Methodologies Used for Evaluation of Video Tools and Algorithms in MPEG-4

- Invited Paper -

Jörn Ostermann, AT&T Labs - Research, 101 Crawfords Corner Rd, Holmdel, NJ 07733-3030
Tel.: ++908 949 6683, Email: ostermann@research.att.com

Abstract

MPEG-4 issued two calls for proposals requesting submission of algorithms and tools relevant to standardization of MPEG-4. This paper reports on the evaluation of tools submitted for evaluation in November 1995 and January 1996. Complete video coding schemes submitted in January 1996 are also covered. The goal of the evaluation was to cluster the tools according to the technical areas they address, to evaluate them according to the issues relevant to the standardization process, and finally to suggest areas of core experiments to improve a video verification model (VM) as soon as the VM becomes available. Altogether, MPEG evaluated 87 tools and 19 complete coding algorithms, most of them highlighted in this paper. During the evaluation, 19 areas for core experiments were identified. Each core experiment is targeted at different functionalities like compression efficiency, content-based coding, error resilience, scalability. This definition of core experiments caused close collaboration and supported mutual fertilization between organizations working on similar tools, which allowed the VM to progress much faster than expected.

1. Introduction

For starting the standardization process of MPEG-4, MPEG did not only call for complete coding algorithms like for MPEG-1 and MPEG-2 but also for tools which address areas relevant to functionalities to be provided by MPEG-4 [6][7]. Hence, a tool is assumed to be a module that would enable a MPEG-4 encoder or decoder to perform operations or improve on available operations required for certain functionalities. MPEG-4 called for proposals to be evaluated and tested in November 1995. For video coding algorithms, proposers had to encode given video sequences at given bit rates between 10 kbit/s and 1 Mbit/s. They had to submit a description of encoder and decoder, bitstreams, a decoder and the decoded sequences on a D1-tape. The performance of the coding algorithms was subjectively tested in November 1995. For the submission of video tools, proposals had to provide a thorough description of their tool and results showing the performance of the tool. These tools were subject to an evaluation by experts. Since it was not possible to provide common test conditions for tools covering vastly different technical areas, subjective or objective testing was not planned.

The reason for asking for tools submissions was mainly to include new techniques from research institutes and universities which normally do not work on video coding but who nevertheless do work very relevant to MPEG-4. Especially due to the new aspects of content-based coding, MPEG-4 looked for ideas outside of the *traditional* block-based hybrid DCT encoding scheme. Areas like image analysis, image synthesis and computer graphics were thought to be relevant. Furthermore, it was hoped that recent progress in still image coding would help to increase coding efficiency for video coding.

On a non-technical side, it was expected that institutions with a solid tool proposal would see the merit of introducing their tools in the final MPEG-4 standard and thus help MPEG-4 video to broaden its technology base and to incorporate new technology into the standard. As seen later, MPEG-4 was very successful in attracting new parties and new relevant technology.

Initially, interested parties were asked to submit video tools for evaluation in November 1995. This call resulted in the submission of 70 tools from more than 50 companies [1]. However, due to changes in the MPEG-4 1996 work plan, a second call for video tools and algorithms was issued in November 1995 [7]

which resulted in the submission and evaluation of another 17 tools from 13 organizations and 19 algorithms from 16 organizations in January 1996 [2].

In order to ensure a thorough and timely evaluation of the video tools, MPEG-4 employed two ad hoc groups. Ad hoc groups continue the work of MPEG between two consecutive meetings. Any member of MPEG can sign up in such a group. The groups work according to their mandate. Decisions are achieved by consensus. The mandate of the ad hoc group on the "Evaluation of tools for non tested functionalities of video submission" was “to evaluate tools and algorithms submitted for MPEG-4 non-tested functionalities”. The ad hoc group on "Evaluations of Tools and Algorithms of video submissions for MPEG-4 in January 1996" had the similar mandate of “to define means for the evaluations and to perform evaluation of tools and algorithms submitted for MPEG-4 in January 1996”. Both ad hoc groups were chaired by the author of this paper.

This paper describes in Section 2 the concepts of tools and core experiments. In Section 3, evaluation criteria for video tools and video coding algorithms are defined. Section 4 addresses the issue of gathering impartial teams of experts for the evaluation. Section 5 gives an overview over the technical areas addressed by the tools and indicates where they can be used in the implementation of the MPEG-4 video coding standard. The evaluation of algorithms is covered. The main results and recommendations of the evaluation process are also provided in Section 5. This paper ends in Section 6 with a critical evaluation of the achieved results and its influence on the current developments within MPEG-4 video.

2. Tools and Core Experiments

It was foreseen that MPEG-4 video will be developed using a core experiment process already known from MPEG-1 and MPEG-2. This approach assumes that a codec consists of connected and configured tools. After defining an initial verification model (VM) that describes a complete video codec addressing the envisioned functionalities of the standard, this verification model would be improved by experiments replacing existing parts of the encoder/decoder with new tools or adding new tools to the codec. These experiments are called *core experiments*. Core experiments evaluate the performance of tools in the framework of the current VM. A core experiment defines the changes to the VM, including syntax and the test conditions for the experiment. At least two independent companies have to carry out a core experiment. If consistent results are achieved and the performance of the video codec is improved due to the tools introduced by the core experiment, the tools are considered for being a mandatory part of the VM. This decision is prepared by the video group of MPEG-4 and approved at the plenary of MPEG.

Let us take H.263 as an example to highlight the definition of tools. There are several possibilities of describing a H.263 decoder by tools. They distinguish themselves by the level of detail. On a high level, we could define major tools like a motion compensation tool (overlapped block motion compensation including motion vector decoding) and a texture decoding tool (Huffman decoding, fixed length decoding, inverse quantization, inverse zigzag scanning, inverse DCT). On the other hand, we could also define H.263 using minor tools as done for the definition of the major texture decoding and motion compensation tools above. Whereas at the beginning of the standardization process, core experiments can investigate major tools (i.e. should the texture decoding tool be replaced by a tool using wavelets and zerotree coding) and minor tools (i.e. should the quantizer be replaced by a new one), it is expected that at a later time in the development process the core experiments are more concerned with the fine tuning of the chosen technology, thus only looking into optimizations of tools already part of the VM.

3. Evaluation Criteria

3.1 Video Tools

The purpose of the evaluation of video tools was to provide an overview of the submitted technology, classify tools according to the technical areas they address, and evaluate tools according to issues relevant to the standardization process. Furthermore, this evaluation should enable the definition of core experiments and of the context for these experiments. In the case of several tools addressing the same

technical area like shape coding, the core experiments would allow a fair comparison of the different tools in order to choose the best.

The evaluation process was prepared using email reflectors for discussion prior to the actual evaluation meetings. Before knowing the tool submissions, we agreed upon the evaluation criteria that are relevant for the standardization process. For each tool we filled out a template with the following 10 items:

1. Functionality: Which functionality is addressed? Here the major functionalities like compression efficiency, content-based scalability, content-based access, and error resilience should be given.
2. Efficacy: How effective is the tool at meeting the goal of the functionality it addresses? This item asks whether the performance of the tool was demonstrated, or how the tool distinguishes itself from other tools addressing the same technical area. For scalability, the number of supported layers would be of interest.
3. Encoder/Decoder: Is the tool to be used in the encoder or decoder? Some tools like rate control or motion estimation are only required at the encoder. These tools will not be in the mandatory part of the standard.
4. Adaptability: Is the tool applicable to a wide range of scenes, bitrates, error conditions, resolutions, delay? For example, tools addressing camera motion will only show their benefits in scenes with camera motion. Similarly, tools relying on adaptive arithmetic coding tend to be less error resilient than fixed length coding.
5. Coding environment: 1.) Does the success of the tool depend on particular coding schemes? Some tools like a multi-resolution vector quantizer tool might require a subband codec in order to show their benefits. 2.) Has the tool/algorithm been presented as part of a coding scheme? It was expected that some companies submit a complete coding algorithm for subjective testing by MPEG-4 and also submit the innovative parts of the algorithm as a tool.
6. Standardization: Does the tool require standardization? Only decoder tools require standardization. However, a decoder might use post processing tools that certainly do not require standardization.
7. Syntax: Does the tool require or benefit from special syntactic elements? Some tools might require syntactic structures not known from current image coding standards. For example, knowledge of the decoder might be used by the encoder to change the syntax of the bitstream without requiring signaling of this change (Example: Based on the current decoded image, the decoder automatically generates an scene adaptive 2D mesh for motion compensation of the next image. Each node of the mesh requires one motion vector. Since the encoder can run the same algorithm for mesh generation, it does not have to transmit how many motion vectors will be in the bitstream).
8. Added value: What are the merits of the tool compared to other submitted or known tools? Given the provided information and common knowledge, the tool should be compared to other tools. However, since no comparative tests were conducted, these comparisons are vague unless this information was provided in the tool description.
9. Margins for improvement: What are the margin and time frame for improvements? It was foreseen that some tools would be declared as preliminary results. Here, the areas of improvement (i.e. implementation complexity, coding efficiency, adaptation to a wider range of scenes) should be named, and if possible, the required time frame should be given.
10. Complexity: What is the implementation complexity? Here a rough comparison to known tools of a standard hybrid DCT codec with block motion compensation is examined. Also, the influence of scene contents or bitrate on the complexity is of relevance.

It was understood that the evaluation of tools according to these criteria could not be used for a ranking of tools. Table 1 shows typical evaluation results for a well-documented tool submission. In case of missing information in the documentation, this would be stated in the evaluation form of the tool.

Table 1 Evaluation results based on a well-document tool submission (here MPEG 95/423). Example taken from [1].

1. Functionality Addressed	1. Content-Based Multimedia Data Access Tools 2. Content-Based Manipulation and Bitstream Editing 3. Hybrid Natural and Synthetic Data Coding 5. Improved Coding Efficiency 8. Content-Based Scalability
2. Efficacy	The shape coding tool is effective in meeting the goals of the functionalities it addresses. However the efficiency of this tool should be compared with similar tools in this class.
3. Encoder/Decoder	The proposed tool is used in the encoder and decoder.
4. Adaptability	This tool is applicable to a wide range of scenes, bitrates and resolutions.
5. Coding Environment	This scheme is dependent on particular coding schemes in the sense that the quality of the decoded contour is highly dependent on the quality of the coded texture. However, the scheme has no explicit restriction to any texture coding technique.
6. Standardization	Yes, this tool requires standardization. The texture coding technique and the contour recovering process has to be standardized.
7. Syntax	No special syntactic element is necessary
8. Added Value	Method of recovering contour from the coded texture. Solves elegantly and simply the contour coding problems.
9. Margins for improvement	There is room for improvement in the selection of the background color and the composition operation.
10. Complexity	The method does not require explicit modeling of the contour contrary to other shape coding methods so the complexity is mainly that of the texture coding technique used. Additional complexity may be required for the background color computation.

3.2 Video Algorithms

Video coding algorithms were only evaluated during the Munich ad hoc group meeting in January 1996. The call for submissions in January 1996 was issued in November 1995 [7]. As for the 1995 subjective tests, algorithm proposers had to encode given video sequences at given bit rates between 10 kbit/s and 1 Mbit/s [6][8]. They had to submit a description of an encoder and a decoder, bitstreams, a decoder and the decoded sequences on a D1-tape. For the purpose of comparison, all the sequences were also encoded using an optimized H.263 encoder for the lower bitrates and an MPEG-1 encoder for bitrates of 320 kbit/s and above. These encoded sequences were referred to as
. They were part of the November 1995 test and also available during the January 1996 evaluation. Whereas the video coding algorithms submitted to MPEG-4 in 1995 were evaluated using properly conducted subjective tests requiring the editing of the submitted tapes, no subjective tests could be carried out in January 1996 mainly due to the short advance notice of the evaluation. Therefore, the goal of the algorithm evaluation in January 1996 was not to achieve a ranking but to classify algorithms according to their technology, introduce the major tools of the algorithm into the core experiment process, and recommend which video tapes should be shown to the MPEG-4 video group.

Prior to the ad hoc group meeting, we agreed on 9 evaluation criteria for video coding algorithms. These criteria are mainly adapted from the evaluation criteria of tools:

1. Functionality addressed.

2. Picture quality (Better, similar, worse than the anchor), frame rate of tape, buffer control, dominant artifact.
3. Efficacy: How well are functionalities other than compression addressed?
4. Adaptability: Range of scenes, bitrates, error conditions, resolutions, delay.
5. Algorithm characterization: Motion estimation, motion compensation, texture coding, shape coding, syntax, others.
6. List of relevant core experiments: List core experiments as established in [5].
7. Implementation complexity: Encoder (benchmark), decoder (benchmark), real time implementation available (Video DSP, ASIC, PC).
8. Additional advantages.
9. Margin and time-frame for improvement.

Item 2, picture quality, is certainly the most important and also most difficult item to evaluate. Whereas frame rate and buffer control algorithms are objective measures, getting an agreement on whether an algorithm introduces a dominant artifact like ringing or blocking is difficult. The most challenging question, however, is picture quality compared to the anchor sequences. Whenever required, the anchor sequences and the sequences of an algorithm proposal were shown to the evaluation group in sequence. The viewing conditions, like distance to the monitor and room illumination, were not specified. For each algorithm, the group had to decide whether the proposal achieves a picture quality better, similar or worse than the anchor. Again, these evaluations of picture quality cannot be compared to the thorough subjective tests conducted in November 1995. Furthermore, the judgment of picture quality compared to the anchor is not necessarily stable and thorough subjective tests might give different results. However, within our limited capabilities, we tried to achieve a fair evaluation. Table 2 shows a typical evaluation result for a video coding algorithm.

Table 2 Typical evaluation result of a video coding algorithm (example for MPEG 95/639) from [2].

1. Functionality Addressed	compression/content based functionality; SNHC
2. Picture quality Frame rate of tape Buffer control Dominant artifact	Better than or similar to the anchor >=15 frames per second No Shape of objects
3. Efficacy	SNHC - very good; spatial scalability not fully demonstrated; temporal scalability not demonstrated
4. Adaptability: Range of scenes Bitrates Error conditions Resolutions Delay	 Generic >= 48 kb/s No Independent of resolution Depending upon method for sprite generation, could be much greater than MPEG-1
5. Algorithm Characterization: Motion estimation Motion compensation Texture coding Shape coding Syntax	 Affine block based Same as classical methods (not classical block based) H.263 Simplified chain codes + polygonal approx. + 8-bit alpha coding VOP based using H.263
6. Accommodated by the following Core Experiments:	P4, S1, S2, O1
7. Implementation Complexity Encoder (Benchmark) Decoder (Benchmark) Real time implementation (Video DSP, ASIC, PC)	 More complex than H.263 for natural VOPs; less for synthetic VOPs As above No
8. Additional advantages	Flexibility of representation
9. Margin and time-frame for improvement	Real time sprite accretion in 6 months, better shape and alpha channel coding

4. Evaluation Teams

The evaluation was carried out by approximately 80 members of MPEG video during a two-day meeting of the ad hoc groups hosted by Hughes Aircraft Company in Los Angeles and Deutsche Telekom in München just before the MPEG meetings in November 1995 and January 1996, respectively.

Due to the large number of submitted tools, they had to be evaluated in parallel groups. There were basically two criteria for clustering the tools, namely MPEG-4 functionality addressed and technical area. We decided to cluster the tools into different technical areas since this allowed for a technical evaluation carried out by experts of the specific area.

In November 1995, we evaluated 70 proposals. In order to allow for an efficient evaluation, we established 8 groups addressing the following technical areas:

1. Pre/Postprocessing (Touradj Ebrahimi, EPFL)
2. Texture encoding: Subband (Ya Qin Zhang, David Sarnoff Research Center)
3. Texture encoding: Segmentation/VQ (George Campbell, C-Cube)
4. Shape coding (Thiow Keng Tan, Matsushita)
5. Motion estimation (Takahiro Fukuhara, Mitsubishi)

6. Motion segmentation (Raj Talluri, Texas Instruments)
7. Coding efficiency, segmentation, and background estimation (Peter Gerken, University of Hannover)
8. Error robustness, stereo (Jim Brailean, Motorola)

The chairs of these groups are given in brackets.

In January 1996, 17 tools and 19 algorithms had to be evaluated. The evaluation was carried out in four groups:

1. Object functionality (Touradj Ebrahimi, EPFL)
2. H.263++ (Boon Choong Seng, Matsushita)
3. Error resilience and others (Jim Brailean, Motorola)
4. Wavelet and mesh (Ya Qin Zhang, David Sarnoff Research Center)

This clustering of technical areas reflected the smaller number of tools/algorithms to be evaluated as well as the outcome of the November 1995 subjective tests which seemed to favor codecs based on hybrid DCT codecs with motion compensation like H.263. Each group had to evaluate tools as well as algorithms.

In order to allow a thorough evaluation of a tool/algorithm, the proposer was present during the discussion of his proposal, which allowed for questions and answers beyond the information provided by the description or an accompanying video demonstration. In many cases, the proposer was a regular member of the group evaluating his proposal. If this was not possible, he would join from another group just for the evaluation of his proposal. Due to the very technical nature of all discussions, it was felt that proposers did not introduce a bias to the evaluation of their proposals.

5. Evaluation Results

This section provides an overview of the evaluation results. Section 5.1 describes the different video tools evaluated in November 1995 and in January 1996. They were grouped into core experiments in order to allow their comparison in a VM. Section 5.2 compares different algorithms addressing functionalities like coding efficiency, scalability and error resilience. Algorithms were only evaluated in January 1996.

5.1 Video Tools

As a result of the tools evaluation, we were able to cluster the tools into technical areas as well as to recommend combination of tools into core experiments. The following two sections 5.1.1 and 5.1.2 give an overview of these results.

5.1.1 Technical Areas

The evaluated tools can be clustered into 10 technical areas (Table 3). Most contributions were received for texture coding, motion estimation and shape coding. Especially the submissions dealing with shape coding and coding of texture for arbitrarily shaped regions are of significant importance to MPEG-4 in order to enable the content-based functionalities. Several tools are concerned with segmentation. Segmentation is important for building an encoder which allows a content-based encoding of a sequence without having access to the segmentation information. The following sections give a brief overview of shape coding, texture coding, motion estimation, segmentation and others.

Table 3: Technical areas addressed by tools.

Category		MPEG Document Number
Shape coding	Geometrical representation (polygons, splines ...)	355, 360, 447, 461,332,565
	Implicit coding	423
	Bit map	340, 445,459
	Chain coding	318, 344
Texture coding	Wavelet transform and coding	437,323,333a,378,439,441,310, 620
	DCT	489,555,596
	Vector quantization	326,410a, 410b, 410c
	Still image segmentation	326,557
	Texture prediction	327, 394, 410.c,617,623
	Bit allocation	372, 436,566
	Entropy coding	371,410b
	Arbitrary-shaped regions	378,596,555,620,624,423
Motion estimation	Block Based	364, 334, 438, 309, 552
	Mesh-based	411, 444,571, 602, 650,333b
	Parametric Motion Estimation	407, 460
	Global Motion Estimation	440
Segmentation	Spatial segmentation	356,369,541,343,571,595
	Motion segmentation	406a,406b,329
	Spatial/motion segmentation	540,325,319,366
	Background estimation	305,406c,653
Error Resilience	Error detection and correction	308, 313,600,616,
MSDL	Video compositor	544
Stereoscopic images	Analysis, synthesis, coding	367, 368, 487
Postprocessing	Spatial Postprocessing	357, 358, 370
	Content based postprocessing	328, 424, 443
Preprocessing	Color space conversion	539
SNHC	Face synthesis	464

5.1.1.1 Shape Coding

Two classes of shape coding information are considered: Binary shapes define the silhouette of an object in its current view using a binary mask. Alpha maps also define the silhouette of an object in its current view. However, an alpha map includes blending information allowing for the object to be partially translucent or for the object boundaries to be blended with the background of a composed image. Typically, each pel of an alpha map is PCM-coded with 8 bit.

Figure 1 shows an image where the outline of the shape of the person is superimposed on the coded image. An MPEG-4 video encoder would transmit two bitstreams: One bitstream would describe the background that is static in this *Akiyo* sequence and one would describe the moving person. The latter bitstream would include motion, texture and shape information.

Figure 1: Outline of the binary shape superimposed on the coded image (from [318]).

Shape coding has not been investigated by video coding standardization bodies prior to MPEG-4. Therefore, the 11 proposed shape coding tools describe the method and provide results without comparison to other schemes. Different measures for computing the fidelity of the approximated shape with respect to the original shape are proposed. In [360], the Euclidean distance between each point of the original contour and the approximated shape is measured (Figure 2). The maximum of this value describes the shape approximation quality. Another proposed measure is the relative number of pels that are misrepresented as being inside or outside the original shape due to lossy shape coding. All the submitted shape coding tools deal with encoding of binary shape information. Encoding of alpha maps is only addressed by one algorithm submitted in January 1996 [639]. The tools for coding binary shapes can be clustered into four categories:

Geometrical Representation: These tools approximate the boundary of a shape using a polygon [447] (Figure 2) or a polygon/spline approximation (Figure 3) [360].

4/9/2010

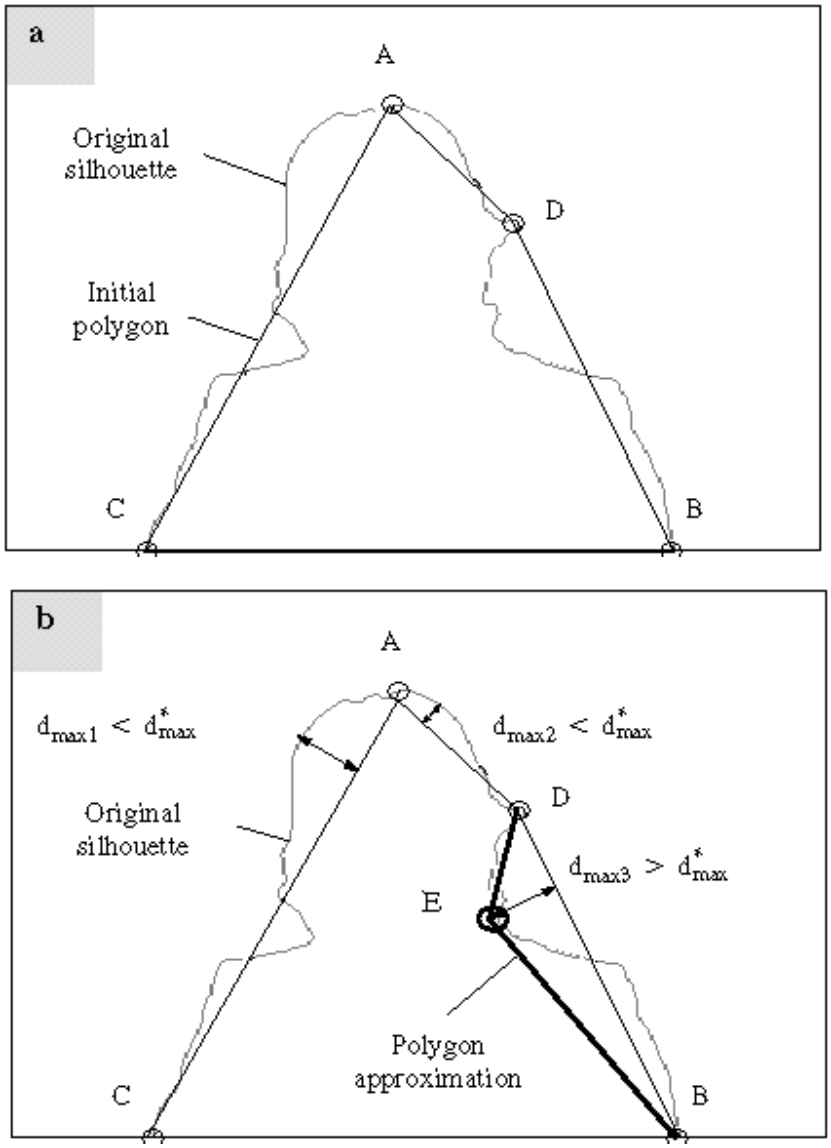


Figure 2: Polygon approximation of a binary shape: a) Initial polygon ABCD. b) The approximation error between the original contour and line DB exceeds the threshold d^*_{max}(from [15])**.**

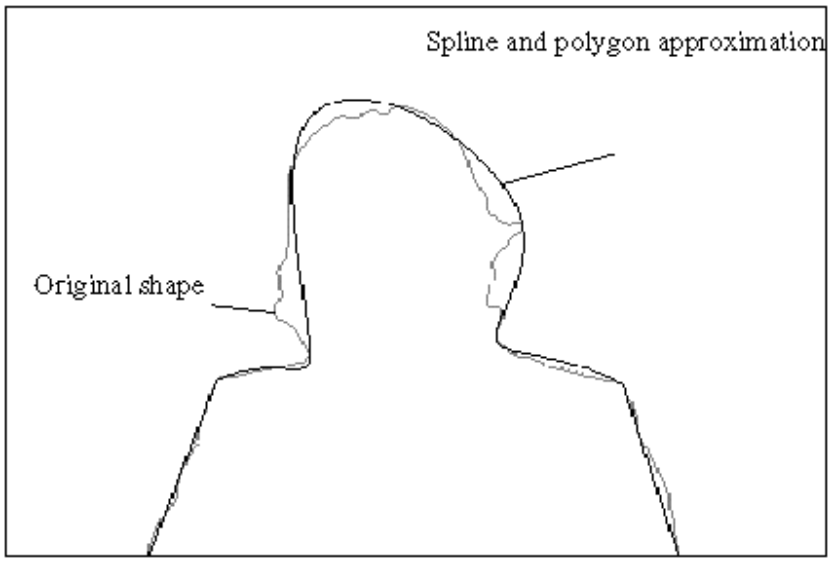


Figure 3: Combination of a polygon/spline approximation for a quality measure d^*_{max}=15 pel (from [15])**.**

Using this geometrical representation it is straightforward to generate a coarse or a fine approximation of the shape. Whereas some proposals suggest to achieve a fine approximation by introducing more polygon/spline points [360][447], other approaches favor an explicit encoding of the shape approximation error using a transformation like the Discrete Sine Transform (DST)[565] or Discrete Cosine Transform (DCT) (Figure 2). These methods can be used for temporal predictive coding of shape using motion compensation of the vertices that represent the approximated object shape.

Bit Map: These tools propose a shape coding method which is based on the macro blocks of current image coding standards. For each macroblock, the object shape is encoded using a bitmap. This bitmap can be lossless or generated from some predefined patterns. In [459], it is proposed to adapt the position of the macro-block grid such that the number of macro blocks covered by the object is minimized (Figure 4). Hence, this technique reduces the overall bitrate for encoding an arbitrarily shaped object up to 6% for the sequences in the MPEG-4 test set.

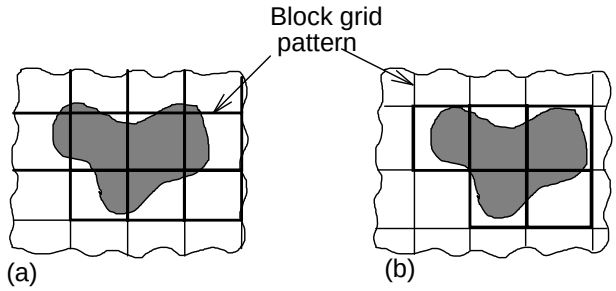


Figure 4: Schematic illustration of the representation of an object shape with respect to the macro block grid. a) Due to the fixed macro-block grid, the object covers 10 macro blocks. b) After adapting the macro-block grid, the object covers only 5 macro blocks (from [459]).

Chain Coding: Chain coding is a technique that describes the contour of a shape. Beginning with a starting point on the contour and a starting direction, the relative changes in direction, required to go from the current contour point to the next neighbor on the contour, are coded. Several variations exist. A distinction is for example whether a point on a rectangular grid is considered to have 4 or 8 direct neighbors. Traditionally, chain coding is used for lossless encoding of shape. The proposals suggest improvements in

statistical modeling of the event to be encoded. In order to allow a lossy encoding of shape, the shape is prefiltered using morphological operations like dilation and erosion that result in smooth contours allowing for efficient encoding [318]. This contour filtering does not necessarily results in a visual degradation of the shape (Figure 5).

Figure 5: On the left the foreground object as defined by the original segmentation, on the right after filtering the object shape (noise reduction) (from [318]).

Implicit Shape Coding using Texture Coding: One tool [423] suggested placing the object to be encoded on a background with a known and unique color. This frame consisting of the object and the background is encoded with a conventional technique like H.263. The decoder decodes the image and extracts the object using the background color as a chroma key. It is expected that this method outperforms the other coding methods for encoding of complicated shapes. However, a separate rate control for shape and texture coding bits is not possible using this implicit coding technique. Here the quality of the shape approximation and of the texture is controlled by the step size of the quantizer for the DCT coefficients only.

It is important to note that scalability issues of shape coding were not sufficiently addressed by the proposed tools.

5.1.1.2 Texture Coding

Texture coding has been investigated by standardization bodies for a long time. Currently, all important image coding standards like JPEG, H.261, MPEG-1, MPEG-2, and H.263 subdivide an image into square blocks. Each square block is encoded using a DCT. Naturally most tools submitted in November 1995 suggested improvements for frame-based and block-based texture coding. The tools can be classified into 8 groups: Wavelet transform and coding, DCT, vector quantization, segmentation assisted texture coding, texture prediction, bit allocation and rate control, and finally entropy coding. From a functionality point of view, texture encoding methods for encoding arbitrarily-shaped regions and scaleable texture encoding methods are of most importance. Whereas several tools address the problem of spatial scalability, only one tool [489] deals with temporal scalability. It is interesting to note that all but one of the tools dealing with texture coding for arbitrary-shaped regions were submitted in January 1996. Some of these tools were part of an algorithm for the November 1995 test. In the following, tools for wavelet transforms and arbitrarily shaped regions will be described in more detail.

Wavelet Transform and Coding: The largest group of texture coding tools dealt with wavelets (

Table 3). The proposals include wavelet decompositions and different zerotree entropy encoders. The wavelet decompositions are proposed for coding intra-frame signals as well as prediction error signals. Also a subband encoder for small independent square blocks is proposed for coding displaced frame difference signals. Several tools address spatial scalability by proposing to assign different bands of the wavelet decomposition to different levels of spatial detail. Two tools propose to adapt wavelets and embedded zero trees for encoding arbitrarily-shaped regions.

Arbitrarily-Shaped Regions: The adaptation of wavelets and zero-trees for encoding texture signals of arbitrarily-shaped regions is proposed [378][620]. Several other contributions deal with extrapolating the texture of the shape to a square block and then perform a DCT on the signal [555][596]. The art lies in finding an extrapolation method which keeps the rate for encoding this *boundary block* small. In [639], repetitive padding is proposed. The texture of pels in a boundary block outside of the object is set to the value of the closest pel inside the object (Figure 11). Some tools claim to compare favorable to a shape adaptive DCT [9]. As mentioned before, the tool using texture coding for implicit shape coding also encodes the texture of an arbitrarily-shaped region. Temporal and spatial scalability issues were not addressed.

5.1.1.3 Motion Estimation

For motion estimation, four categories can be identified: Block-based, mesh-based, parametric and global motion estimation. Since motion estimation is a well known technique, a lot of tools compare themselves with full search block matching. Although motion estimation itself will not be part of the standard, the parameters transmitted for motion compensation depend to a large extent on the motion model used for motion estimation. Therefore, motion estimation is very relevant for the standardization process.

Block-based motion estimation: Here, the main theme is a variation on full search block matching using variable block size, as compared to the 8x8 and 16x16 block sizes in H.263. A joint optimization of variable block size segmentation and residual error encoding is also suggested. Another tool reports improvements by using ¼ pel accuracy for motion vectors and a special antialiasing filter for motion compensation [552]. Furthermore, making overlapped block motion compensation adaptive on a block basis is proposed. The idea of zerotrees for encoding motion vectors is used to encode dense motion-vector fields.

Mesh-based motion estimation: Using adaptive triangles or quadrilateral meshes for motion estimation, motion compensation and object tracking is proposed. On an prediction error basis, these algorithms outperform block-based motion compensation. One proposal [411] suggests to generate an adaptive mesh based on information the decoder has in order to avoid transmission of node position information. This intelligent exploitation of decoder knowledge requires a flexible syntax that is controlled by the image contents. Currently, MPEG-4 Systems Descriptive Languages (MSDL) [10], which define bitstream parsing, decoding and other system aspects, are not supporting such a syntax [3].

Parametric motion estimation and global motion estimation: Here, an affine motion model with 6 parameters is favored. It is used for global motion estimation, as well as for object motion estimation. A similar method is also proposed for background mosaicking (Figure 6) [653].

5.1.1.4 Segmentation

Segmentation is mainly driven by the need to allow content-based encoding of image sequences which do not have any segmentation information available. Most of the segmentation tools segment images based on texture and motion. Some segment motion only whereas others consider texture to increase the accuracy of the motion segmentation. Affine motion models and dense displacement vector fields with some statistical model like a Markov random field are used for segmentation. Since the segmentation has to be described using a shape coding tool and the motion models are similar to the ones proposed in the previous section, these segmentation tools will not have a major influence on the standardization development. However, there is one exception:

Background segmentation: Several tools try to segment foreground objects from the background objects in order to automatically fill a background memory at the decoder. One proposal even considers camera motion and generates a background memory larger than any frame of the sequence by mosaicking [653].

Background mosaicking assumes a rigid scene background (Figure 6). Only part of this background is visible in the current frame of the video sequence. More of the background becomes visible as the camera pans across the scene. For each given frame, the technique first identifies background and foreground regions (Figure 7). The background is defined as the regions with motion coherent to the dominant motion (camera motion). For the background regions, the camera motion is estimated using a global motion model with six or eight motion parameters. Using these motion parameters, the background motion can be compensated with respect to the previous image or with respect to the background mosaic of the previous frames. The motion parameters allow for a seamless integration into a large background image (Figure 8). Please note that the street light in Figure 8b-d appears distorted. However, when synthesizing an image from the background mosaic, the motion parameters revert this distortion such that the light appears in its original shape as seen in Figure 7a. The assumption of a static background can be relaxed. Limited local motion in the background can be neglected by the algorithm. For fast camera panning, the missing local motion within the background, like the people sitting in a sports arena, might not be caught by the human eye [639].

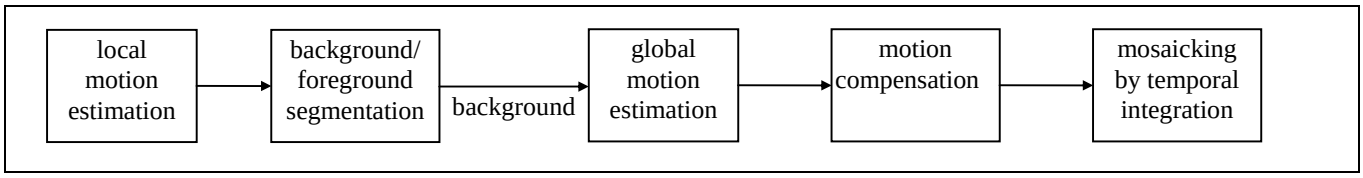


Figure 6: Background Mosaicking (from [653]).

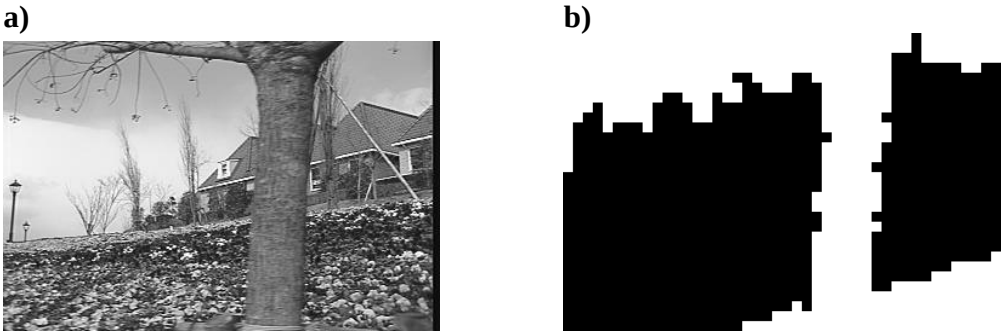


Figure 7: a) A frame of the test sequence *Flower Garden*. The camera is panning from left to right. b) shows the segmentation of the image into foreground (white) and background (black) regions (from [653]).

a)

b)

c)

d)

Figure 8: *Flower Garden*: a) background mosaic after one frame, b) background mosaic after 25 frames, c) background mosaic after 50 frames, and d) background mosaic after 100 frames (from [653]).

5.1.1.5 Other Tools

For *error resilience*, two tools [308][616] propose signaling over a back channel to tell the encoder which information was lost. Tool [616] describes an implementation that does not increase delay. It signals to the transmitter which macro blocks are received with errors. The transmitter uses this information in order to code in Intra mode those macro blocks that have a temporal reference to erroneous data at the receiver. One tool suggests to use error correcting codes and interleaving to deal with random and burst errors [313]. Fixed length codes are more error resilient than variable length codes, but can differ in their error resilience. A method for developing an error resilient code is proposed [600] extending the concept of Gray codes. The so-called Baang Code considers the amplitude represented by a code word with respect to all other code words within a Hamming distance of 1.

Other tools cover diverse areas from *preprocessing* over *stereoscopic images* to *post processing*. Post processing was mainly concerned with removing artifacts introduced by a coding algorithm. One tool proposed TrueGaze for video phone applications [443]. TrueGaze compensates for the parallax distortion due to the misalignment of the camera and monitor axis thus enabling eye contact in video conferencing. [544] describes the architecture of a video compositor. The task of the compositor is to allow real-time and interactive composition of video streams and computer graphics objects into a scene. Currently, MPEG-4 MSDL deals with composition of video, audio, and graphics. One tool [464] demonstrated the *animation* of an artificial face model from a person in a video sequence (Figure 9). The area of face animation and its interface to text-to-speech systems [14] is currently covered by the MPEG Synthetic Natural Hybrid Coding (SNHC) group.

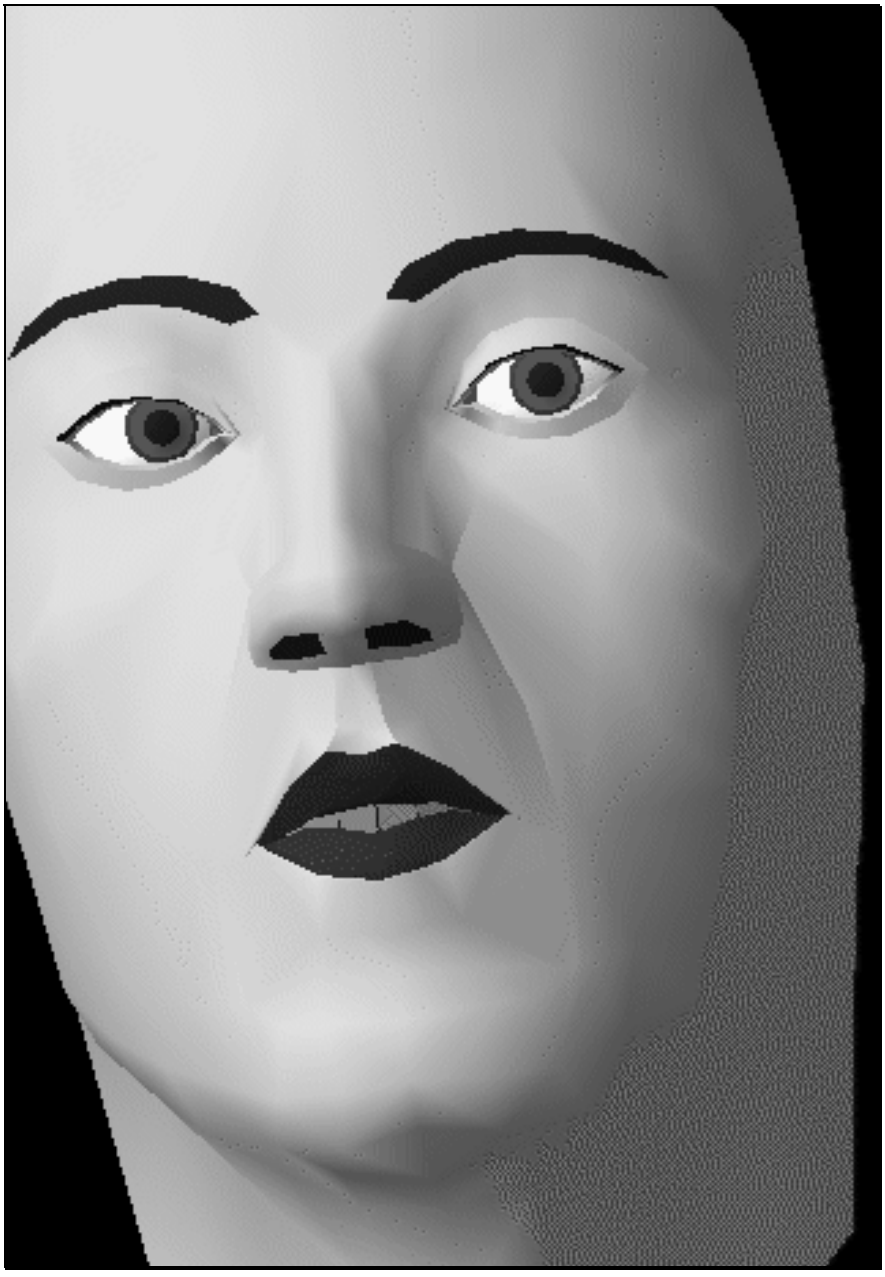


Figure 9: Synthetic face driven from a talking person in a video sequence (from [464]).

5.1.2 Core Experiments

Based on the analysis of the tools, the ad hoc group recommended core experiments in those areas where at least one of the submitted tools provided evidence for improvement or added functionalities compared to state of the art image sequence coders (H.263, MPEG-1, MPEG-2). In this paper, only the grouping of tools into core experiments is given. Tools were grouped into core experiments in order to improve mutual fertilization of similar proposals. For details, the reader is referred to the ad hoc group reports [1][2] or the current list of core experiments as provided in the video verification model [4].

During the first evaluation in November 1995, 15 core experiments were suggested:

1. Wavelet transform (323, 378, 437, 333a)
2. Zero-Tree encoding (439, 441)
3. Intra-frame coder (410.a, 326)
4. Interframe texture prediction (327, 394, 410.c)
5. Arbitrary shaped region texture coding (326, 356, 459, 369)
6. Shape coding (355, 360, 447, 461, 423, 340, 445, 318, 344)
7. Motion estimation with variable-size block-based motion estimation (334, 364, 438, 309), 2D triangle mesh motion estimation (411, 444, 333b), and parametric and global motion estimation (407, 460, 440)
8. Segmentation with temporal segmentation (319, 406, 329), spatial segmentation (343, 541, 319), and spatio-temporal segmentation (319, 406, 406, 325, 540)
9. Object-tracking (366)
10. Background memory (406, 305)
11. Bit allocation (372)
12. Rate control (436)
13. Error-resilience (308, 313)
14. Coding of multiple concurrent data streams (367, 368, 487)
15. Post processing (358, 370, 357, 328, 424, 443)

Based on this list and the algorithms submitted for subjective testing, a comprehensive list of core experiments was defined before the January 1996 MPEG meeting [5].

Due to the evaluation in January 1996, four new core experiments were suggested:

1. Object-based temporal scalability (582).
2. Modulated lapped transform (MLT) (646, 557)
3. Automatic sprite generation (653).
4. Adaptive inter-coded I-frame (623).

The other submitted tools were assigned to the previously defined core experiments.

5.2 Video Algorithms

Video algorithms submitted to MPEG-4 in November 1995 were subjectively tested providing a firm ranking of different proposals [8][11]. A document evaluating the technology of the submitted algorithms is not available. The evaluation of algorithms in January 1996 provided an excellent overview of the techniques submitted. However, subjective tests were not carried out.

For the algorithm evaluation in January 1996, 14 organizations submitted one algorithm addressing one or more functionalities. One company submitted 4 proposals that were based on H.263 but had major components like texture coding or motion compensation changed in the different proposals. Several proposals were a refined version of a submission to the November 1995 subjective tests [8]. Others showed a tool submitted for November 1995 incorporated into a video codec. Table 4 summarizes the different algorithms. Most algorithms addressed coding efficiency, the traditional area of video compression.

However, the emphasis on content-based functionalities like object scalability, content-based spatial and temporal scalability increased compared to the November 1995 submissions.

At the November 1995 meeting the preliminary conclusion was drawn that H.263 could be a good starting point for developing a MPEG-4 video verification model (VM). This conclusion was corroborated during this evaluation. 14 algorithms were based on H.263. This means, that all used an H.263 like syntax, and all used either H.263 motion compensation or texture coding or both. The following sections give an overview of the submissions addressing coding efficiency, object scalability and error resilience. Finally, the recommendation of the ad hoc group regarding video algorithms is presented.

Table 4 Overview of evaluated algorithms (Func = functionality addressed, Cps = compression, ObjSlb = object scalability, ErrRes = error resilience, MC = motion compensation). An x in the columns Mot. and Tex. means that motion compensation and texture coding according to H.263 is used, respectively.

DocNo	Func	Mot.	Tex.	Comments
553	Cps			1 quadtree defines block-size for MC and DC update
564	"			affine block-based MC, segmentation of prediction error based on HVS, DCT
586	"		x	adaptive mesh/block MC selection
592	"	x	(x)	Intra-frames: vector wavelet with lattice VQ. Inter-frames: H.263
599	"		x	adaptive affine/block MC selection, background memory
609	"		x	MC modes: 8x8 block, 16x16 block, average of 8x8 and 16x16 block, or affine
615	"		x	global and local MC using affine 6 parameter model, overlapped MC
625	"			region-based affine MC, chain coding, shape adaptive DCT
637	"	x		wavelet, zerotree
646	"	x		modulated lapped transform
654	"	x	x	additional long term frame memory for prediction
659	"	x	x	quantizer selection according to identified interesting regions
554	ObjSlb	x	x	shape coding using polygon approximation and DST
573	"			quadtree for object shape and selection of compression technique: MC, DCT, fractal, bi-level signal update
582	"	x	x	temporal object scalability, higher frame rate for bounding box of object of interest
583	"	x		object wavelet, zerotree
639	"			affine motion, sprites; chain code, polygon approx. and DCT; wavelets, DCT. Object, temporal and spatial scalability
587	ErrRes	x		wavelet, error correction, sync words
601	"			no MC, hybrid DCT, frame memory with DCT coefficients, inter/intra coding of coefficients, low complexity

5.2.1 Compression Efficiency

The 12 algorithm submissions can be grouped into 4 categories:

New: The codec described in [553] codes I-frames like H.263. For coding of P frames at CIF resolution, the current image is partitioned into macro blocks of size 32x32 luminance pels and 16x16 chrominance pels. Each quadrant of a macroblock can be subdivided further. The quadrant is then considered to be a macro block. This partitioning of a macroblock can be done recursively until the minimum block size of 4x4 pels is reached (Figure 10). This partition is used to convey the structural information for motion compensation as well as for texture updates. One motion vector is assigned to each macro block of the partition. Overlapped block motion compensation is employed considering the variable block size of the partition. The texture prediction error of each block can only be updated by a DC value quantized to 7 bits. This scheme could be extended to cover object-based functionalities. The authors presented the algorithm as being fractal where each iteration produces one frame of the decoded sequence. For many sequences, the picture quality was perceived to be better than the anchor sequences.
In [564], affine block-based motion compensation is used. A variable block-size DCT codes a prediction error signal which is modified according to the human visual perception. In [625], the image is segmented into regions of homogenous affine motion and texture. The segmentation and the texture are encoded using a chain code and a shape adaptive DCT, respectively.

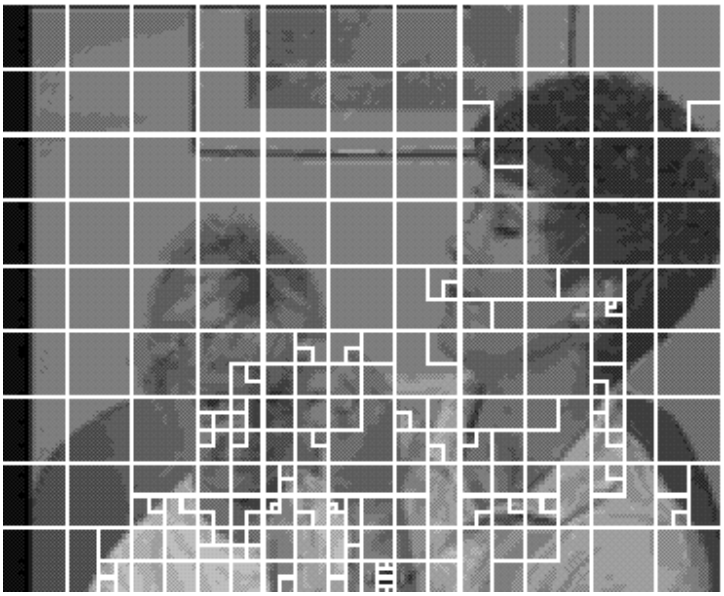


Figure 10: Frame 64 of the sequence *Mother and Daughter* with the partition into macro blocks of sizes 32x32 down to 4x4 pels superimposed (from [12]).

Motion compensation like H.263: For texture coding, a modulated lapped transform as well as a wavelet codec using a zerotree for entropy coding are presented.

Texture coding like H.263: 3 proposals suggest modifying the motion compensation in H.263 by introducing at least one additional mode on a block basis that would allow a block-wise affine motion compensation [609][599][615]. The use of a background memory is also recommended [599]. One proposal uses affine motion compensation for global and local motion compensation and shows improvements in the case of fast global motion [615].

Motion compensation and texture coding like H.263: In proposal [592], a codec is proposed that only differs in the encoding of Intra-frames from H.263. For encoding the intra-frame, a vector transform with adaptive lattice VQ for quantization and Huffman coding for entropy coding is suggested. For intra frames only, improvements of 3-8 dB are reported compared to the intra frame coding results of H.263 at 16 kbit and 20 kbit for QCIF and CIF frames, respectively. Depending on the image signal, significant improvements can be seen in still images as well as for sequences where I-frames are coded using this proposal and P-frames are coded according to H.263. However, a new (old?) kind of distortion is introduced: Instead of blocking artifacts, the images exhibit a granular noise at low data rates. This noise can be compared to noise known from regular TVs with a poor antenna. There is not yet a conclusion on which artifact is less disturbing.
One proposal suggests introducing a second frame memory for prediction in order to preserve a long-term memory of the image signal [654]. Furthermore, it is suggested to allow larger changes of the quantizer step-size between macro blocks in H.263.

5.2.2 Object Scalability

Object scalability actually refers to more than just one functionality. It includes the basic object scalability that encodes an object independently of others. Furthermore, MPEG-4 addresses spatial and temporal scalability on an object level. Whereas three proposals address basic object scalability, one proposal

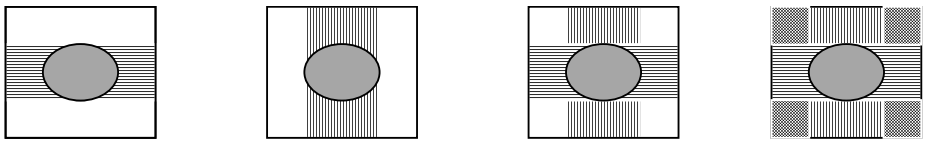
4/9/2010

enables object-based temporal scalability and one seems to be suited for both temporal and spatial scalability. As far as technology is concerned, the same categories as above can be applied:

New: Proposal [639] is a refinement of an algorithm submitted to the November 1995 subjective tests. Each object is encoded in a separate bitstream using H.263 extended by shape coding. Binary shapes are encoded by chain codes or polygonal approximations. Alpha maps are encoded in three phases:

- Get a binary shape by binarizing the alpha map.
- Code the outline of the binary shape using chain code or polygonal approximation.
- Code the alpha map values inside the coded binary shape using a transform like wavelets or DCT.

The concept of *sprites* is introduced. Sprites are 2D-objects with slow temporal variation of their shape and texture. The temporal variation is primarily generated by the affine transformation of the sprite into the image plane. Sprites are defined for each video scene. Examples can be text overlays, planes, graphics objects, backgrounds, etc. Sprites can be seen as an extension of mosaicking to non-background but still quasi-rigid objects (Figure 8). These objects/sprites are encoded and transmitted to the decoder. The decoder receives also the motion trajectory of the object/sprite. The motion trajectory is defined using an affine transformation. The shape of a sprite is defined by a binary or gray-level alpha plane. Hence, the decoder is able to compose a video scene by overlaying several sprites on top of a conventionally encoded video sequence. The proposal uses a wavelet transform for coding the texture of sprites. However, that is not essential to the technique. This algorithm is capable of basic object scalability as well as temporal scalability. Most algorithms for coding texture require the texture to be defined in a rectangular shape. In order to allow for an efficient encoding, a repetitive *padding* technique is used (Figure 11). This technique extrapolates the texture of an object by repeating the value of the boundary pixels in horizontal and vertical lines (Figure 11 a, b). If a pel gets a value according to Figure 11a and b, the average is assigned to the pel (Figure 11 c). Finally, the remaining pels are defined by averaging the neighboring extrapolated pels (Figure 11 d). The picture quality of this proposal was rated similar to H.263 for sequences that could not take advantage of the sprite concept. If sprites could be used for coding parts of the scene, the algorithm outperformed H.263.



(a) (b) (c) (d)

Figure 11: Illustration of some steps of repetitive padding: The gray oval defines the object shape. The texture of the object is extrapolated horizontally (a) and vertically (b). The remaining areas are defined by averaging the extrapolated horizontal and vertical neighbors (c and d) (from 639).

In proposal [573], a quadtree is used to describe the object shape. Within the object, the same quadtree is used to select the most efficient compression method for blocks of an object (Figure 12). For coding of each block inside the object with size and location defined by the quadtree, the encoder selects from the following coding methods: Motion compensation with DCT of the prediction error, DCT with different quantizers, a fractal transform and a bi-level signal update. Each coding mode is suited for the efficient encoding of different texture signals thus allowing an adaptive coding of the image signal.

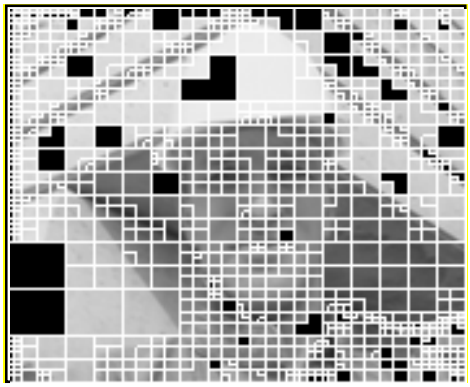


Figure 12: Segmentation of Foreman using a quadtree, 1st frame, for coding at 112kbit/s; white blocks are coded using fractals (from [13]).

Motion compensation like H.263: For texture encoding, a shape-adaptive wavelet transform with zerotree is shown in a codec [583]. This method was proposed as a tool in November 1995. Picture quality was judged similar to the anchor sequences.

Motion compensation and texture coding like H.263: One proposal uses H.263 and augments this codec with a shape codec to achieve object scalability. The shape codec uses a polygon approximation and an optional Discrete Sine Transform (DST) encoding of the shape approximation error [554]. Again, this shape codec was proposed as a tool in November 1995. One proposal achieves temporal scalability in a fairly rough manner. The shape of an object is approximated by a circumscribing rectangle of macro blocks [582]. Using H.263, this area is encoded at a higher temporal frequency than the remaining parts of the image. At the decoder, it is possible to decode at the higher or the lower temporal frequency. In sequences with a fast moving camera, inconsistencies in the relative motion of a foreground object with respect to the background are noticeable.

5.2.3 Error Resilience

For submissions in this category, proposers had to encode video sequences at given bitrates. Then, the bitstreams were corrupted using a given error pattern with single bit errors and burst errors. The picture quality of the decoded corrupted bitstream was evaluated. 2 algorithms were submitted.

One proposal distinguished itself by a very low complexity. It is a hybrid DCT scheme. Due to the lack of motion compensation, the frame memory can store DCT coefficients. As far as temporal prediction is concerned, lower frequency DCT coefficients are encoded using temporal prediction whereas higher frequency components are encoded without prediction. The other proposal was based on H.263 motion compensation and a wavelet transform for texture coding. Error correction and frequent synchronization words as well as transmission of wavelet coefficients without any temporal reference achieve error robustness.

5.2.4 Recommendations

During the evaluation of algorithms in January 1996, there was no possibility to carry out subjective tests. However, there were several hours allocated to informal viewing of submitted test sequences as well as references provided by encoding the standard image sequences using H.263 and MPEG-1. For algorithms addressing compression only, they had to outperform the anchor or provide inherent functionalities which are not or only difficult to be achieved using H.263 or MPEG-1. From the range of proposals that satisfied the conditions above, examples were selected which cover the main ideas presented. The ad hoc groups recommended that MPEG-4 video look at all the coded MPEG-4 test sequences submitted by Microsoft (object scalability, sprites, alpha maps) [639], Iterated Systems (coding efficiency, variable blocks for motion and DC update) [553], Sharp (object scalability, shape-adaptive wavelet with zerotree) [583], and Lehigh University (coding efficiency, vector wavelet with lattice VQ for Intra-frame) [592].

The evaluation of the algorithms confirmed that H.263 was a good candidate to provide basic video compression functionality. Furthermore, it was shown that content-based functionalities can be achieved without a significant loss in picture quality. Especially for scenes with graphics objects like text overlay, content-based encoding of video allows for a significant improvement in picture quality.

6. Conclusions

This paper describes the evaluation process and results of the MPEG-4 tools evaluation in November 1995 and January 1996. Furthermore, the evaluation of algorithms in January 1996 is covered.

87 tools covering all aspects of image sequence coding like compression, content-based functionalities like object scalability, temporal scalability were evaluated according to the functionality they address.

Especially properties of a tool like efficacy, adaptability to different scenes and bitrates as well as syntax requirements were reviewed. To compare the tools in a coding environment, the tools based on similar techniques were grouped into 19 areas of core experiments. This grouping eases the comparison of tools. Furthermore, it helps collaboration and allows a mutual stimulation for further improvement of tools. Several months after the evaluation, the achievements of this can already be seen. Especially shape coding techniques have evolved and outperform any algorithm known in January 1996. A similar example is encoding of texture for arbitrarily shaped regions. In the area of texture coding, DCT coding as well as wavelet transform coding have evolved and it is currently not foreseeable which technique is the better performer in terms of coding efficiency.

Initially the call for the submission of tools to MPEG-4 was controversial. However, at this point in time there is the general agreement that due to this evaluation of tools, progress in the development of an MPEG-4 video algorithms was accelerated significantly. Additionally, several organizations new to MPEG were attracted, thus broadening the area of expertise in MPEG-4 video.

The evaluation of algorithms provided an excellent overview of the techniques submitted in January 1996. Unfortunately, this overview is not available for the subjective tests in November 1995. However, the November 1995 tests provided a firm ranking of different proposals which is missing for the algorithm submissions in January 1996. The evaluation in January 1996 confirmed that H.263 was a good choice for providing basic image compression functionality. Starting from H.263, the MPEG-4 VM now contains many tools from MPEG-1, MPEG-2 and more importantly, all the video tools like shape coders and region-based texture coders required for achieving content-based functionalities.

Several proposed techniques require more computational power than existing image coding standards like H.261, H.263 or MPEG-1. They achieve higher compression or allow for additional functionalities and increased flexibility. Since image coding is not as computation-bound as it used to be, limitations of today's image coding standards due to complexity have to be reconsidered.

7. Acknowledgment

The author would like to thank Dr. Amy Reibman and Prof. F. Pereira for their reviews. Thomas Wiegand kindly provided Figure 10. A special thank you goes to the numerous people who submitted proposals for evaluation and helped in the process of evaluating them.

8. References

8.1 General

[1] Jörn Ostermann, “Report on the Ad Hoc Group on the Evaluation of Tools for non tested Functionalities of Video Submissions”, ISO/IEC JTC1/SC29/WG11 N1064, Dallas meeting, November 1995.

[2] Jörn Ostermann, “Report on the Ad Hoc Group on Evaluation of Tools and Algorithms of video submissions for MPEG-4 in January 1996”, ISO/IEC JTC1/SC29/WG11 N1162, München meeting, January 1996.

[3] “MSDL working draft 1.2”, ISO/IEC JTC1/SC29/WG11 N1331, Tampere meeting, July 1996.

[4] “MPEG-4 Video Verification Model Version 4.0”, ISO/IEC JTC1/SC29/WG11 N1380, Chicago meeting, October 1996.

[5] Atul Puri, “Report of ad hoc group report on coordination of future experiments in MPEG-4 video”, MPEG96/669, München meeting, January 1996.

[6] “MPEG-4 test and evaluation procedures”, ISO/IEC JTC1/SC29/WG11 N0999, Tokyo meeting, July 1995.

[7] “MPEG-4 additional call for proposals and proposal package description”, ISO/IEC JTC1/WG11 N1108, Dallas Meeting, November 1995.

[8] T. Alpert, V. Baroncini, D. Choi, L. Contin, R. Koenen, F. Pereira and H. Peterson, "Subjective evaluation of MPEG-4 Video codec proposals: methodological approach and test procedures", Signal Processing: Image Communication, this issue.

[9] T. Sikora, S. Bauer, B. Makai, "Efficiency of shape-adaptive transforms for coding of arbitrarily shaped image segments", IEEE Trans. on Circuits and Systems for Video Technology, Vol. 5, No.3, June 1995.

[10] O. Avaro, "The MPEG-4 Systems and Description Languages: A Way ahead in Audio Visual Information Representation”, Signal Communciation: Image Communication, this issue.

[11] H. Peterson, “Report of the ad hoc group on MPEG-4 video testing logistics”, ISO/IEC JTC1/SC29/WG11 MPEG95/532, Dallas meeting, November 1995.

[12] Thomas Wiegand, “Result from MPEG-4 video core experiment P5”, personal communication, October 1996.

[13] Touradj Ebrahimi, Franck Bossen, Roberto Castagno, Carmen De Sola, Corinne Le Buhan, Laurent Piron, Emmanuel Reusens, Vincent Vaerman, “Dynamic coding of visual information”, ISO/IEC JTC1/SC29/WG11 MPEG95/320, Dallas meeting, November 1995.

[14] Jörn Ostermann, Marc Beutnagel, “An interface for the animation of human heads from text”, ISO/IEC JTC1/SC29/WG11 MPEG96/1197, Chicago meeting, July 1996.

[15] Jörn Ostermann, “Object-based analysis-synthesis coding based on the source model of moving rigid 3D objects”, Signal Processing: Image Communication, No. 6, pp. 143-161, 1994.

8.2 Contributions to ISO/IEC JTC1/SC29/WG11 Dallas Meeting (MPEG 95/)*

0305 Eishi Morimatsu, Kimihiko Kazui, “Technical description of background estimation tool for video content based manipulation”.
0308 F. Seytter, B. Wimmer, “Hybrid ARQ-type II error protection tool”.
0309 Klaus Illgner, Frank Mueller, “Hierarchical very low bit rate video coding; part I - motion estimation and displacement vector field coding”.
0310 Frank Mueller, Klaus Illgner, “Hierarchical very low bit rate video coding; part II - displaced frame difference coding”.
0313 Raj Talluri, “Error protection scheme for transmitting data over noisy communication channels”.
0318 Bossen Frank, Ebrahimi Touradj, “Region shape coding”.
0319 Gu Chang, Ebrahimi Touradj, Kunt Murat, “Morphological moving object segmentation and tracking for content-based video coding”.
0323 Andreas Hütter, Deyu Qian, “Block-based subband coding”.
0325 Weiping Li, H.Q.Cao, S.Li, W.Li, F.Ling, S.A.Segan, H.Sun, J.P.Wus, Y.Q.Zhang, “Motion compensation for video coding”.
0326 Weiping Li, H.Q.Cao, S.Li, W.Li, F.Ling, S.A.Segan, H.Sun, J.P.Wus, Y.Q.Zhang,, “Vector-based intraframe video coding”.
0327 Weiping Li, H.Q.Cao, S.Li, W.Li, F.Ling, S.A.Segan, H.Sun, J.P.Wus, Y.Q.Zhang,, “Vector-based interframe video coding”.

0328 Weiping Li, H.Q.Cao, S.Li, W.Li, F.Ling, S.A.Segan, H.Sun, J.P.Wus, Y.Q.Zhang, "Segmented motion-compensated interpolation of coded video".
0329 Raj Talluri, "A motion segmentation tool".
0332 Klaus Stuhlmüller, "Contour description with parabolic-blending-curves".
0333 Jens-Rainer Ohm, "Three-Dimensional Subband Coding with Motion Compensation".
0334 Graham Martin, Roger Packwood, Injong Rhee, "Variable size block matching motion estimation".
0340 Kohtaro Asai, Takahiro Fukuhara, "Object-based video coding with hybrid segment-pattern MC".
0343 Nobuya Suzuki, Takeshi Mogi, Kouta Fujimura, "Hierarchical color segmentation method for low bit-rate video coding".
0344 Nobuya Suzuki, Takeshi Mogi, Kouta Fujimura, "Hierarchical region representation and adaptive image reconstruction".
0355 Yuji Itoh, "Edge-oriented progressive image coding".
0356 Yuji Itoh, "Edge-based multilayering scheme".
0357 Yuji Itoh, "Detail preserving nonlinear filter using binary index".
0358 Roberto Castagno, "A nonlinear adaptive technique for the removal of blocking effect in image sequences coded at very low bitrate".
0360 Peter Gerken, Michael Wollborn, Stefan Schultz, "Polygon/spline approximation of arbitrary image region shapes as proposal for MPEG-4 tool evaluation - Technical description".
0364 Delmot, Thierry, "Adaptive block matching algorithm".
0366 Peter List, L. Ott, "Object tracking tool for non-rigid objects".
0367 A. Kopernik, B. Chupeau, "View-point synthesis".
0368 A. Kopernik, B. Chupeau, "Depth based segmentation".
0369 Thierry Delmot, "Split and merge algorithm".
0370 Thierry Delmot, "Postprocessing by overlapping interpolation functions in multiresolution decompositions of images".
0371 Thierry Delmot, "Multi-Huffman entropy coder".
0372 Thierry Delmot, "Interesting regions determination".
0378 Tomoko Aono, Norio Ito, Hiroyuki Katata, Hiroshi Kusao, "Object based Wavelet Transform (OWT) tool for video".
0394 Minoru Etoh, "Matsushita proposal for video tools and algorithm".
0406 Christine Guillemot, "Motion analysis tools".
0407 Henri Sanson, "Region-based parametric motion estimation".
0410 Giorgio Parladori, Jose Mir, Giancarlo Calvagno, Roberto Rinaldo, Carla Zecchinato, "Video coding tools proposal for MPEG-4".
0411 Olivier Avaro, Marie Dudon, Gerard Eude, Christian Roux, "Active Triangular Meshes for Motion Estimation And Representation".
0423 Tsuhan Chen, "Coding of segmented regions for content-based scalability".
0424 Tsuhan Chen, "Better rendition of lip synchronization".
0436 T.Chiang, Y.-Q.Zhang, J.Lee, S.Martucci, H.Peterson, I.Sodagar, R.Suryadevara, C.Wine , "A rate control scheme using a rate-distortion model".
0437 I.Sodagar, T.Chiang, J.Lee, S.Martucci, H.Peterson, R.Suryadevara, C.WiQ.Zhang, "A flexible wavelet transform package for image and video representation".
0438 J.Lee, Y.-Q.Zhang, T.Chiang, S.Martucci, H.Peterson, I.Sodagar, R.Suryadevara, C.Wine, "Variable-block-size motion estimation for video compression".
0439 I.Sodagar, T.Chiang, J.Lee, S.Martucci, H.Peterson, R.Suryadevara, C.Wine, Y.-Q.Zhang, "Embedded zero-tree wavelet coding for video compression".
0440 R.Suryadevara, M.Irani,T.Chiang,J.Lee,S.Martucci,H.Peterson,I.Sodagar,C.Wine,Y.-Q.Zhang, "a global motion estimation/compensation scheme".
0441 S.Martucci,T.Chiang,J.Lee,H.Peterson,I.Sodagar,R.Suryadevara,C.Wine,Y.-Q.Zhang, "A zero-tree entropy coding tool for wavelet compression of video".
0443 Homer H. Chen, Barry G. Haskell, "Description of AT&T TrueGaze tool".
0444 A. Murat Tekalp, Yucel Altunbasak, "Object-scaleable, content-based 2-d mesh design and tracking for object-based video coding".
0445 Raj Talluri, "Shape coding scheme for object-based video encoder".

0447 Kevin O'Connell, Damon Tull, “Motorola MPEG-4 contour-coding tool technical description”.
0459 J. H. Moon, S. M. Chun, G. H. Park, J. H. Lee, “Effective shape adaptive region partitioning(sarp) methods by varying block grid positions”.
0460 J. H. Moon, K. Y. Yoo, S. H. Lee, J. K. Kim, “Progressive motion estimation based on 6- and 2-motion parameters for object-based video coding”.
0461 J. H. Moon, J. W. Chung, J. K. Kim, “Shape information reduction based on contour prediction and shape coding type”.
0464 Eric Petajan, Hans Peter Graf, “Face feature analysis and communication for face animation in MPEG-4”.
0487 Atul Puri, Barry G. Haskell, Richard V. Kollarits, “Gain corrected stereoscopic coding using sbasic for mpeg4 multiview concurrent streams”.
0489 Atul Puri, Bob Schmidt, “3D-DCT coding tool”.
0539 E. Martinez-Uriegas, “Chromaplex color conversion for MPEG”.
0540 S-C Han, “Joint motion estimator/segmentator”.
0541 P. Letray et al., “Enhanced segmentation using the cooperative perceptive grouping process based on a simulation of the visual cortical system”.
0544 P. Chou et al., “A video synthesizer tool and related objects”.

8.3 Contributions to ISO/IEC JTC1/SC29/WG11 München Meeting (MPEG 96/)*

0552 Ulrich Benzler, Oliver Werner, “Motion and aliasing compensating prediction with quarter-pel accuracy and adaptive overlapping blocks as proposal for MPEG-4 tool evaluation - Technical description”.
0553 Michael Zeug, John Muller: “MPEG-4 Video Submission Technical Description”.
0554 C. S. Park, J. R. Kim, J. I. Kim, J. T. Lim, D. D. Hwang, J. H. Kim, H. S. Kim, K. H. Chang: “Daewoo algorithm for object scalability”.
0555 C. S. Park, J. R. Kim, J. I. Kim, J. T. Lim, D. D. Hwang, J. H. Kim, H. S. Kim, and K. H. Chang, “Daewoo proposal for region texture coding”.
0557 R.A. Beuker, R. Heusdens, A. Kalker, H. Theunis, “Adaptive filter banks with segmentation”.
0564 Jae-Seob Shin, Shi-Hwa Lee, Sung-Gul Ryoo, Seong-Jin Kim, Yang-Seock Seo, “Video Compression Algorithm using Motion Segmentation and Color Perception”.
0565 Yu-Shin Cho, Shi-Hwa Lee, Jae-Seob Shin, Yang-Seock Seo, “Shape Coding Tool: Using polygonal approximation and reliable residue error sampling method”.
0566 Sung-Gul Ryoo, Seong-Jin Kim, Yang-Seock Seo, “Rate Control Tool: Based on Human Visual Sensitivity for Low Bitrate Coding”.
0571 G.Russo, S.Colonnese, A.Neri, P.Talone, “Moving objects versus still background classification: a spatial temporal segmentation tool for MPEG-4”.
0573 Touradj Ebrahimi, Sushil Bhattacharjee, Franck Bossen, Roberto Castagno, Carmen De Sola, Corinne Le Buhan, Laurent Piron, Emmanuel Reusens, Vincent Vaerman, “Improved Implementation Dynamic Coding of Visual Information”.
0582 Hiroyuki Katata, Norio Ito, Hiroshi Kusao, “Video coding algorithm for compression efficiency”.
0583 Tomoko Aono, Norio Ito, Hiroyuki Katata, Hiroshi Kusao, “Wavelet based video coding algorithm for object scalability”.
0586 Keiichi Hibi, Nobuyuki Ema, Seiji Sato, “Video coding algorithm using adaptive 2D-triangle mesh based prediction”.
0587 Keiichi Hibi, Nobuyuki Ema, Seiji Sato, “Wavelet based layered algorithm for error resilience”.
0592 Weiping Li, H.Q.Cao,S.Li,W.Li,F.Ling,S.A.Segan,H.Sun,J.P.Wus,Y.Q.Zhang: “A Video Coding Algorithm Using Vector-Based Techniques”.
0595 Shinichi Sakaida,Yoshiaki Shishikui, “Spatial Segmentation using K-means algorithm”.
0596 Yoshiaki Shishikui,Shinichi Sakaida, “Region Support DCT(RS-DCT)”.
0599 Yoshihiro Miyamoto, Yotaka Yokoyama, “Video Coding Algorithm using Adaptive Warping Prediction - Technical Description”.
0600 Harald.Brusewitz, Goran.Bang, “A tool for generating bit error resilient fixed length code tables”.

0601 Harald.Brusewitz, “Low complexity encoder and error resilience”.
0602 Karsten Schröder, Holger Hornig, “Grid Based Motion Compensation and Shape Coding Using Curved Triangles”.
0609 Y. Nakaya, A. Date, S. Misaka, Y. Suzuki, “Technical description of Hitachi’s proposed video coding algorithm Ver. 2”.
0615 Hirohisa Jozawa, Kazuto Kamikura, Atsushi Sagata, “Technical description of video proposal for MPEG-4 algorithm evaluation”.
0616 Gisle Bjontegaard, “An error resilience method based on back channel signaling and FEC”.
0617 Gisle Bjontegaard, “A simple edge loop filter to reduce blocking and mosquito noise”.
0620 T. Ebrahimi, O. Egger, M. Kunt, “Arbitrarily shaped region interior coding using embedded zerotrees”.
0623 Faramarz Azadegan“Use of inter-block compression to improve coding efficiency of intra-coded frames”.
0624 Faramarz Azadegan“Alternative to Border Extension in Image/Object Filtering”.
0625 Jean-Francois Vial, Edouard Francois: “Technical description of the video coder”.
0637 Iraj Sodagar, Ya-Qin Zhang, ”Very Low Bit Rate Video Coder (Algorithm submission to MPEG4)”.
0639 Ming-Chieh Lee, Wei-ge Chen, Chuang Gu, Bruce Lin, Steve Zabinsky, Rick Szeliski, “Microsoft Proposal for MPEG4 2nd Evaluation Phase”.
0646 Gary J. Sullivan, Rick S. Grinnell, Henrique S.Malvar, Wilson C. Chung, “An MPEG-4 Video Coding Proposal Using a Modulated Lapped Transform Enhancement of H.263”.
0650 R. Nagarajan, R. R. Burns, P. Au, “Low Bit Rate Object Based video coding using Warping techniques”.
0653 Frederic Dufaux, “Background mosaicking”.
0654 Kohtaro Asai, Takahiro Fukuhara, “Core Experiments of Video coding with block-partitioning and adaptive selection of two frame memory (STFM / LTFM)”.
0657 Thierry Delmot, “Region-enhanced H.263 coder : A Trial”.