NATURAL AND SYNTHETIC VIDEO IN MPEG-4

Jörn Ostermann and Atul Puri

AT&T Labs - Research
100 Schultz Drive
Red Bank, NJ 07701, USA
Email: {osterman, apuri}@research.att.com

ABSTRACT

The ISO MPEG committee, after successful completion of the MPEG-1 and the MPEG-2 standards, has recently completed the Committee Draft for MPEG-4, its third standard. MPEG-4 is designed to be an object-based standard for multimedia coding. The visual part of the standard specifies coding of both natural and synthetic video. The MPEG-4 visual standard supports coding of natural video not only in a conventional manner (using frames) but also as a collection of arbitrary shape objects (using video object planes). Further, it supports functionalities such as spatial and temporal scalability, both conventional and with arbitrary shape objects. It also supports error resilient coding for delivery of coded video on error prone channels. MPEG-4 visual standard also supports coding of synthetic video which includes still texture maps used in 3D graphics models, mesh geometry for object animation, and parameters for facial animation.

1. INTRODUCTION

The ISO MPEG committee, after successful completion of the MPEG-1 and the MPEG-2 standards, has recently [1,2] completed the Committee Draft for MPEG-4, its third standard. The current specification is referred to as version 1 and is expected to undergo clarification and minor refinement leading to the final stage as the International Standard in January 1999. The MPEG-4 standard is designed to address the requirements of a new generation of highly interactive multimedia applications while supporting traditional applications as well. Such applications, in addition to efficient coding, also require advanced functionalities such as interactivity with individual objects, scalability of contents, and a high degree of error resilience. The MPEG committee is addressing these new generation applications and their requirements via MPEG-4, an object-based standard for multimedia coding.

The MPEG-4 standard, similar to its predecessors consists of a number of parts, the primary parts being systems, visual and audio. The visual part of MPEG-4 includes coding of both natural and synthetic video. in addition to the tools included in the visual part of MPEG-4, some tools of MPEG-4 systems are also relevant to our discussion as they offer additional capabilities to tools included in the visual part of the standard. In this paper we present an overview of the visual part of the MPEG-4 standard with focus on tools enabling new functionalities such as content based interactivity, content-based scalability, improved robustness in error prone environments and increased coding efficiency. Further, we also discuss tools for coding and animation of synthetic video, in particular, object mesh and facial wireframe models.

We now discuss the organization of this paper. In section 2, a high level coding structure and the coding tools for natural video supported are discussed. Section 3 describes the coding tools for synthetic video. Finally, section 4 summarizes the discussed attributes of the MPEG-4 visual standard

2. CODING OF NATURAL VIDEO

In this section we discuss tool and techniques for coding of natural video supported in MPEG-4 visual standard. Although MPEG only standardizes the decoding process and the bitstream syntax, for the purpose of explanation it is necessary to discuss the coding process.

2.1 Basic Video Coding

Many of the MPEG-4 functionalities require access not only to entire sequence of pictures, but to an entire object, and further, not only to individual pictures, but also to temporal instances of these objects within a picture. A temporal instance of a video object can be thought of as a snapshot of arbitrary shaped object that occurs within a picture, such that like a picture, it is intended to be an access unit, and, unlike a picture, it is expected to have a semantic meaning. The concept of Video Objects (VOs) and their temporal instances, Video Object Planes (VOPs) is central to MPEG-4 video. A VOP can be fully described by its texture variations (a set of luminance and chrominance values) and its shape.

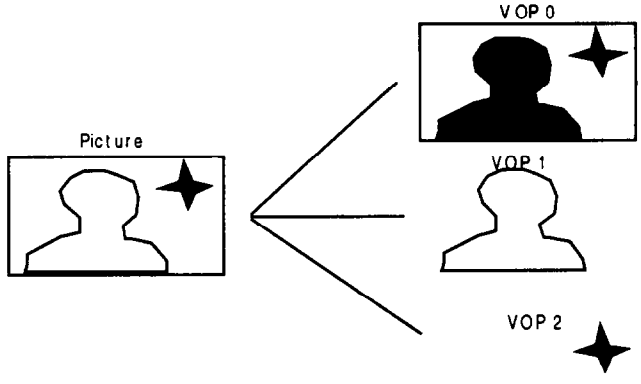

Figure 1 Semantic segmentation of video frame into VOPs

In Figure 1, we show the decomposition of a picture into a number of separate VOPs. The scene consists of two objects

(head and shoulders view of a human, and a logo) and the background. The objects are segmented by semi-automatic or automatic means and are referred to as VOP1 and VOP2, while the background without these objects is referred to as VOP0. Each picture in the sequence is segmented into VOPs in this manner. Thus, a segmented sequence contains a set of VOP0s, a set of VOP1s and a set of VOP2s, in other words, in our example, a segmented sequence consists of VO0, VO1 and VO2.

To consider how MPEG-4 video coding takes place consider a sequence of VOPs. Now, extending the concept of intra (I-) pictures, predictive (P-) and bidirectionally predictive (B-) pictures of MPEG-1/2 is to VOPs, I-VOP, P-VOP and B-VOP result. If, two consecutive B-VOPs are used between a pair of reference VOPs (I- or a P-VOPs), the resulting coding structure is as shown in Figure 2.

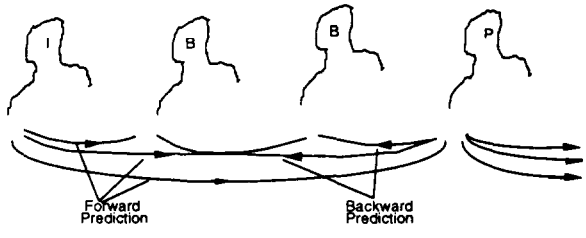

Figure 2 An example prediction structure using I, P and B-VOPs

In Figure 3 we show the internal structure of an encoder for which codes VOs of a scene. Its main components are: Motion Coder, Texture and Shape Coder. The Motion Coder uses macroblock and block motion estimation and compensation, similar to H.263 and MPEG-1/2 but modified to work with arbitrary shapes. The Texture Coder uses block DCT coding based on H.263 and MPEG-1/2 but much better optimized, further it is also adapted to work with arbitrary shapes. An entirely new component is Shape Coder. The partial data of VOs (such as VOPs) is buffered and sent to Systems Multiplexer.

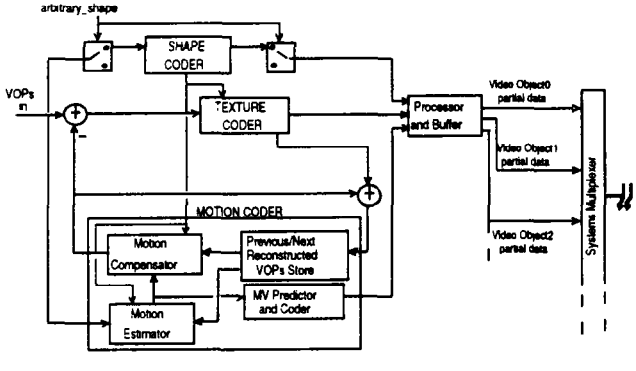

Figure 3 Structure of video object encoder

2.1.1 Shape Coding

Since, content based access requires that each of the video objects in a scene be coded independently, not only motion and texture but also shape of object needs to be coded. Two types of shape can be distinguished: binary shapes (outline of an object) and grayscale alpha maps (variation of the transparency of an object within its outline). While the method for binary shape coding has been finalized, work on finalization of grayscale shape coding is in progress.

For binary shape coding, a rectangular bounding box enclosing the arbitrary shape VOP is formed such that its horizontal and vertical dimensions are multiples of 16 pels. Each block of size 16x16 pels within this bounding box is called binary alpha block (BAB). Three types of binary alpha blocks are distinguished and signaled to the decoder: transparent block (outside the object), opaque blocks (entirely inside the object) and boundary blocks (cover part of the object as well as part of the background). For boundary blocks a context-based shape coder has been developed.

For intra mode, the transparency of each pel within a boundary block is coded using a fixed arithmetic coder with a context of 10 pels (Figure 4a).

For inter mode, the shape of the current BAB is predicted from the previous frame using integer pel motion compensation for 16x16 blocks. For each boundary block, a shape motion vector is encoded predictively by choosing the first suitable vector from neighboring blocks. In case the motion compensated prediction error is large, the transparency of each pel in the BAB is arithmetically encoded using a fixed template of 9 pels. The template considers the motion compensated location in the previously coded VOP and the already coded pels of the current BAB (Figure 4b).

For lossy shape coding and rate control purposes, the shape of a macroblock can be subsampled by a factor of 2 or 4 prior to coding. The decoder decodes the shape parameters and upsamples the shape accordingly.

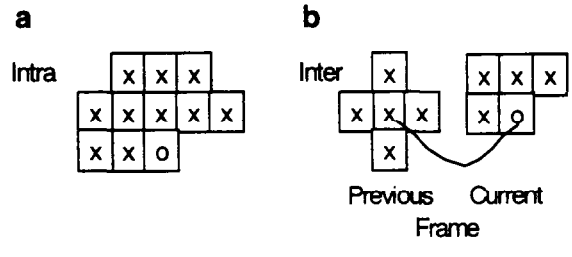

Figure 4 Templates for defining the context of the pel to be coded (o), a)the intra mode context, b) the context when coding in inter mode.

2.1.2 Motion Coding

The motion coder consists of a block motion estimator, block motion compensator, VOPs store and motion vector predictor and coder. In case of P-VOPs, the motion estimator computes motion vectors using current VOP and the previous reconstructed VOP (from VOP store). In case of B-VOPs, motion estimator

computes motion vectors using the current VOP and the temporal previous reconstructed VOP, as well as the temporally next reconstructed VOP, both the reconstructed temporally previous and next VOPs are available in the VOP store. Motion vector predictor generates motion vector difference signal for coding by motion vector coder. For each macroblock motion vectors can be computed on a 16x16 or 8x8 block basis similar to H.263. Furthermore, motion compensation modes such as overlapped block prediction and use of unrestricted range are allowed. Since, motion compensation is performed on blocks whereas objects of arbitrary shapes may have to be coded, padding is needed to extend the active area of arbitrary shape to block boundary.

For motion compensated prediction of current VOP, the reference VOP is motion compensated using overlapped block motion compensation. In order to guarantee that every pel of the current VOP has a value to predict, some or all of the boundary and transparent blocks of the reference VOP have to be padded. Boundary blocks are padded using repetitive padding repeating: first boundary pels are replicated in horizontal direction, then in vertical direction making sure that if a pel can be assigned a value by both padding directions it is assigned an average value. Since this repetitive padding puts a computational burden on the decoder, a simpler padding is used in a second step. Transparent macroblocks bordering boundary blocks are assigned an average value determined by the neighboring pels.

2.1.3 Texture (DCT) Coding

The texture coder codes the luminance and chrominance variations of blocks forming macroblocks within a VOP. The block that lie inside the VOP boundary are coded using 8x8 DCT coding similar to that in H.263 but optimized for MPEG-4. The blocks that lie on the boundary are first padded and then DCT coded similar to the blocks that lie inside the VOP. The remaining blocks are transparent and are not coded. The sequence of operations in the texture coder are: DCT of original or prediction error blocks, quantization of block of DCT coefficients, scanning of quantized coefficients, and finally, variable length coding of quantized coefficients. Each block belonging to non-intra coded macroblocks, uses H.263 like coding. However, for efficient coding of intra blocks a number of new tools such as adaptive DC coefficient prediction, nonlinear quantization of DC coefficients, adaptive AC coefficient prediction and adaptive scanning of DCT coefficients have been developed leading to notable improvements.

2.2 Scalable Video Coding

Scalability of video is the property that allows a video decoder to decode portions of the bitstream to generate decoded video of quality commensurate with the amount of data decoded. Scalable coding offers a means for decoders to better match time varying computational/memory resources to the data to be decoded. Scalability inherently also offers increased error resilience. MPEG-4 video supports both the traditional (frame based) scalability as well as content (VOP) based scalability. Content based scalability allows improved tradeoffs since the scalable representation of an object selected for decoding can be better matched to the requirements of the application and constraints of available bandwidth/decoding resources.

MPEG-4 video supports both spatial and temporal scalability. Temporal scalability offers decoders a means to increase temporal resolution of decoded video using decoded enhancement layer(s) when combined with decoded base layer. Spatial scalability offers decoders a means to decode and display the base layer or the enhancement layer; for example the base layer may use one-quarter of the resolution of the enhancement layer. MPEG-4 general scalability framework employs modified B-VOPs that only exist in enhancement layer to achieve both temporal and spatial scalability. The modified enhancement layer B-VOPs use the same syntax as normal B-VOPs but with semantics modified to allow a number of interlayer prediction structures needed for scalable coding.

MPEG-4 video version 1 supports the traditional frame based temporal and spatial scalabilities in addition to object based temporal scalability; object based spatial scalability is expected to be supported in version 2.

2.3 Sprite Coding

Sprites – also known as mosaic or world image – are suited for describing rigid objects that can be rendered by applying at each time instant an affine transform to the sprite image in order to generate the approximate rendering of object. In MPEG-4 video, sprites are employed as a representation of background image which can be used for very efficient coding of scenes involving camera pan and zoom. In such cases, sprites are much larger than the size of image being coded and thus correspondingly require more memory.

2.4 Robust Video Coding

Truly robust video coding requires a diversity of strategies such as error localization, error recovery and error concealment. MPEG-4 systems offers capability of assigning priorities to objects which can be used to provide higher priorities or graceful degradation of some objects. Furthermore, MPEG-4 video offers coding tools to reduce error propagation and facilitate data recovery. For instance, three tools - resynchronization, data partitioning, and reversible VLCs are supported. To facilitate resynchronization MPEG-4 video allows insertion of resync markers at resynchronization points within each VOP; resync markers are unique 17-bit codes that normally can not be emulated. When higher error resilience is needed, MPEG-4 video coded data is arranged as video packets, the header of such packet contains not only resync marker but also macroblock number, quantizer value and (optional) timing information. Further, data partitioning if employed allows shape, motion and texture data to be separated within a packet. For instance, coded video data of a packet is arranged as shape data followed by a shape marker followed by macroblock mode and motion vector data followed by a motion marker followed by coded texture DCT data. Additionally, reversible VLCs when used for entropy coding of DCT coefficients can offer partial recovery from errors. Lastly, MPEG-4 video, does not standardize error concealment.

3. CODING OF SYNTHETIC VIDEO

The MPEG-4 systems standard allows definition of animated synthetic 2D and 3D objects and scenes including animated text,

2D and 3D meshes as well as synthetic faces, whereas the MPEG-4 visual standard includes tools for animation of objects using 2D meshes, facial animation and coding of still texture maps.

3.1 Object Mesh Coding

Mesh based representation of an object is useful for a number of functionalities such as animation, content manipulation, content overlay, merging natural and synthetic video and others. To generate a 2D mesh based representation of a natural or synthetic video object, at its first appearance in the scene, it is tasselated with triangular patches resulting in an initial 2D mesh. The node points of this initial mesh are then tracked as the VOP moves in the scene. The 2D motion of video object can thus be compactly represented by the motion vectors of the node points of the mesh and motion compensation can be achieved by warping of texture map corresponding to patches by affine transform from a VOP to next. Texture used for mapping on to object mesh models or facial wireframe models are either derived from video (coded with DCT) or still images (coded with wavelet transform, to be discussed.

3.2 Face Animation

An MPEG-4 terminal supporting facial animation is expected to include a default face model. The systems part of MPEG-4 provides means to customize this face model by means of face definition parameters (FDP) or to replace it with one downloaded from the encoder. The definition of a scene including 3D geometry and face model can be sent to the receiver using Binary Format for Scenes (BIFS). The visual part of MPEG-4 defines how to animate these models.

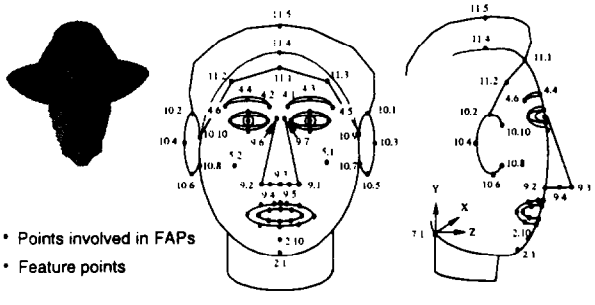

Figure 5 Feature points used for facial animation

Three groups of facial animation parameters (FAP) are defined. First, for low-level facial animation, a set of 66 FAPs is defined. These include head and eye rotations as well as motion of feature points on mouth, ear, nose and eyebrow deformation (Figure 5). Since these parameters are model independent their amplitudes are scaled according the proportions of the actual animated model. Second, for high-level animation, a set of primary facial expressions like joy, sadness, surprise and disgust are defined. Third, for speech animation, 14 visemes define mouth shapes that correspond to phonemes. Visemes are transmitted to the decoder or are derived from the phonemes of the Text-to-Speech synthesizer of the terminal.

The FAPs are linearly quantized and entropy coded using arithmetic coding. Alternatively, a time sequence of 16 FAPs can also be DCT coded. Due to efficient coding, it takes only about 2 kbit/s to achieve lively facial expressions.

3.3 Still Texture Coding

For coding still texture maps, Discrete Wavelet Transform (DWT) coding was selected for the flexibility it offered in spatial and quality scalability (while maintaining good coding performance). It is envisaged that applications involving interactivity with texture mapped synthetic scenes require continuous scalability. In DWT coding, texture map image is first decomposed using a 2D separable decomposition using Daubechies (9,3)-tap biorthogonal filter. Next, the coefficients of the lowest band are quantized, coded predictively using implicit prediction (similar to that used in intra DCT coding) and entropy coded using arithmetic coding. This is followed by coding of coefficients of higher bands by use of multilevel quantization, zero-tree scanning and arithmetic coding. The resulting bitstream is flexibly arranged allowing a large number of layers of spatial and quality scalability to be easily derived.

4. DISCUSSION

We have reviewed the MPEG-4 visual standard which supports coding of both natural and synthetic video objects. For the case of natural video, a scene is coded as composed of individual objects. Moreover, traditional video (frame-based) coding is also supported as a special case. Natural video objects are coded using block-based motion compensation, shape coding using context information and arithmetic coding, and texture coding using block DCT. Temporal and spatial scalability of video objects is also supported. Using error resilience tools, coding can be made highly robust. Thus, for natural video, a number of object based functionalities are enabled while maintaining good coding efficiency. Further, general synthetic video objects can be animated/manipulated using mesh based coded representation while an efficient mechanism to represent/animate faces is also included. Still texture maps used in object/face animation are coded by wavelet coding. Thus, a number of functionalities utilizing animation of faces and generic objects become possible leading to new applications in communication and entertainment.

5. REFERENCES

[1] ISO/IEC JTC1/WG11 Text of Visual CD, Nov. 1997.

[2] ISO/IEC JTC1/WG11 Text of Systems CD, Nov. 1997.

[3] N. Brady, F. Bossen, N. Murphy, "Context-based arithmetic encoding of 2D shape sequences", Special session on shape coding, ICIP 97, Santa Barbara, 1997.

[4] A. Puri and A. Eleftheriadis, "MPEG-4: An object-based standard for multimedia coding" invited chapter to appear in Visual Communications book, Marcel & Dekker, 1998.

[5] J. Ostermann, E. Haratsch, "An animation definition interface", International Workshop on synthetic - natural hybrid coding and three dimensional imaging, pp. 216-219, Rhodes, Greece, September 5-9, 1997.