2—1

S/W-Based Video Codec Systems for Intranet and Internet Multimedia Services

Yong Han Kim, Nam Ik Cho, Kichul Kim

School of Electrical Engineering
Seoul City University
90 Jeon Nong Dong, Dong Dae Moon Ku, Seoul 130-743, KOREA
Email : yhkim@tina.scu.ac.kr, nicho@tina.scu.ac.kr, kkim@scucc.scu.ac.kr

Abstract : This paper describes two different S/W-based video codec systems. One is a frame-based video codec with fixed structure based on ITU H.263 standard and the other an object-based video codec with flexible architecture based on ISO MPEG-4 standard currently under specification and planned to be finalized in 1998. These codecs are an experimental implementations for examining the feasibility of real-time and/or flexible S/W-based video codecs operating in intra and/or internetworking environments.

1 Introduction

Two different implementations of S/W video codecs are described in this paper. The purpose is to demonstrate the feasibility of S/W-based video codecs for intranet and/or internet multimedia applications.

First, computational complexity of H.263[1] is analyzed for estimating the system specifications suitable for real-time software video codecs. As a result, it is concluded that more than 4 frames of QCIF images can be coded and decoded using 90MHz Pentium desktop computer if we use 3 step search for ME/MC and there is no bottleneck for the data transfer from image grabber to the memory of the computer. Then we implement the H.263 based codec using 90 MHz Pentium notebook computers. PCMCIA image grabbers and sound cards are used as input devices, and wireless LAN cards are used for communication with mobility, which is one of the main advantages of the notebook computers.

The motivation of the second work in this paper is to demonstrate the feasibility of the platform-independent flexible architecture of video codecs. This type of codec architecture is currently under heavy study within one of the ISO projects called MPEG-4 [2]. In this paper the feasibility of the reconfigurable decoder architecture with downloading of new decoding tools is demonstrated by implementing the Java encapsulation[3] of the MPEG-supplied S/W decoders written in C language[4]. The particular implementation assumes one way communication such as the web browsing of MPEG-4 compressed video data although the full MPEG-4 specification will provide two way communication capabilities for user interactions.

A part of this work was supported by Electronics and Telecommunications Research Institute, Korea, through contract No. ETRI 96-51.

2 Video Phone for Notebook Computers

Most of application programs for desktop computers can be used for notebooks without modification. However, in case of video phone programs, it needs much modification because the video overlay or capture boards for notebooks are not so popular and have limitations on speed and space. Also, while desktop computers are usually wired on modem or Ethernet, the notebook computers need wireless communication channel for keeping the portability.

For the implementation of software video codec on notebook computers, efficient software modules for I/O and network are necessary in addition to the coder and decoder programs. Figure 1 shows the structure of the program modules required for the implementation of the software codec. In Fig. 1, the image grabber is the module for receiving NTSC signal and converting it into digital YUV format. We used NogaTech's PCMCIA Capture Vision for receiving image inputs, and used its 16 bit API (application program interface) for converting RGB into YUV format. The image coder/decoder are programmed based on the H.263 standards. The audio coder and decoder accommodates many kinds of speech compression algorithms. The data multiplexer/demultiplexer combines or separates the bit streams from audio and video coder/decoder. The output from the multiplexer enters into the packet driver modules and transmitted through wireless LAN cards. AT&T's 2.4GHz WaveLAN cards are employed in our system. The packet driver modules control the wireless LAN cards, and the communication protocol is based on the TCP/IP.

Since the 16 bit API is used for the control of image inputs, whereas the rest is programmed in 32 bit mode, the image grabber modules is not synchronized with the rest of the program. As a result, the coding efficiency is lower than expected. The system can code and decode 2 frames of sub-QCIF images per second, much less than the estimated frame rates. In future studies, we are going to develop 32 bit program for the image grabber or employ MFC (Microsoft Foundation Class) libraries.

3 Platform-Independent Codec Architecture

The basic architecture is described in Figure 2 The decoder can have 3 different types of decoding tools: resident, user-installed, and downloaded tools. Every MPEG-4 compliant decoders should be equipped with all the resident tools defined by MPEG-4. These are allowed to be platform-dependent. User-installed tools are those that can be installed by the users of the decoders and are the extensions to the resident tools. They are not necessarily platform-independent and some restrictions may apply to the versatility of the decoders. This means that the decoders avoid of the user-installed tools cannot decode the encoded bitstreams produced under the assumption of those tools. The downloaded tools are those that can be downloaded through the network. These tools should be platform-independent unless the encoder has all the executable binaries for different decoder platforms. The availability of the downloaded tools enables the encoder to define any new tools as desired. This results in flexibility and extensibility of the decoder, which differentiates the MPEG-4 decoder from the conventional schemes such as H.261, H.263, and MPEG-2.

According to the MPEG-4 Draft Specifications, a particular decoding mode or algorithm consists of a set of decoding tools. If the decoder is to operate in one of the selectable modes, it uses only the resident tools. This mode is called "Flex-0", which is quite similar to the profile formation in MPEG-2. If the encoder is to define a new decoding mode by reconfiguring the resident tools, the reconfiguration information should be delivered to the decoder as well as the compressed data. This mode is called "Flex-1." Finally the "Flex-2" permits downloading of new tools. The downloadable tools should be platform-independent modules. One approach to achieve such capability is to provide the tools in Java bytecodes that can be interpreted by any Java-enabled platforms.

Figure 3 illustrates the block diagram of the implementation that allows all the three "Flex" modes. The main program of the decoder is implemented using Java so that the downloaded tools can be configured. It is a "Java application" rather than a "Java applet." A web browser such as Netscape is used to browse the HTML files at the server, which has links to the MPEG-4 compressed bitstream files. If the client user click one of the links, the browser invokes a supplementary program such as a "plug-in" in case of Netscape and the plug-in initiates the MPEG-4 Java decoder process. The plug-in makes possible the in-screen display of the reconstructed video and provides the data transfer channel between the browser and the Java decoding process. This data transfer can be implemented through temporary files as well as "pipes." Note that for security reasons direct access from "Java applets" is not allowed to the client hard disks. The MPEG-4 encoded bitstream includes the configuration information as well as the downloaded tools, if any. All the downloaded tools are Java bytecodes so that any Java-enabled platform can use them after the on-line reconfiguration of the decoder functionality. The main program of the MPEG-4 decoder interprets the configuration information and call the appropriate decoding tools. If new tools are downloaded, the decoder main program places them in a predefined directory for further use in the decoding process. Right before the decompression, the decoder main process invokes another process that will display the reconstructed video frame by frame whenever a complete frame is decompressed.

The S/W-based MPEG-4 decoder is implemented on a IBM-compatible PC which is a network client. The server has MPEG-4 compressed bitstreams obtained by off-line encoding of test video material according to the MPEG-4 Verification Model[5]. For the test of "Flex-0" and "Flex-1" capability, two different quantization matrices are used at the choice of the encoder. And for the test of "Flex-2", one of the simplest tools is programmed in Java and downloaded within the bitstream. As a result, it has been demonstrated that the idea of flexible platform-independent decoder architecture is feasible.

4 Conclusions

The software codecs based on H.263 standards has been implemented using PCMCIA image grabbers, wireless LAN cards, and 90 MHz Pentium notebook computers for mobile video phones and other applications. They can code and decode more than 2 frames of sub-QCIF images per second. Since the image grabber software module is separated from others, the frame rate of the codec is much lower than the estimated. For further performance improvements, 32 bit interface program or program modules based on MFC (Microsoft Foundation Class) libraries is being developed.

The other contribution of this paper is in demonstrating the feasibility of the platform-independent decoder architecture that can execute downloaded decoding tools in a combination of the resident tools. Based on this basic architecture, further research is required for a full implementation of the MPEG-4 decoder.

References

[1] International Telecommunication Union, Line Transmission of Non-telephone Signals, Draft ITU-T Recommendation H.263, April 1995.

[2] ISO/IEC JTC1/SC29/WG11 N1401, "MPEG-4 MSDL Specification Version 1.3," Sep., 1996.

[3] Ken Arnold and James Gosling, *The Java Programming Language*, Addison-Wesley Publishing Co., 1996.

[4] ISO/IEC JTC1/SC29/WG11 MPEG96/M1089 "MoMuSys C implementation of the VM 2.2," July. 1996.

[5] ISO/IEC JTC1/SC29/WG11 N1277, "MPEG-4 Video Verification Model Version 3.0," July 1996.

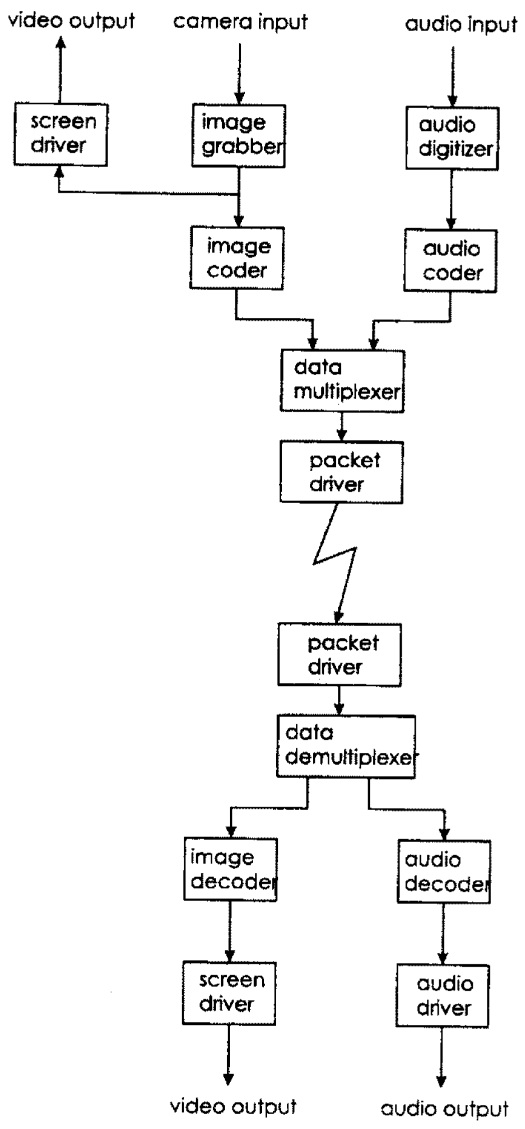

Figure 1: Structure of the software for video phone.

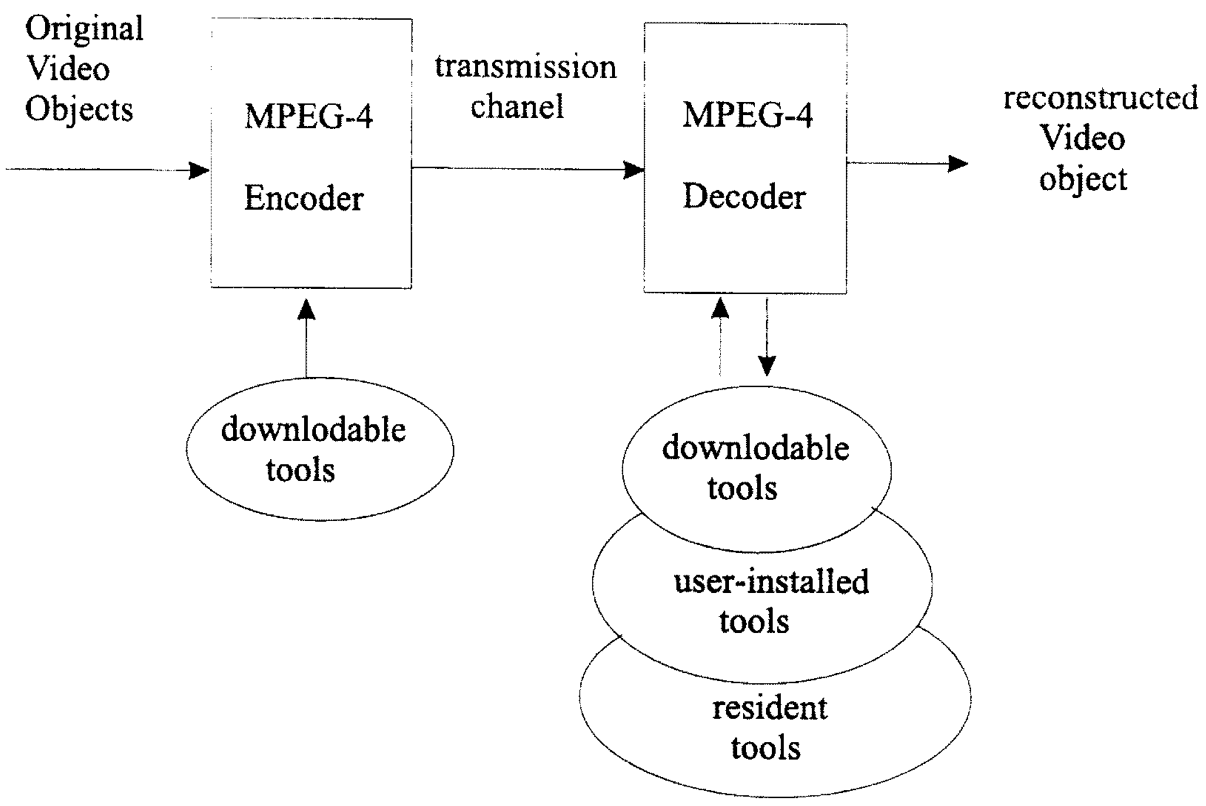

Figure 2: Basic architecture of the platform independent video decoder.

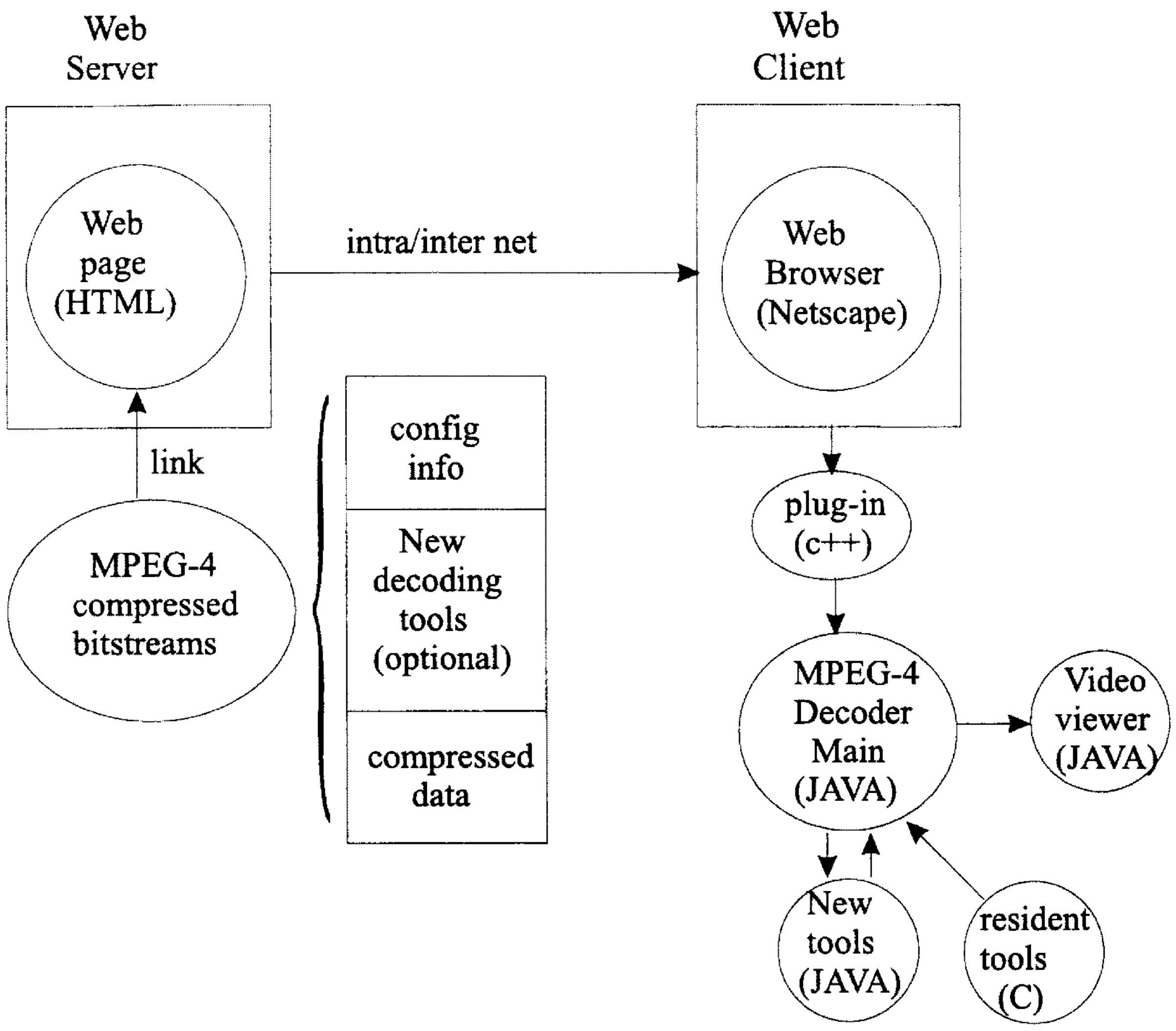

Figure 3: Block diagram of the implementation of the S/W-based MPEG-4 decoder.