To appear in SMPTE Journal, Dec. 1995

Object-Oriented Television

By V. Michael Bove, Jr.
Media Laboratory, Massachusetts Institute of Technology
Room E15-324, 20 Ames Street
Cambridge MA 02139 USA
(617) 253-0334, Fax (617) 258-6264
Internet: vmb@mit.edu

ABSTRACT

In search of more compression, researchers have recently sought to describe digital video of real scenes not as sequences of frames but rather as collections of objects that are rendered and combined according to scripting information. Depending upon the application and the scene analysis tools available, representations may range from two-dimensional layers to full three-dimensional computer-graphics-style data bases. The significance of these more meaningful representations goes beyond compression, however, enabling new forms of interactivity and personalization, as well as new degrees of freedom in post-production. This paper proposes a computational framework for a television receiver that can handle digital video in forms from "traditional" motion-compensated transform coders to sets of three-dimensional objects and discusses the requirements for a scripting language to control such a receiver. It is also noted that the concept of scalability can be expanded to include "intelligently resizable video," where the originator of a video sequence can specify how the scene is to be composed and cut for displays of differing sizes and aspect ratios.

INTRODUCTION

Television's technological history to date is closely tied to image sequences. A camera projects a continuously moving, three-dimensional scene onto an image plane and samples it on a spatiotemporal grid. The primary goal of television system design has been to store and transmit the image information so that — given a constrained channel bandwidth — the receiver reproduces, as accurately as possible, the sampled light intensities on the image plane of the camera.

Digital video systems can, of course, capture, store, transport, and reproduce images more efficiently and accurately than the analog systems they are now replacing. But since powerful computational devices now mediate between the camera and the display, digital video also points toward an alternative model for a television system; perhaps such a system should be more appropriately thought of as capturing sufficient information about a moving scene so that the receiving device can synthesize imagery in an appropriate fashion for the size, aspect ratio, frame rate, resolution, and dimensionality of the display, as well as the specific requirements of the viewer.

An initial step in this direction came with the ideas of scalability and open architecture,[1-5] where the intent was to produce a digital signal representation that allowed different displays to show images at a variety of resolutions and frame rates. In essence, receivers were not to reproduce the samples taken by a camera at a particular spatial resolution and frame rate, but rather were to try to approximate the samples that would have been taken had the continuous image been sampled on various different grids. As digital channels proliferate and consumer video viewing equipment expands beyond the traditional TV set to include projectors, panels, portable communicators, resizable windows on personal computers, and perhaps other devices, such flexibility becomes not just important but mandatory. But the intent of scalable, open architecture video systems is to show precisely the same pictures — optimally resampled — on all displays. Composing and editing moving pictures to be attractive and understandable under all the anticipated viewing circumstances is nearly impossible, and it may be appropriate to broaden the concept of scalability to include somewhat different imagery across diverse displays.

Image-based coding algorithms, whether interoperable or not, incur fundamental compression inefficiencies in failing to take into account the actual structure of the scene represented by the video frames. In essence, their model of the the information to be coded — square arrays of pixels that translate in a plane — is not always a good match to what is happening in front of a camera. The model mismatch likewise restricts degrees of freedom in display, and also minimizes the interactivity and automated scene-understanding options. Such considerations underlie the recent interest in representations that are more physically and semantically meaningful than sequences of ordinary images (or motion vector arrays and transform-coded error signals). Authors have used differing terms — structured video, model-based video, or analysis-synthesis video — but all refer to the representation of moving scenes in terms of component objects that are assembled according to scripting information to produce images for viewing. Object-oriented video coding methods can range from variations on motion-compensated coding, in which camera images are simply segmented into regions or layers based upon motion (perhaps in combination with other cues like color and texture),[6] to the fitting of three-dimensional object models to image sequences.[7] The added compression efficiency of structured video generally comes about because the more accurate transmitted model and the receiver's greater computational ability permit better prediction. In layered or region-segmented 2-D coders, for example, the transmitted motion parameters are intended to be a more correct approximation for the image-plane projections of real-world objects than would result from (x, y) vectors computed on an arbitrary square grid. Added memory in the decoder also eliminates the need to retransmit information about occluded and revealed regions.[8] Yet more computationally intensive algorithms can, under certain circumstances, estimate camera or object motion in space.[9]

Decoding some of the more sophisticated representations is computationally equivalent to synthetic computer graphics rendering — the major research challenge lies largely in the encoding, which involves the application of machine vision techniques. But while scene-analysis methods continue to be developed, in many cases the needed information may already exist. Data on the approximate shapes and arrangements of objects and on camera motions might also be provided by computer previsualization tools used in program production;[10] this might prove useful in coding or supporting viewer interaction or personalization. Many video editing systems now permit effects such as compositing. And increasingly, video content is created on computer-graphics systems.

Even if the structured scene representation is used only within the studio, new production and post-production possibilities emerge (see, for instance, Reference [11]). Our research, however, looks into transmitting the scene description to the receiver.

OBJECTS AND OPERATIONS

The intent of the research described herein is not to dictate a single algorithm for handling video, nor even to establish how to determine the "best" description for a particular application or moving scene. Transform-based encoders will soon be ubiquitous and inexpensive, and they involve a short and deterministic encoding delay. It is likely that JPEG, MPEG, and the like will be with us for a long time. Thus, any system we consider designing has to be able to deal with these "traditional" digital video representations as well as higher-level ones.

Given a sufficiently powerful and flexible decoder, the way in which a scene is described, and the forms of the constituent objects, represent an originator-specified tradeoff among a number of considerations:

- *Interactivity/personalization:* Clearly, the more individually-described objects in the scene, and the more the representation is like a computer-graphics data base, the more presentation options are enabled. Conversely, the originator of a program can force all viewers to see precisely the same presentation by appropriate design of the script and selection of objects.

- *Compression efficiency:* Exactly how the choice of representation affects the reconstructed image quality at a given bit rate remains an active research area in analysis/synthesis video. In any event, where image quality is the most important consideration, a flexible decoding pipeline permits an encoding process to select the most appropriate representation, even on a scene-by-scene or object-by-object basis.

- *Analysis ability:* More sophisticated scene descriptions require more computation, and indeed some scenes simply may not lend themselves to high-level descriptions given current analysis methods and encoding hardware capabilities. If the desire for real-time encoding and transmission is foremost, ordinary 2-D motion-compensated image sequences (or just sequences of interframe-coded images) will be sufficient and appropriate.

In order to unify several current research projects on higher-level moving scene descriptions, to explore the computational and scripting requirements of such representations, and to enable prototyping of services and applications, we have begun examining frameworks for object-oriented video decoders. One possibility for creating a very open decoding and display environment might be based upon a very powerful digital signal processing chip (providing at least two orders of magnitude more operations per second than those currently available) running a standardized operating system; another strategy is to use electrically programmable logic devices so that specialized hardware functionality can be downloaded like software. In this paper, we will instead attempt to identify a common core of operations and incorporate these into a flexible — if not totally reprogrammable — processing pipeline capable of handling a broad range of image and scene representations.

To design our decoding pipeline, we should examine the different types of information (the "objects") that constitute the compressed data resulting from various types of algorithms. For example, a hybrid predictive coder can be thought of as producing three kinds of objects: intracoded frames, arrays of motion vectors, and error signals, where the first and the third are quantized and statistically-coded in the transform domain for coding efficiency. The bitstream information identifying a set of data as, for instance, an I-frame, might be considered a very simple sort of script. Similarly, a layered two-dimensional coder produces a series of independent objects (which might be represented as polygons or other mathematically-described shapes, onto which an array of pixels is mapped, or as transform-coded rectangular arrays perhaps with transparent regions — in the latter