VuSystem Performance Measurements *

Christopher J. Lindblad †

Telemedia Networks and Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology

Abstract

In this paper I discuss some performance measurements made of the VuSystem, a programming system for the software-based processing of audio and video data. The VuSystem is designed to run on ordinary Unix workstations with no specific support for the manipulation of multimedia data. Measurements made of processing times of representative filter modules demonstrate the viability of the approach.

Introduction

There is a class of multimedia applications in which the computer performs tasks requiring the direct processing of multimedia data, as well as the capture, storage, retrieval, and display tasks of traditional multimedia applications. Members of the class are best called *computer-participative* multimedia applications, because in them the computer directly participates in the interpretation of the multimedia data. These applications require more support than is provided by traditional multimedia toolkits. They require an extensible in-band media processing component.

To support their development, I designed and built the VuSystem [1, 2], a prototype software environment for applications that directly manipulate temporally sensitive data. The system provides simple scheduling and resource management functions to allow intelligent media-processing applications to run on ordinary Unix workstations. Because it includes an easy-to-program extensible in-band processing component, it is uniquely suited for rapid development of applications that perform intelligent processing of live media.

VuSystem applications [3] combine intelligent media processing with traditional capture and display. Some process live video for more responsive human-computer interaction. Others digest television broadcasts in support of content-based retrieval. Both classes demonstrate the utility of network-based multimedia systems that deliver audio and video data all the way to the application.

Following the construction of the VuSystem prototype, I performed a series of experiments that demonstrate the system can manipulate digital video with low overhead and high throughput. In this paper, I report some of the measurements made.

---

*This research was supported by the Advanced Research Projects Agency of the Department of Defense, monitored by the United States Air Force (AFSC, Rome Laboratory) under contract No. F30602-92-C-0019, and by a grant from Nynex.

†The author can be reached at: MIT Laboratory for Computer Science, Room 504, 545 Technology Square, Cambridge, MA 02139; Tel: +1 617 253 6042; Email: cjl@lcs.mit.edu.

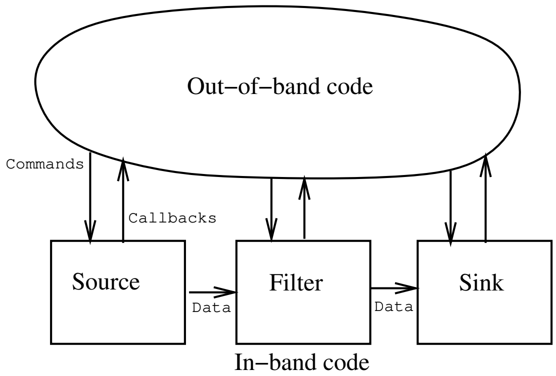

Figure 1: The VuSystem Approach.

The VuSystem

VuSystem applications include The Room Monitor, The Whiteboard Recorder, and The Video Rover. They provide more responsive human-computer interaction through the intelligent processing of live video. Other applications that manipulate pre-recorded video include The News Browser, The Joke Browser, and The Sports Highlight Browser. They demonstrate the practicality of automatic extraction to support content-based retrieval of produced video.

VuSystem applications have two components: one which does traditional *out-of-band* processing and one which does *in-band* processing. Out-of-band processing is that which performs the event-driven functions of a program. In-band processing is that performed on every video frame and audio fragment. In-band code is more elaborate in the VuSystem than in traditional multimedia systems such as Apple Quicktime or Microsoft Video for Windows because VuSystem applications perform sophisticated analysis of their input media data.

In the VuSystem, the in-band processing component is arranged into processing *modules* that pass dynamically-typed data *payloads* through input and output *ports*. The out-of-band component of the VuSystem is programmed in the Tool Command Language, or Tcl [8], an interpreted scripting language. Application code written in Tcl is responsible for creating and controlling the network of in-band media-processing modules, and controlling the graphical user-interface of the application.

The VuSystem is implemented on ordinary Unix workstations as a program that interprets an extended version of Tcl. In-band modules are implemented as C++ classes and are linked into this Tcl shell. Simple applications that use the default set of in-band modules are written as Tcl scripts. More complicated applications leverage customized modules that are linked into the shell.

VuSystem programs have a *media-flow* architecture: code that directly processes temporally sensitive data is divided into processing *modules* arranged in data processing *pipelines*. This architecture is similar to that of some visualization systems [9, 10], but is unique in that all data is held in dynamically-typed time-stamped *payloads*, and programs can be reconfigured while they run. Timestamps allow for media synchronization, and dynamic typing and reconfiguration allows programs to change their behavior based on the data being fed into them.

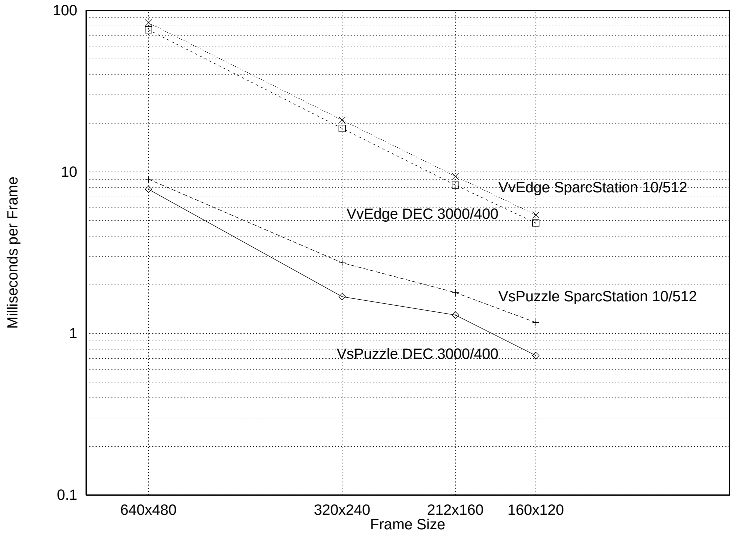

Figure 2: Processing Times of Representative Filter Modules.

frame size	puzzle frames/sec	edge frames/sec
640x480	12	6.67
320x240	30	25
212x160	15	15
160x120	30	30

Table 1: Rates of VuSystem Applications.

Performance

To verify that the overhead of the module data protocol is low, I measured the amount of time a simple transparent filter took to process a payload. It was approximately 12 microseconds on the Sun SparcStation 10/512 and approximately 3 microseconds on the Digital DEC 3000/400. Similarly, to verify that the run-time scheduler has low overhead, I measured the amount of time a mimimum filter takes to process a payload. It was approximately 150 microseconds on the Sun SparcStation 10/512 and approximately 115 microseconds on the Digital DEC 3000/400.

To demonstrate that media processing modules in the VuSystem are able to perform their functions with perceptual-time granularity, I measured the amount of time two representative filter modules took to process a video frame (Figure 2). On both the Digital DEC 3000/400 and the Sun SparcStation 10/512, a filter module that rearranges video frames for a puzzle program can scramble a 320 by 240 pixel, or half-sized, frame in approximately 2.5 milliseconds. An edge-detecting filter module can highlight edges in a half-sized frame in approximately 20 milliseconds. These times indicate that elaborate pixel-based operations can be performed efficiently on standard computer workstations.

I measured the total system throughput of two VuSystem programs based on two representative filter modules (Table 1). The video puzzle example application can process half-size live video at fully 30 frames per second with system capacity to spare. The edge highlighting application can process 25 frames per second of half-size live video.

Conclusion

In this paper I briefly discussed some performance measurements made of two VuSystem filter modules on two popular Unix workstation models. These measurements hint that it is possible to perform in-band media processing and still retain the temporal sensitivity that multimedia requires. VuSystem applications that perform visual processing can easily do so at 15 half-resolution frames per second. This is an acceptable level of performance for today, and will improve with advances in workstation technology as the system is portable.

References

[1] C. J. Lindblad, D. J. Wetherall, D. L. Tennenhouse, "The VuSystem: A Programming System for Visual Processing of Digital Video," *Proceedings of ACM Multimedia 94*, October 1994.

[2] C. J. Lindblad, "A Programming System for the Dynamic Manipulation of Temporally Sensitive Data," MIT/LCS/TR-637, MIT Laboratory for Computer Science, Cambridge, MA, August 1994.

[3] C. J. Lindblad, D. J. Wetherall, W. F. Stasior, J. F. Adam, H. H. Houh, M. Ismert, D. R. Bacher, B. M. Phillips, D. L. Tennenhouse, "ViewStation Applications: Intelligent Video Processing Over a Broadband Local Area Network," *Proceedings of the 1994 USENIX Symposium on High Speed Networking*, August 1994.

[4] D. L. Tennenhouse, J. Adam, D. Carver, H. Houh, M. Ismert, C. Lindblad, W. Stasior, D. Wetherall, D. Bacher, and T. Chang, "A Software-Oriented Approach to the Design of Media Processing Environments," *Proceedings of the International Conference on Multimedia Computing and Systems*, May 1994.

[5] J. F. Adam, H. H. Houh, M. Ismert, and D. L. Tennenhouse, "A Network Architecture for Distributed Multimedia Systems," *Proceedings of the International Conference on Multimedia Computing and Systems*, May 1994.

[6] W. Stasior, "Visual Processing for Seamless Interactive Computing," *The ViewStation Collected Papers*, MIT/LCS/TR 590, MIT Laboratory for Computer Science, Cambridge, MA, November 1993.

[7] J. F. Adam, "The Vidboard: A Video Capture and Processing Peripheral for a Distributed Multimedia System," *Proceedings of the ACM Multimedia Conference*, August 1993.

[8] J. K. Ousterhout, "Tcl: An Embedded Command Language," Computer Science Division (EECS), University of California, Berkeley, CA, January 1990.

[9] C. Williams and J. Rasure, "A visual language for image processing," *IEEE Computer Society Workshop on Visual Languages*, Skokie, Illinois, 1990.

[10] C. Upson, T. Faulhaber, Jr., D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, A. van Dam, "The Application Visualization System: A computational environment for scientific visualization," *IEEE Computer Graphics and Applications*, 30-42, July 1989.