Real-time Estimation of Subjective Utility Functions for MPEG-4 Video Objects

Paul Bocheck, Yasuyuki Nakajima, and Shih-Fu Chang

{bocheck, sfchang}@ctr.columbia.edu

Department of Electrical Engineering

Columbia University, New York, U.S.A.

nakajima@spg.lab.kdd.co.jp

Multimedia Communications Laboratory

KDD Research Laboratories

Saitama, Japan

Abstract

In this paper, we propose an original system for a subjective *utility function* estimation suitable for real-time video applications. In video networking applications, utility function is defined as a video quality metrics representing a user satisfaction index as a function of bandwidth. The real-time performance is achieved in our system by (1) using novel content classification techniques based on visual features directly extracted from compressed video streams and (2) employing machine learning techniques for classification of utility functions. Our extensive experiments, based on MPEG-4 encoded video streams, indicate that high accuracy of 80%-85% in estimating utility functions can be achieved. These results demonstrate that the proposed system represents a promising approach to real-time utility function estimation. The real-time performance is instrumental in achieving higher network resource utilization leading to flexible and less expensive interactive multimedia services. The proposed technique is general and can be applied to other video compression standards such as MPEG-1, MPEG-2, and H.263. An important application of the proposed system is in future low bandwidth and wireless networks that can support and benefit from content-aware media scalability.

Keywords: bandwidth utility function, MPEG-4, subjective video quality

1 Introduction

Multimedia has become one of the most dynamic fields in today's information technology. Widespread availability of Internet has escalated development of new, network-oriented applications. Additionally, digital video encoding standards, such as MPEG-1, MPEG-2, H.261, and H.263, have opened the previously undiscovered world of real-time video communications. While H.261 and H.263 are successfully used in several video communication applications, these standards are frame-based and their suitability for low bit-rate and wireless communications is somewhat limited. New functionalities that are directly applicable for low bit-rate and high error-rate communications are explicitly provided in emerging MPEG-4 encoding standard; they include content-based scalability and error control. MPEG-4 is suitable for a wide range of applications. It was designed according to requirements of both storage and communication applications. The most distinctive feature of MPEG-4 is its ability to independently encode video objects (VO) appearing in a scene. Object-based treatment of video sequences enables employment of a new type of media scaling technique called *content-based scalability*.

It is generally accepted that video quality can be improved by exploiting video content scalability through rate control coupled with media scaling techniques [13]. Additionally, significant increase of network utilization can be achieved using dynamic bandwidth allocation [4]. These techniques are well suited for transport of video in conditions of time-varying bandwidth limitations that can be found in the wireless networks and Internet.

In networks that support adaptive bandwidth allocation techniques (e.g., content-aware media scaling and re-negotiation), video applications can characterize its multimedia content adaptability to bandwidth variations through *utility functions* (UF). Utility function is defined as a video quality metrics (e.g., subjective or objective) representing a user satisfaction index as a function of bandwidth. Consequently, applications can effectively describe its media scaling capability by using utility functions. Utility functions, previously used in economics research, are successfully applied for utility-fair network resource allocation [1, 14, 9]. Additionally, subjective utility functions can be efficiently used for (1) bit rate control and (2) dynamic rate shaping (DRS) [12] in network nodes.

Utility functions of various video components comprising single application are combined in a media-scaling profile. By the video component we mean any audio-visual multimedia object that is part of a multimedia presentation and can be delivered, in general, to the receiver by a separate stream. Additionally, the media-scaling profile contains media scaling preferences that are set individually by each application, priority of each video object in terms of their relative importance, preferred video quality for each priority, etc. Such content description facilitates hierarchical network bandwidth allocation model based on two layers: (1) utility-fair resource allocation between independent video applications and (2) optimal allocation of available bit budget among various video components of each application [15].

However, efficiency of utility-fair bandwidth allocation depends on (1) actual quality metrics

used and (2) feasibility of real-time utility function generation. Although previous studies in QoS indicated the importance of utility function as a soft media scaling indicator [1, 14, 9], little attention was given to actual generation of utility function online, especially in the context of subjective video quality.

In this paper, we propose a novel system for a real-time subjective *utility function* estimation suitable for future MPEG-4 networking applications. In Section 2, we discuss in detail a subjective video quality evaluation metrics and an effective method of utility function generation in the context of MPEG-4. In Section 3, we introduce a novel framework for utility function classification. In particular, we use video content to classify video objects into different utility classes, each of which is associated with distinctive utility functions. In the last section, we discuss our experimental results of content-based approaches for utility function generation and compare several classification models developed for MPEG-4.

For simplification, in the rest of this paper, we use the term "video object" synonymously to refer to video object plane (VOP) in MPEG-4 terminology.

2 Utility function

The MPEG-4 video-encoding standard gives an efficient representation of audio-visual objects of arbitrary shape. It supports most of the functionality already provided by MPEG-1 and MPEG-2 in its core video compression mechanism that is based on block-based DCT algorithms. One of the most notable new features supported by MPEG-4 is separate encoding of video objects appearing on the scene. Especially this new feature promotes new media scaling functionality, namely content-based scalability.

In MPEG-4, content-based scalability can be employed on top of quality, spatial and temporal scalability. In fact, each individual video object in MPEG-4 can be independently scaled to the point that it is replaced by a static icon or it is completely removed. Consequently, total bit budget assigned to the video scene (e.g., scene containing more than one video object) is distributed among individual video objects such that optimal subjective quality is achieved.

Given subjective quality of individual video objects, subjective quality of video scenes can be estimated from subjective quality of each individual video object and its priority. Priority corresponds to user's subjective estimation of object's relative importance in terms of visual quality. In practice, video object priority may be determined based on heuristics that take into account its content features (e.g., object position, speed, complexity, etc.). Alternatively, video object priority can be explicitly indicated by user. In this paper, our work focus on utility function estimation of individual video objects. The utility function of these individual video objects can be used to optimally allocate network resources assigned to application.

Video compression causes degradation in quality of video content. User's satisfaction with the

quality is measured by "quality indicators", also called utility functions. The simplest quality indicator is the binary (e.g., "yes/no") indicator that expresses only satisfaction/dissatisfaction with currently offered quality of service [1]. In general, utility functions have a functional form depending on many variables. Specifically, in video networking applications the quality indicator, called bandwidth utility function, is defined to be a function of a single variable: bandwidth. In that case, the bandwidth utility function represents relationship between user's satisfaction index and the network bandwidth. For example, 5-level satisfaction index called mean opinion score (MOS) was defined by ITU as a measure of subjective video quality [3]. The MOS-based utility function allows simple and efficient expression of media scalability. For simplification, in the rest of the paper, we use term "utility function" (UF) to refer to "bandwidth utility function".

In general, estimation of utility functions for compound video scenes and individual video objects is difficult. For simplification, utility functions are usually specified in parameterized form. Consequently, they can be estimated from limited number of sampling points. Resulting utility function can be, if necessary, computed by interpolation. Each individual sampling point is estimated using particular quality model. Unfortunately, simple quality models such as SNR (signal to noise ratio), PSNR (peak signal to noise ratio), and WSNR (weighted SNR) do not correspond to visual quality perceived by humans well. On the other hand, quality models that are based on human visual system can achieve results that are more accurate. However, high computational complexity makes their real-time implementation difficult. Additionally, these models typically require presence of both original and encoded video objects side by side for human to compare their quality. In addition, to estimate subjective utility function, it is necessary to compute video quality multiple times with different encoder parameters such as quantization steps. This process of multiple calculation also causes additional complexity in utility function evaluation. To our best knowledge, a system that allows efficient estimation of subjective utility function in real time does not exist. We will demonstrate feasibility of such a system using the proposed content-based classification technique.

To measure the subjective utility functions of MPEG-4 video objects, we adopted "Picture Quality Assessment System" developed by Hamada [2]. Their hardware-assisted standalone system computes the picture quality for the compressed video stream in real-time. The quality estimation process is based on human visual system that is modeled at three-layers including object, texture and local noise. The distortion residuals after comparing original and encoded pictures are weighted at each layer and aggregated to the higher layers. At the object layer, the weighted noise is summed over the whole object and converted into distortion scale recommended by ITU-R Rec. 500-7 [3]. It was reported that the assessment conducted using the system on three different codecs was highly correlated (0.8 - 0.9 correlation) with ITU standardized subjective tests with 20 viewers.

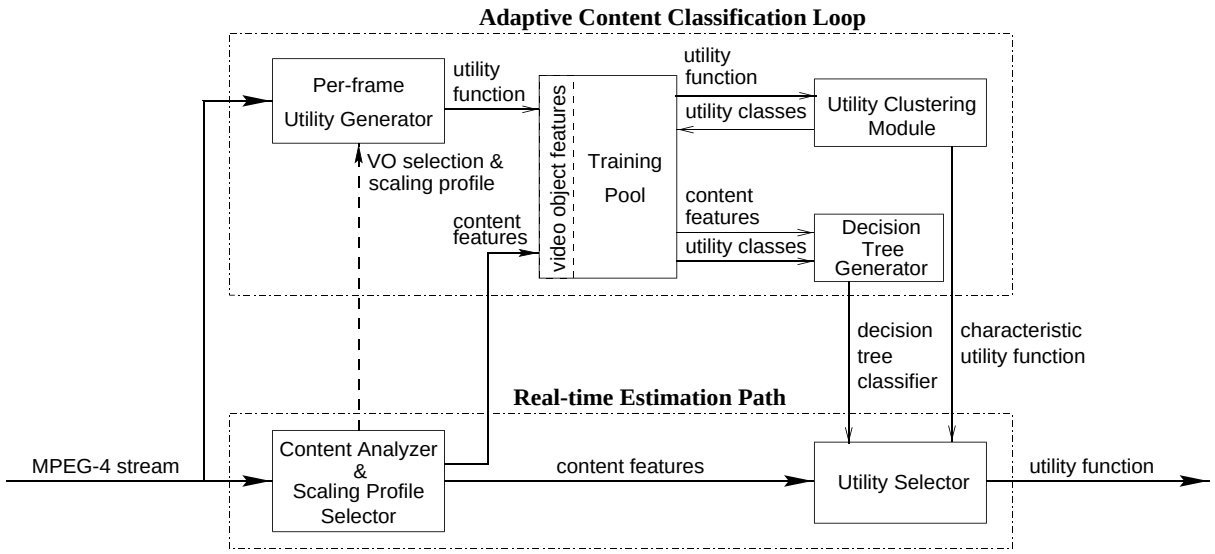

Figure 1: Content-based utility function estimator.

3 Content-based utility function estimator

In general, direct evaluation of bandwidth utility function is relatively complex and requires large amount of processing power. With currently available technology, direct frame-by-frame utility function generation is difficult to achieve in real-time. Consequently, new approaches simplifying the utility function computation are needed. In this section, we describe a novel technique intended for *estimation* of utility functions based on video content. The proposed technique does not rely on explicit generation of the utility function for each video object. Instead, it classifies video content in real-time to classes with distinctive utility functions. For simplification, in this section, we use the term "video object" synonymously to refer to either video frame (for frame-based encoding schemes such as MPEG-1, MPEG-2, H.263, etc.) or video object plane (VOP) in MPEG-4 terminology.

One possible role of such a system in the network is in source nodes. The utility function, generated by the system, will be used for utility fair bandwidth allocation between scalable streams [9]. In that case, the estimated utility function will be sent to other network nodes in the transmission path. There it will be used for bandwidth allocation and scheduling. A discussion of corresponding networking protocol can be found in [9].

Figure 1 illustrates the proposed architecture for real-time MPEG-4 subjective utility function estimation. With some modifications, the system is also appropriate for a variety of compression techniques including MPEG-1, MPEG-2, H.263 etc. The system architecture comprises two main entities: an *adaptive content classification loop* (also referred to as adaptation loop) and a *real-time estimation path*. The adaptive content classification loop is relatively complex and consists of the utility generator, training pool, utility clustering module, and decision-tree generator. However, the real-time estimation path comprises the content analyzer and the utility selector only.

The architecture of the system was designed to be flexible and to allow smooth system adaptations over various video content types. Both the adaptive content classification loop and the real-time estimation path operate asynchronously. In particular, based on online extracted content features, the real-time estimation path estimates utility function for each video object. In contrast, an adaptation loop is intermittently activated to re-compute decision tree parameters for the use in the real-time estimation path. Explicit computation of the utility function is performed at the adaptation loop for selected video objects only. This way, the architecture decouples the adaptation loop that is computationally intensive from the real-time estimation path; that is, the system facilitates operation in real-time by avoiding explicit per-object generation of utility functions that would otherwise require intensive computation.

3.1 Real-time estimation path

A real-time estimation path allows estimation of utility function on a frame-by-frame basis. The estimation is based on content features extracted by the content analyzer that operates on the compressed video stream. The operation of the real-time estimation path can be summarized in the following way.

Assume that the system is initialized and the adaptation loop has already estimated decision tree parameters needed by the real-time estimation path. First, content-related and MPEG-4 specific features are extracted in real-time from the compressed video stream by the content analyzer. Second, the utility selector (as illustrated in Figure 1) uses these features to determine the *utility class* of the current video object. The selection is based on the decision tree that is periodically updated in the adaptation loop. Finally, once the class is determined, *characteristic utility function* for that class (determined initially during the training period) is used as an estimator of the utility function for the video object. Note that since content features are extracted from the compressed video stream in real time, the system is able to estimate utility function correspondingly in real time. Following is a detailed description of the content analyzer and the utility selection modules.

The content analyzer parses the encoded video stream and extracts meaningful visual content information for each video in real time. The content information comprises visual and encoder-specific features such as object size, visual complexity, color, object speed, peak-SNR, average background and object motion, etc. It is possible with the current technology to extract content features (both visual and encoder-related) from the compressed video stream in real-time [10]. The use of content features for video traffic modeling was also proposed in [4]. Content-modulated video traffic modeling is based on close relation between video content and bandwidth requirements.

Additionally, the content analyzer selects some video objects for further evaluation in the adaptation loop. The selected video objects are (1) directly evaluated for computing utility function and (2) placed together with its content features in the training pool. The algorithm that selects which video object will used in the adaptation loop is out of the scope of this paper. In general, the video

object that indicates "substantial change" in its visual content will be used.

The decision tree is executed in the utility selector that is part of the real-time path. Once the utility class of video object is determined, the characteristic utility function for that class is used as an estimator. Additionally, the utility estimator may forecast utility function at different time scales by combining and filtering estimations for a given window size (for example GOP in MPEG). The predicted utility functions are transmitted along with the compressed video stream to the network nodes. There, utility functions are used for network resource management and media scaling in the case of bandwidth variations.

3.2 Adaptive content classification loop

The operation of the adaptation loop is computationally intensive and difficult to apply to every video object. Consequently, the adaptation loop is invoked only intermittently and is separated from the real-time estimation path. The adaptation loop uses only a subset of video objects stored in the training pool to re-compute the decision tree as illustrated in Figure 4. Given the nature and accuracy of algorithms for subjective or objective estimation of video quality, it is reasonable to include only some characteristic video objects in the training pool. The selection of these objects is based on heuristics. Video objects could be selected periodically, or when their content features have substantially changed. For example, video objects that are selected from different scenes tend to have their content features substantially different from each other. The selection of video objects that are included in the training pool is controlled by the content analyzer. The chosen video object is first evaluated for its utility function in the utility generator and later placed together with its content features in the training pool.

A training pool contains video objects that were selected and evaluated by the content analyzer. Each video object in the training pool is associated with three different sets of parameters: (1) content features, (2) parameters describing utility function and (3) their utility class. Content features are evaluated by the content analyzer. Parameterized utility function is computed by the utility generator. Utility class is obtained from the utility clustering module. The utility clustering module uses only parameters describing utility function in its operation.

In our experiments, the utility generator adopted method for subjective object quality assessment. The method is based on a model of human visual system introduced by Hamada in [2]. The generator computes the utility function according to the encoding method and scalability profile used for the video object. The scalability profile indicates user's preferences in terms of which scalability options are enabled, their sequential invocation, etc. Alternatively, the utility generator can be based on objective quality estimator as presented in [11]. The utility generator uses the original video stream for computation of utility function. If the original video stream is not available, the utility generator may only scale-down the video stream. In that case, the utility function is determined under additional assumptions about the current subjective quality.

An important component of the whole adaptation loop is the utility clustering module. It operates autonomously on the training pool, and clusters utility functions of different video objects stored in the pool into a small number of utility classes. The automated clustering is performed by using unsupervised classification algorithms operating on selected features of the utility function [6]. Examples of several classification schemes are discussed in the next section. Derived directly from the utility functions within the same utility class, the *characteristic utility function* of each class is used as a utility function estimator of that class in the real-time estimation path. A smooth adaptation is assured by continuous replacing and re-clustering new video objects in the pool. Every time when clustering is completed, all video objects in the training pool are marked according to the class they belong to. The algorithm for generation of the decision tree then uses the marked video objects in the pool as training data to form the decision tree for the utility selector.

The decision-tree generator starts its operation every time after re-clustering is completed. The operation is also based on machine learning techniques, in particular, on supervised classification algorithm [7]. The generator determines the decision tree by using utility classes derived by the utility clustering module and content features extracted by the content analyzer. Note that its does not use parameters describing utility function. Once a decision tree is formed, the utility class of a particular video object can be determined by using only the content features of the video object. This operation, in fact, is performed in the utility selector that uses the decision tree to estimate the utility function. In this case, the characteristic utility function of the class serves as the estimation of the utility function characterizing the video object. Content features are extracted from the compressed video stream in real time thus enabling real-time utility estimation.

4 Utility function estimation experiment

In this section, we discuss our implementation of a content-based utility function estimator and several experiments that were used to verify its function and, in general, feasibility of the proposed system. For simplification, the experiments were performed offline based on an original MPEG-4 encoded video sequence. We focus our discussions on utility clustering module, decision tree generator and utility selector. In particular, we compare two structurally different implementations of the utility clustering module based on machine learning techniques.

4.1 Experiment setup and implementation details

4.1.1 MPEG-4 video object stream preparation

Initially, we have prepared seven MPEG-4 video object streams in the following way. Original video sequence was concatenated from 38 high quality CIF video shots of 100 to 300-frames long each. First, each shot was segmented into two or more arbitrary-shaped video objects using a system

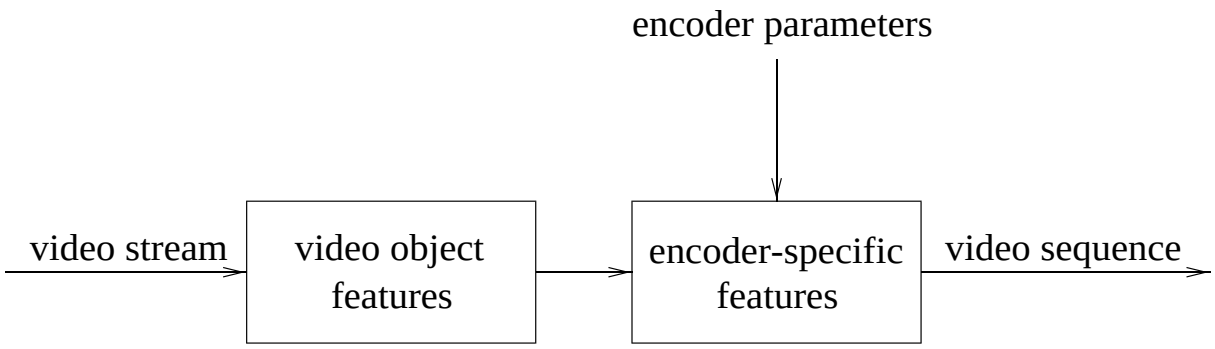

Figure 2: Video content features.

for semi-automatic video segmentation, described in [5]. Second, each segmented video object was encoded using MPEG-4 VM8 software [8]. Finally, the resulting streams corresponding to each video object were concatenated to form a single 10734-frame long video object stream. Seven such video object streams were prepared, each corresponding to different DCT quantization coefficient $q = 1, 5, 10, 15, 20, 25, 31$.

Video object streams were VBR-encoded with following parameters: lossless shape coding (alpha threshold = 0, changeCRdisable = 0), binary alpha channel, 8 bits/pixel, and the following options enabled: error resilience, data partitioning, reversible VLC, AC/DC prediction and SADCT coding. In our experiment, we selected a lossless shape because each video object was encoded independently. Lossy encoding of independently encoded objects belonging to the same original frame created shape artifacts at the composition layer due to misalignment of edges of objects.

In our offline simulation experiments of content-based utility estimator, one set of video objects was used for training the utility clustering module and generation a decision-tree. The second set was used to obtain the classification results in the utility selector. We have randomly selected half of the video objects and placed them in the training pool thus bypassing the video object selection algorithm described in Section 3.

4.1.2 Content analyzer

In general, the content analyzer can extract (1) video object features and (2) encoder-related features (Figure 2). Video object features describe uniquely video object characteristics that do not change if different encoding technique is used (e.g., video object size, speed, etc.). In contrary, encoder-related features can change depending on encoding technique and encoder parameters (e.g., frame type, average energy of AC DCT coefficients, number of bits for various encoder-specific stream components, etc.). While some of the features can be extracted directly from compressed video streams (object size, picture type, etc.), other features can be obtained from the encoder only (e.g., PSNR). Consequently, selection of appropriate content features depends on location of content analyzer (e.g., at the server or network node).

Accurate estimation of video object features is difficult. It typically requires evaluation of the original video sequence at spatial domain. However, in networking applications high accuracy of estimation of these features is generally not necessary. Consequently, these features can be estimated with limited accuracy directly from compressed video streams.

We have simulated function of the content analyzer offline. Content features were extracted directly from each one of the MPEG-4 encoded video object streams and associated encoder logs. Following video object features were used: object size (in pixels), average and variance of motion vectors, and average energy of AC DCT coefficients. Our experiments have shown that accuracy of the correct recall can be improved by including the encoder-related features in the feature vector. The following MPEG-4 encoder-specific features and parameters were used: quantization parameter q, frame type (I, P, B); number of bits for shape, motion, texture, and headers. These features can be extracted relatively easily from the compressed video stream at both server and network nodes in real time. Additionally, we have used PSNR that was computed by encoder. This feature can be used only when original video stream is available (i.e., at the server side).

4.1.3 Utility generator

A parameterized form of subjective utility function was obtained for each video object directly from MPEG-4 video object streams by the modified Picture Quality Assessment System [2]. To support utility estimation corresponding to individual MPEG-4 video objects, the original Picture Quality Assessment System was modified in the following way.

The modified system use three CIF streams as an input: (1) an original video sequence, (2) a decoded MPEG-4 video object sequence corresponding to MPEG-4 video object stream encoded at given quantization coefficient and (3) a binary video object mask sequence obtained from semi-automatic video segmentation system [5]. The binary video object mask selects the region over which the quality score was computed. The output from the system is mean opinion score (MOS) for MPEG-4 video object in a range from 1 (lowest) to 5 (highest) [3]. Similarly, utility values (i.e., quality) corresponding to different quantization parameters q are obtained by using the MPEG-4 encoded video object stream (i.e., stream at input 2) with different q. Quality estimations corresponding to video object streams obtained using different quantization coefficients served as sampling points for parameterized utility function.

In our experiment, utility function was defined by seven sampling points x_q corresponding to seven quantization parameters:

$$\mathcal{U} = \{x_q;\ q = 1, 5, 10, 15, 20, 25, 31\},\ x_q = \{r(q), u(q)\} \tag{1}$$

where $r(\cdot)$ denotes rate, $u(\cdot)$ utility, and q the quantization parameter.

4.1.4 Utility clustering module

The utility clustering module clusters the "shape" of utility function of video objects into a set of *utility classes*. The "shape" was specified by the feature vector consisting of discrete values corresponding to sampling points of utility function. We have developed two different methods based on two different utility classification models: composite and joint models.

The first model, called the *composite model*, consists of two independent 9-class classifiers: rate-only and utility-only. According to utility function defined in Equation 1, rate-only classifier is used for classification of rate $r(q)$ and utility-only classifier is used for classification of utility $u(q)$; both $r(q)$ and $u(q)$ are indexed by quantization parameter q. Consequently, the rate-only model uses only a subset of parameters from $\mathcal{U}$, namely its rate components $r(q)$ as a feature vector. Similarly, the utility-only model uses utility components $u(q)$ only. The classification results from both independent classifiers are ortogonally combined to create 81-state composite model. For example, assume that the video object's rate $r(q)$ is classified into class a_r and its utility $u(q)$ is classified into class b_u. In that case, video object classification according to composite model is expressed as $\{a_r, b_u\}$.

The second model, called the *joint model* uses all 14 parameters defining utility function $\mathcal{U}$ as a feature vector. Similarly to the composite model, the joint model classifies utility function into 81 classes. There is a distinction and tradeoff between the use of composite and joint models. Given the same number of classes, the joint model can obtain clustering results more accurately, because it uses full set of utility function parameters as the feature vector. Additionally, it can be directly used for estimation of utility function for video streams under dynamic rate shaping adaptation. On the other hand, the composite model is better suited for encoder controlled adaptation since both of its compound models (rate only and utility only) are indexed in terms of quantization coefficient q.

For flexibility, in our experiments we have used general, publicly available, machine learning tools. In particular, the utility clustering module was realized by Autoclass III [6]. Autoclass III is Bayesian unsupervised classifier that estimates class membership given unlabeled test cases. The classifier was used to find "clusters" in the feature space that correspond to different models. Although Autoclass III chooses number of classes automatically, we override this option by selecting a fixed number of classes. Specifically, we used 9 classes for rate-only and utility-only models and 81 classes for the joint model.

In addition to classification, the utility clustering module computes characteristic utility function for each one of the utility classes. The characteristic utility function of each class is used as a representative prediction for other utility curves in the same class.

4.1.5 Decision-tree generator, classifier and utility selector

After the classification of utility functions, all video objects at the training pool are labeled by class membership. Class membership and content features are used as an input to the decision-tree generator. The generator is based on OC1 software [7], a supervised machine learning system based on oblique decision trees. Note that contrary to the utility clustering module, OC1 uses for classification content features that are easy to obtain from compressed video object streams. Decision trees of this form consist of a linear combination of the attributes at each internal node and can be viewed simply as a more general form of axis-parallel univariate decision trees. Result of classification, namely a decision tree, is used in the utility selector. Once the video object utility class is determined by decision tree in the utility selector, the characteristic utility function of the class is used as an estimator of utility function for that video object. Detailed description of the tools can be found in [6] and [7].

4.2 Results

Figure 3 depicts 500-frame long trace obtained from original 10734-frame long video object sequence that was used in our simulations. Seven traces of the same original video sequence that are shown from top to bottom correspond to different quantization parameters $q = 1, 5, 10, 15, 20, 25, 31$ respectively. Rate variations are shown on the left of Figure 3 and corresponding subjective quality utility variations are shown on the right. For example, top-left graph depicts rate variations in bits/pixel of the stream encoded with $q = 1$. A video stream that is encoded using small quantization parameter results in high bit-rate and high subjective quality. This is illustrated at the top-right graph of Figure 3 by constant utility estimated at level 5 for the whole length of the 500-frame period. On the other hand, bit rate variations that are shown at the bottom-left graph of Figure 3 correspond to stream encoded with quantization parameter $q = 31$. In this case, although the video sequence was encoded as VBR, constant quantization parameter did not result in constant subjective quality. This is illustrated at the bottom-right graph. Figure 3 depicts an important observation: encoding with fixed quantization coefficient (open loop VBR) does not imply constant video quality encoding. Exploration of this property for more efficient encoding that is characterized by constant subjective video quality is beyond the scope of this paper.

In the following, we focus on implementation of the utility clustering module. In particular, at each one of the three experiments we compare two utility clustering models: composite and joint models. In the first two experiments, we include two distinct models using I-frames and P-frames only. In particular, in the first experiment we cluster video objects using I-frames only and in the second experiment, we cluster video objects using P-frames only. In the third experiment, we apply the clustering models to include both I- and P-frames.

As we have already mentioned, composite model consists of two intermediate utility-only and

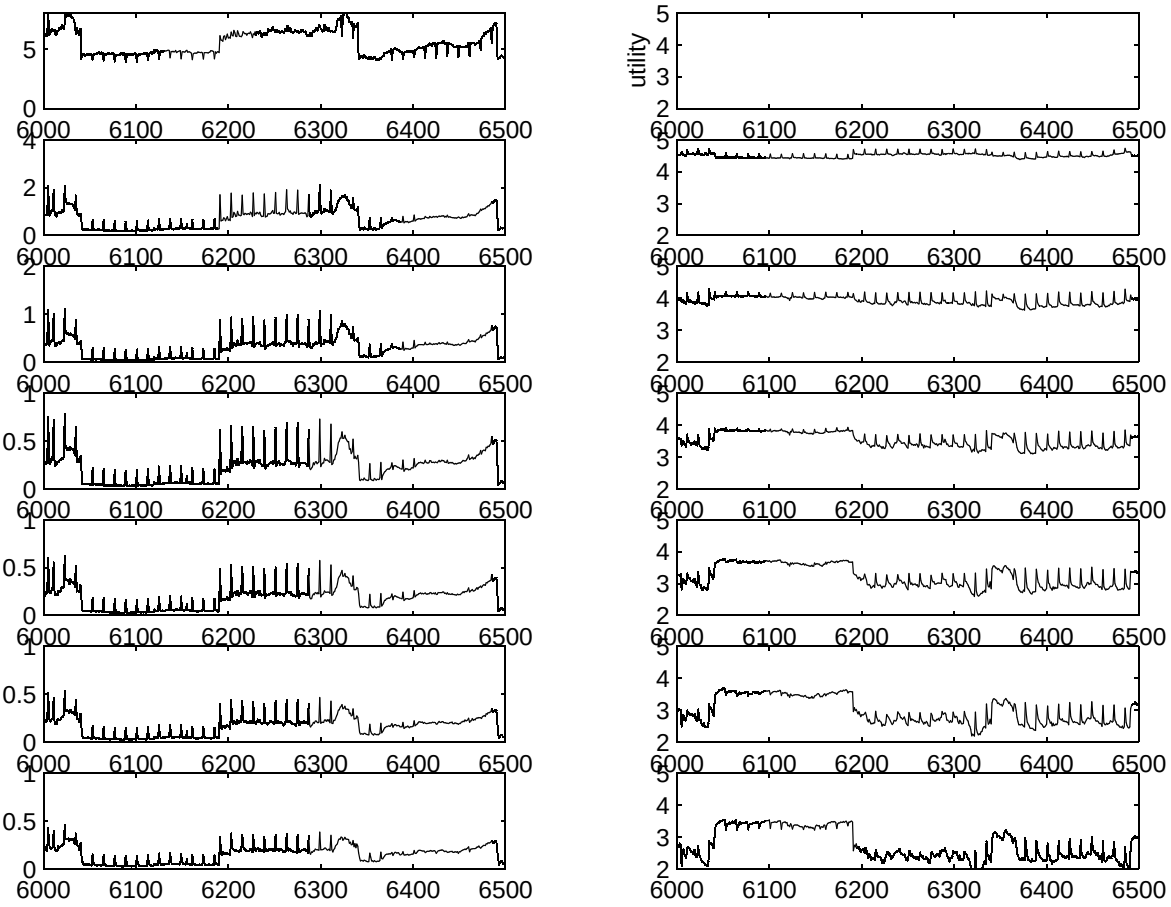

Figure 3: MPEG-4 stream bit-rate and corresponding utility estimation.

rate-only clustering models that are independent of each other. Figure 4 depicts example of clustering results that were obtained using these models; utility-only model is depicted on the left and rate-only model is depicted on the right. Each graph illustrates nine different classes of distinct shape. Each class is shown by its mean value that is computed among all members of particular class. In our example, nine classes were obtained using the Autoclass III software. Note that each clustering model is based on different feature vector. In particular, the feature vector of the utility-only model consists of seven utility values for each video object corresponding to different values of q. Correspondingly, rate-only model consists of seven rate values.

Both utility-only and rate-only models are combined to form composite model. Figure 5 depicts resulting composite model consisting of 81 utility classes. The model is based on previously obtained rate-only and utility-only models. Each utility class is shown by its mean utility function $\mathcal{U}$. Figure 5 illustrates different shapes of utility function corresponding to different utility classes.

On the other hand, the joint model is formed directly from all 14 features of utility function $\mathcal{U} = \{x_q;\ q = 1, 5, 10, 15, 20, 25, 31\}$. For comparison, joint model is depicted in Figure 6. Similarly,

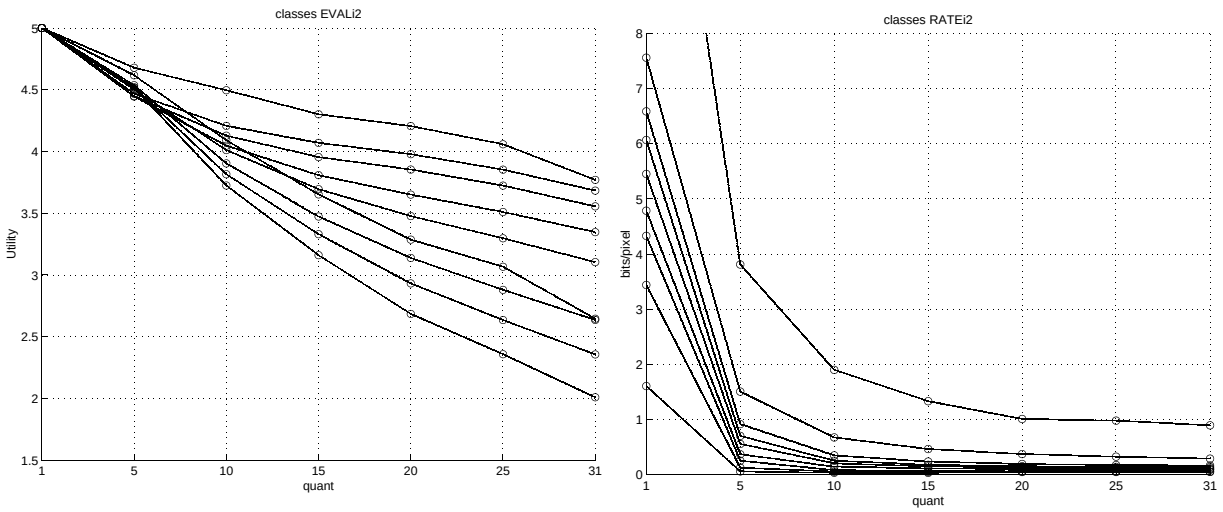

Figure 4: Results of UT9 utility-only (left) and RT9 rate-only (right) independent clustering models.

each utility class is shown by its mean utility function $\mathcal{U}$.

In general, clustering that is obtained using composite and joint models are different. Because composite model is obtained as combination of two independent models, some utility classes corresponding to particular rate-only and utility-only classes may not be populated (e.g., no video object is classified into that particular utility class). Consequently, it does not typically represent optimal clustering. On the other hand, joint model can obtain clusters that are not possible to capture by ortogonally projected rate-only and utility-only models of composite model. This feature leads in general to better clustering results compared to composite model. Our results, summarized in Table 1, indicate that this is also the case in our experiments.

The results obtained by composite and joint utility clustering models are summarized in Table 1. In our experiments, the goodness of clustering is measured by *classification consistency* and mean square error (MSE). Classification consistency of model $\mathcal{P}$ is defined as $\mathcal{F}\{\mathcal{P}\} = 10 \log S\{\mathcal{P}\}/G\{\mathcal{P}\}$, where $G(\mathcal{P})$ is *degree of grouping* and $S(\mathcal{P})$ is *degree of separation* [4]. In general, higher values of $\mathcal{F}$ are related to model in which better clustering is obtained (e.g., small distances between members in the same class and large distances between classes). Mean square error (MSE) is measured using distance between actual utility function of each video object and characteristic utility function (e.g., mean utility function for each class) of the class into which the particular video object is classified. As expected, better clustering, indicated by higher values of $\mathcal{F}$, favor the joint model. Similarly, better clustering, with the exception of the I-frame test case, is confirmed by lower MSE values for joint model, compared to composite model.

The results obtained by decision-tree classifier are summarized in Table 2. Classification accuracy measures percentage of video objects that were classified into correct classes. In our experiments, composite and joint decision-tree classifier is constructed for each I-, P-, and I- & P-frame test case. They directly correspond to composite and joint utility clustering models in Table 1.

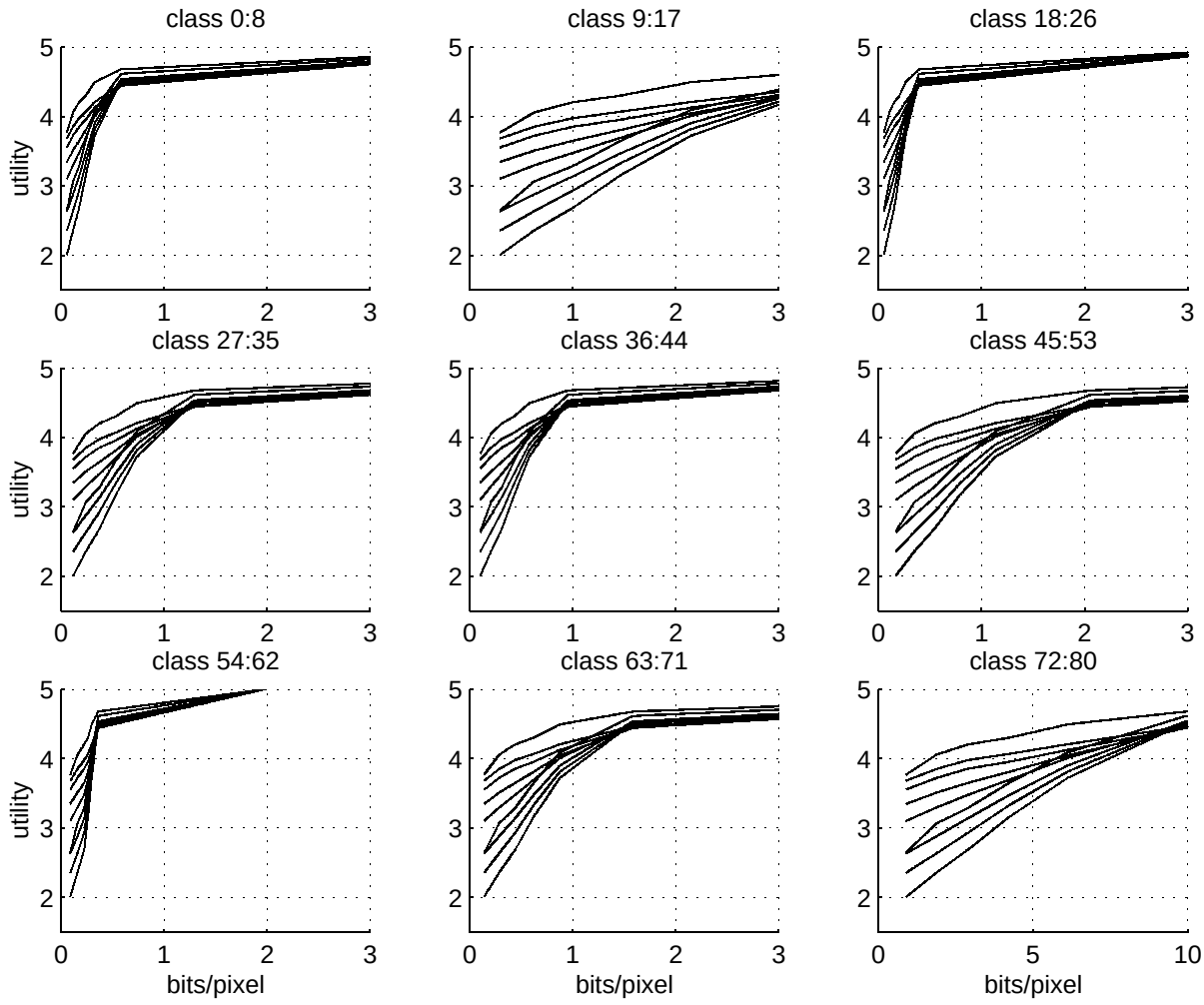

Figure 5: Composite utility clustering model for I-frames.

The results were obtained using video objects that were not used to train the adaptive content classification loop (i.e., to cluster utility functions and generate decision-tree classifier). Because of (1) decision-tree classifier is based on sample content feature vector and (2) optimizations are performed in decision-tree classifier to reduce complexity of decision tree, some of the video objects are not classified correctly into the utility class they belong to. This fact is illustrated by classification accuracy and, with exception of joint model for I-frame, by decrease in $\mathcal{F}$ and increase of MSE compared to results in Table 1.

Although classification accuracy is similar for both composite and joint models, it is somewhat higher for composite model and I- and P-frame test cases. In contrary, with exception of I-frame case, joint models give better results in terms of MSE. In all cases, joint model achieved better classification consistency $\mathcal{F}$ compared to composite model.

Based on our results, it appears that while composite models perform slightly better for homogeneous test cases (i.e., for I-frames and P-frames test cases) in terms of classification accuracy,

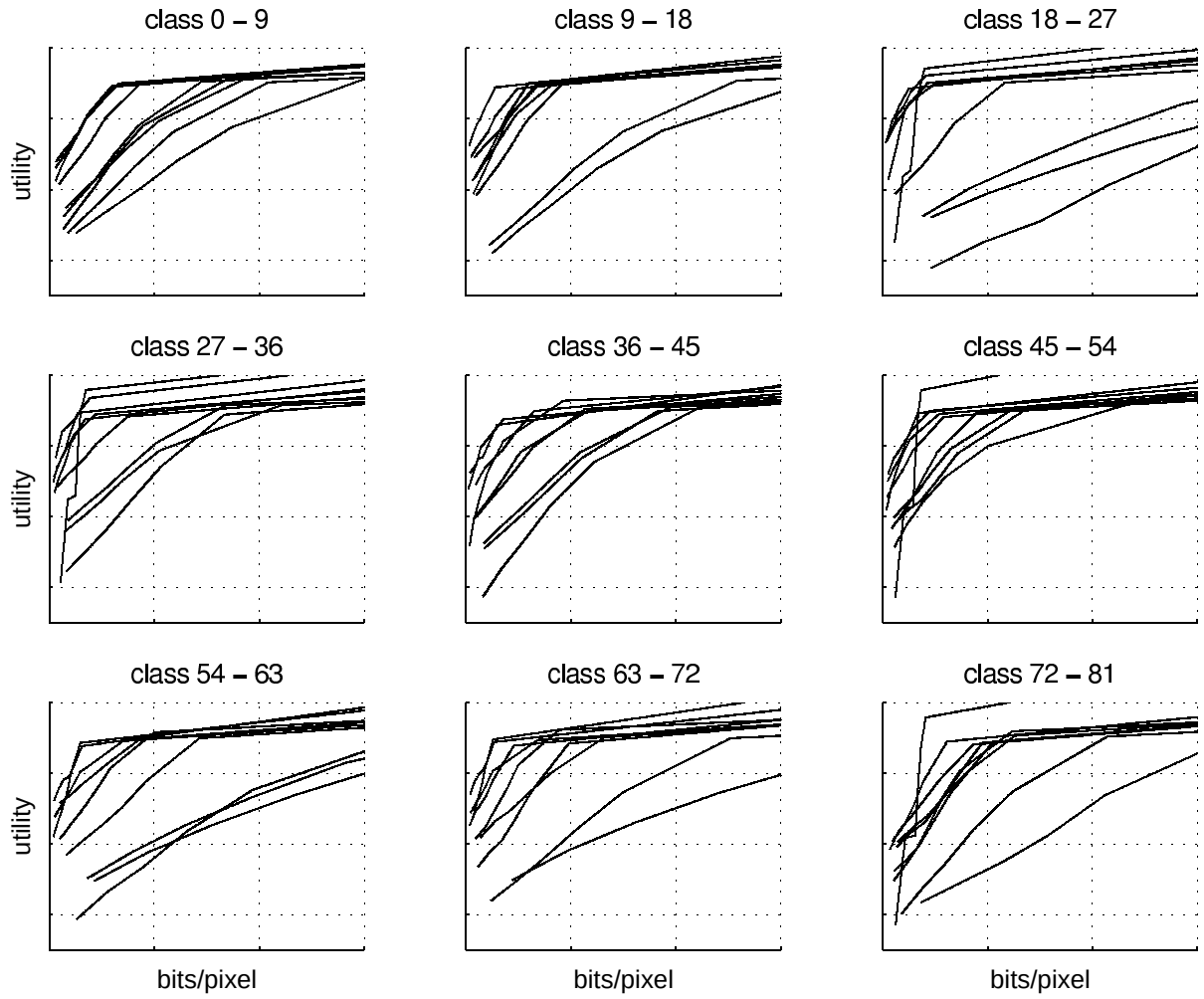

Figure 6: Joint utility clustering model for I-frames.

joint model perform better for both P-frame and combined I- & P-frame test case in terms of both classification consistency and MSE. Consequently, models based on combination of simple classifiers (i.e., composite models) may be sufficient to construct decision-tree classifiers that operate on video objects of the same type (i.e., I- or P-frames only). In contrary, models based on joint rate-utility feature vectors (i.e., joint models) are more appropriate in cases where the training pool contains video objects of different types (i.e., I- & P-frames combined).

5 Conclusion

In this paper, we presented a novel framework for real-time subjective utility function estimation. We demonstrated that the system for real-time subjective utility function estimation could be constructed using a unique content-aware video classification approach and general machine learning algorithms. The utility functions for video objects can be used for media scaling in future low

	I-frame		P-frame		I- & P-frame	
	comp	joint	comp	joint	comp	joint
F	26.61	28.49	25.21	31.90	25.07	32.13
MSE	1.102	3.22	3.22	2.60	3.32	3.01

Table 1: Comparison of composite (comp) and joint utility clustering models.

	I-frame		P-frame		I- & P-frame	
	comp	joint	comp	joint	comp	joint
accuracy (%)	88.62	78.93	80.51	78.75	83.89	86.78
F	25.84	30.25	24.43	30.93	24.54	31.74
MSE	1.57	4.07	4.26	3.77	3.58	3.31

Table 2: Comparison of composite (comp) and joint decision-tree classifiers.

bandwidth and wireless networks. Our results indicated the feasibility and relative high accuracy of such a system.

6 Acknowledgement

This work was completed while the first author was visiting KDD Multimedia Communications Group, Japan. The authors would like to thank all members of the group for their continuous encouragement and support. We also thank Dr. Hamada for his constructive suggestions, especially with modifications of Picture Quality Assessment System and Mr. Komatsu and Mr. Aoyagi for their help with software implementation.

References

[1] D. Reininger and R. Izmailov, "Soft Quality of Service with VBR$^+$ Video", *Proceedings of International Workshop on Audio-Visual Services over Packet Networks (AVSPN'97)*, Aberdeen, Scotland, UK, September. 15-16, 1997, pp. 207-211.

[2] T. Hamada, S. Miyaji, and S. Matsumoto, "Picture Quality Assessment System by Three-layered Bottom-up Noise Weighting Considering Human Visual Perception", *Proceedings of SMPTE*, New York, 1997

[3] Recommendation ITU BT.500-7, "Methodology for the Subjective Assessment of the Quality of Television Pictures"

[4] P. Bocheck and S.-F. Chang, "Content Based Dynamic Resource Allocation for VBR Video in Bandwidth Limited Networks", *Proceedings of the Sixth IEEE/IFIP International Workshop on Quality of Service (IWQoS'98)*, Napa, California, May 18-20, 1998

[5] Di Zhong and Shih-Fu Chang, "AMOS: AN ACTIVE SYSTEM FOR MPEG-4 VIDEO OBJECT SEGMENTATION", *International Conference on Image Processing 98*, October 4-7, 1998, Chicago, Illinois, USA

[6] R. Hanson, J. Stutz, and P. Cheeseman, "Bayesian Classification Theory", *NASA Technical Report FIA-90-12-7-01*

[7] S. K. Murthy, S. Kasif, and S. Salzberg, "A System for Induction of Oblique Decision Trees", *Journal of Artificial Intelligence Research*, 1994

[8] "MPEG-4 VM software, MoMuSys-VFCD-V01-980507", *European ACTS project MoMuSys*, 1998

[9] R. Liao and A. Campbell, "On Programmable Universal Mobile Channel", *Proceedings of ACM Mobicom'98*, Dallas, TX, October, 1998

[10] J. Meng and S.-F. Chang, "Tools for Compressed-Domain Video Indexing and Editing", *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Database*, Vol. 2670, San Jose, February 1996.

[11] R. Liao, P. Bouklee, and A. Campbell, "Online Generation of Bandwidth utility function for Digital Video", *Proceedings of PacketVideo'99*, New York City, Apr. 26-27, 1999.

[12] [3] A. Eleftheriadis and D. Anastassiou, "Dynamic Rate Shaping of Compressed Digital Video", *Proceedings of 2nd IEEE International Conference on Image Processing (ICIP95)*, Arlington, VA, Oct. 1995.

[13] N. Yeadon, F. Garcia, D. Hutchison, and D. Shepherd, "Filters: QoS Support Mechanisms for Multipeer Communications", *IEEE Journal on Selected Areas in Communications, Special Issue on Distributed Multimedia Systems and Technology*, Vol. 14, No. 7, Sept. 1996, pp. 1245-1262.

[14] K. Lee, "Adaptive Network Support for Mobile Multimedia", *Proceeding of ACM Mobicom'95*, Berkeley, CA, Nov. 1995.

[15] , R. Liao, P. Bocheck, A. Campbell, and S.-F. Chang *Content-aware Network Adaptation for MPEG-4*, submitted to NOSSDAV'99, June 1999