VIDEO POSTCARD
A Tool for Exploring the New Method in Postcard Communication

by **Erhhung Yuan**

Submitted to the Department of Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology
on May 18, 1994
for the fulfillment of the Advanced Undergraduate Project (6.199)

ABSTRACT

Since its existence, the postcard has remained a two dimensional entity that embodies the pictorial and textual descriptions of a given moment. Its visual expressions belong solely to the eyes of the photographer or artist who framed that instant in time. The sender can at best describe in words his associations with the picture, but they are never integrated in a way the sender really prefers. If he is given the opportunity to start from scratch and be the photographer, his final product would, for sure, be much more personal. Just as film introduced a definite sense of time and life into still photography and allowed time to be molded as a material, the advent of digital video and high speed, high bandwidth computer networks in a world of rapidly growing connectivity has stretched the expressive potential of a paper postcard into the temporal dimension. This technology not only allows the sender to participate in the creative process of designing personalized descriptions, hence improving personal communication, but also gives him the power of non-linear presentation in an interactive environment of integrated audio, video, and text. It is thus the goal of Video Postcard to provide the necessary tools for transforming the traditional method of communication by postcard into an interactive movie experience available to all.

Project Supervisor:

Glorianna Davenport
Assistant Professor of Media Technology,
Director of the Interactive Cinema Group
MIT Media Laboratory

Introduction

Sending a postcard is a simple and routine task that usually involves nothing more than selecting a piece of ready-made picture, writing about personal experiences on the other side of that picture, and mailing it. This simplicity is the reason why the process has never evolved since its existence and people have never given much thought into improving it. This is not to say that the traditional method of sending postcards has not been an efficient and effective means of communication; rather, because this task is so well suited for the digital medium, its potential ought to be explored in depth.

Video, in the most basic sense of the word, is simply a series of photographs, which, when digitized and shown in a coherent order, tells a story. If this were the only definition, then a video postcard is really just a stack of paper postcards that can be viewed like a flip book but little else. Fortunately, video offers a material that is more easily manipulated than is possible with photographs. A picture can only *suggest* time, it cannot physically *demonstrate* nor *interact* with time. By using digital video, which may also include audio and text, one not only can alter the temporal attributes of a scene captured in real time, such as speed, direction, and transition between cuts, but also interact with the scene by dynamically changing its visual and temporal attributes as well as the flow of the story. These techniques have been used in a variety of entertainment and educational software developed by Philips Consumer Electronics in the last couple of years, in a genre which they dubbed as the *Compact Disc Interactive*, or *CD-I*, and will foreseeably be the key to the success of interactive television when it becomes widely available in everyone's home. One can even imagine some time in the near future when people will be sending each other virtual realistic postcards. In such a scenario the viewer would become the participant in the scene and discovers for himself the experiences that another has experienced. But, for now, Video Postcard tries to take a step in that direction by bringing the design aspect of interactive video into everyone's hands.

Application Overview

Video Postcard is developed as an X Windows/Motif application on UNIX based systems. The particular platform on which the application is written is a Digital DECstation 5000/200 running X11R4, Motif 1.1.4, and Ultrix 4.2. Video clips are stored in the MPEG (Motion Picture Experts Group) format, and uses the encoding/decoding algorithms developed by the Regents of the University of California.

The MPEG format was chosen primary because of its excellent compression ratio. Since digital video is inherently large in size (on the order of tens of megabytes for just a few minutes), it is imperative that the size be reduced significantly without seriously limiting the amount of video that can be incorporated into a postcard. The real concern for file size is due to the fact that mail servers which use the standard SMTP (Simple Mail Transfer Protocol) cannot reliably handle messages larger than 100 kilobytes. Even with compression, it is necessary to segment each piece of video into manageable blocks and convert them from binary to ASCII format before giving them to the mail handler. Therefore, having to dealing with raw video would just be too slow and cumbersome.

Opting for the MPEG format is, however, not without cost. Since such sophisticated encoding/decoding algorithms impose a high load on the system, playback quality will suffer if the CPU cannot keep up with the intended frame rate. In general, the playback quality is directly proportional to the number of video clips that must be played concurrently; hence, the quality may vary from one scene to the next for the same postcard.

The SMTP was chosen because it is used by nearly all mail servers on the Internet. It also seemed logical that a video postcard should be handled like any other piece of e-mail, that is, it would be addressable by a user's e-mail address, and it can be placed into the appropriate mailbox in the user's account.

Using the Application

The first step in creating a video postcard is gathering the necessary video clips and converting them into the MPEG format. This can be done by using a capture board to digitize the analog video source, and then passing the digitized video through an MPEG encoder, such as *mpeg_encode*, on a frame by frame basis, to generate an MPEG file rated at 30 fps (since software is available to do this type of conversion, Video Postcard does not currently provide it as a feature).

Once the video clips are available, the user can begin the design process by starting the application. Select *Create* from the *Postcard* menu to bring up a large construction window containing the command button bar, the canvas area, the time line, and the message bar (see figure 1). The canvas area is where video and text clips are imported, adjusted, and sequenced; it is also the viewport where playback eventually occurs. In edit mode, each clip is displayed with a frame and two handles that can be clicked-and-dragged to adjust its position and dimensions. In addition, the user may choose to show the *snap placement grid* as a guide in the layout process (see figure 2); activating the *snap placement* option will also restrict the adjustments to the grid lines. The time line shows the total duration of the postcard and an indicator positioned at the current time. It is also used to jump directly to a specific frame if clicked with the mouse. There are several shortcut keys which facilitate the editing process, and they are listed on table 1. In most instances, simply moving the mouse pointer over a specific button, arrow, or region of the construction window will display a short description about the purpose of the item in question on the message bar.

Click on the *Insert Video* button to pop up the MPEG selection dialog. Here the user chooses an MPEG movie to add to the canvas. After making a selection, the dialog disappears and a clip frame containing the MPEG filename is placed at the center of the

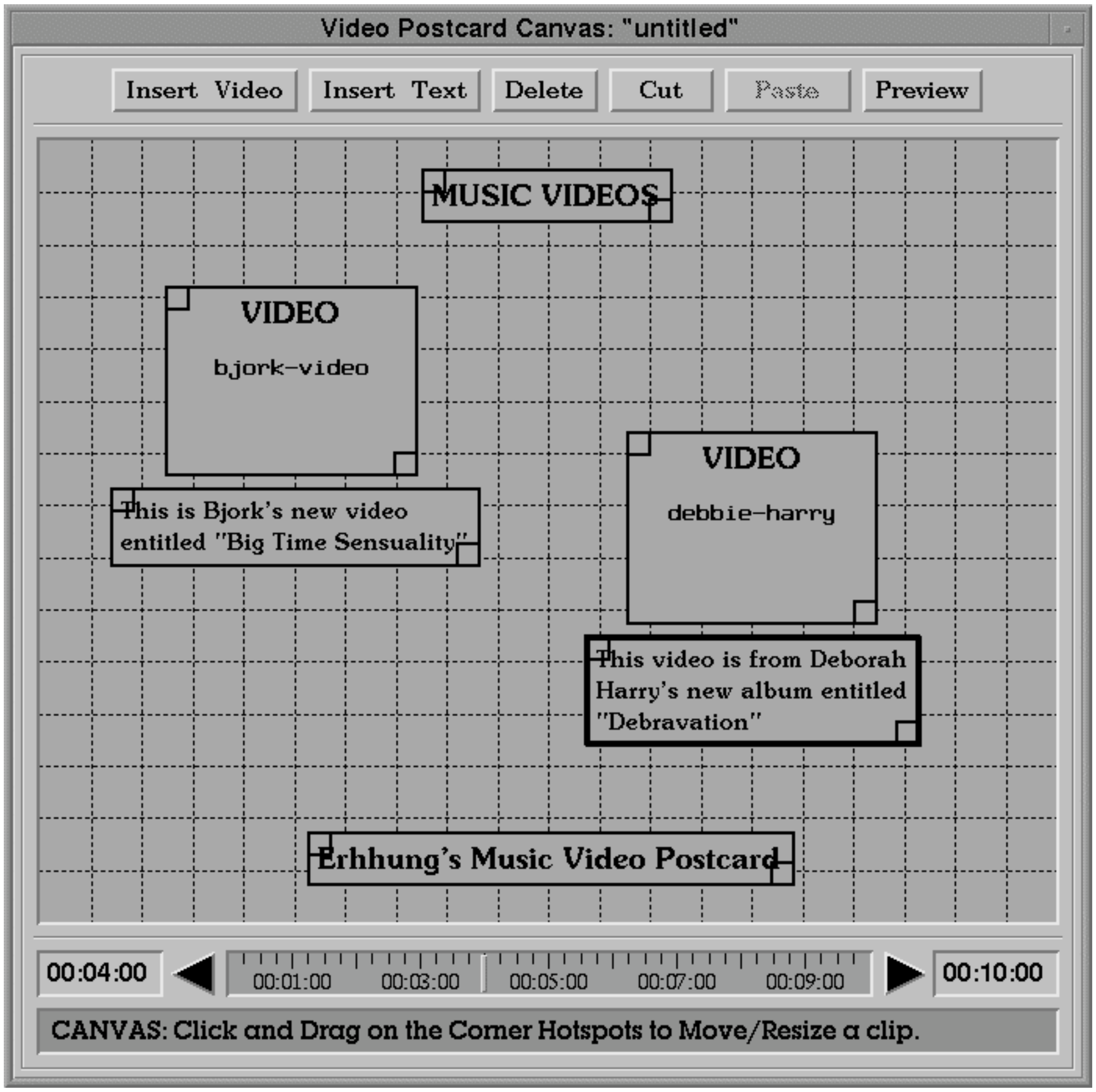

Video Postcard Canvas: "untitled"
Insert Video
Insert Text
Delete
Cut
Paste
Preview
MUSIC VIDEOS
VIDEO
bjork-video
This is Bjork's new video entitled "Big Time Sensuality"
VIDEO
debbie-harry
This video is from Deborah Harry's new album entitled "Debravation"
Erhhung's Music Video Postcard
00:04:00
00:01:00
00:03:00
00:05:00
00:07:00
00:09:00
00:10:00
CANVAS: Click and Drag on the Corner Hotspots to Move/Resize a clip.

Figure 1: The construction window.

canvas. At this time the user may move and resize [1] the clip, insert another video clip, or insert a text clip. Text clips can be used as captions to accompany video clips, or as titles if a large font is used, or just as plain text to describe the designer's personal experiences. Text is entered in the box labeled

[1] Since an MPEG movie is originally encoded with specific dimensions, resizing the video clip to a different set of dimensions will require dynamic scaling during playback, which can be very costly on the system load and hence degrade the quality of the video.

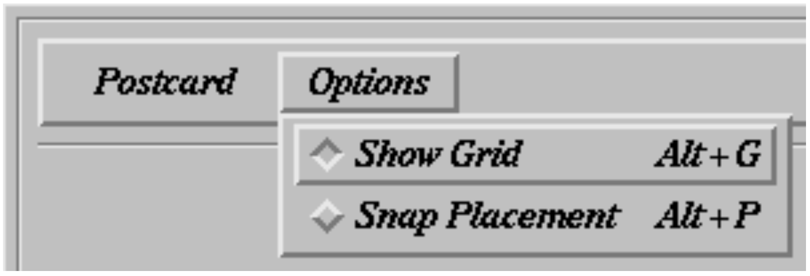


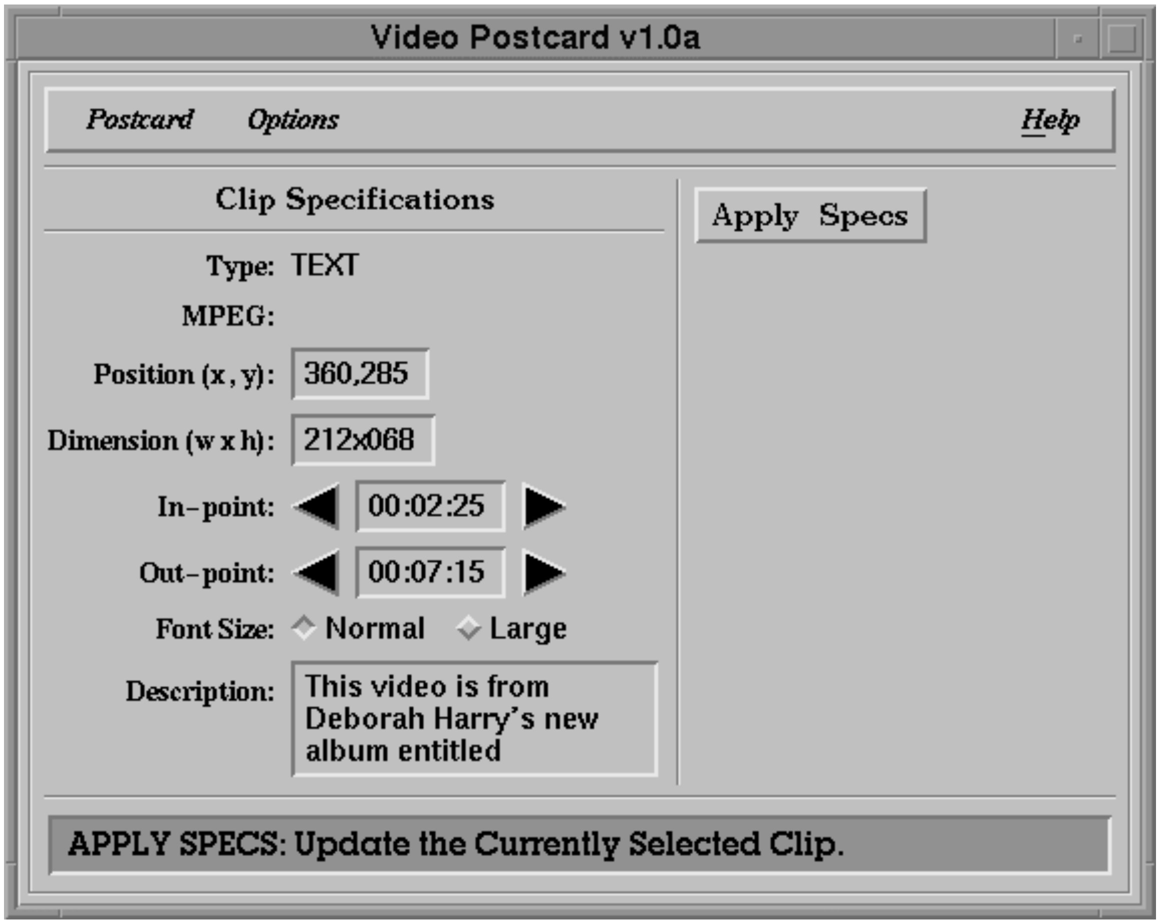
Figure 2: The options menu.

KEY	PURPOSE
v	Insert video
t	Insert text
d	Delete
c	Cut
p	Paste
r	Preview
meta-g	Show grid
meta-p	Snap placement
left-arrow	Decrease current time
right-arrow	Increase current time
shift left-arrow	Decrease total time
shift right-arrow	Increase total time
control left-arrow	Jump to frame zero
control right-arrow	Jump to last frame

Table 1: Shortcut keys.

Description on the *Clip Specifications* page (see figure 3). The specifications page allows the user to directly specify the position and dimensions of the currently selected clip (shown with a thick border), as well as its in and out points. After changes have been made, clicking on the *Apply Specs* button will update the clip on the canvas. For a text clip, Video Postcard will try to fit as much text as possible within the given dimensions, and word-wrap if the text does not fit on a single line. Figure 1 is an example of a very simple video postcard.

Editing features that are currently supported in Video Postcard are *Delete*, *Cut*, and *Paste*. Since there is no undo command, deleting the currently selected clip is a non-reversible operation. A cut, on the other hand, preserves the deleted clip, and is recoverable by doing a paste. The delete command, which can only be used when the currently selected clip is visible, is useful if the user wants to delete one or more clips before pasting the saved one at the same position. Pasting is effectively the same as directly modifying the in and out points of the cut clip, with its new in-point set to the current time, and its new out-point set to a time such that the original duration remains unchanged.



Clip Specifications

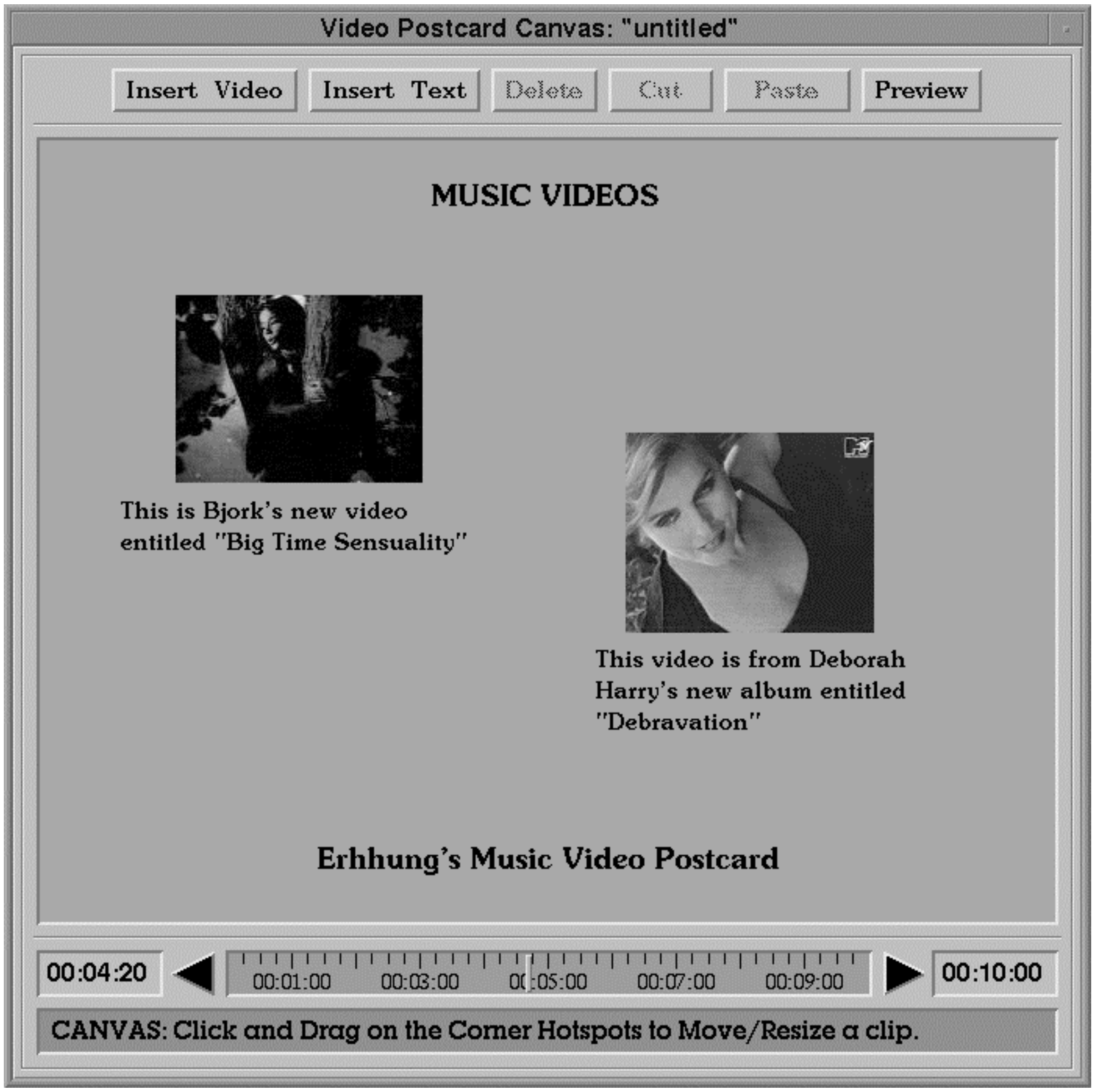
Type: TEXT
MPEG:
Position (x, y): 360,285
Dimension (w x h): 212x068
In-point: 00:02:25
Out-point: 00:07:15
Font Size: Normal Large
Description: This video is from Deborah Harry's new album entitled

APPLY SPECS: Update the Currently Selected Clip.

Figure 3: The clip specifications page.

Click on the *Preview* button to take a brief look at the current progress of the postcard. In preview mode, playback begins at the current time, and proceeds in real-time. This means that the application may have to skip frames in order to maintain the proper frame rate. In addition, grid lines, text frames, and all adjustment handles are removed during playback the only frames that remain are the video frames, which are still labeled with their MPEG filenames since no video is shown. Click on the *Preview* button again to stop the playback while it is in progress and return to edit mode.

To see the finished postcard in its entirety, select *View* from the *Postcard* menu, then choose one of the two subitems to indicate the desired playback mode. Video Postcard is currently unable to conduct formal viewing in real-time mode due to difficulties in dealing with the MPEG decoding routines. Hence, viewing is possible only in non-skipping mode. This means that a 5 second clip rated at 30 fps may only play for half its intended length if the CPU can only decode the data stream at 15 fps. Figure 4 is a snapshot of the video postcard shown in Figure 1 being played in non-skipping mode.



MUSIC VIDEOS

This is Bjork's new video entitled "Big Time Sensuality"

This video is from Deborah Harry's new album entitled "Debravation"

Erhhung's Music Video Postcard

CANVAS: Click and Drag on the Corner Hotspots to Move/Resize a clip.

Figure 4: Viewing a finished video postcard.

To save the finished postcard, select *Save* from the *Postcard* menu. If a filename has not already been given, the user will be prompted for the new filename; otherwise, he or she will be asked to confirm the ensuing overwrite operation. It is very important that the MPEG files stay in the same directory and their names be unchanged so that the application can find them later when the postcard is delivered to the addressee. Video postcard files are saved in ASCII format so they can be easily identified and modified. Using ASCII files will also simplify the software development process should more specifications be incorporated in the future. Since audio capability is not supported at this time, the addition of audio clips is an excellent example of this kind of upgrade.

It is now time to send a video postcard. Select *Send* from the *Postcard* menu to pop up the destination address prompt dialog. Here the user enters the standard Internet e-mail address of the recipient, which Video Postcard will attempt to verify by using the UNIX *finger* command. The reason for doing this is to avoid the possibility of having *mhmail* return to the sender tons of undeliverable messages if the address was incorrectly entered [2]. If verification passes, the application proceeds to prepare the postcard file and its associated MPEG files for delivery. If verification fails, the user can either enter a new address for Video Postcard to verify again, cancel the send operation, or click on the *Force Send* button to ignore the warning.

Since each MPEG file needs to be *uuencoded* into ASCII and *split* into 100-kilobyte blocks, Video Postcard performs these rather time consuming operations in the background and let's the user continue with other tasks. Periodically, the message bar will show the current status of the preparation process. When all is completed, a formal status report will be generated; this includes the number of messages that were sent, the time spent on each, etc.

On the receiver's side, the user can check for incoming video postcards by selecting *Scan* from the *Postcard* menu. The scanning process requires several steps, the first of which is invoking the mhmail command *inc* to incorporate any new mail into the user's *inbox* . The next step is parsing the subject line of each message in *inbox* which is numbered after the pointer *~/Mail/postcards/.mh_sequences*. Files which contain the identifier "VIDEO POSTCARD" on the subject line are then moved into the *postcards* folder, where parsing the *Sequence* field [3] in each determines the final assembling order. And in this order, all files are *uudecoded* to form the original MPEG files (component files are deleted to conserve disk space). The postcard file, which remains an ASCII file, is simply renamed to its proper name, with all header information kept as record. Once the scanning process is done, viewing the postcard is just a matter of selecting *Load* and *View* from the *Postcard* menu.

[2] In many instances the *finger* command will fail to locate the recipient even though the given address is valid because the *fingerd* program at the destination site is either not running or is refusing certain kinds of requests. Hence, a failed verification is by no means fatal: it is simply meant to alert the user and to ask him or her to double check the address before asserting the send operation.

[3] Video Postcard inserts a field named *S equence* in the body of each message at the time of delivery, containing the name of the postcard, a unique time stamp (as generated by the *ftim e()* function call) identifying all files which belong to that postcard, the MPEG filename (or ".header" if it is the postcard file), and the sequence number (eg. "01/04" meaning part 1 of 4).

Application Design

A major goal in the development of Video Postcard is increasing the readability and maintainability of the code through the use of procedural abstraction and specification, as well as structured data types. The application is written in C, but, nonetheless, tries to imitate the style of object-oriented programming by grouping the functions together with the data inside coherent structures. Furthermore, all functions include the *requires* and *effects* clause in their specifications, making it easier for future maintainers to modify and debug them if necessary. Modularity was especially driven by the desire to eventually replace the poorly designed MPEG routines, which sacrificed procedural abstraction and modularity for speed, with ones that better interface with the application.

Conclusion

As there are still quite a few program modules that need to be implemented or finished, such as routines to handle the sending, receiving, saving, and loading of postcards, there has been very little feedback regarding future enhancements at this time. The immediate goal is making these features available so that everyone can experience the entire process as described above. Although Video Postcard could be useful in its present state, it lacks one major component that was proposed in the original design: making the video postcard an interactive experience through the use of hyperlinked clips. This would make Video Postcard somewhat like the Mosaic browser for the World Wide Web, but the fundamental difference is that it does not depend on user intervention. No matter what the viewer chooses to do, the video postcard is always moving in time. And when Video Postcard is finally capable of delivering such an experience, we can really say that it has enhanced our personal communication skills.