ca at Client 1 (not connected to the proxy) and Client 2 (connected to the proxy). We adjusted the resolution and number of colors so that the clients simulate a Windows CE device with 640 x 240 pixel display. We recorded the characteristics for each component within this page and captured a screen shot from each computer. We also calculated the download time for every component. Figure 6.11 and Figure 6.12 show a screen shot for the page at client 1 (not connected to the proxy) and client 2 (connected to the proxy). Table 6.7 illustrates the recorded characteristics for these components and the calculated download time in both cases.

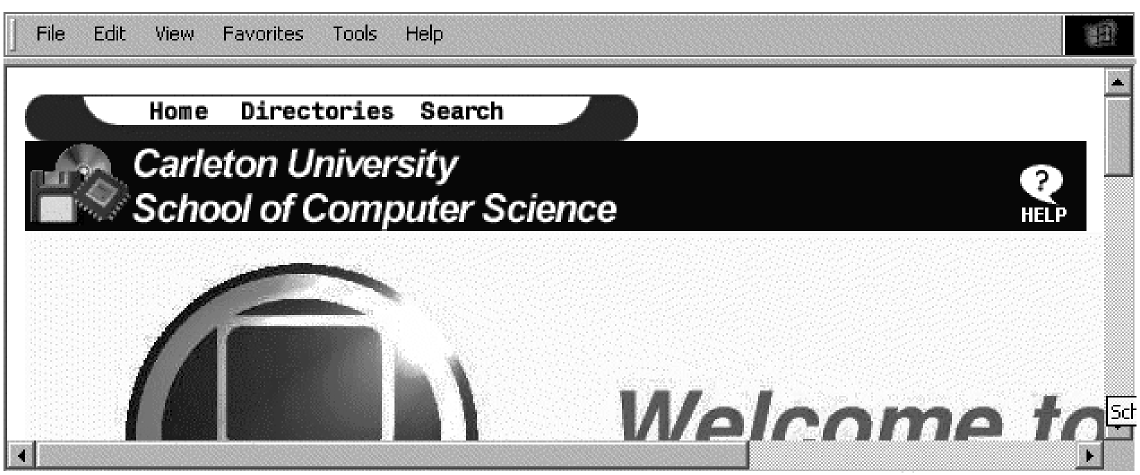


Figure 6-11: A Screen Shot of the Computer Science Dept Website Without Installing the Proxy

Figure 6-12: A Screen Shot of the Computer Science Website Using Our Proxy

Device:	Resolution:	640 x 240 pixel
	Colors supported: 256	
Bandwidth:	6.9 kbps	

File Name	File Size (bytes)	Transcoded File Size (bytes)	File Size Reduction (%)	Resizing Factor Applied	Quality Factor Applied	Colors Reduction Factor Calculated	Transcoding Time in seconds (Proxy)	Decoding Time in seconds (Client)	Download Time in seconds (without proxy)	Download Time in seconds (after proxy)
splash.jpg	35563	7416	79.15%	0.60	0.20	N/A	0.21	0.31	5.6996	1.6203
people_on.gif	1,127	152	86.51%	0.60	N/A	0.25	0.09	0.00	0.5545	0.1132
header.gif	5,440	1,027	81.12%	0.60	N/A	0.25	0.12	0.00	1.2138	0.2743
courses_on.gif	1,428	1,104	22.69%	0.60	N/A	0.25	0.09	0.00	0.5984	0.2514
research_on.gif	1,421	1,098	22.73%	0.60	N/A	0.25	0.09	0.00	0.5974	0.2506
button_bar.gif	1,299	317	75.60%	0.60	N/A	0.25	0.09	0.00	0.5796	0.1372
publications_on.gif	1,261	169	86.60%	0.60	N/A	0.25	0.08	0.00	0.5640	0.1058
*.html	7599	7599	0.00%	N/A	N/A	N/A	N/A	0.00	1.1089	1.1089
	63,584	21,733	65.82%				1.32		10.5987	4.8433

Table 6.7: Output Image Characteristics and Operation Factors Performed

As shown in Figure 6.11, this page is designed for at least 800 x 600 pixel resolution. Using the browser of Windows CE device will show only the top left corner of this page. The user has to scroll right and down to see the other part of the page such as the right side menu. Using the transcoding proxy (Figure 6.12), all images were resized to fit the device display. The total bytes of the downloaded files were reduced to 34% of the original page. The download time is reduced to 46% of the time it takes to download the original page over a 6.9 kbps link.

The Computer Science website is a good example to apply our transcoding proxy to without affecting the page layout or loosing any functionality of the web. The only drawback here is the loss of the transparency property of the blue banner at the top as shown in Figure 6.12. This problem is considered as one possible topic of future work.

We applied our proxy on some other Web pages and recorded the transcoded file characteristics with respect to the original characteristics for each component. Tables 6.8, 6.9 and 6.10 summarize the results. For more details see Appendix G.

File Name	Original File Size	Transcoded File Size in bytes (Palm 320x200 pixel)	% File Size Reduction (Palm 320x200 pixel)	Transcoded File Size in bytes (CE Device 640x240 pixel)	% File Size Reduction (CE Device 640x240 pixel)	Transcoded File Size in bytes (Laptop 800x600 pixel)	% File Size Reduction (Laptop 800x600 pixel)
Jean1.gif	27,153	1,210	95.54%	2,426	91.07%	9,755	64.07%
Jean2.gif	113,538	3,005	97.35%	6,782	94.03%	37,617	66.87%
kunz.gif	6,031	482	92.01%	935	84.50%	2,543	57.83%
kunz_new.gif	23,154	300	98.70%	839	96.38%	2,664	88.49%
pictures.html	1,458	1,458	0.00%	1,458	0.00%	1,458	0.00%
pictures.html	1,458	1,458	0.00%	1,458	0.00%	1,458	0.00%
soccer1.gif	65,851	2,102	96.81%	4,715	92.84%	20,243	69.26%
soccer2.gif	50,360	3,710	92.63%	9,244	81.64%	37,717	25.11%
Total	**289,003**	**13,725**	**95.25%**	**27,857**	**90.36%**	**113,455**	**60.74%**

Table 6.8: Transcoded Image Characteristics for Kunz Website
http://kunz-pc.sce.carleton.ca/pictures.html

File Name	Original File Size	Transcoded file Size in bytes (Palm 320x200 pixel)	% File Size Reduction (Palm 320x200 pixel)	Transcoded File Size in bytes (CE Device 640x240 pixel)	% File Size Reduction (CE Device 640x240 pixel)	Transcoded File Size in bytes (Laptop 800x600 pixel)	% File Size Reduction (Laptop 800x600 pixel)
b-all.gif	6,298	206	96.73%	762	87.90%	3,138	50.17%
b-order.gif	3,536	151	95.73%	548	84.50%	1,962	44.51%
dr-alert.gif	8,704	375	95.69%	1,100	87.36%	4,601	47.14%
dr-harsp.gif	11,140	366	96.71%	1,234	88.92%	5,201	53.31%
dr-heron.gif	5,807	312	94.63%	848	85.40%	2,949	49.22%
dr-wduck.gif	11,197	377	96.63%	1,007	91.01%	4,318	61.44%
ospr-txt.gif	9,649	237	97.54%	977	89.87%	3,715	61.50%
p1.gif	1,594	71	95.55%	388	75.66%	844	47.05%
w-he-ovl.gif	2,632	159	93.96%	479	81.80%	1,283	51.25%
wild.html	4,724	4,724	0.00%	4,724	0.00%	4,724	0.00%
wild1.html	5,380	5,380	0.00%	5,380	0.00%	5,380	0.00%
wild-tx.gif	5,335	204	96.18%	828	84.48%	2,920	45.27%
Total	**75,996**	**12,562**	**83.47%**	**18,275**	**75.95%**	**41,035**	**46.00%**

Table 6.9: Transcoded Image Characteristics (http://www.ospreynest.com/wild.html)

File Name	Original File Size	Transcoded file Size in bytes (Palm 320x200 pixel)	% File Size Reduction (Palm 320x200 pixel)	Transcoded File Size in bytes (CE Device 640x240 pixel)	% File Size Reduction (CE Device 640x240 pixel)	Transcoded File Size in bytes (Laptop 800x600 pixel)	% File Size Reduction (Laptop 800x600 pixel)
back426.jpg	7,957	7,957	0.00%	7,957	0.00%	7,957	0.00%
Bubbite.gif	8,216	314	96.18%	1,019	87.60%	4,277	47.94%
Cessiywn.gif	31,516	1,226	96.11%	2,541	91.94%	13,638	56.73%
dec.gif	30,804	741	97.59%	5,049	83.61%	20,864	32.27%
dedegirl.gif	87,742	1,582	98.20%	19,022	78.32%	34,409	60.78%
delta.gif	88,832	1,217	98.63%	2,680	96.98%	16,774	81.12%
einstein.gif	21,998	979	95.55%	2,013	90.85%	11,735	46.65%
FlyingCat1.gif	21,981	943	95.71%	1,677	92.37%	9,060	58.78%
fun.gif	10,658	1,679	84.25%	2,217	79.20%	7,225	32.21%
KITT5.gif	18,222	593	96.75%	1,632	91.04%	8,131	55.38%
ktkr.gif	3,938	187	95.25%	870	77.91%	2,578	34.54%
lance2.gif	19,343	831	95.70%	1,940	89.97%	9,160	52.64%
outside.gif	18,365	581	96.84%	1,499	91.84%	7,913	56.91%
pawbar.gif	1,903	906	52.39%	955	49.82%	1,510	20.65%
purrfect.gif	58,198	500	99.14%	1,148	98.03%	5,701	90.20%
shadywaving.gif	11,976	646	94.61%	1,466	87.76%	7,222	39.70%
sorrell.gif	22,386	701	96.87%	1,534	93.15%	9,388	58.06%
Spiceet.gif	34,190	348	98.98%	854	97.50%	3,447	89.92%
yoda.gif	32,004	1,160	96.38%	2,389	92.54%	12,425	61.18%
Total	**530,229**	**23,091**	**95.65%**	**58,462**	**88.97%**	**193,414**	**63.52%**

Table 6.10: Transcoded Image Characteristics (http://lilhartz.com/PHOTO.htm)

Referring to tables 6.8, 6.9, and 6.10, applying our proxy on different devices resulted in different reductions in file size. The low the resolution of the device, the high the reduction. For Palm (320x200 pixel), file size reduction ranges from 83% to 95%. For Windows CE device (640x240 pixel), this value ranges from 75% to 90%. For Laptop (540x480 pixel), the reduction ranges from 45% to 63%.

6.2.6 Case Study 6: Dependent Images

When a webpage contains images formed by combining multiple sub-images, the proxy performs the transcoding on each one individually. Although it applies the same geometry resizing on all images, the transcoded images will not be combined to a smaller image. This problem is due to applying the transcoding only on images, not on all page elements such as the HTML table containing these images. Figures 6.13 and 6.14 show this problem when we download one of the WindRiver website pages (http://www.wrs.com/windword/) before and after transcoding.

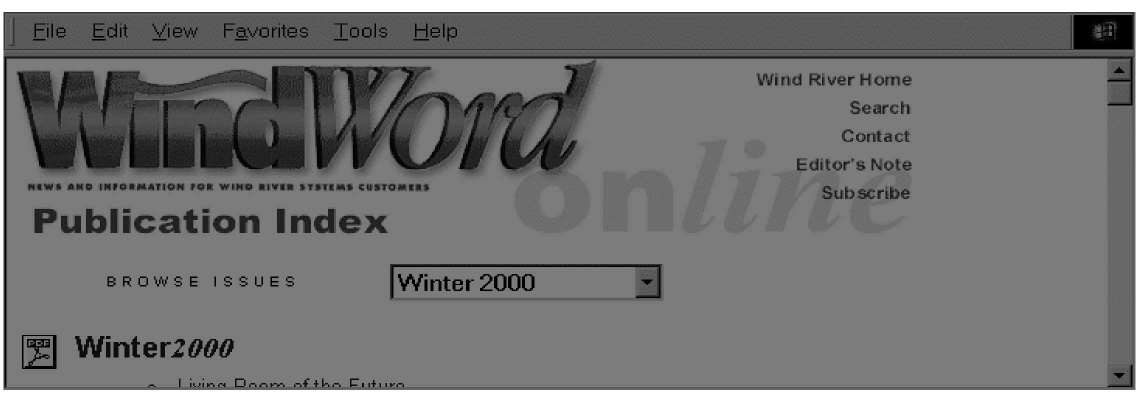


Figure 6-13: A Screen Shot of a Site Containing Dependent Images Without Installing the Proxy

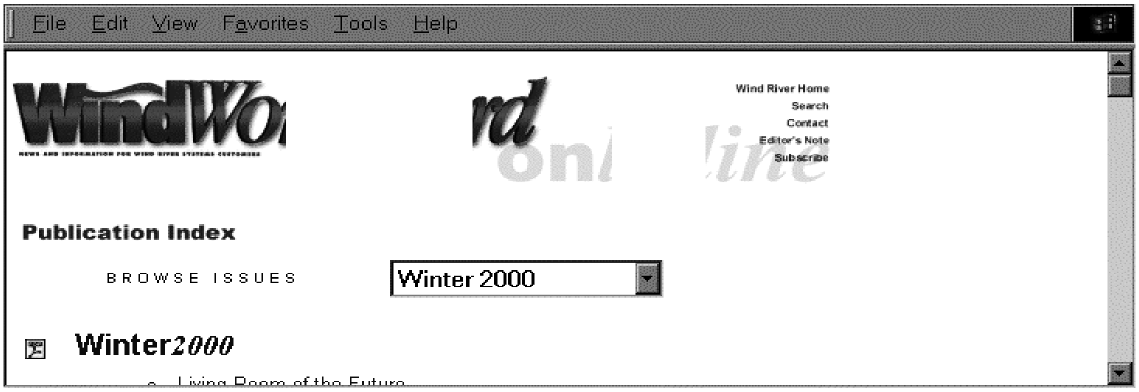


Figure 6-14: A Screen Shot of a Site Containing Dependent Images Using our Proxy

In general, web pages are carefully designed, much attention is being given to layout, including how elements relate to each other. A more complete transcoding proxy will have to consider such interdependencies, not just single elements.

6.3 Discussion

We believe that our transcoding proxy is very flexible in performing transcoding operations compared to most of the high-level filters used for the Internet. Although our proxy applies the most common compression methods such as resizing, JPEG quality reduction and reducing the number of colors for GIF images, it does not use fixed parameters when applying such compressions. It dynamically performs the most suitable parameters to match the current device and network bandwidth available. The user can also customize the proxy behavior by changing the default values of the user preferences such as freezing the animated GIF images, using thumbnail, default JPEG quality factor and removing background images.

From the user point of view, we experimentally found that the quality, size and download time of the transcoded image are reasonable. Our transcoding proxy takes into account these constraints every time it performs an operation to the image by verifying that the reduction factors applied are still within our experimentally determined user satisfaction range. Although the transcoded image file size could be slightly bigger in some cases compared with other proxies, we guarantee that the image received is acceptable to the user.

Based on many experiments on our transcoding proxy, we found that the average file size reduction ranges from 15% to 25% of the original file size for a typical web page, resulting in a significant reduction in download time. However, we have to take into account the other factors affect the download time and consequently the performance of the proxy such as the characteristics of the user device and the computer that runs the proxy server, such as the processor speed and memory available.

As our proxy transcodes only images, it performs good with websites that contain independent images. However, if the page contains combined images, it will break down all the images, resulting in an unacceptable rendering of page. As a future work, we will study all the web components and perform suitable transcoding operations to each element while keeping the layout of the original page. We have already some preliminary ideas we will discuss in the future work section.

Some additional case studies are available in Appendix G as well as at: http://chat.carleton.ca/~smohame2/demo/MainScreen.html

Chapter 7: Conclusion and Future Work

The ultimate goal of our work is to improve the performance of accessing the WWW over low bandwidth networks and using devices have different capabilities, without affecting the important information. Based on some basic statistics measuring data traffic at the client side, we found and verified that GIF and JPEG images represent the bulk of data on the net. Therefore, we currently focus on JPEG and GIF images.

We have studied a sample of typical images available on the Internet with respect to their characteristics after applying different optimization operations. Geometry resizing, JPEG quality reduction, GIF colors reduction and GIF to JPEG conversion are suitable candidates for image transcoding.

We introduced a relationship between the device display characteristics and the user image quality satisfaction level. We found that the lowest image quality that can be accepted by the user depends on the device display resolution and the color depth supported by this device. We characterized the minimum transcoding operation factor applied on an image with respect to the device displays. Quality factor reduction for JPEG images and colors reduction factor for GIF images range from 40% for high quality displays to 20% for low quality displays. However, geometry resizing factor ranges from 30% for high quality displays to 60% for low quality displays.

We presented a proxy server for transcoding images in order to adapt, on the fly, these images to the current network bandwidth available and client device display capabilities taking into account the user satisfaction with the transcoded image.

We evaluated our proxy server by applying some case studies for different image categories and complete websites. We recorded the characteristics of the transcoded images/contents and the download time. We found that the download time varies in small margins resulting in stable performance at the different network bandwidths.

One of the potential areas of our future work is studying all the web page components and perform suitable transcoding operations to each element while keeping the layout of the original page. So far, our proxy deals only with GIF and JPEG images. However, web pages contain other elements such as text, tables, Java Applets and Java script. As we studied the most effective and suitable transcoding operations for images, we will study the most effective operations to be applied to these elements. One of our current ideas is to capture the whole web page as an image, or more than one image, adapt this image based on the device capabilities and the network bandwidth as we discussed, and eventually, mapping all the hyper links to the transcoded image.

Another requirement from our proxy server is to detect on the fly the network bandwidth available. Right now, our proxy uses this parameter and performs the calculations based on it. We need to measure the current bandwidth in real time. Some software can do this job by installing a program at the client side and measure roundtrip times for messages of different rises.

We may also add more features and flexibility to the user preferences. For example, a user can interactively select the suitable parameters for the image when he/she downloads the image by using an attached dropdown menu.

References

[1] R. Han, P. Bhagwat, R. Lamaire, T. Mummert, V. Perret, and J. Rubas. *Dynamic adaptation in an image transcoding proxy for mobile web browsing.* IEEE Personal Communications, page 8-17, December 1998.

[2] J. Smith, R. Mohan and C. Li. *Content-based Transcoding of Images in the Internet.* Proc. Int'l. Conf. Image Processing, (ICIP-98), Oct. 1998, pages 7-11.

[3] Fox, S. D. Gribble, Y. Chawathe, and E. A. Brewer. *Adapting to network and client variation using active proxies: lessons and perspectives.* IEEE Personal Communication, Vol. 5, No. 4, page. 10-19, August 1998.

[4] H. K. Choi, and J. O. Limb, *A Behavioral Model of Web Traffic.* Proceedings of the Seventh Annual International Conference on Network Protocols, Institute of Electrical and Electronics Engineers, 1998.

[5] J. E. Pitkow and M. M. Recker. *A simple yet robust caching algorithm based on dynamic access patterns*. In Proceedings of 2nd International WWW Conference, pages 1039-1046, October 1994.

[6] S. Chandra, A. Gehani, C. S Ellis, and A. Vahdat. *Transcoding characteristics of web images.* Multimedia Computing and Networking 2001 conference, Procedure 4312 - Page 13, January 23, 2001.

[7] C. Cunha, A. Bestavros, and M. Crovella. *Characteristics of www client-based traces*. Technical Report TR-95-010, Boston University Department of Computer Science, April 1995.

[8] Abdulla et al, G. Abdulla, E. Fox and M. Abrams. *Shared user behavior on the World Wide Web*, WebNet97, Toronto, October 1997.

[9] M. El Shentenawy, A. Gaddah, Q. Guo, T. Kunz and R. Hafez, Image transcoding for proxy Internet wireless access, Proceedings of the 12th International Conference on Wireless Communications, Calgary, Alberta, Canada, page 542-552, July 2000.

[10] A. Gaddah, M. El Shentenawy, T. Kunz and R. Hafez, *Image Transcoding Proxy for Mobile Internet Access,* (In progress), The ninth ACM Multimedia Conference will be held in Ottawa, Ontario, Oct., 2001.

[11] http://www.imagemagick.org, ImageMagick: is a robust collection of tools and libraries to read, write, and manipulate an image in many image formats.

[12] M. D. Schroeder, *JPEG Compression Algorithm and Associated Data Structures.* http://www.cs.und.edu/~mschroed/jpeg.html

[13] http://www.faqs.org/faqs/compression-faq, Compression Frequently Asked Question.

[14] http://www.cywarp.com/FAQ_TimeCapsules.asp, Digital Photography FAQ.

[15] M. Nelson, *LZW Data Compression*, Dr. Dobb's Journal October, 1989.

[16] H. Bharadvaj, A. Joshi and S. Auephanwiriyakul. *An Active Transcoding Proxy to Support Mobile Web Access.* Proceedings of the 17th IEEE Symposium on Reliable Distributed Systems, Institute of Electrical and Electronics Engineers, Inc., page 118-123, 1998.

[17] W.Y. Ma, I. Bedner, G. Chang, A. Kuchinsky, and H. Zhang. *A Framework for Adaptive Content Delivery in Heterogeneous Network Environments.* Hewlett-Packard Laboratories.

[18] S. Acharya and B. Smith. *MiddleMan: A Video Caching Proxy Server.* The 10th International Workshop on Network and Operating System Support for Digital Audio and Video, Chapel Hill, North Carolina, USA, June 26-28 2000.

[19] M. Hori, G. Kondoh, K. Ono, S. Hirose, S. K. Singhal. *Annotation-based Web content transcoding.* WWW9 / Computer Networks 33(1-6): 197-211 (2000)

[20] C. H. Chi, J. Deng, and Y. H. Lim. *Compression Proxy Server: Design and Implementation.* National University of Singapore, 2nd USENIX Symposium on Internet Technologies and Systems (USENIX), Boulder, Colorado, USA, October 11-14, 1999.

[21] ProxiNet, http://www.proxinet.com.

[22] Fox et al., The TranSend Service, *TranSend service homepage*, Available at transend.cs.Berkeley.edu, 1998.

[23] Gozilla: http://www.gozilla.com, Software for downloading and categorizing predefined file types from the Internet.

[24] M. Liljeberg, H. Helin, M. Kojo, and K. Raatikainen. *Enhanced Services for World Wide Web in Mobile WAN Environmen.* Univ. of Helsinki CS Tech Report C-1996-28; in Proc. IMAGE'COM 96, , Bordeaux, France, May 20-24, 1996.

[25] Rabbit HTTP proxy: http://www.nada.kth.se/projects/prup98/web proxy.

[26] Apache: http://www.dsdelft.nl/~apache

[27] DeleGate: http://wall.etl.go.jp/delegate

[28] A. Gaddah, *WWW Browsing Using a Transcoding Proxy*, Thesis (M.C.S), Carleton University, dec. 2000.

[29] R. E. Baxter and E. Davis, *The Penguin Dictionary of Economics, © Graham Bannock 1998.*

Appendix A: Saving Identical Images with Different Image Processing Software

http://chat.carleton.ca/~smohame2/TestImages.html

Test (1): Saving the same image using different software:

File Size: 16246 byte	12705 byte	13017 byte	7126 byte
19997 byte	14240 byte	14235 byte	8517 byte
9545 byte	8950 byte	6673 byte	4800 byte
9407 byte	7781 byte	5869 byte	4481 byte

Test (2): Designing the same image using different software:

7781 byte	7531 byte	9024 byte

Appendix B: Accessing and Downloading Cache Files

The main objective of the **Accessing and Downloading Cache Files** survey is to measure the maximum cache limit ratio at the client side when accessing the Internet. This can be obtained by determining the number and percentage of files that are accessed more than one time by the user. We assume that the user accesses the main server when he/she downloads the file for the first time. For the second or third time he/she accesses the same file from the cache storage, the proxy server in our case.

As a starting point, we sent an email to 100 people with different interests asking them to forward their local caches every 2 days after clearing their previous ones. 12 of them continued to forward their caches for over 20 days. Although they are not a large number of users for obtaining accurate results, it gives us an indication of the advisable cache performance. Table A.1 shows a sample of the results for one user over one testing period.

File Format	Total Bytes Repeated	Number of Repeated Files
ASP	115,499	6
CSS	19,446	7
DB	180,224	1
GIF	789,612	329
HTM	1,130,019	10
JAR	50,224	1
JPG	13,964	3
JS	10,770	4
All File Formats	**2,309,758**	**361**

Table A.1: Sample of Results

Referring to the overall results, we deduced that GIF images represent the largest percentage of files that are accessed by the user more than once.

Over 12 days, our survey resulted in the following results:

- GIF files: 2,228 (2,927,685 bytes) repeatedly accessed files among 11,247 files (12,178,547 Bytes).
- HTML files: 89 (704,837 bytes) repeatedly accessed files among 425 files (3,222,642 Bytes).
- JPEG files: 45 (2,091,792 bytes) repeatedly accessed files among 427 files.
- TXT files: 28 (1,122,825 bytes) repeatedly accessed files.
- JS files: 6 (1,225,581 bytes) repeatedly accessed files.
- The rest of the accessed files contribute a negligible amount.

Tables (A.2), (A.3) and (A.4) summarize these results for the 3 file formats must frequently accessed from the cache.

From	To	Number of Files	Total Bytes	Number of Repeated Files	% Files Repeated	Total Bytes Repeated	% Bytes Repeated
06/18/00	06/19/00	290	687,266	0	0.00%	0	0.00%
06/20/00	06/21/00	401	1,174,496	52	12.97%	154,316	13.14%
06/22/00	06/23/00	329	789,612	92	27.96%	111,163	14.08%
06/24/00	06/25/00	217	476,799	76	35.02%	76,765	16.10%
06/26/00	06/27/00	1,247	3,767,456	130	10.43%	546,461	14.50%
06/28/00	06/29/00	521	832,337	211	40.50%	286,431	34.41%
06/30/00	07/01/00	989	1,648,757	431	43.58%	786,541	47.71%
07/02/00	07/03/00	4,443	2,549,780	974	21.92%	865,868	33.96%
07/04/00	07/05/00	2,498	181,012	211	8.45%	80,176	44.29%
07/06/00	07/07/00	312	71,032	51	16.35%	19,964	28.11%
	Total	**11,247**	**12,178,547**	**2,228**	**19.81%**	**2,927,685**	**24.04%**

Table A.2: Accessing GIF Files

From	To	Number of Files	Total Bytes	Number of Repeated Files	% Files Repeated	Total Bytes Repeated	% Bytes Repeated
06/18/00	06/19/00	18	27,399				
06/20/00	06/21/00	46	198,987	4	8.70%	19,678	9.89%
06/22/00	06/23/00	71	630,019	11	15.49%	95,897	15.22%
06/24/00	06/25/00	25	87,264	5	20.00%	17,521	20.08%
06/26/00	06/27/00	81	489,754	18	22.22%	110,273	22.52%
06/28/00	06/29/00	19	397,421	4	21.05%	93,451	23.51%
06/30/00	07/01/00	34	498,586	10	29.41%	134,561	26.99%
07/02/00	07/03/00	131	893,212	37	28.24%	233,456	26.14%
Total		**425**	**3,222,642**	**89**	**20.94%**	**704,837**	**21.87%**

Table A.3: Accessing HTML Files

From	To	Number of Files	Total Bytes	Number of Repeated Files	% Files Repeated	Total Bytes Repeated	% Bytes Repeated
6/18/00	06/19/00	18	229,541				
6/20/00	06/21/00	33	1,060,929	4	12.12%	49,541	4.67%
6/22/00	06/23/00	93	9,913,964	8	8.60%	489,020	4.93%
6/24/00	06/25/00	16	715,405	1	6.25%	52,343	7.32%
6/26/00	06/27/00	127	8,993,210	12	9.45%	709,821	7.89%
6/28/00	06/29/00	98	12,254,491	14	14.29%	609,765	4.98%
6/30/00	07/01/00	12	1,485,932	2	16.67%	97,436	6.56%
7/02/00	07/03/00	30	2,948,703	4	13.33%	83,865	2.84%
Total		**427**	**37,602,175**	**45**	**10.54%**	**2,091,791**	**5.56%**

Table A.4: Accessing JPEG Files

These results seem reasonable because of the fact that GIF images are usually used for designing banners, lines, buttons and most of the decorative entities on a web page. Every time a user accesses the same website, he/she actually downloads the same buttons and decorative items unless there are some changes in the web design. For the other content formats such as HTML, JPEG, or JS, they might be changed or recreated periodically. These results could be very helpful when thinking about caching strategies, and can also be used to design representative user scenarios.

Appendix C: Survey

http://www.chat.carleton.ca/~smohame2/survay/Form01.htm

Name (Optional)
Email Address (Optional)

Monitor settings

Color Palette

True color (32 bit)	High color (16 bit)	256 colors	16 colors	Other (Please specify):

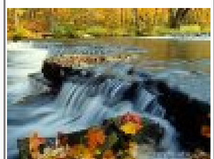
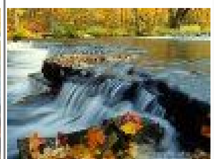
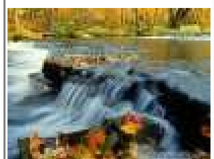
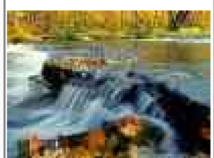
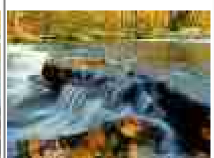
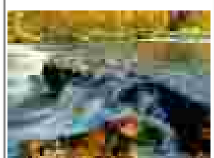
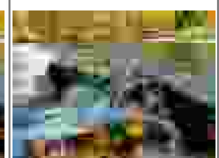
Desktop area

1280 x 1024	1024 x 768	800 x 600	640 x 480	Other (Please specify):

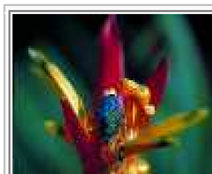
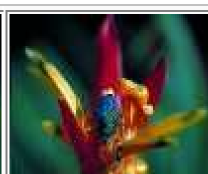
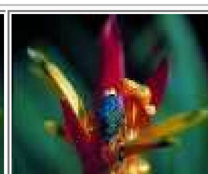
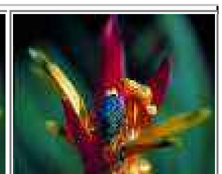
Please, click each image to see it in full size.
Click back from your browser and check if you accept it as a quality and loading time or not.
Thank you.

Trace #1: Quality Reduction Test

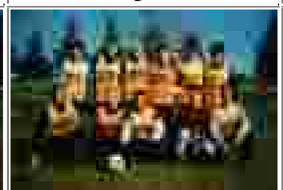
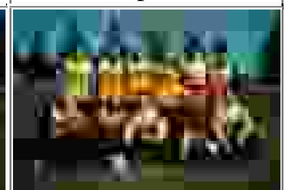
Quality Test #1

 Original Acceptable Not Acceptable	 80 % Acceptable Not acceptable	 60% Acceptable Not acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

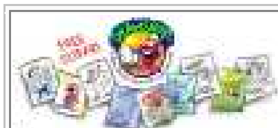
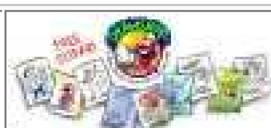
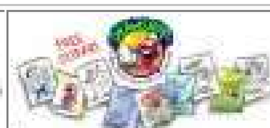
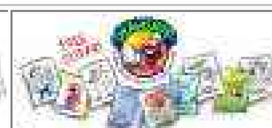
Quality Test #2

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

Quality Test #3

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

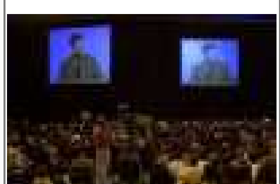
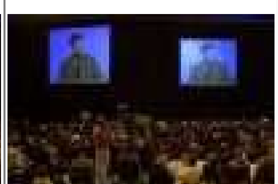
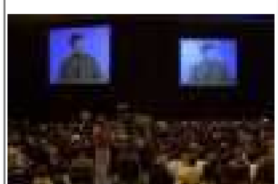
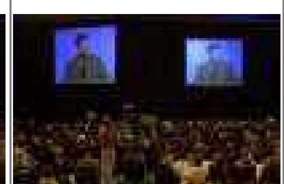
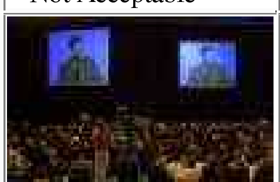
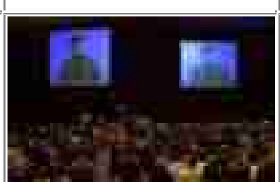
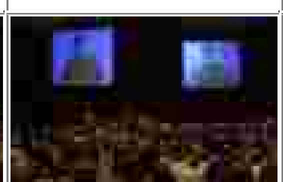
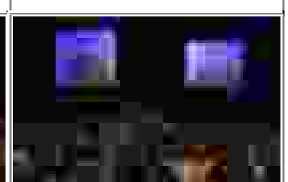
Quality Test #4

 Original Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

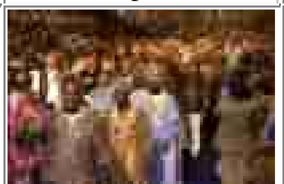
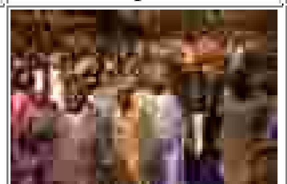
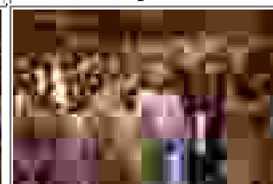
Quality Test #5

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

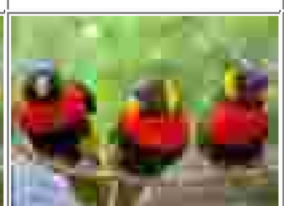
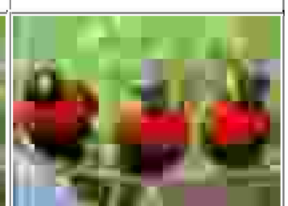
Quality Test #6

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

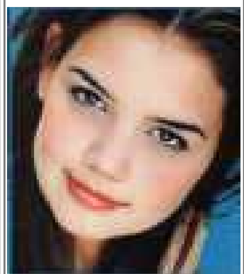
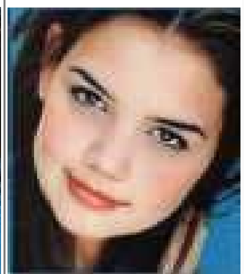
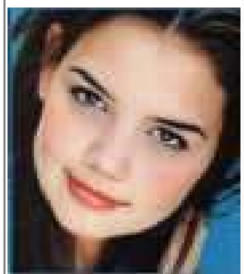
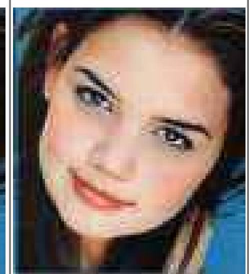
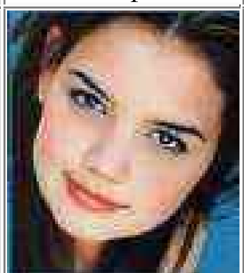
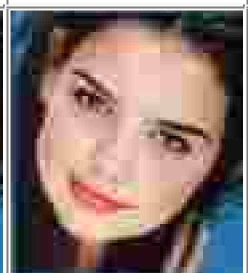
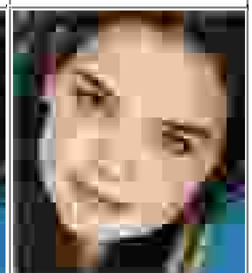
Quality Test #7

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

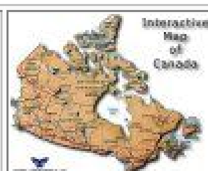
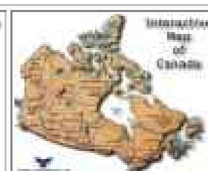
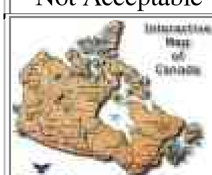
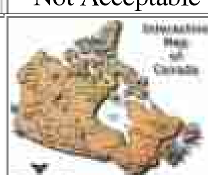
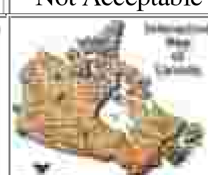
Quality **Test #8**

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

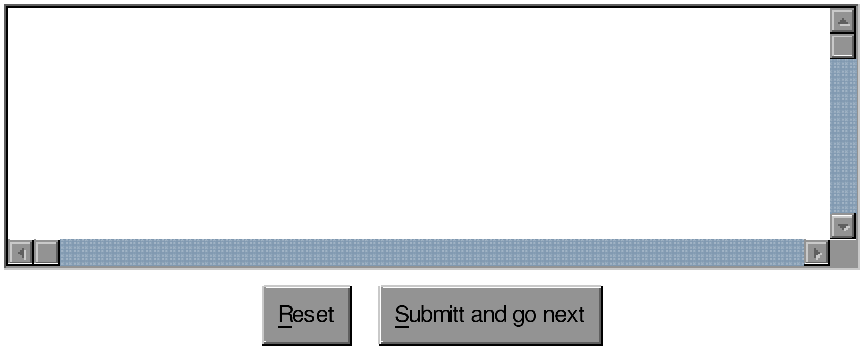
Quality Test #9

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

Quality Test #10

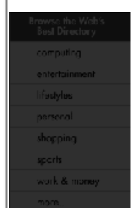
 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 30 % Acceptable Not Acceptable	 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable

Comments

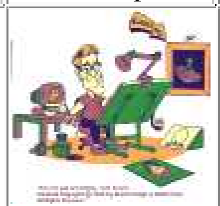


Trace #2: Colors Reduction Test

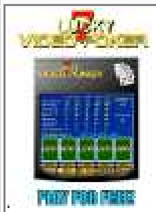
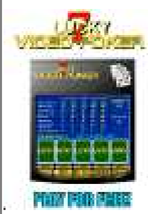
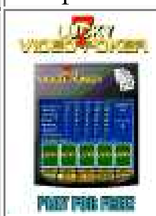
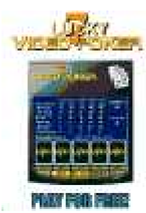
Colors Test #1

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable 'Not Acceptable
 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable	 03 % Acceptable Not Acceptable

Colors Test #2

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable	 03 % Acceptable Not Acceptable

Colors Test #3

 Original Acceptable Not Acceptable	 80 % Acceptable Not Acceptable	 60% Acceptable Not Acceptable	 40% Acceptable Not Acceptable
 20 % Acceptable Not Acceptable	 10 % Acceptable Not Acceptable	 05 % Acceptable Not Acceptable	 03 % Acceptable Not Acceptable

Colors Test #4

Original Acceptable Not Acceptable	80 % Acceptable Not Acceptable	60% Acceptable Not Acceptable	40% Acceptable Not Acceptable
20 % Acceptable Not Acceptable	10 % Acceptable Not Acceptable	05 % Acceptable Not Acceptable	03 % Acceptable Not Acceptable

Comments

Reset   Submitt and go next

Trace #3

GIF to JPEG Test

Original GIF Images	 GIF 1	 GIF 2	 GIF 3	 GIF 4
Converted to JPEG	 Acceptable Not Acceptable	 Acceptable Not Acceptable	 Acceptable Not Acceptable	 Acceptable Not Acceptable

Comments

Reset   Submitt and go next

Appendix D: JPEG Quality Factor Survey for High Resolution Displays

Resolution	1280x1024 or 1024x768 pixel
Colors	32 bit
No. of images under test	10
Confidence Intervals	95%
N (No. of observations)	39
C (Constant)	1.96

% Quality Factor	100	80	60	40	20	10	5	3
User 01	10	10	10	10	9	8	2	0
User 02	10	10	10	5	3	0	0	0
User 03	10	10	10	10	7	3	2	0
User 04	10	10	10	10	5	4	4	0
User 05	10	10	10	10	7	5	4	0
User 06	10	10	10	6	6	4	2	0
User 07	10	10	8	5	4	3	0	0
User 08	10	10	6	4	3	0	0	0
User 09	10	10	10	10	8	5	0	0
User 10	10	10	10	10	10	7	6	0
User 11	10	10	10	7	4	3	2	0
User 12	10	10	8	5	1	0	0	0
User 13	10	10	9	8	6	4	0	0
User 14	10	10	8	6	6	0	0	0
User 15	10	10	9	6	4	0	0	0
User 16	10	9	7	4	4	2	0	0
User 17	10	10	9	6	5	3	0	0
User 18	10	10	10	3	0	0	0	0
User 19	10	10	8	4	4	3	3	0
User 20	10	10	8	7	5	5	3	0
User 21	10	10	10	9	6	6	2	1
User 22	10	10	8	4	3	2	1	0
User 23	10	10	7	3	1	1	0	0
User 24	10	10	10	9	9	9	2	0
User 25	10	10	9	5	2	2	1	0
User 26	10	10	10	9	6	6	1	0
User 27	10	10	7	5	2	0	0	0
User 28	10	10	8	6	5	3	0	0
User 29	10	10	10	10	10	8	7	1
User 30	10	10	10	9	8	5	0	1
User 31	10	10	9	4	4	2	0	0
User 32	10	10	10	7	7	3	0	0
User 33	10	10	10	9	8	7	0	0
User 34	10	10	9	4	4	4	0	0
User 35	10	10	10	7	7	4	0	0
User 36	10	10	10	5	5	5	0	0
User 37	10	10	10	6	5	5	0	0
User 38	10	10	9	6	5	4	0	0
User 39	10	10	8	5	5	0	0	0
No. of respondents	**39.00**	**38.90**	**35.40**	**25.80**	**20.30**	**13.50**	**4.20**	**0.30**
Percentage of respondents	**100.00%**	**99.74%**	**90.77%**	**66.15%**	**52.05%**	**34.62%**	**10.77%**	**0.77%**
Lower Limit	**0.9103**	**0.9057**	**0.7771**	**0.5046**	**0.3691**	**0.2168**	**0.0436**	**0.0005**
Upper Limit	**1.0000**	**0.9999**	**0.9652**	**0.7895**	**0.6683**	**0.5031**	**0.2421**	**0.1032**

JPEG Quality Factor Survey for Medium Resolution Displays

Resolution	800x600 pixel
Colors	16 bit
No. of images under test	10
Confidence Intervals	95%
N (No. of observations)	86
C (Constant)	1.96

% Quality Factor	100	80	60	40	30	20	10	5
User 01	10	10	10	10	10	10	6	0
User 02	10	10	10	10	9	2	0	0
User 03	10	10	10	10	10	4	0	0
User 04	10	10	10	10	10	5	4	0
User 05	10	10	10	10	10	6	5	0
User 06	10	10	10	10	9	6	0	0
User 07	10	10	10	5	4	1	0	0
User 08	10	10	6	4	3	2	2	0
User 09	10	10	10	9	8	5	0	0
User 10	10	10	10	10	7	6	6	0
User 11	10	10	9	7	4	3	2	0
User 12	10	10	8	5	5	2	0	0
User 13	10	10	9	8	6	4	2	0
User 14	10	10	8	6	6	4	1	0
User 15	10	10	9	6	4	2	0	0
User 16	10	10	10	10	9	3	3	0
User 17	10	10	10	9	8	3	1	0
User 18	10	10	10	7	4	2	1	0
User 19	10	10	10	10	5	3	0	0
User 20	10	10	10	10	7	7	3	0
User 21	10	10	10	9	6	6	2	1
User 22	10	10	10	5	4	1	0	0
User 23	10	10	8	2	0	0	0	0
User 24	10	10	10	10	10	0	0	0
User 25	10	10	10	9	6	2	1	0
User 26	10	10	10	9	7	1	0	0
User 27	10	10	10	10	10	10	3	2
User 28	10	10	10	10	8	4	0	0
User 29	10	10	10	10	10	8	7	1
User 30	10	10	10	10	10	7	1	0
User 31	10	10	9	4	4	2	0	0
User 32	10	10	10	7	7	3	0	0
User 33	10	10	10	9	8	7	0	0
User 34	10	10	9	4	4	4	0	0
User 35	10	10	10	7	7	1	0	0
User 36	10	10	10	5	2	0	0	0
User 37	10	10	10	10	7	2	2	0
User 38	10	10	10	8	2	0	0	0
User 39	10	10	10	5	5	2	0	0
User 40	10	10	10	9	7	2	0	0
User 41	10	10	10	9	8	5	0	0
User 42	10	10	10	10	8	3	0	0
User 43	10	10	10	10	10	10	8	5
User 44	10	10	10	9	5	4	0	0
User 45	10	10	10	8	4	3	0	0
User 46	10	10	10	9	5	0	0	0
User 47	10	10	10	10	8	5	1	0

% Quality Factor	100	80	60	40	30	20	10	5
User 48	10	10	10	8	7	3	0	0
User 49	10	10	10	10	7	4	0	0
User 50	10	10	10	10	10	10	2	0
User 51	10	10	10	10	10	9	1	0
User 52	10	10	10	10	10	6	2	0
User 53	10	10	9	6	0	0	0	0
User 54	10	10	10	10	10	10	0	0
User 55	10	10	10	10	10	10	10	0
User 56	10	10	10	10	9	7	1	0
User 57	10	10	10	10	10	8	2	0
User 58	10	10	10	10	10	5	0	0
User 59	10	10	10	10	9	6	0	0
User 60	10	10	10	10	9	6	1	0
User 61	10	10	10	10	8	8	1	0
User 62	10	10	10	10	8	6	0	0
User 63	10	10	8	0	0	0	0	0
User 64	10	10	10	6	2	0	0	0
User 65	10	10	10	10	10	6	0	0
User 66	10	10	9	6	6	0	0	0
User 67	10	10	10	6	5	5	1	0
User 68	10	10	9	8	5	4	0	0
User 69	10	10	10	10	9	3	0	0
User 70	10	10	10	10	8	6	1	0
User 71	10	10	10	10	10	7	0	0
User 72	10	10	9	6	4	0	0	0
User 73	10	10	10	10	9	6	0	0
User 74	10	10	10	10	10	7	1	1
User 75	10	10	10	10	10	3	0	0
User 76	10	10	10	10	10	5	1	0
User 77	10	10	10	10	5	3	0	0
User 78	10	10	10	10	10	5	0	0
User 79	10	10	10	6	2	0	0	0
User 80	10	10	10	10	10	5	1	0
User 81	10	10	10	10	10	6	2	0
User 82	10	10	10	10	10	10	10	3
User 83	10	10	10	10	9	8	1	0
User 84	10	10	10	10	8	5	1	0
User 85	10	10	10	10	5	0	0	0
User 86	10	10	10	9	6	0	0	0
No. of respondents	**86.00**	**86.00**	**83.90**	**73.40**	**61.00**	**36.40**	**10.00**	**1.30**
Percentage of respondents	**100.00%**	**100.00%**	**97.56%**	**85.35%**	**70.93%**	**42.33%**	**11.63%**	**1.51%**
Lower Limit	**0.9572**	**0.9572**	**0.9174**	**0.7637**	**0.6060**	**0.3243**	**0.0644**	**0.0032**
Upper Limit	**1.0000**	**1.0000**	**0.9931**	**0.9130**	**0.7947**	**0.5288**	**0.2010**	**0.0685**

JPEG Quality Factor Survey for Low Resolution Displays

Resolution	640 x 480 pixel or lower
Colors	8 bit or lower
No. of images under test	10
Confidence Intervals	95%
N (No. of observations)	41
C (Constant)	1.96

% Quality Factor	100	80	60	40	30	20	10	5
User 01	10	10	10	10	10	9	0	0
User 02	10	10	10	10	10	5	0	0
User 03	10	10	10	10	10	9	1	0
User 04	10	10	10	10	10	9	0	0
User 05	10	10	10	10	10	7	0	0
User 06	10	10	10	10	9	4	0	0
User 07	10	10	10	10	10	1	0	0
User 08	10	10	10	9	3	0	0	0
User 09	10	10	10	10	10	3	0	0
User 10	10	10	10	10	10	6	0	0
User 11	10	10	10	10	10	6	2	0
User 12	10	10	10	10	10	10	4	0
User 13	10	10	10	10	10	10	5	1
User 14	10	10	10	10	10	4	0	0
User 15	10	10	10	10	10	10	1	0
User 16	10	10	10	10	10	10	10	5
User 17	10	10	10	10	10	9	4	0
User 18	10	10	10	10	10	9	2	0
User 19	10	10	10	10	5	3	0	0
User 20	10	10	10	10	7	7	3	0
User 21	10	10	10	10	10	9	2	0
User 22	10	10	10	10	10	3	0	0
User 23	10	10	10	8	5	5	3	0
User 24	10	10	10	10	10	9	2	0
User 25	10	10	10	10	10	5	0	0
User 26	10	10	10	9	5	0	0	0
User 27	10	10	10	10	10	10	10	0
User 28	10	10	10	10	10	2	0	0
User 29	10	10	10	10	10	8	0	0
User 30	10	10	10	10	10	10	0	0
User 31	10	10	10	9	8	8	0	0
User 32	10	10	10	10	8	0	0	0
User 33	10	10	10	10	10	10	8	0
User 34	10	10	10	10	10	10	10	0
User 35	10	10	10	10	10	1	0	0
User 36	10	10	10	10	4	0	0	0
User 37	10	10	10	10	10	10	5	0
User 38	10	10	10	10	10	10	7	0
User 39	10	10	10	10	10	10	0	0
User 40	10	10	10	10	10	10	9	4
User 41	10	10	10	10	10	10	3	0
No. of respondents	**41.00**	**41.00**	**41.00**	**40.50**	**37.40**	**27.10**	**9.10**	**1.00**
Percentage of respondents	**100.00%**	**100.00%**	**100.00%**	**98.78%**	**91.22%**	**66.10%**	**22.20%**	**2.44%**
Lower Limit	**0.9143**	**0.9143**	**0.9143**	**0.8933**	**0.7868**	**0.5079**	**0.1218**	**0.0043**
Upper Limit	**1.0000**	**1.0000**	**1.0000**	**0.9987**	**0.9669**	**0.7864**	**0.3697**	**0.1260**

Appendix E: Colors Reduction Factor Survey for High Resolution Displays

Resolution	1280x1024 or 1024x768
Colors	32 bit
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	39
C (Constant)	1.96

% Colors Reduction Factor	100	80	60	40	20	10	5	3
User 01	4	4	4	4	4	3	1	0
User 02	4	4	4	4	2	0	0	0
User 03	4	4	4	4	2	0	0	0
User 04	4	4	4	3	3	1	0	0
User 05	4	4	4	4	4	4	1	0
User 06	4	4	4	4	4	4	0	0
User 07	4	4	3	2	2	0	0	0
User 08	4	4	3	2	1	0	0	0
User 09	4	4	4	4	4	0	0	0
User 10	4	4	4	4	3	3	0	0
User 11	4	4	4	4	2	1	1	0
User 12	4	4	3	1	0	0	0	0
User 13	4	4	4	4	3	0	0	0
User 14	4	4	3	2	2	0	0	0
User 15	4	4	3	3	1	0	0	0
User 16	4	4	4	4	4	4	2	0
User 17	4	4	4	4	4	4	4	0
User 18	4	4	3	2	0	0	0	0
User 19	4	4	4	4	4	2	0	0
User 20	4	4	4	4	3	2	0	0
User 21	4	4	4	4	2	2	1	1
User 22	4	4	4	4	3	1	0	0
User 23	4	4	4	4	2	1	0	0
User 24	4	4	4	4	4	4	2	0
User 25	4	4	4	1	1	0	0	0
User 26	4	4	4	4	3	0	0	0
User 27	4	4	4	1	0	0	0	0
User 28	4	4	4	4	4	4	1	0
User 29	4	4	4	4	4	3	0	0
User 30	4	4	4	4	3	0	0	0
User 31	4	4	4	2	2	0	0	0
User 32	4	4	4	3	3	1	0	0
User 33	4	4	4	4	4	3	0	0
User 34	4	4	4	4	2	0	0	0
User 35	4	4	4	3	2	0	0	0
User 36	4	4	4	3	0	0	0	0
User 37	4	3	2	0	0	0	0	0
User 38	4	4	4	4	2	0	0	0
User 39	4	4	4	2	0	0	0	0
No. of respondents	**39.00**	**38.75**	**37.00**	**31.50**	**23.25**	**11.75**	**3.25**	**0.25**
Percentage of respondents	**100.00%**	**99.36%**	**94.87%**	**80.77%**	**59.62%**	**30.13%**	**8.33%**	**0.64%**
Lower Limit	**0.9103**	**0.8990**	**0.8311**	**0.6589**	**0.4403**	**0.1806**	**0.0299**	**0.0004**
Upper Limit	**1.0000**	**0.9996**	**0.9858**	**0.9013**	**0.7347**	**0.4576**	**0.2115**	**0.1010**

Colors Reduction Factor Survey for Medium Resolution Displays

Resolution	800x600 pixel
Colors	16 bit
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	86
C (Constant)	1.96

% Colors Reduction Factor	100	80	60	40	30	20	10	5
User 01	4	4	4	4	4	4	0	0
User 02	4	4	4	4	4	4	2	0
User 03	4	4	4	4	4	0	0	0
User 04	4	4	4	4	4	1	0	0
User 05	4	4	4	4	4	4	2	0
User 06	4	4	4	4	4	2	0	0
User 07	4	4	4	4	3	1	0	0
User 08	4	4	4	3	3	1	1	0
User 09	4	4	4	4	4	0	0	0
User 10	4	4	4	4	3	1	0	0
User 11	4	4	4	4	4	4	2	1
User 12	4	4	4	4	4	1	0	0
User 13	4	4	4	4	1	0	0	0
User 14	4	4	4	4	4	3	0	0
User 15	4	4	4	4	2	1	0	0
User 16	4	4	4	4	4	4	4	0
User 17	4	4	4	4	4	3	0	0
User 18	4	4	4	4	4	2	0	0
User 19	4	4	4	4	2	0	0	0
User 20	4	4	4	4	4	0	0	0
User 21	4	4	4	4	0	0	0	0
User 22	4	4	3	0	0	0	0	0
User 23	4	4	4	4	4	4	1	0
User 24	4	4	4	4	4	4	4	0
User 25	4	4	4	4	4	3	3	0
User 26	4	4	4	4	3	0	0	0
User 27	4	4	4	4	4	2	0	0
User 28	4	4	4	4	4	4	0	0
User 29	4	4	4	4	4	4	2	0
User 30	4	4	4	4	4	3	1	1
User 31	4	4	4	2	0	0	0	0
User 32	4	4	4	4	4	1	0	0
User 33	4	4	4	4	4	1	0	0
User 34	4	4	4	4	2	0	0	0
User 35	4	4	4	0	0	0	0	0
User 36	4	4	4	3	0	0	0	0
User 37	4	4	4	4	4	4	3	0
User 38	4	4	4	4	4	4	4	2
User 39	4	4	4	4	4	1	0	0
User 40	4	4	4	4	0	0	0	0
User 41	4	4	4	4	4	4	4	0
User 42	4	4	4	4	4	3	1	0
User 43	4	4	4	4	4	4	0	0
User 44	4	4	4	4	2	1	0	0
User 45	4	4	4	4	3	1	0	0
User 46	4	4	4	4	4	3	3	1

% Colors Reduction Factor	100	80	60	40	30	20	10	5
User 47	4	4	4	4	4	4	1	0
User 48	4	4	4	4	3	0	0	0
User 49	4	4	4	4	4	3	0	0
User 50	4	4	4	4	4	1	0	0
User 51	4	4	4	4	4	3	0	0
User 52	4	4	4	4	4	1	0	0
User 53	4	4	4	4	4	4	4	0
User 54	4	4	4	4	4	0	0	0
User 55	4	4	4	4	4	3	1	0
User 56	4	4	4	2	0	0	0	0
User 57	4	4	4	4	3	0	0	0
User 58	4	4	0	0	0	0	0	0
User 59	4	4	4	4	1	1	0	0
User 60	4	4	4	4	4	4	2	0
User 61	4	4	4	2	0	0	0	0
User 62	4	4	4	4	4	4	1	0
User 63	4	4	4	4	4	3	2	0
User 64	4	4	4	3	1	0	0	0
User 65	4	4	4	4	1	0	0	0
User 66	4	4	4	4	1	0	0	0
User 67	4	4	4	3	1	0	0	0
User 68	4	4	4	4	3	1	0	0
User 69	4	4	4	4	2	0	0	0
User 70	4	4	4	4	4	4	0	0
User 71	4	4	4	4	4	3	0	0
User 72	4	4	4	4	3	0	0	0
User 73	4	4	4	4	4	2	0	0
User 74	4	4	4	4	4	1	0	0
User 75	4	4	4	3	2	0	0	0
User 76	4	4	4	4	1	1	0	0
User 77	4	4	4	4	2	0	0	0
User 78	4	4	4	4	0	0	0	0
User 79	4	4	4	4	4	2	1	0
User 80	4	4	4	4	4	4	2	2
User 81	4	4	4	4	4	4	1	0
User 82	4	4	4	4	4	2	0	0
User 83	4	4	4	4	4	1	0	0
User 84	4	4	4	4	4	2	1	0
User 85	4	4	4	4	4	3	0	0
User 86	4	4	2	0	0	0	0	0
No. of respondents	**86.00**	**86.00**	**84.25**	**79.25**	**64.00**	**37.00**	**13.25**	**1.75**
Percentage of respondents	**100.00%**	**100.00%**	**97.97%**	**92.15%**	**74.42%**	**43.02%**	**15.41%**	**2.03%**
Lower Limit	**0.9572**	**0.9572**	**0.9235**	**0.8450**	**0.6429**	**0.3308**	**0.0928**	**0.0052**
Upper Limit	**1.0000**	**1.0000**	**0.9948**	**0.9619**	**0.8246**	**0.5356**	**0.2450**	**0.0765**

Colors Reduction Factor Survey for Low Resolution Displays

Resolution	640x480 or lower
Colors	8 bit or lower
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	41
C (Constant)	1.96

% Colors Reduction Factor	100	80	60	40	30	20	10	5
User 01	4	4	4	4	4	4	1	0
User 02	4	4	4	4	4	4	2	0
User 03	4	4	4	4	4	4	0	0
User 04	4	4	4	4	4	2	0	0
User 05	4	4	4	4	4	1	0	0
User 06	4	4	4	4	4	0	0	0
User 07	4	4	4	4	4	4	1	0
User 08	4	4	4	4	4	4	0	0
User 09	4	4	4	4	4	4	3	0
User 10	4	4	4	2	0	0	0	0
User 11	4	4	4	4	4	1	0	0
User 12	4	4	4	4	4	4	1	0
User 13	4	4	4	4	4	4	3	2
User 14	4	4	4	4	4	1	0	0
User 15	4	4	4	4	4	1	0	0
User 16	4	4	4	4	0	0	0	0
User 17	4	4	4	4	4	4	3	0
User 18	4	4	4	4	4	4	4	3
User 19	4	4	4	4	4	4	3	0
User 20	4	4	4	4	4	2	0	0
User 21	4	4	4	4	4	4	4	4
User 22	4	4	4	4	4	4	3	0
User 23	4	4	4	4	4	4	3	0
User 24	4	4	4	4	4	2	0	0
User 25	4	4	4	4	4	3	1	0
User 26	4	4	4	4	0	0	0	0
User 27	4	4	4	4	4	4	3	0
User 28	4	4	4	4	4	3	2	0
User 29	4	4	4	4	4	4	0	0
User 30	4	4	4	4	4	2	1	0
User 31	4	4	4	4	4	1	0	0
User 32	4	4	4	4	4	4	4	1
User 33	4	4	4	4	4	4	2	0
User 34	4	4	4	4	4	4	0	0
User 35	4	4	4	4	4	3	3	0
User 36	4	4	4	4	4	2	0	0
User 37	4	4	4	4	4	4	2	0
User 38	4	4	4	4	4	4	2	0
User 39	4	4	4	4	4	4	4	0
User 40	4	4	4	4	4	4	2	0
User 41	4	4	4	4	4	4	4	1
No. of respondents	**41.00**	**41.00**	**41.00**	**40.50**	**38.00**	**30.00**	**15.25**	**2.75**
Percentage of respondents	**100.00%**	**100.00%**	**100.00%**	**98.78%**	**92.68%**	**73.17%**	**37.20%**	**6.71%**
Lower Limit	**0.9143**	**0.9143**	**0.9143**	**0.8933**	**0.8057**	**0.5807**	**0.2410**	**0.0221**
Upper Limit	**1.0000**	**1.0000**	**1.0000**	**0.9987**	**0.9748**	**0.8431**	**0.5248**	**0.1862**

Appendix F: Geometry Resizing Factor Survey for High Resolution Displays

Resolution	1280x1024 or 1024x768 pixel
Colors	32 bit
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	39
C (Constant)	1.96

% Geometry Resizing	100	80	60	40	30	20	10	5
User 01	4	4	4	4	4	4	3	0
User 02	4	4	4	4	3	1	1	0
User 03	4	4	4	4	2	0	0	0
User 04	4	4	4	4	4	3	1	1
User 05	4	4	4	4	4	4	3	2
User 06	4	4	4	4	4	3	3	1
User 07	4	4	4	4	3	1	0	0
User 08	4	4	4	4	2	1	0	0
User 09	4	4	4	4	4	3	1	0
User 10	4	4	4	4	4	3	3	2
User 11	4	4	4	4	4	4	1	0
User 12	4	4	4	4	4	3	2	0
User 13	4	4	4	4	4	3	0	0
User 14	4	4	4	4	4	4	4	0
User 15	4	4	4	4	4	4	2	1
User 16	4	4	4	4	4	2	0	0
User 17	4	4	4	4	4	4	4	2
User 18	4	4	4	4	4	4	4	0
User 19	4	4	4	4	4	3	1	0
User 20	4	4	4	4	4	4	2	0
User 21	4	4	4	4	4	3	3	0
User 22	4	4	4	4	4	3	0	0
User 23	4	4	4	4	4	1	0	0
User 24	4	4	4	4	4	4	2	0
User 25	4	4	4	4	3	2	1	0
User 26	4	4	4	4	3	0	0	0
User 27	4	4	4	4	3	3	2	1
User 28	4	4	4	4	4	4	2	0
User 29	4	4	4	4	4	4	2	0
User 30	4	4	4	4	4	3	2	1
User 31	4	4	4	4	4	2	2	0
User 32	4	4	4	4	4	1	0	0
User 33	4	4	4	4	4	1	0	0
User 34	4	4	4	4	3	3	1	0
User 35	4	4	4	1	0	0	0	0
User 36	4	4	4	4	2	2	0	0
User 37	4	4	4	4	1	0	0	0
User 38	4	4	4	4	2	0	0	0
User 39	4	4	4	4	3	2	2	0
No. of respondents	**39.00**	**39.00**	**39.00**	**38.25**	**33.50**	**24.00**	**13.50**	**2.75**
Percentage of respondents	**100.00%**	**100.00%**	**100.00%**	**98.08%**	**85.90%**	**61.54%**	**34.62%**	**7.05%**
Lower Limit	**0.9103**	**0.9103**	**0.9103**	**0.8781**	**0.7177**	**0.4590**	**0.2168**	**0.0232**
Upper Limit	**1.0000**	**1.0000**	**1.0000**	**0.9972**	**0.9359**	**0.7511**	**0.5031**	**0.1948**

Geometry Resizing Factor Survey for Medium Resolution Displays

Resolution	800x600 pixel
Colors	16 bit
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	86
C (Constant)	1.96

% Geometry Resizing	100	80	60	40	30	20	10	5
User 01	4	4	4	4	4	2	0	0
User 02	4	4	4	4	4	4	0	0
User 03	4	4	4	4	4	4	2	0
User 04	4	4	4	4	1	0	0	0
User 05	4	4	4	1	0	0	0	0
User 06	4	4	4	0	0	0	0	0
User 07	4	4	4	4	4	4	4	2
User 08	4	4	4	4	4	4	4	0
User 09	4	4	4	4	0	0	0	0
User 10	4	4	1	0	0	0	0	0
User 11	4	4	1	0	0	0	0	0
User 12	4	4	4	4	2	1	0	0
User 13	4	4	4	4	4	4	4	1
User 14	4	4	4	4	4	4	0	0
User 15	4	4	4	4	1	0	0	0
User 16	4	4	4	4	2	0	0	0
User 17	4	2	0	0	0	0	0	0
User 18	4	4	0	0	0	0	0	0
User 19	4	4	4	4	4	0	0	0
User 20	4	4	4	4	4	3	0	0
User 21	4	4	4	4	2	1	0	0
User 22	4	4	4	2	2	0	0	0
User 23	4	4	4	4	4	1	0	0
User 24	4	4	4	4	4	2	0	0
User 25	4	4	4	4	2	0	0	0
User 26	4	4	4	4	3	0	0	0
User 27	4	4	4	4	4	1	0	0
User 28	4	4	4	4	4	1	0	0
User 29	4	4	4	4	3	0	0	0
User 30	4	4	4	4	3	2	0	0
User 31	4	4	4	4	0	0	0	0
User 32	4	4	4	2	0	0	0	0
User 33	4	4	4	4	4	2	0	0
User 34	4	4	4	4	2	0	0	0
User 35	4	4	4	0	0	0	0	0
User 36	4	4	4	2	0	0	0	0
User 37	4	4	4	4	4	4	0	0
User 38	4	4	4	4	4	4	4	0
User 39	4	4	4	2	1	0	0	0
User 40	4	4	4	4	4	4	2	0
User 41	4	4	4	4	4	4	0	0
User 42	4	4	4	4	4	2	1	0
User 43	4	4	4	4	4	1	0	0
User 44	4	4	4	4	1	0	0	0
User 45	4	4	2	0	0	0	0	0
User 46	4	4	4	4	4	0	0	0

% Geometry Resizing	100	80	60	40	30	20	10	5
User 47	4	4	4	4	4	0	0	0
User 48	4	4	2	0	0	0	0	0
User 49	4	4	4	4	4	4	0	0
User 50	4	4	4	4	1	0	0	0
User 51	4	4	4	4	3	2	0	0
User 52	4	4	4	4	4	4	0	0
User 53	4	4	4	4	4	4	0	0
User 54	4	4	4	4	0	0	0	0
User 55	4	4	4	4	4	1	0	0
User 56	4	4	4	4	0	0	0	0
User 57	4	4	4	4	1	0	0	0
User 58	4	4	4	0	0	0	0	0
User 59	4	4	4	4	0	0	0	0
User 60	4	4	4	4	4	4	4	0
User 61	4	4	2	0	0	0	0	0
User 62	4	4	4	4	4	1	0	0
User 63	4	4	4	4	4	2	0	0
User 64	4	4	4	4	4	0	0	0
User 65	4	4	4	4	4	0	0	0
User 66	4	4	4	4	2	0	0	0
User 67	4	4	4	4	4	2	0	0
User 68	4	4	4	4	3	0	0	0
User 69	4	4	4	4	4	1	0	0
User 70	4	4	4	4	4	2	0	0
User 71	4	4	4	4	4	4	4	2
User 72	4	4	4	4	2	0	0	0
User 73	4	4	4	3	0	0	0	0
User 74	4	4	4	4	1	0	0	0
User 75	4	4	4	2	0	0	0	0
User 76	4	4	4	4	4	2	0	0
User 77	4	4	4	4	0	0	0	0
User 78	4	4	4	1	0	0	0	0
User 79	4	4	4	4	4	0	0	0
User 80	4	4	4	4	4	4	0	0
User 81	4	4	4	4	4	4	0	0
User 82	4	4	4	4	4	0	0	0
User 83	4	4	4	4	4	3	0	0
User 84	4	4	4	4	4	2	1	0
User 85	4	4	4	4	1	0	0	0
User 86	4	4	0	0	0	0	0	0
No. of respondents	**86.00**	**85.50**	**80.00**	**70.75**	**51.75**	**26.25**	**7.50**	**1.25**
Percentage of respondents	**100.00%**	**99.42%**	**93.02%**	**82.27%**	**60.17%**	**30.52%**	**8.72%**	**1.45%**
Lower Limit	**0.9572**	**0.9467**	**0.8560**	**0.7287**	**0.4961**	**0.2180**	**0.0439**	**0.0030**
Upper Limit	**1.0000**	**0.9994**	**0.9676**	**0.8891**	**0.6987**	**0.4091**	**0.1658**	**0.0676**

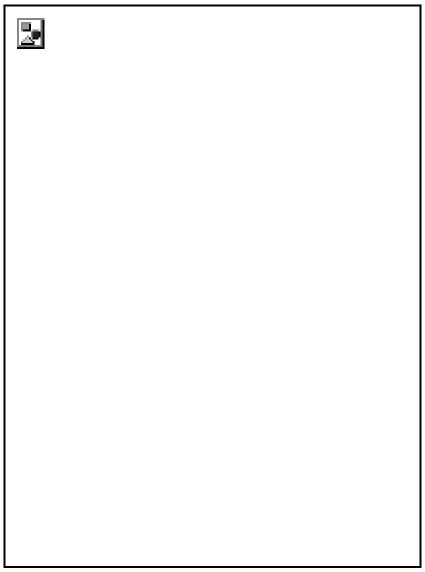
Geometry Resizing Factor Survey for Low Resolution Displays

Resolution	640x480 pixel or lower
Colors	8 bit or lower
No. of images under test	4
Confidence Intervals	95%
N (No. of observations)	41
C (Constant)	1.96

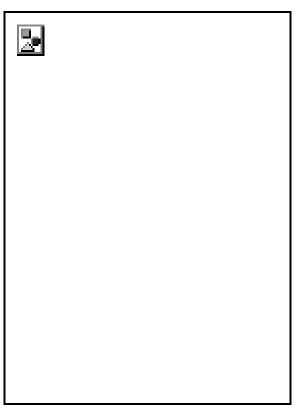
% Geometry Resizing	100	80	60	40	30	20	10	5
User 01	4	4	4	4	4	2	0	0
User 02	4	4	4	4	4	3	0	0
User 03	4	4	4	4	4	0	0	0
User 04	4	4	4	4	1	0	0	0
User 05	4	4	4	4	1	0	0	0
User 06	4	4	4	0	0	0	0	0
User 07	4	4	4	2	0	0	0	0
User 08	4	4	4	0	0	0	0	0
User 09	4	4	4	4	4	4	2	0
User 10	4	4	4	3	0	0	0	0
User 11	4	4	4	1	0	0	0	0
User 12	4	4	4	4	3	0	0	0
User 13	4	4	4	4	3	0	0	0
User 14	4	4	0	0	0	0	0	0
User 15	4	4	2	0	0	0	0	0
User 16	4	4	4	4	3	0	0	0
User 17	4	4	4	4	4	1	1	0
User 18	4	4	4	4	4	1	0	0
User 19	4	4	1	0	0	0	0	0
User 20	4	4	4	4	4	2	0	0
User 21	4	4	4	4	4	3	0	0
User 22	4	4	4	4	1	0	0	0
User 23	4	4	4	4	0	0	0	0
User 24	4	4	4	3	0	0	0	0
User 25	4	4	4	4	3	0	0	0
User 26	4	4	4	3	0	0	0	0
User 27	4	4	4	4	4	4	0	0
User 28	4	4	4	4	1	0	0	0
User 29	4	4	4	4	2	0	0	0
User 30	4	4	4	4	3	0	0	0
User 31	4	4	4	4	4	4	1	0
User 32	4	4	4	4	4	3	1	0
User 33	4	4	3	0	0	0	0	0
User 34	4	4	4	3	1	0	0	0
User 35	4	4	4	4	4	0	0	0
User 36	4	4	4	4	0	0	0	0
User 37	4	4	4	4	3	3	0	0
User 38	4	4	4	4	0	0	0	0
User 39	4	4	3	0	0	0	0	0
User 40	4	4	4	4	4	1	0	0
User 41	4	4	4	4	4	4	0	0
No. of respondents	**41.00**	**41.00**	**38.25**	**31.75**	**20.25**	**8.75**	**1.25**	**0.00**
Percentage of respondents	**100.00%**	**100.00%**	**93.29%**	**77.44%**	**49.39%**	**21.34%**	**3.05%**	**0.00%**
Lower Limit	**0.9143**	**0.9143**	**0.8138**	**0.6263**	**0.3481**	**0.1156**	**0.0063**	**0.0000**
Upper Limit	**1.0000**	**1.0000**	**0.9779**	**0.8755**	**0.6408**	**0.3604**	**0.1351**	**0.0857**

Appendix G: http://kunz-pc.sce.carleton.ca/pictures.html

(No Proxy)



Myself, summer 1994



Myself, October 1997

Jean (1)
By the way, this is not a photo out of a magazine to show off my dream woman, this actually is my wife.... :-)
(Someone who checked out the pages advised me to add this, to make sure that people won't mistakenly believe that these are just any arbitrary model photos, and I gladly picked that suggestion up)

Jean (2)

Hobbies: I enjoy playing squash and soccer. The above picture shows the team that won the ‘C’ league championship at the University of Waterloo in the Fall 1995 term. Guess which one of the players is me.....

This is the Division 7 Spitfires team for 1999 (Ottawa-Carleton Soccer League). Unlike the previous year, we actually managed to win at least one game this season (though I was not present for this historic moment, which some tend to interpret their own way ....)

http://kunz-pc.sce.carleton.ca/pictures.html

Device = (Palm 320x200 pixel)

Myself, summer 1994

Myself, October 1997

Jean (1)
By the way, this is not a photo out of a magazine to show off my dream woman, this actually is my wife.... :-)
(Someone who checked out the pages advised me to add this, to make sure that people won't mistakenly believe that these are just any arbitrary model photos, and I gladly picked that suggestion up)

Jean (2)

Hobbies: I enjoy playing squash and soccer. The above picture shows the team that won the ‘C’ league championship at the University of Waterloo in the Fall 1995 term. Guess which one of the players is me.....

This is the Division 7 Spitfires team for 1999 (Ottawa-Carleton Soccer League). Unlike the previous year, we actually managed to win at least one game this season (though I was not present for this historic moment, which some tend to interpret their own way ....)

---

http://kunz-pc.sce.carleton.ca/pictures.html
Device = CE Device 640x240 pixel

Myself, summer 1994

Myself, October 1997

Jean (1)
By the way, this is not a photo out of a magazine to show off my dream woman, this actually is my wife.... :-)
(Someone who checked out the pages advised me to add this, to make sure that people won't mistakenly believe that these are just any arbitrary model photos, and I gladly picked that suggestion up)

Jean (2)

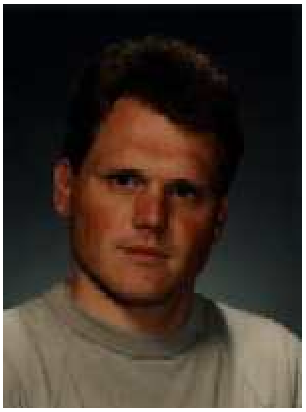
Hobbies: I enjoy playing squash and soccer. The above picture shows the team that won the 'C' league championship at the University of Waterloo in the Fall 1995 term. Guess which one of the players is me.....

This is the Division 7 Spitfires team for 1999 (Ottawa-Carleton Soccer League). Unlike the previous year, we actually managed to win at least one game this season (though I was not present for this historic moment, which some tend to interpret their own way ...)

---

http://kunz-pc.sce.carleton.ca/pictures.html

Device = Laptop 800x600 pixel



Myself, summer 1994

Myself, October 1997

Jean (1)
By the way, this is not a photo out of a magazine to show off my dream woman, this actually is my wife.... :-)
(Someone who checked out the pages advised me to add this, to make sure that people won't mistakenly believe that these are just any arbitrary model photos, and I gladly picked that suggestion up)

Jean (2)

Hobbies: I enjoy playing squash and soccer. The above picture shows the team that won the ‘C’ league championship at the University of Waterloo in the Fall 1995 term. Guess which one of the players is me.....

This is the Division 7 Spitfires team for 1999 (Ottawa-Carleton Soccer League). Unlike the previous year, we actually managed to win at least one game this season (though I was not present for this historic moment, which some tend to interpret their own way ....)
