OPTIMAL VIDEO ADAPTATION FOR RESOURCE CONSTRAINED MOBILE DEVICES BASED ON UTILITY THEORY

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖZGÜR DENİZ ÖNÜR

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences

_____

Prof. Dr. Canan Özgen

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. Mübeccel Demirekler

Head Of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science

_____

Assoc. Prof. Dr. A.Aydın Alatan

Supervisor

Examining Committee Members

Prof. Dr. Kemal Leblebicioğlu _____

Assoc. Prof. Dr .A. Aydın Alatan _____

Assoc. Prof. Dr. Gözde Bozdağı Akar _____

Assist. Prof. Dr. Cüneyt Bazlamaçcı _____

Assoc. Prof. Dr. Yasemin Yardımcı _____

ABSTRACT

Optimal Video Adaptation for Resource Constrained Mobile Devices Based on Utility Theory

Önür, Özgür Deniz

MSc., Department of Electrical and Electronics Engineering

Supervisor: Associate Professor A. Aydın Alatan

September 2003, 10406 pages

This thesis proposes a novel system to determine the best representation of a video in the sense that, a user watching the video reaches the highest level of satisfaction possible, given the resource capabilities of the viewing device. Utility theory is used to obtain a utility function representing the user satisfaction as a function of video coding parameters, and the viewing device capabilities. The utility function is formulated as the weighted sum of three individual components. These components are chosen such that, the satisfaction on any one of the components is independent of the satisfaction on every other component. The advantage of such decomposition is the ability to express individual components as simple mathematical relations, modeling user satisfaction. Afterwards, the unknown parameters of these models are determined by results of subjective tests, performed by a multitude of users. Finally, simulated annealing is utilized to find

the global optimum of this utility function representing the user satisfaction. Simulation results based on subjective viewing tests on a resource limited mobile device indicate a consistent user satisfaction by the determined optimal encoding parameters of the video.

Keywords: video adaptation, utility, universal multimedia access

ÖZ

Kaynakları Sınırlı Cihazlara Yönelik Fayda Kuramına Dayanan Video Adaptasyonu

Önür, Özgür Deniz

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi : Doçent Dr. A. Aydin Alatan

Eylül 2003, 10406 sayfa

Bu çalışma, bir kullanıcının, herhangi bir videoyu izlerken, izlemek için kullandığı cihazın kaynaklarının izin verdiği ölçüde, en yüksek memnuniyeti elde edebilmesini sağlamak için, videonun hangi şekilde kodlanması gerektiğini tespit edebilen, özgün bir sistemin tasarımı ve gerçeklemesini içermektedir. Tez ayrıca, bir kullanıcının memnuniyetini, birbirinden bağımsız üç bileşene ayrıştırabilmek için bir metod önermektedir. Bu şekildeki bir ayrıştırmanın faydası, bileşenlerin çeşitli video ve cihaz parametrelerinin fonksiyonu olarak, basit denklemler halinde ifade edilebilmesini sağlamasıdır. Denklemlerin tam olarak belirlenebilmesi için, birden fazla kullanıcı tarafından yapılan subjektif teslerin sonuçları, belirli eğrilerle, sonuçlarla eğriler arasındaki fark en az olacak şekilde modellenmiştir. Bunların ardından, fayda kuramı kullanılarak, kullanıcıların memnuniyetini belirten denklemin bütünü, üç bileşenin çeşitli katsayılarla çarpılarak toplanması suretiyle elde edilmiştir. Son olarak “ısıl işlem benzetimi”

kullanılarak, adı geçen denklemin evrensel optimum noktası bulunmuştur. Sınırlı kaynaklara sahip bir cihazda yapılan ve subjektif değerlendirmeye dayanan simulasyon sonuçları, sistemin belirlediği video kodlama parametreleriyle kodlanan videoların, her defasında kullanıcıları memnun ettiğini göstermektedir.

Anahtar Kelimeler: video adaptasyonu, fayda, çoğul ortam verisine evrensel erişim

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to A. Aydın Alatan, who has always managed to provide the necessary insight, and the motivation that was crucial to complete this thesis. Without him, this thesis would not be near to what it has turned out to be now.

Heart felt thanks goes to my buddies Medeni Soysal and Yağız Yaşaroğlu, with them this endeavor has been nothing but a pleasure. I would especially like to thank Oktay Onur Kuzucu for showing me the undisputable truth that one could always find something better to do than write a thesis. Ekin Dino and Çağlar Karasu have made this effort far easier than I imagined, but I will make them suffer when it is their turn.

Finally, my family deserves more credit than I can possibly give here. They have been patient and loving throughout my studies and provided everything one could possibly need.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

The last decade has observed the explosive increase in the usage of mobile communication devices throughout the world. It all began with the introduction of first generation (1G) analog networks, the so-called mobile phones of this first generation were heavy and their cumbersome size rendered them impractical. Nevertheless people adopted and used these devices, especially for mobile communication in cars.

Later the second generation (2G) digital mobile networks were introduced employing the early cellular phones. These circuit-switched networks were only capable of voice communications and simple text-based services like the Short Messaging Service (SMS) [1].

The most widely known example is the GSM (Global System for Mobile Communications) network. Mobile person-to-person speech communication has become extremely popular and has reached 1 billion users today [1].

After the 2G markets saturated mobile phone manufacturers started producing multimedia enabled cellular phones which operated in coordination

with packet based 2.5G technologies, such as the GPRS (General Packet Radio System) to deliver more appealing content than their predecessors. Multimedia enabled at this stage meant delivering still images or downloading video clips to hand sets. However for fully multimedia enabled mobile communications featuring rich video content with high quality sound "streamed" to mobile terminals the consumer had to wait for the 3G networks.

The world's first fully commercial 3G network was launched by the Japanese wireless operator NTT DoCoMo in Oct 1st 2001[2]. The 3G system called FOMA (freedom of mobile multimedia access) was able to deliver data at speeds up to 384 Kb/s.

WCDMA and cdma2000 networks are being deployed today in Europe and Japan; these are expected to deliver data rates from 1.25 Mbps up to 3.75 Mbps. When the transition to 3G networks is completed the typical mobile terminal user will be able to enjoy data transfer rates more than enough to access high quality video and audio.

As the first 3G networks emerge multimedia content providers and mobile terminal producers face a serious problem. There is a bulk of multimedia content available in the market, but in very different formats, many companies have proprietary codecs and any given device on the market can only access a very limited percentage of the available content. In other words the existing formats and the existing user terminals are not interoperable. This prevents the augmented usage of the multimedia content, related services and the advanced mobile terminals, thus greatly reducing the revenue expected from the wireless market.

1.2 Scope of the Thesis

A significant percentage of the research going on in the field of digital multimedia is aimed at making the video content in the market available to all kinds of devices regardless of the format or the resource requirements of the media clip.

The MPEG committee observing the above conditions proposed the MPEG-21 standard which is expected to reach IS (International Standard) status by 2004. The MPEG-21 standard aims to define a multimedia framework to enable the transparent and augmented use of multimedia resources across a wide range of networks and devices used by different communities [12], especially taking into account the Digital Rights Management (DRM) issues.

The concept of enabling multimedia delivery through heterogeneous networks, different user preferences and varying terminal capabilities is named as Universal Multimedia Access (UMA). As it can be easily seen from the definitions the purpose of MPEG-21 overlaps with that of UMA. One can view MPEG-21 as a standard embodying UMA and DRM. It is useful to note that in the context of MPEG, DRM is termed as Intellectual Property Management and Protection (IPMP).

The resource requirements of a video clip is changed through a process called *video adaptation* in which a single data source (i.e. High Bit Rate Video) is modified to obtain different representations of a video clip taking into account the capabilities of any given user terminal.

The aim of the thesis is to develop a methodology to obtain the optimal

adaptation scheme given the properties of the end terminal requesting the video data.

This can be accomplished with the help of the *utility theory* which deals with making optimal decisions given the pros and cons of each alternative choice to be considered. The theory, originally proposed to model the satisfaction of people as a function of their income, has been successfully applied to non-monetary issues as well.

The satisfaction the user gets from watching a certain video clip is modeled as a utility function depending on certain spatial and temporal properties of the video, and the resource capabilities of the user terminal that the user intends to watch the video on. Then, this function is used to maximize the user satisfaction from the multimedia experience. This is accomplished with the help of a stochastic optimization algorithm called "simulated annealing". Thus the representation of the video in terms of the afore mentioned spatial and temporal properties that is going to result in highest user satisfaction is obtained.

1.3 Thesis Outline

In the following chapter Video Adaptation, the modification of video content according to the properties of the end-terminals will be introduced and the possible types of the adaptation process will be discussed.

In Chapter 3 the newly emerging mpeg standard MPEG-21 is introduced and the Universal Multimedia Access (UMA) concept and its application to the MPEG-21 standard are outlined.In Chapter 4 the fundamentals of Utility Theory

are briefly discussed together with the application of utility theory to non-monetary issues and examples.

The next chapter, Chapter 5 deals with video adaptation based on utility theory and explores the proposed system in detail together with the experiments performed.

Chapter 6 concludes the thesis with a section on possible future work.

CHAPTER 2

VIDEO ADAPTATION

Video Adaptation is transforming a given representation of a video into another representation. Video Adaptation is a crucial technique that enables transparent media resource delivery to diverse users having different network connectivity, terminal capabilities, decoders and physical locations.

2.1 Types of Adaptation

Video adaptation is usually classified into four main types. In this section, these different types are explained in detail. Figure 1, illustrates the types of video adaptation.

2.1.1 Syntax Conversions

Syntax or format conversions may be required according to the decoding abilities of the user terminal. The terminal may not have the adequate hardware (CPU, memory) to decode complex bit streams, so only a marginally compressed video would be more appropriate.

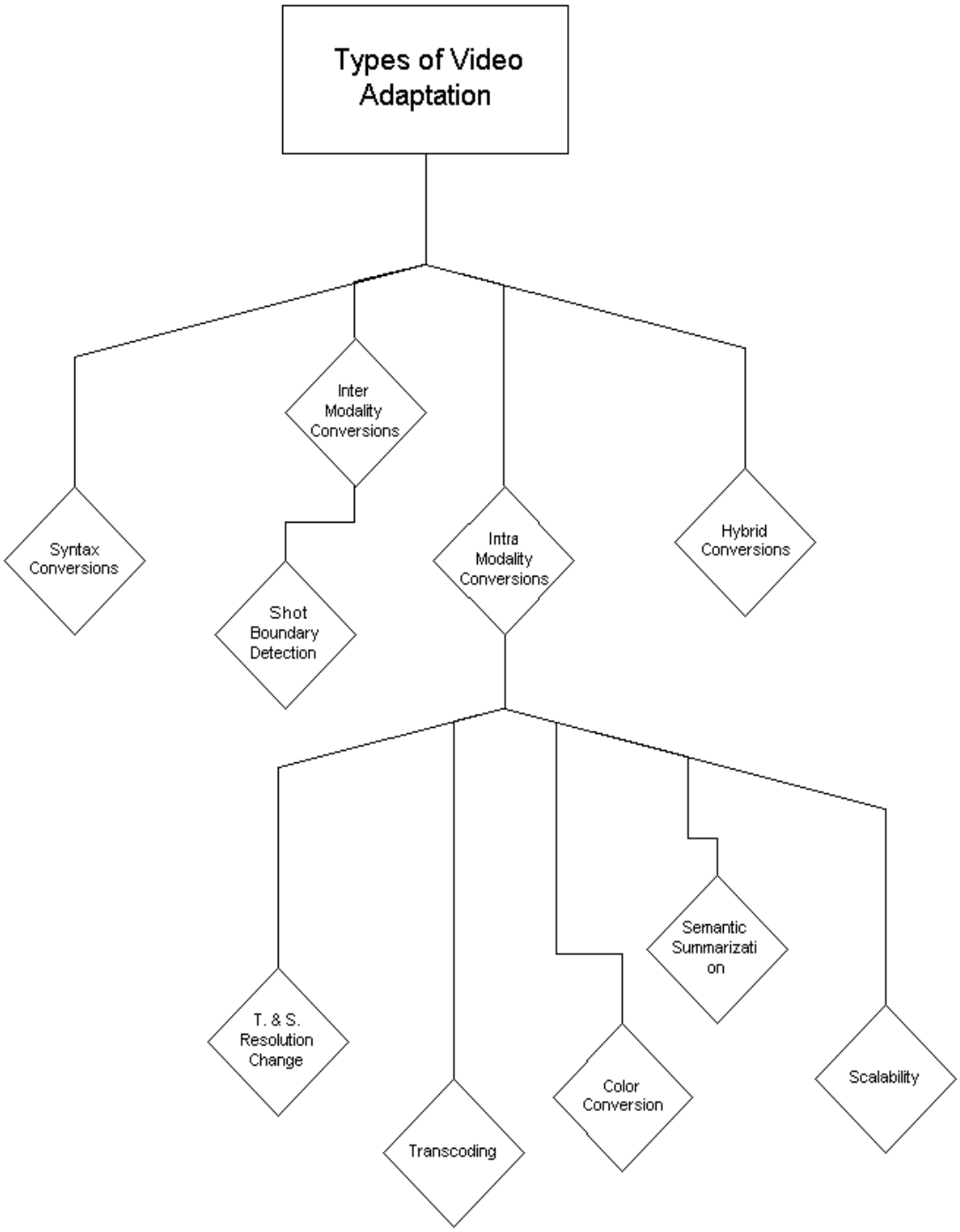

Figure 1 – Types of Video Adaptation

The format may require changes according to the application, for example some formats are not suitable for streaming applications, such as MPEG-1, whereas others while being suitable for streaming are not suited to editing, such as the ‘asf’ format of Microsoft.

Format conversions may also be required simply because an end terminal does not have the necessary decoders to decode that specific bit stream. Downloading the necessary decoder, if it is available, might be a better idea than performing a format conversion due to the time it takes to perform the conversion.

2.1.2 Inter-Modality Conversions

Multimedia content is usually in multiple media formats or modalities. A video clip may contain the raw data from a video, multiple channels of audio and closed captions as text. New modalities can be created from the existing modalities of a video clip. For example, by speech to text conversion the audio information associated with the video can be used to create a textual modality or vice versa. One can also choose representative frames from a video clip to create an image modality of a video, which is composed of key frames capturing the most important parts of the media clip [3].

Inter modality conversions are especially useful for devices which are severely resource constrained. Instead of sending a motion picture to a PDA, it is more appropriate to send still images, one can send only the audio modality of the movie to a simple cellular phone and finally it is better to send only the text information to a severely resource constrained legacy mobile terminal.

2.1.2.1 Shot Boundary Detection

A *shot* is defined as a continuous recording of the camera [6]. From a signal processing point of view, it can be described as temporally adjacent frames of a video without significant content change between successive frames [4]

Video clips are composed of many shots placed sequentially in time. Two types of transitions are possible between shots:

- Abrupt transitions
- Gradual transitions

Abrupt transitions or cuts occur between two consecutive frames. Cut transitions are easy to detect using frame difference metrics which will be described shortly. Gradual transitions on the other hand are transitions in which the first shot is gradually replaced by the proceeding shot over a span of many frames (usually at most 100 frames are used for gradual transitions). The most common type of gradual transition is the *dissolve*. In a dissolve transition, the intensity of the first shot is slowly decreased while the intensity level of the next shot is gradually increased. The longer the transition period the harder it is to detect dissolve transitions.

Temporal video segmentation can be considered as an initial step to adapt video modality into image modality, since it involves the determination of a video's stationary parts to obtain "good" representative frames. Detecting shot boundaries (i.e. determining the start and end points of shots) has been a subject of serious research especially in the late 90s. Many algorithms have been proposed for automatic shot boundary detection and well defined techniques exist.

One of the most widely accepted algorithms for shot boundary detection is the twin-comparison algorithm [5], which uses two different thresholds for detecting cuts and gradual transitions(Tb,Ts) in the histogram difference metric. Histograms and the histogram difference metric can be briefly explained, as follows:

The histogram of an image for any color channel is an *n*- dimensional vector $H_i(j)$, j=1,..,n where n is the number of possible values in that particular channel and $H_i(j)$ is the number of pixels in frame i having a color value of j. The histogram difference metric then can be defined as

$$D(i,i+1)=\sum_{j=1}^{n}\left|H_i(j)-H_{i+1}(j)\right| \qquad \textbf{(1)}$$

If the histogram difference D (i, i+1) is larger than Tb then a cut is detected in the frame i+1, if it is less than Tb but more than Ts then a cumulative difference metric is computed and yet another threshold is used in this new metric to determine if there is a gradual transition in the corresponding frame [5].

The implemented algorithm in this thesis, is the shot boundary detection using multiple cues. The method uses the K-means unsupervised clustering algorithm [15] with K=2.

2-means clustering algorithm can be briefly outlined as follows; *N* dimensional feature vectors are separated into 2 clusters via an iterative algorithm that starts with initial cluster centers, set by the user, and then separates the feature vectors into two clusters according to their Euclidian proximity to the user set

centers. In the next iteration, cluster centers are recomputed as the arithmetic mean of the feature vectors which were assigned to the corresponding cluster by the previous iteration and the feature vectors are clustered again using the new cluster centers. The algorithm ends when the difference between cluster centers computed by consecutive iterations is below a certain threshold.

In the proposed algorithm 2-dimensional feature vectors are used, one of the features being the histogram difference and the other being the pixel difference metric. The pixel difference metric can be described as:

First the number of pixels that have changed more than a certain threshold are determined.

$$DP(i,i+1,x,y)=\begin{cases}1 & if \quad \left|P_i(x,y)-P_{i+1}(x,y)\right|\geq T_1\\ 0 & otherwise\end{cases} \qquad \textbf{(2)}$$

Then, the number of pixels that have changed are summed to obtain the metric [6].

$$D(i,i+1)=\frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}DP(i,i+1,x,y)}{X.Y} \qquad \textbf{(3)}$$

After the frames are separated into two clusters as either shot boundary and non-shot boundary, there is an elimination step which depends on a heuristic observation that shot boundaries always form a local maximum in the histogram difference metric. Hence the frames that are not local maxima in the histogram difference are eliminated, even if they are in the shot boundary cluster.

Histogram difference and the pixel difference metrics used in the algorithm complement each other quite well. While the histogram difference metric captures the substantial changes in the color composition of a frame, it is insensitive to object motion in the frame. This prevents false alarms for sequences containing significant motion. The pixel difference metric helps in determining shot boundaries at which the color composition does not change considerably; for example, shots from cameras positioned at different angles in a soccer pitch.

The primary reason for preferring this algorithm over the twin-comparison algorithm is that it achieves unsupervised clustering and its performance does not depend on the thresholds used (the user set cluster centers which are needed in the first iteration do not effect the end result noticeably). Thresholds can be tuned to obtain near-perfect results for any specific video sequence; on the other hand, obtaining acceptable results for all possible video sequences is very difficult in threshold dependent algorithms.

After the shot boundaries are detected, a frame from each shot needs to be chosen as a representative key frame, to be able to complete the adaptation of the video modality to image modality. Key frames can be chosen as the first frame, the last frame, or the frame in between the first and the last frames of a shot; however a more logical choice would be to choose the frame that maximizes the histogram difference metric within a shot, as the key frame.

2.1.3 Intra Modality Conversions

Intra modality conversions are conversions for which the modality of the video remains unchanged, but the resource requirements for displaying the video can be changed significantly.

2.1.3.1 Temporal and Spatial Resolution Change

Changing the resolution of a video is one of the fundamental approaches that can be classified as an intra modality conversion. The resolution can be changed either temporally or spatially.

Temporal resolution changes may have two forms[7]:

- Frame rate reduction
- Time condensation

Frame rate reduction is achieved by dropping some subset (for example, one can drop the B frames in an MPEG coded sequence) of frames from a video sequence while keeping the playback duration constant. The gaps formed due to the absence of frames, are filled either by duplicating the frame before the gap position or by interpolating the neighboring frames. The average bandwidth required for transmitting the video can be reduced with this technique.

Time condensation is also performed by removing a subset of frames from a video clip but this time the frame rate is kept constant and the playback time is reduced. This operation is also named as *video skimming* [7]. A subset of shots instead of frames can also be removed.

Three types of spatial resolution changes are possible:

- Display area reduction
- SNR adjustment
- Omission of objects

The display area of a video clip can be reduced by simply down sampling the image pixels, or replacing a group of pixels by a single pixel having an intensity value equal to the average of the pixels that it represents. This technique also reduces the bandwidth required for the transmission of the particular video.

SNR adjustment is quite similar to compressing a video sequence. As in many compression standards, some of the high frequency components of the DCT of a frame can be dropped to obtain a frame having a lower SNR. In this way, one can trade necessary transmission bandwidth with signal quality.

Finally, if the video or an image is coded using a codec that performs object coding such as MPEG-4 or JPEG 2000, then low-priority video objects can be dropped again resulting in a reduction of overall bandwidth requirement [7].

2.1.3.2 Transcoding

The process of converting between different compression formats and/or further reducing the bit rate of a previously compressed signal is known as transcoding [9]. Many of the adaptation operations described above can be considered as some form of transcoding, the key point here is that those operations can only be named as a kind of transcoding, if they are performed using an already compressed bit stream.

The most straightforward type of transcoding is done by fully decoding the bit stream and then re-encoding it with the desired parameters (bit rate, display

resolution etc.) as seen in Figure 2 . However, transcoding is usually used in situations where real-time processing performance is required and decoding followed by re-encoding of the bit stream is not acceptable in terms of computational complexity. Therefore most of the research in the field of transcoding has been directed towards obtaining a bit stream with desired properties by minimally decoding the bit stream.

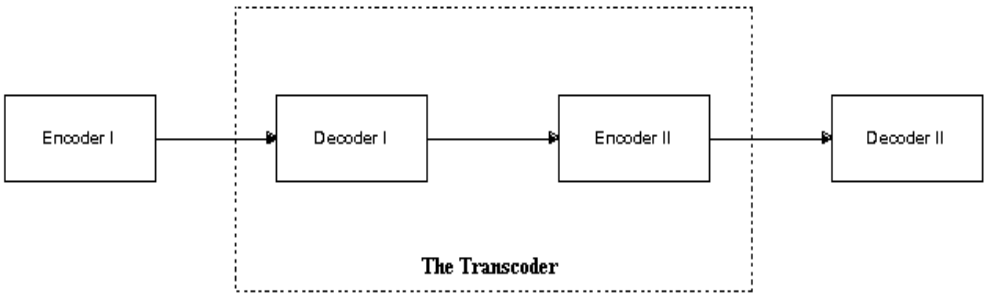

Figure 2 – The Basic Transcoder

There is an important difference between first generation coding and transcoding, the transcoder has only access to an already coded bit stream as opposed to the raw data available to the first generation coder. This means that the data available to a transcoder already contains considerable amount of quantization noise.

Consider the transcoder in Figure 3, the bit stream is first variable length decoded and then inverse quantized to obtain the DCT coefficients. These coefficients are re-quantized to obtain the desired bit rate and then variable length coded to obtain the bit stream. This open-loop architecture is prone to an error

called as the *drift error* [9]. The reason for drift error is simply the loss of high frequency information. The transcoder in Figure 3 discards the high frequency information starting from an I frame and continuing with the following frames whose blocks are coded as the difference between the I frame (or a P frame if the reference is taken as a P frame) and the preceding frames. Hence, when a decoder gets the transcoded bit stream, it decodes the reduced quality anchor frame and stores it in memory. Later, in order to reconstruct the following P or B frames, the reduced quality I frame is added to the reduced quality residual components of the frames to be predicted. Moreover, these already degraded P frames are used to predict the remaining frames of the video. As a result the drift error progressively increases resulting in reconstructed frames becoming severely degraded. The key point in avoiding drift error is to recompute the residue using the motion compensated anchor frames having some amount of quantization error due to the cascaded quantization, although the quantization error is not removed by this technique, since the residue is computed using the reduced quality anchor frames the error does not increase progressively.

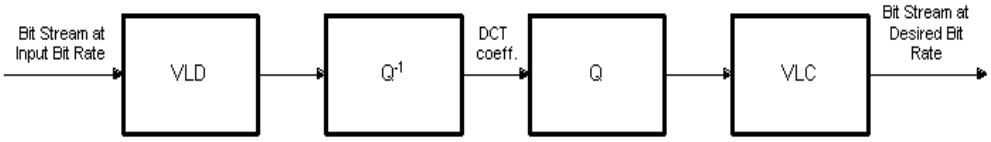

Figure 3 – Open Loop Transcoder

The most straightforward way to avoid the drift error is to use the cascaded encoder decoder structure, but this structure is not acceptable because of its computational complexity, as explained above. Some simplified architectures that approximates the cascaded structure also exist and are commonly used [10].

A substantial reduction in computational complexity can be achieved if the cascaded encoder uses the same picture types and the same motion vectors as the original encoder. There are three reasons for the obtained reduction [11] :

1. The decoded pictures are not yet in the display order in the output of the decoder of the transcoder; they need to be reordered, and furthermore when they are

re-encoded, they will need to be put back into the bit stream order by the encoder. If the picture types selection of the first generation encoder will not be changed, both reorderings (first in the decoder of the transcoder and then in the encoder of the transcoder) can be omitted.

2. The motion estimation can be omitted, since the previously obtained motion vectors are going to be used. This results in a very substantial decrease in computational complexity, since motion estimation is one of the most rigorous procedures of video coding.

3. Since most of the incoming data from the first generation encoder will be used in the transcoder, there is no need to fully decode the pictures in the transcoder. Hence the amount of memory required to store the frame data can be reduced.

Detailed block diagrams of the simplified transcoding architectures can be found readily in the literature [9] [10] [11].

2.1.3.3 Scalability

Scalability or scalable video coding is, encoding a video in a multi layered manner, in such a way that, the individual layers contain representations of the video having different qualities. The layers are present as sub-streams in the video bit stream.

Advanced video codecs allow scalable video coding. For example in MPEG-2, video can be coded in a multilayered manner. The first layer is the *base layer* [8]. The base layer can be decoded independently of other layers. The layers coded on top of the base layer are called *enhancement layers* [8] . To be able to decode an enhancement layer the layers that are below the particular enhancement layer in the hierarchy need to be decoded beforehand [8]. Each enhancement layer coded on top of the base layer brings higher signal quality and also higher transmission bandwidth requirement. The video can easily be scaled according to the abilities of a user terminal; i.e. to a resource constrained device only the base layer is sent and as the capabilities of the requesting user terminal are increased more and more enhancement layers can be delivered to the device.

2.1.3.4 Color conversion

The color information associated with a video frame can be changed to obtain a more resource conservative video. The most basic color conversion would be to convert the image to gray scale. Afterwards, instead of using eight bits to code each color channel, the video frame is represented with intensity levels ranging from 0 to 255 by a total of 8 bits. Another possible color conversion can involve re-quantizing each individual color channel to represent them with a smaller

number of bits. For example, each channel can be represented with four bits instead of eight, resulting in a twelve bit representation overall.

2.1.3.5 Semantic Summarization

If one can determine high level features of a video from some low level descriptors (i.e. determine the important moments of a soccer game, for example using the motion vectors of the video together with the audio content) then this information can be used to filter out the relatively unimportant points of the video and perform time condensation adaptation. In a different scenario, one can encode the important points of the video with a high bandwidth and use only key frames for the less important parts, as mentioned in Section 2.1.2. This summarization technique results in a reduction of bandwidth requirement, while allowing a multimedia consumer to enjoy the important parts of a media clip with full quality [7].

2.1.4 Hybrid Conversions

The final types of video adaptation that will be discussed, are hybrid conversions. Hybrid conversions, enjoy the advantages of different adaptation schemes.

Obviously, any one of the adaptation processes described above can be applied together with one of the other adaptations. Such conversions containing more than one adaptation process are called hybrid conversions. Hybrid conversions are used to reduce the resource requirements of a media clip, even further than a single adaptation process can achieve. For example, a video clip can have a display area reduction together with SNR adjustment or one can perform frame rate reduction together with omission objects.

Hybrid conversions can also contain multiple intra modality conversions as well as an inter modality conversion coupled with one or more of the intra modality conversions (for example a video clip can be converted to a slideshow of key frames and the display area of the key frames can also be changed).

CHAPTER 3

UNIVERSAL MULTIMEDIA ACCESS (UMA) and MPEG-21

As mentioned earlier UMA and MPEG-21 are closely related concepts. In fact the main aim of MPEG-21 is to make UMA a reality with the least possible human intervention while maximally protecting individual rights.

Interoperability is the most important issue for UMA. On the other hand, for realization of interoperability, existence of open standard recommendations is mandatory.

The next sections are an in-depth treatment of UMA and MPEG-21.

3.1 What is UMA?

As the name implies, the goal of UMA is to make access to multimedia data possible from any device, any location and anytime. The primary focus of UMA is to enable terminals with limited communication, processing and storage capabilities to access rich multimedia content. One of the challenges that the UMA has to deal with is the delivery of multimedia content across heterogeneous networks in a way that will suit individual user preferences (This is where transcoding comes into play). Figure 4 illustrates the UMA concept.

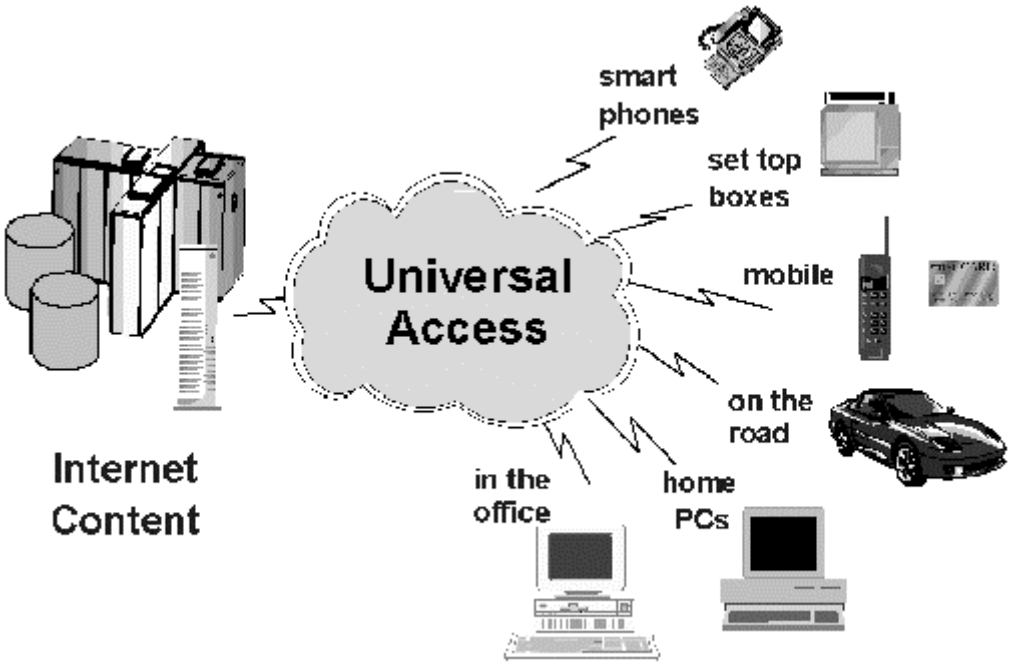

Figure 4 - The UMA concept. The vision of UMA is to make access to rich multimedia data possible across heterogeneous networks with resource constrained devices

UMA will be the driving force behind the development of services in 3G environments, since users will only be able to use UMA enabled services ubiquitously. UMA enabled services are actually expected to be the reason why the end users will adopt emerging communication systems over the current ones [12] .

3.2 MPEG-21 Fundamentals

MPEG-21 aims at developing a multimedia framework, to enable the delivery of multimedia content across heterogeneous networks to terminals with different capabilities, specifically taking into account Digital Rights Management (DRM).

The two most important concepts in MPEG-21 are the *Digital Item* and the *user*. The MPEG-21 framework is an open standard that defines the relationship of *users* with digital items. As it is stated in [12] "The DIs can be considered the *what* of the multimedia framework and the users can be considered the *who* of the multimedia framework".

The *User* can be defined as any entity that exchanges, accesses, consumes, trades or manipulates DI's. MPEG-21 makes no distinction between a content provider or a consumer; both are users[13].

MPEG-21 can be most basically defined as a multimedia framework in which users interact with other users and the object of that interaction is the *digital item*.

The digital item requires a more detailed explanation and is the subject of the next section.

3.2.1 Digital Items

A digital item is a structured object composed of 3 sub-entities

- Multimedia Resource: This part is the multimedia data itself. It can exist in any modality, i.e. audio, video, text.
- Associated Metadata: This is the data that describes the media data and the overall structure of the Digital Item. The Metadata is composed of a set *descriptors,* each associating certain information with the DI or a component within the DI.
- Identification : A form of identification that uniquely distinguishes the Digital Item from others(e.g. a Globally Unique Identifier(GUID))

Summing up, the DI is a complete digital representation of some work that has an intellectual value because it is the result of a creative effort; therefore the DI is the unit that is acted upon i.e. collected, exchanged, managed etc.

3.2.2 Digital Rights Management

The aim of the DRM parts of the MPEG-21 standard is to provide a framework that allows Users to express their rights, interests and agreements related to Digital Items and also assures them that these rights, interests and agreements will be persistently and reliably managed and protected regardless of the transmission mediums and the end-devices that the content reaches[13].

The concept of DRM is realized in 3 parts in MPEG-21.

1. Intellectual Property Management and Protection (Part 4 of the standard)
2. Rights Expression Language(REL) (Part 5 of the standard)
3. Rights Data Dictionary(RDD) (Part 6 of the standard)

IPMP part extends on the IPMP capabilities introduced in MPEG-4[18]. This part includes the retrieval of IPMP tools from remote locations and their authentication. It also aims to standardize exchanging messages between the IPMP tools and the terminals. Managing DRM issues in conjunction with the REL and RDD are also addressed in this part.

The Rights Expression Language is intended to provide flexible and interoperable mechanisms to provide transparent and augmented use of digital

resources, such as computer software, digital music, movies etc. in a way that protects digital contents and honors the rights of the respective owners of the content. The REL is a machine interpretable language; it (together with the RDD) is intended to be used by the "intelligent agents" which are programs that negotiate the access and DRM issues on behalf of a User they represent with other intelligent agents representing other Users. The REL is also responsible from making sure that personal data (such as a User's preferences regarding a certain DI) is processed in accordance with the rules governing the personal privacy issues. On the other hand, RDD is a dictionary containing all the key words necessary to describe the rights of the Users who control the DI's and the permissions granted by them. Using the RDD, the above mentioned rights and permissions can be unambiguously expressed using a standard syntactic convention (defined by REL) across all domains in which rights and permissions need to be expressed. That is, using the keywords present in the RDD, a document written in REL can be generated automatically. This document is then used by the intelligent agents in determining access rights and other DRM related issues. This automatic generation concept is especially useful in transforming REL documents across different domains that may require different formatting or different content in rights management documents.

3.3 MPEG-21 in UMA Context

UMA deals with the delivery of media resources to terminals with different capabilities through heterogeneous networks in compliance with user specified preferences. As stated earlier the goals of MPEG-21 and UMA are quite similar.

The main difference is the absence of IPMP in the context of UMA. However, one should note that while UMA is a concept, MPEG-21 is a tool that is designed to implement UMA.

In light of the above, one can say that, the requirements of UMA and the rights management issues can be implemented using the DI's in MPEG-21.

In order to be able to meet the requirements of UMA, that is to be able to deliver rich multimedia content to resource constrained devices across heterogeneous networks, it may be necessary to adapt the DI's according to the requirements of the data delivery scenario. Note that it is not only the Multimedia Resource of the DI that is being adapted; the associated metadata and the identification are also undergoing the adaptation process. The adapted DI is always considered as a new DI, with new metadata and identifiers. Figure 5 illustrates the adaptation process. The implementations of the Resource and the Descriptor adaptation engine are non-normative. However, MPEG-21 Digital Item Adaptation (DIA) descriptors which will be outlined shortly are standardized.

DIA is another part of the MPEG-21 standard that deals with the adaptation of digital items. The specifics of the multimedia delivery scenario, such as the network conditions, user preferences, terminal capabilities etc. are called as the *usage environment* in the context of MPEG-21. The adaptation of the DI's are performed according to the usage environment that is described by the *usage environment descriptors* defined in the DIA section of MPEG-21. The standard requires that MPEG-21 should describe the usage environment at least in terms of terminal, network, delivery, User and natural environment characteristics[13].

What follows is a detailed outline of the groups of standardized usage environment descriptors as mentioned above.

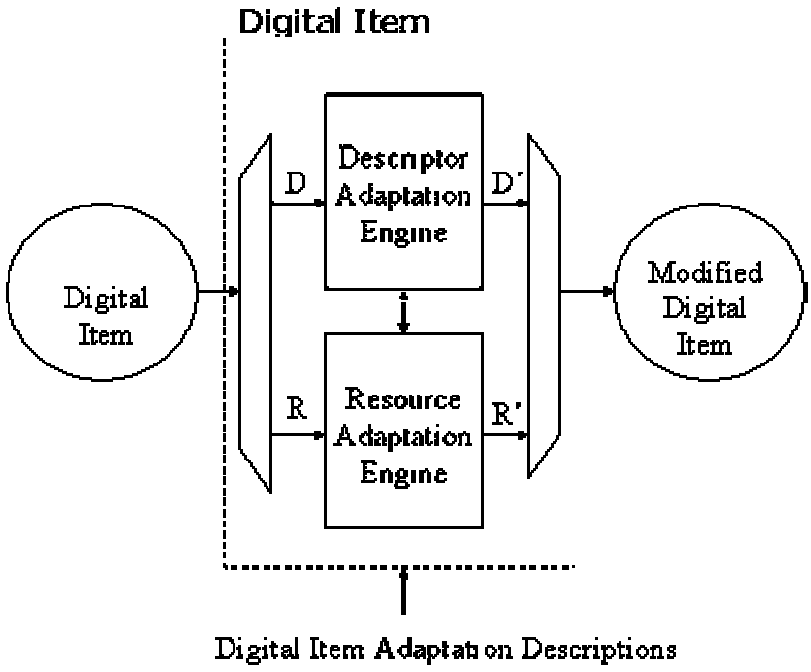

Figure 5 - The adaptation of a digital item. Both the media resource and the descriptors are adapted by the help of MPEG-21 DIA descriptors

The terminal characteristics that are normatively defined are [13]

- Acquisition Properties: capture resolution, spatial and temporal formats
- Device Type: encoder, decoder, gateway, router, camera etc.
- Device Profile: MPEG decoders specified by profile/level parameters
- Output Properties: specific rendering capabilities for each media modality

- Hardware Properties: processor speed, power consumption, memory architecture
- Software Properties: browser plug-in, operating system
- System Properties: configuration options, ability to scale the media resources and manage multiple resources simultaneously
- IPMP related capabilities: RDD and REL support, encryption, certificate authentication

The physical network characteristics that are normatively defined are [13]

- Delay characteristics: end-to-end delay, one-way delay, variation of delay
- Error Characteristics: BER, burstiness
- Bandwidth Characteristics: available bandwidth, variation of available bandwidth

The delivery layer characteristics that are normatively defined are [13]

- Types of transport protocols supported: TCP/IP, RTP/RTSP
- Types of connections supported: broadcast, multicast, unicast

The standardized User characteristics are [13]

- User Preferences: filtering and search preferences, QoS preferences, preferences regarding different media modalities
- Demographic Information: gender, age
- Accessibility and personalization attributes: auditory and visual impairments
- Mobility characteristics of the Users

The standardized Natural Environment characteristics are [13]

- Location: global positioning coordinates, country, state, city.
- Type of location: indoor, outdoor, public place, home, office
- Available access networks in a given area
- Velocity of user or terminal
- Illumination properties affecting user or terminal

These 5 groups are the descriptors that need to be input to the adaptation engine so that the adaptation process can create a new DI to suit the needs of the scenario at hand.

Usually using a smaller subset of the above descriptors is sufficient for demonstrating the usage and functionality of the standard.

The DIA section of the MPEG-21 standard contains some additional requirements. The two most relevant of these are :

- The MPEG-21 DIA descriptors should provide some form of mapping between descriptions and the characteristics of a media source to be adapted. For example given the usage environment descriptors of terminal CPU and memory, a way of determining the media resource properties that can be handled (i.e. the maximum bit rate that can be displayed by a terminal having a CPU of say 300MHz) by this configuration should also exist in the MPEG-21 DIA.

- MPEG-21 DIA descriptors should support an IPMP system that can control the types and degrees of adaptations permitted on DIs.

CHAPTER 4

UTILITY THEORY

When faced with making a decision, usually various alternatives are considered and the one that satisfies the requirements of a particular scenario to the highest extent is selected. However, it is not always easy to identify the best possible path especially in problems involving risk and multiple objectives. In this chapter, various methods for identifying the best possible decision will be introduced, and the virtues of utility theory among others will be demonstrated. The reason for making such an analysis is to devise a method to optimize the "value" of a multimedia content in terms of resource parameters.

4.1 EMV and Introductory Examples

The simplest criterion for making a decision is the Expected Monetary Value (EMV). The EMV of an alternative is equal to the probability of each outcome multiplied by the revenue expected from that outcome, summed across all possible outcomes associated with that alternative

.

$$EMV[x] = p_1x_1 + p_2x_2 + ... + p_kx_k \quad \textbf{(4)}$$

where

P_i = The probability of outcome i

X_i = The expected revenue from the outcome i

k = The number of outcomes associated with the alternative.

In order to clear this point consider an example:

A company is about to make a decision concerning the launch of an expensive advertising campaign for their new product. The advertising campaign will cost the company 1million *US$.*

Their forecasters predict that without the advertisement campaign their sales will amount to*1 million US$ with a probability of 75*%, however if all turns out well it will reach *5 million US$ but this has a small probability of 25%.* On the other hand, if the advertising campaign is carried out the probability that the sales will be limited to *1 million US$ drops to 40 %* and the increased sales of *5 million US$ are more probable with 60%*

Notice that the alternatives considered in this problem are marketing the product without the advertising campaign or marketing the product with the advertising campaign.

The EMV of the first alternative (without the ad. campaign) is

EMV[1] = $1.000.000 X 0.75 + $5.000.000 X 0.25 = $2.000.000

The EMV associated with the other alternative (with the ad. camp.) on the other hand is

EMV[2] = $1.000.000 X 0.4 + $5.000.000 X 0.6 - $1.000.000 X 1 = $2.400.000

Note that since the advertising campaign will certainly cost $1.000.000 its probability appears as one in the EMV equation of the second alternative.

As a result if the company uses the EMV as their decision mechanism in this problem, they should go ahead with the advertising campaign.

EMV measure can be used reliably in many cases where the marginal returns of the alternatives are similar, i.e. making 100,000 dollars vs. making 200,000 dollars with respective probabilities. However there are situations where this is inadequate, in the sense that it does not reflect the true 'values' of the alternatives. This is especially true in cases where the alternatives are substantially different in terms of the money or the prize gained.

Two examples which are common-place in everyday life will be given to illustrate this point:

Consider the car insurance companies, as a car owner one pays about the 3%-5% of the value of a car to have the car insured for a year. Actually the probability of having an accident is much less than that percentage (even in under-developed countries). However, people are willing to pay that amount due to the prospect of having to pay a large amount of money in case of an accident worries

you. This is actually why the insurance companies always profit by a comfortable margin.

The same applies to lotteries as well. The EMV of a lottery is actually much less than the amount one pays for participating in it. However, considering even the slightest possibility of being a millionaire, people are driven to lotteries. Consider the case someone is asked to pay a dollar for the prospect of winning 1million US$ with a probability of one in 10 million. The associated EMV is ten cents, but one will easily pay the requested amount due to of the exciting probability of winning a million dollars.

Assume that now the prize money in 10 times the previous one, 10 million US$. The probability of winning remains the same. Will anyone show the same eagerness to pay 10 dollars to attend this lottery? The answer is probably no, even though the associated EMV is exactly the same. The reason is the additional satisfaction obtained from successive millions of dollars does not tend to increase one's happiness as much as the first million. This phenomenon is referred to as the *diminishing marginal returns to utility* by the economists [14]. Diminishing marginal returns to utility is illustrated by the graph in Figure 6 which depicts the satisfaction obtained from the winnings in a lottery as a function of the amount of money won. The two cases above clearly indicate that EMV does not always yield accurate results as a decision making tool. A more suitable theory needs to be developed that can help a decision maker in the above cases

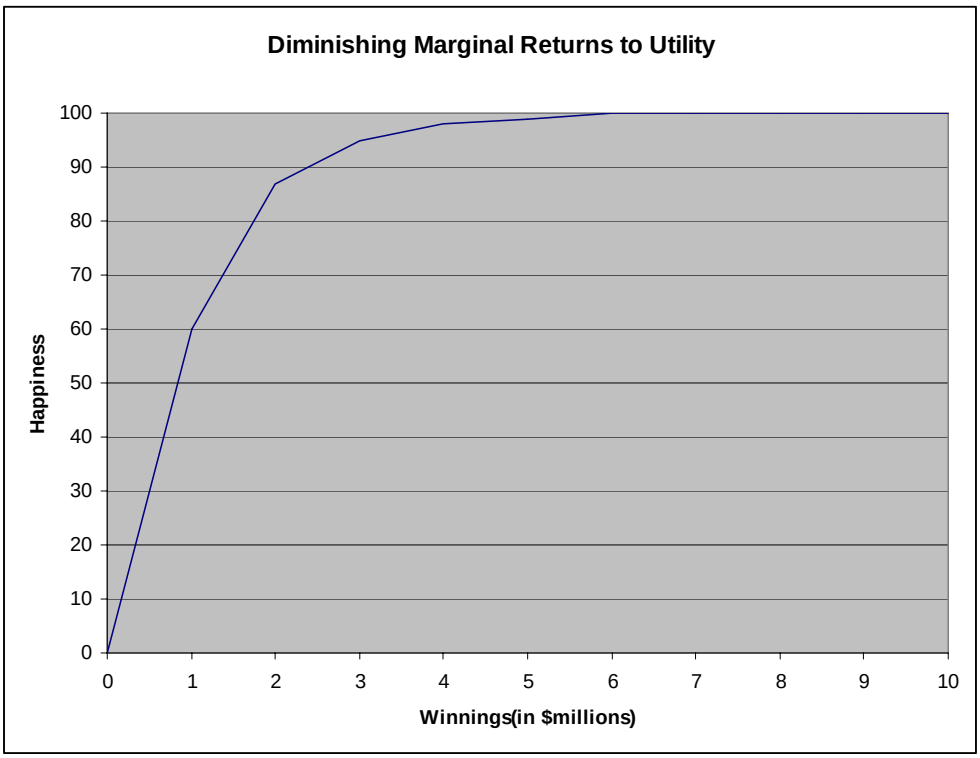

Figure 6 – Diminishing Marginal Returns to Utility

.

4.2 Fundamentals of Utility Theory

Before the alternative theory can be introduced, some new terms should be defined.

The maximum amount a person is willing to pay in order not to face a risk, that is the negative value of a risk to an individual is called the *certainty equivalent* (CE) for the risk. The same term is also used for the maximum amount, a person is willing to pay to be qualified to enter a lottery, in this case the CE is of course positive.

The difference between a decision makers CE and the EMV is called as the *risk premium*[14]

$$Risk\ \Pr emium = EMV - CE \tag{5}$$

There are three basic risk attributes that can be inferred from the sign of the Risk Premium[14].

- POSITIVE SIGN : When one is willing to pay more than its EMV to avoid a risk then the risk premium is positive. Notice that in this case both the EMV and the CE are negative. But since CE is more negative than EMV the overall sign of risk premium is positive. This type of behavior is termed as *risk averse* behavior. Larger Risk Premium indicates more risk aversive behavior.
- ZERO RISK PREMIUM : A zero risk premium indicates that the decision maker takes EMV as the main reference while making the decision and the value of a risk or lottery is exactly equal to its EMV. This type of behavior is *risk neutral* behavior.
- NEGATIVE SIGN : when one is willing to pay more money to enter a lottery than the associated EMV the risk premium becomes negative. Note that both the EMV and the CE are positive for this case. This type of behavior is called *risk seeking*. The more the magnitude of the risk premium, the more risk seeking is the behavior.

By now, it should be clear that the satisfaction (dissatisfaction), also called *utility* is not always proportional to the prize or money won(lost). When this is the

case, a more appropriate decision rule than the EMV, is to choose the alternative that maximizes the *expected utility* [EU]. The EU is defined as the average utility associated with an alternative. The main goal here is to obtain the satisfaction associated with an outcome as a function of the prize.

4.3 Determining the Utility Function

The relationship between the amount of money somebody won, x, and the associated satisfaction or utility by the same person is called as the utility function U(x). There are several methods for determining the utility function [14]. In all the methods, the aim is to identify the decision maker's relative utility for the individual amounts of money, one at a time, over the range of all possible values. The convention is to express the utilities on a range from 0 to 100. The best alternative corresponding to the highest possible prize money is given a 100 utility, whereas the worst alternative corresponding to the lowest possible prize money is given 0 utility. Finding the utility values in between is the remaining goal . There are several approaches for eliciting the utility values from the decision maker, these involve asking the decision maker the relative satisfaction obtained from different values of money, as compared to the maximum and minimum values, however a more reliable approach is to assume that the utility function has a certain form like

$$U(x) = x^c \tag{6}$$

OR

$$U(x)=(1-e^{-x/c}) \tag{7}$$

and try to estimate the parameter c.

When the utility function is determined, the decision makers risk attitude can be resolved from the shape of the utility function curve. This is illustrated in Figure 7. A person who is *risk neutral,* uses the EMV as the decision mechanism as described previously. As a result the utility curve of a risk neutral decision maker is a straight line, since the value of a certain amount of money is exactly equal to the amount of money itself.

A convex utility curve indicates a decision maker who is not satisfied with the current situation. Subsequent increases in wealth lead to greater and greater increase in satisfaction. The steep increase in utility as wealth increases characterizes a risk seeking behavior. On the other hand, a concave utility curve indicates a decision maker who is generally satisfied with the current situation, subsequent increases in wealth lead to smaller and smaller increases in satisfaction. A concave utility curve characterizes risk averse behavior.

These observations are consistent with the observations regarding the sign of the risk premium. Notice that a concave utility function, having utility values larger than that of the risk neutral utility function, corresponds to a positive Risk-Premium and therefore risk aversive behavior

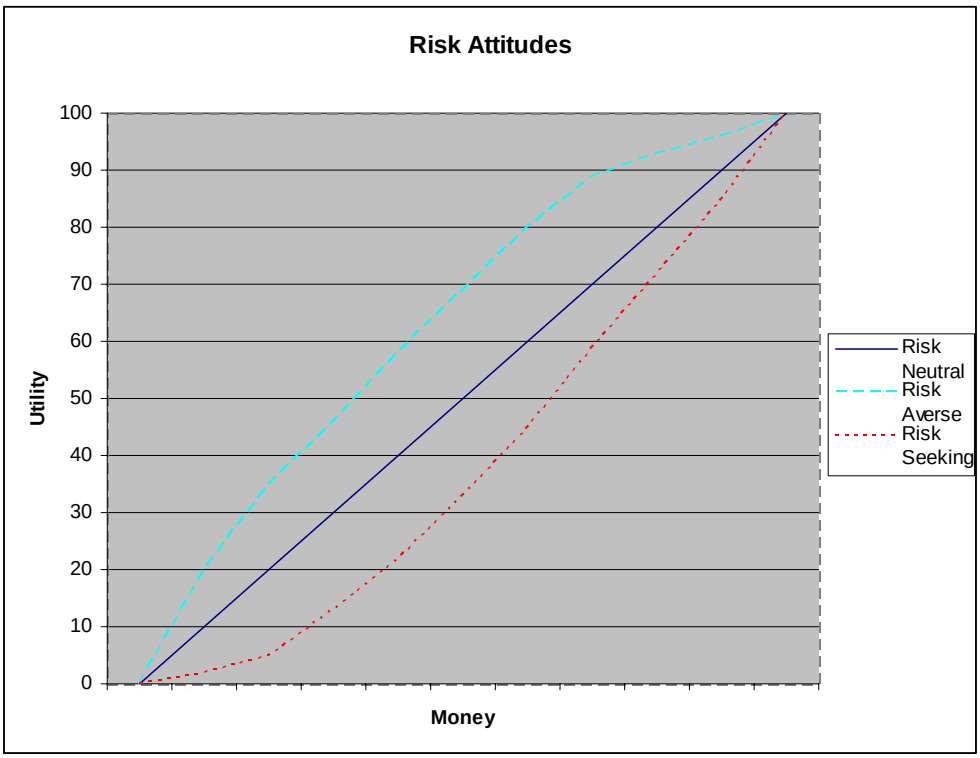

Figure 7 – The Risk Attitudes of a Decision Maker

. Whereas a convex utility function, having utility values smaller than that of the risk neutral utility function, corresponds to a negative Risk-Premium and therefore risk seeking behavior. The utility function curves for various risk attitudes illustrates the points stated above.

Also there are many cases where the utility curve has an S shape , which indicates that a decision maker who is initially dissatisfied by the current situation changes attitude as the amount of money he makes increases above a certain threshold.

4.3 Multiple Objectives and Total Satisfaction

In some cases, it is necessary to consider multiple objectives when dealing with a decision problem. For example, one may want to rent a house in an up-scale neighborhood but also require that the house is as close to his workplace as possible. In this case what determines the location of the house would be the relative importance of the objectives for the decision maker.

The total satisfaction is found by summing the satisfaction obtained from each of the objectives multiplied by the relative importance of each objective according to the following formula:

$$\begin{bmatrix} Total \\ Satisfaction \end{bmatrix} = w_1 U(obj_1) \quad + \quad w_2 U(obj_2) \quad + \ldots + \quad w_n U(obj_n) \tag{8}$$

This formula is called as the *additive utility function*[14], since the utilities of each outcome are weighted and added together.

In order to be able to use the additive utility function, for calculating the total satisfaction, an important assumption needs to be made about the relationship between the objectives.

The satisfaction on each objective needs to be independent of the satisfaction on every other objective. If this assumption does not hold, a more complicated formula is needed to calculate the total satisfaction.

In many cases the assumption of independence is quite reasonable, however there are also situations where this may not be true.

Consider the case when a decision maker is looking for a new job, he has two objectives, to have a decent salary and to work in a nice city. He is

considering two job offers one in Istanbul and the other in Ankara. In this case the satisfaction from the money objective depends on the other objective, because the salary that would provide him a comfortable living in Ankara may not be able to provide the same standard in Istanbul, so the additive utility function can not be used here.

A method of getting around this problem is to eliminate alternatives that cause dependence between the objectives. For the above example eliminate all job offers that pay below a certain threshold in Istanbul and then it can be

reasonably assumed that the satisfaction on individual objectives are independent of each other. This approach is valid in cases where the eliminated alternatives are the ones that are already unlikely to be chosen as the best alternative.

In the next chapter, the concepts borrowed from utility theory will be imported to video adaptation problem, in order to get the most satisfaction from viewed multimedia content on a device with limited resources.

CHAPTER 5.

VIDEO ADAPTATION BASED ON UTILITY THEORY

Utility theory has found many applications especially in the fields of engineering and economy, and has been successfully used for identifying the optimal course of action when faced with a decision making problem [14]

The application of utility theory to video adaptation is a new concept and some preliminary theoretical foundations have been developed [7]. This thesis proposes a more application-oriented and practical approach while maintaining the theoretical consistency.

5.1 Problem Formulation

The main aim of the thesis work is to accurately determine the "satisfaction" of a user resulting from watching a video clip on a resource limited device, as a function of video coding parameters. In the process of video coding, the parameters which should be known prior to encoding the video bit stream are *bit*

rate, *frame rate* and *spatial resolution*. Therefore, this research concentrates on evaluating the user satisfaction, while the above mentioned parameters are varied.While evaluating the user satisfaction, the properties of the end terminal that the video is going to be watched on, need to be taken into consideration. In this thesis, effects of the CPU and the screen size of the terminal on user satisfaction has been investigated. The effect of memory has been left out of the experiments and the calculations. The main reason for this decision is due to existence of a reserved RAM of the devices in POCKET PC class for rendering images to the screen. For the devices in the market, the amount of reserved memory is adjusted in such a way that, the memory never becomes a bottleneck in video applications. Since the system implemented in the thesis targets mainly the POCKET PC devices, memory has not been considered as a factor in determining the user satisfaction.

5.1.1 Subjective Evaluation of Satisfaction

Subjective evaluation of a user satisfaction, while watching a video clip, requires some careful consideration. Obviously, subjective tests should be conducted to quantify satisfaction.

All of the experiments that will be presented in this thesis, are conducted on a SIEMENS POCKET LOOX PDA, whose technical specifications can be found in Appendix B. During these tests, videos having different encoding parameters should be presented to the evaluator in a sequential manner. The evaluators should be asked to grade all the video clips, while specifying a number between 0 and 100, indicating the satisfaction they get from watching that particular clip.

During the evaluation, if all three of the parameters are varied simultaneously between two consecutive video clips, it will be very difficult for the evaluator to decide which clip is better, since one of the clips may have a larger spatial resolution, while the other can have a crisper picture. One might argue that the decision of the evaluator should be valid under all circumstances, since a subjective evaluation is being performed. However, when experiments were initially performed by varying all three of the parameters, the evaluators specified inconsistent grades in that scenario and often requested re-evaluating the previous samples, since they have changed their minds after watching new samples. At the end of such an evaluation, the grades had a random distribution, and lacked a consistent pattern to model the behavior. One approach to solve this problem is to adjust the video coding parameters one at a time. In such a case, the evaluators can grade the videos more accurately.

However, when the evaluation is carried out in this way, another problem arises: Every time the users need to evaluate a new video, they need to determine the contribution of the specific variable that is being changed to the overall utility of the video clip. In other words, when somebody watches a video having smoother motion (corresponding to higher frame rate), the amount of increase in the overall grade of the video needs to be determined, taking into account that the grade, also includes the contribution of the coding parameters that were kept constant.

A novel approach has been proposed, for solving the above mentioned problems that arise when the utility of a video clip needs to be determined subjectively.

5.1.2 Satisfaction Decomposition

The overall utility or the satisfaction a user gets from watching a video clip can be decomposed into three different components. These components are chosen such that the satisfaction, a user gets from one of these components, is independent from the satisfaction the same user gets from every other component. The components will be referred to as sub-objectives, in the remaining parts of this thesis, for the sake of compatibility with the terminology of utility theory.

Considering the fundamental video compression constraints, the sub-objectives are determined as:

1. the "crispness" utility of a video clip
2. the "motion-smoothness" utility of a video clip
3. the spatial resolution utility of a video clip

The determination of the sub-objectives follows naturally from video encoding parameters. The effects of those parameter values on human perception of a video clip or a picture are taken as the basis for such decomposition. In other words, when a high bit-rate video clip is watched, it is perceived as "a crisp" picture, or when a video with high-enough frame rate is viewed, it is perceived as a "motion smooth" video.

5.1.2.1 Crispness

The most accurate measure of the crispness of a video might be the number of bits, per pixel while encoding the clip. This determines the average number of discrete colors (or intensity levels), observable for a particular pixel; i.e. if the

pixel was coded with 24 bits, with 8 bits for each of the three color channels (R, G, B), then this means the number of different colors which can be displayed is equal to $2^{24} = 1{,}703{,}936$. In order to express bits per pixel, in terms of the coding parameters, the bit-rate needs to be divided by both frame rate and spatial resolution. In other words, the first component of the overall utility function should depend on all three video coding parameters. Hence, the first component of the overall utility function, the utility or satisfaction a user will get from the crispness of a video clip, can be formulated as

$$\begin{aligned} U_{crisp}(CBR,CSR,CFR) &= U_{crisp}(CBR/(CFR\ CSR)) \\ &= Ucrisp(coded\ bits\ per\ pixel) \end{aligned}$$

where *CBR* stands for Coded Bit Rate, *CSR* stands for Coded Spatial Resolution, and

CFR stands for Coded Frame Rate.

It should be noted that all the video coding parameters are referred as 'coded' parameters. The reason for such a referral is that when the video is rendered on a client device, the video parameters of the video being played back, may not be exactly the same as the parameters that were originally coded. The reason for this inconsistency is due to the resource constraints of the user terminals. The phrase "coded" is used to emphasize that the values being used here, are the original encoding values of the parameters, forced at the encoder.

5.1.2.2 Motion Smoothness

The motion smoothness of a video clip, is characterized by the coded frame rate of a video. It has been observed that a video coded with 25 frames per second or

higher appears perfectly smooth to the human eye. Actually, it is very difficult to notice any jerkiness in the motion of a video, on clips coded with frame rates higher than 15 fps, unless the video contains high velocity action. Videos are usually coded at 30 fps on NTSC systems and 25 fps on PAL systems, whereas motion pictures are coded at 24 fps.

Another factor required to be taken into account while trying to predict the motion smoothness of a video, is that a video, after being rendered on a client device, might be played back at a different frame rate compared to its original encoding. This is the case, when the limited capabilities of an end terminal are insufficient to accommodate the high frame rates, as explained in the previous section.

Intuitively, the frame rate at which the motion loses smoothness or in other words the frame rate at which, the observed frame rate, deviates from the original coded frame rate depends on the encoded video bit-rate. This is expected, since the bit rate of the video determines the number of bits that CPU of the user terminal needs to process per second (bits/s). After CPU is loaded beyond a certain point due to high bit-rate, it becomes impossible to render the video at higher frame rates. In light of the above discussion, it can be said that the motion smoothness of a video that will be observed on a user terminal, should depend on the frame rate at which the video was originally coded, the bit rate of the video, and CPU of the end terminal. Thus, the second component of the utility function is determined as

$$\mathrm{U}_{\mathrm{smooth}}(CFR, CBR, CPU)$$

where *CPU* stands for the clock speed of the CPU of the terminal on which the video is played back.

5.1.2.3 Spatial Resolution

Intuitively, the utility of the spatial resolution of a video clip should depend on two factors: Spatial resolution that the video is initially coded with and the *screen size* of the user terminal on which the video is to be displayed. One can easily realize that if a video, having a spatial resolution larger than the screen size of the terminal, is transmitted to the terminal, a portion of the video must be clipped in order to display the video on that device. This will inevitably result in reduced user satisfaction and should be avoided, if possible. The final component of the utility function is prototyped as follows:

$$U_{\text{size}}(CSR, ScreenSize)$$

where *Screen Size* stands for the physical screen size of the user terminal.

5.2 Utility Function Generation

The satisfaction for each of the sub-objectives mentioned in the previous section is assumed to be independent of the satisfaction on every other sub-objective. For example, the satisfaction a user gets from the motion smoothness of a video has no relation with the crispness of the same video. Therefore, one can use the *additive utility function* [14] to determine the total satisfaction a user will get from watching a certain video. Thus the final equation for the utility can be obtained as follows:

$$U \equiv w_1 U_{crisp}(\text{CBR}, \text{CSR}, \text{CFR}) + w_2 U_{smooth}(\text{CFR}, \text{CBR}, \text{CPU}) + w_3 U_{size}(\text{CSR}, \text{ScreenSize}) \quad (9)$$

The weights, w_1,w_2 and w_3, associated with the terms of the utility function, are to be determined by simulations, which are explained in Section 5.4.

The first step, in the process for obtaining the utility function, is determining the general forms of the component curves, for each sub-objective.

5.2.1 Crispness term in Utility Function

The satisfaction a user gets from crispness of a video, should increase substantially as the bits/pixel value is increased from zero, but in a similar manner to the economical counterparts, this increase is expected to reach saturation after a certain value of the bits/pixel. The reason for the above expectation is due to reaching human perception of crispness, after the bits/pixel value is increased beyond some point. Hence, increasing bits/pixel value further is not expected to result in a substantial increase in user satisfaction on crispness.

In the light of the above observations and past experience in economical applications of utility [14], it can be asserted that, the utility of crispness curve, should have an exponential form as expressed by the following formula.

$$U_{crisp}(CBR,CSR,CFR) = 1 - e^{-c_1 \frac{CBR}{CFR\ CSR}} \qquad c_1\ \alpha\ CSR \tag{10}$$

where c_1 is a function of CSR.

Equation (10) can be illustrated as in Fig. 3. The reason for the inclusion of the expression c_1, and its relation with CSR, as illustrated in figure, is explained, as follows:

It should be noted that the bits/pixel value obtained by the relation (CBR/CFR*CSR) in the above formula, is the bits/pixel value after compression. The compression algorithms use the redundancy in a picture to compress data. As a result, they have higher compression rates, while compressing pictures with larger spatial resolution, when pictures having more or less the same frequency content are considered. In this case, when pictures, differing only in their spatial resolution are compared, the picture with the larger spatial resolution is compressed much more efficiently, since it contains greater redundancy compared the one with the lower spatial resolution.

Therefore, bits/pixel value required to code a picture with a given crispness, is smaller for the pictures having higher spatial resolutions. In order to account for this fact, a function c_1 has been included in the above formulation. Larger the value of c_1, higher the crispness of a video for a given bits/pixel value. Hence, c_1 should be directly proportional with the CSR. Fig.8 illustrates this fact.

5.2.2 Motion Smoothness term in Utility Function

While increasing CFR, the second component of the utility function, utility of motion smoothness, should increase up to point, in an exponential expression, similar in form, to the utility of crispness, as shown in Eq. (11).

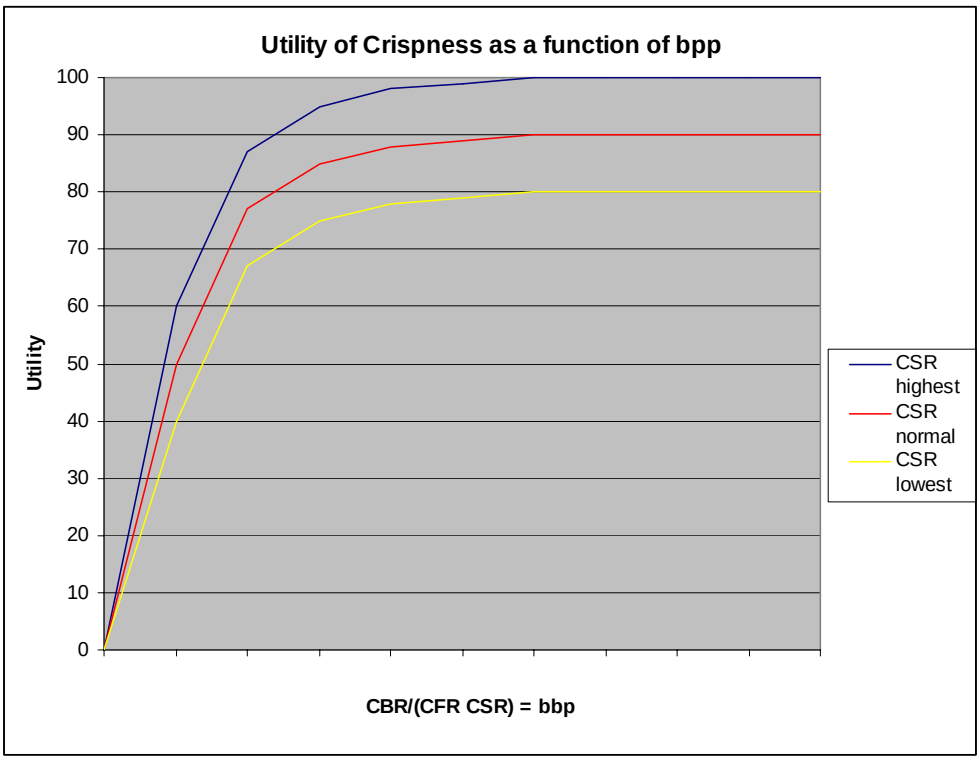

Figure 8 Typical curves for the crispness term in utility function for different CSR values

$$\mathrm{U}_{\mathrm{smooth}}(CFR,CBR,CPU)=\begin{cases}1-e^{-a_0\ CFR} & CFR \le FR_L(CBR)\\ a_1 e^{-c_2(CFR-FR_L(CBR))} & CFR > FR_L(CBR)\end{cases}\Big\}CPU\ Low$$

$$\mathrm{U}_{\mathrm{smooth}}(CFR,CBR,CPU)=\begin{cases}1-e^{-a_0\ CFR} & CFR \le FR_H(CBR)\\ a_1+1-e^{c_3(CFR-FR_H(CBR))} & CFR > FR_H(CBR)\end{cases}\Big\}CPU\ High \quad (11)$$

$FR\ \alpha\ 1/CBR$

$a_1 = 1-e^{-a_0 FR}$

Equation 11 can be illustrated as in Fig. 9.

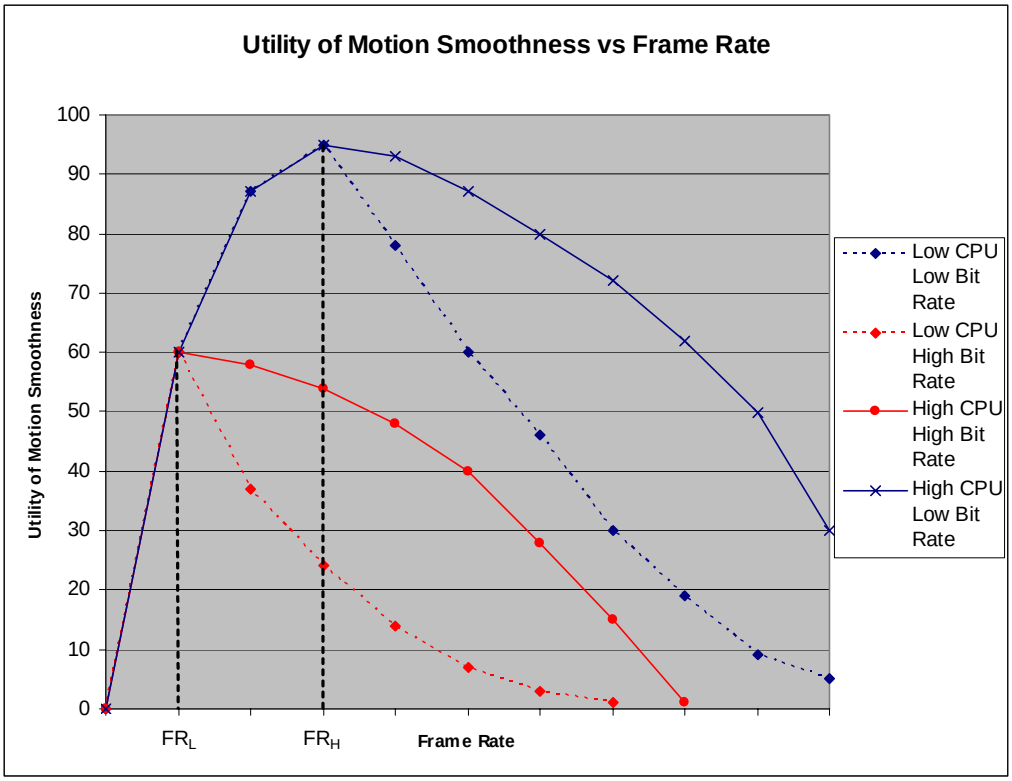

Figure 9 Typical curves for the motion smoothness term in utility function for different CBR and CPU values

The point at which the utility of motion smoothness should start decreasing, due to resource limitations, should depend on the coded bit rate of the video, as stated earlier. In the formulation above, the function FR(CBR) determines the exact location of the point. After the value of bit rate exceeds the value specified by FR, the motion smoothness utility should start dropping. In other words, the resource limited device will not be able to playback the video at the CFR.

The dependence of motion smoothness utility, on the clock-speed of the CPU of the terminal device, is simplified. In other words, the formulation is performed for two different clock speeds only, one representing *high CPU,* and the other representing *low CPU.* Any CPU value input to the system, will be

classified as high or low CPU according to a simple threshold for the sake of simplicity.

Notice that, in Figure 9, the portion of the curve, where the utility is increasing (up to the limit defined by FR), the utilities of the high CPU and the low CPU cases are assumed to be the same. This is reasonable, since the observed and the coded frame rates are the same up to that point, and a particular frame rate, gives the same utility across all platforms unless distorted by the resource constraints. The a_0 term in Eq. (11) is a constant that will be determined based on the results of the performed subjective tests.

FR should be inversely proportional with CBR and this means that as the bit rate is increased, the observed frame rate starts deviating from the coded frame rate, to smaller frame rates. The function FR is different for the cases of high CPU and low CPU, so are the functions c_2 and c_3 which determine the exact form of the utility curve at the declining portion of the curve. Both c_2 and c_3 are functions of CBR. Naturally, one should expect the utilities to drop more severely, as CBR increases, because of the resource constraints of the user terminal, which are discussed previously. Notice that in both curves of Eq. (11), for larger c_2 or c_3 the curve drops faster, so using c_2 and c_3 in direct proportionality to CBR, the desired form for the utility curves can be obtained. The point a_1 in both curves, is the value of utility, at which the functions start decreasing.

5.3.2 Spatial Resolution term in Utility Function

Finally, the utility of the spatial resolution of a video clip is expected to increase in a similar fashion to Eqns. (10) and (11), up to the point where the spatial resolution, becomes equal to the physical screen size of the user terminal. After that point, the utility is expected to decline conforming to the following equation:

$$U_{size}(CSR, ScreenSize) = \begin{cases} 1 - e^{-a_{21} CSR} & CSR \le ScreenSize \\ c_4 e^{-a_{22}(CSR - ScreenSize)} & CSR > ScreenSize \end{cases}$$

$$c_4 = 1 - e^{-a_{21} ScreenSize} \qquad (12)$$

$$a_{21} \quad \alpha \quad 1/ScreenSize$$

$$a_{22} \quad \alpha \quad 1/ScreenSize$$

Eqn. (12) can be illustrated as in Fig. 10.

The functions a_{21} and a_{22} are both inversely proportional with the screen size of the terminal. Note that, larger a_{21} leads to a steeper increase, in the increasing portion of the utility function, and larger a_{22} means a steeper decrease in the declining portion of the utility curve.

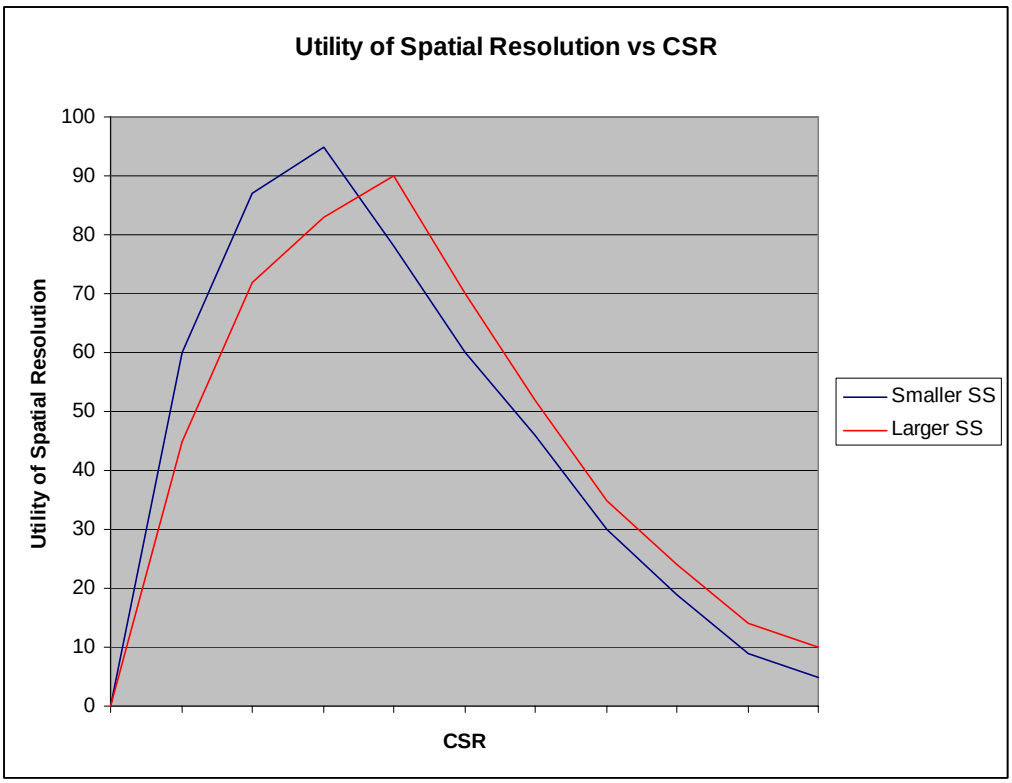

Figure 10 Typical curves for the spatial resolution term in utility function for different Screen Size values

Since the increase and decrease in utility is expected to change more abruptly in devices with smaller screen sizes, the inverse proportionality of a_1 and a_2 with screen size is reasonable. The reason for this expectation might be explained as follows: People tend to watch devices having smaller screens from a smaller distance to the device; as a result small changes in the display tend to affect them more than they would normally effect on devices having larger screens.

5.3 Subjective Tests for Determining the Utility Function

The next step towards determining the components of the utility function uniquely, is a series of subjective evaluation experiments. The experiments are performed by three individuals, who will be referred to as 'evaluators' from this point on in the thesis. The experiments are performed separately for each component. As a result, each evaluator is required to perform a total of 195 tests as illustrated in Figures 11,12,13 and 14. While an experiment on one of the components is being performed, the video coding parameters not affecting the utility of that component are kept constant. The steps followed while performing the experiments are as follows:

The evaluators are first shown the videos that are considered the best and the worst for the particular component of the utility function, for which the experiment is being performed. For example, the video having a frame rate of 1 fps is shown as the worst sample for the utility of the motion smoothness component, whereas the video having a frame rate of 25 fps, is presented as the best sample. While the utility of the worst sample is fixed at 0, and the utility of the best sample is fixed at 100, as required by utility theory. Obviously, the spatial resolution of the sample videos is held constant (at 176x144 pixels) throughout the test performed for motion smoothness. All the experiments are repeated for five different bit-rates, since bit-rate affects the motion smoothness utility, as explained above.

The evaluators are shown videos, coded with different values of the parameter(s) that has an influence on the component of the utility being tested. The evaluators are asked grade those samples with grades ranging between 0 and 100, according to the satisfaction they get from watching that video. The important point here is that they are asked to evaluate the videos, only according to the component being tested. For example, if the motion smoothness was being tested they are asked to express their satisfaction only regarding the motion smoothness of the video, ignoring their satisfaction regarding the crispness or the size of the video.

The tables showing the results of the subjective tests, together with their graphical representations, are shown. Figures 11,12-13 and 14 represent the utilities of crispness, motion smoothness and spatial resolution, respectively.

5.4 Fitting Parameters of Utility Function

The final step for determining the component utilities, is computing the values of the unknown constants or expressions specified in Eqns. (10)-(12). In order to accomplish this goal, the estimates for parameters are obtained first by minimizing the error between the experimental data and the forms of the curves, which are previously stated. As a result of this minimization, a set of data points satisfying the Eqns. (10)-(12) are obtained. Afterwards, a number of curves are fitted to the obtained data points by the help of the MATLAB's curve fitting toolbox.

In order to illustrate the procedure, the determination of the expression c_1 used in Eqn. (10) will be explained, the other expressions and constants are determined in a similar manner.

First by rearranging Eqn. (10),

$$\frac{CFR\ \ CSR}{CBR}\ln(1-U_{crisp})+c_1=0 \tag{13}$$

Eqn. (13) should hold for all the experimental data, if there were no errors. Minimizing sum of squared errors with respect to unknown c_1 gives the optimal c_1 value.

$$\sum_{\substack{experimental\\ utility\ values}}\left(\frac{CFR\,CSR}{CBR}\ln(1-\widetilde{U}_{crisp})+c_1\right)^2 \tag{14}$$

In order to find c_1, which makes Eqn. (14) minimum; first one should take its derivative and then equate to zero before solving for c_1.

$$c_1=\frac{1}{n}\sum_{\substack{experimental\\ utility\ values}}\frac{CFR\,CSR}{CBR}\ln\left(\frac{1}{1-\widetilde{U}_{crisp}}\right) \tag{15}$$

For different values of CSR, the resulting c_1 values of Eqn. (10) are tabulated in Table 5.

In curve fitting phase, MATLAB is utilized for its *least squares fitting algorithm*, to fit curves to the observed data.

Table 1 – The utility of crispness as a function of bits/pixel for different values of spatial resolution

bits/pixel	1.03	1.28	1.55	1.76	1.95	2.13	2.31	2.58	3.01	4.03	9.82	88/72
utility	0	42	56	67	74	79	83	89	95	100	100	
bits/pixel	0.38	0.57	0.78	0.94	1.08	1.21	1.35	1.55	1.89	2.68	7.57	176/144
utility	0	57	70	75	80	86	90	90	93	99	100	
bits/pixel	0.27	0.42	0.57	0.71	0.82	0.93	1.04	1.21	1.5	2.18	6.81	264/216
utility	0	48	70	82	86	86	87	90	97	99	100	
bits/pixel	0.21	0.33	0.48	0.59	0.69	0.78	0.88	1.04	1.31	1.99	6.14	352/288
utility	0	45	60	76	83	88	90	94	98	99	100	

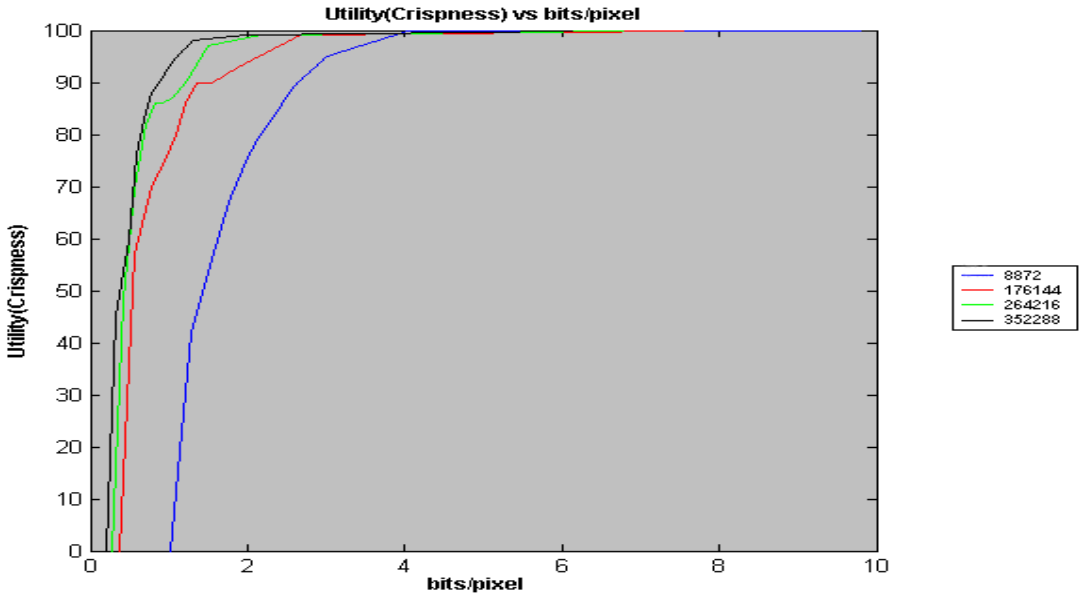

Figure 11 - The utility of crispness as a function of bits/pixel for different values of spatial resolution

Table 2 - The utility of the motion smoothness of a video in terms of the frame rate for different values of bit rate (High CPU case)

	FRAME RATE (fps)												
Bit Rate	1	3	5	7	9	11	13	15	17	19	21	23	25
50k	0	50	70	80	94	100	100	100	97	96	96	93	90
100k	0	50	69	81	94	100	100	100	100	100	100	100	99
150k	0	50	70	80	95	100	100	100	98	96	93	90	77
200k	0	50	70	80	93	88	88	82	79	72	65	52	45
250k	0	50	70	80	95	88	88	82	77	70	70	60	50

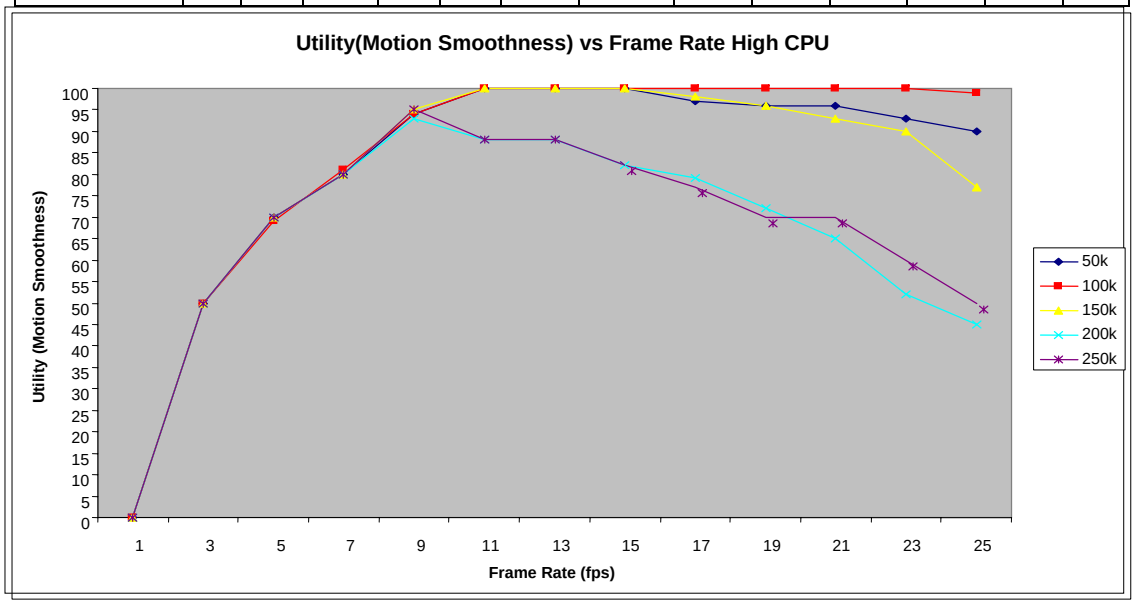

Figure 12 The utility of the motion smoothness of a video in terms of the frame rate for different values of bit rate (High CPU case)

Table 3 - The utility of the motion smoothness of a video in terms of the frame rate for different values of bit rate (Low CPU case)

	FRAME RATE (fps)												
Bit Rate	1	3	5	7	9	11	13	15	17	19	21	23	25
50k	0	50	70	80	94	100	100	100	94	90	90	85	72
100k	0	50	69	81	94	100	100	100	100	100	93	87	68
150k	0	50	70	80	95	100	100	100	98	83	71	67	52
200k	0	50	70	80	90	86	75	65	56	47	40	30	25
250k	0	50	55	62	55	42	30	25	20	20	15	4	0

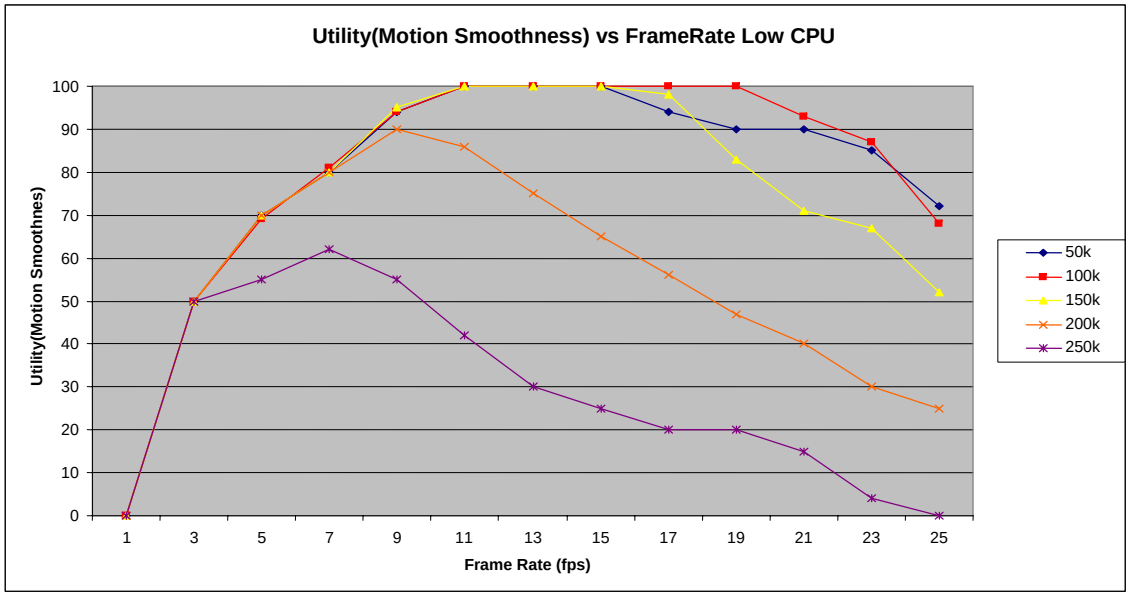

Figure 13 - The utility of the motion smoothness of a video in terms of the frame rate for different values of bit rate (Low CPU case)

Table 4 – The utility of the spatial resolution of a video in terms of the spatial resolution for different values of screen size(High CPU case)

	Spatial Resolution						
Screen Size	88-72	132-108	176-144	220-180	264-216	308-252	352-288
320*240	0	40	57	74	85	95	89
264*216	0	45	59	80	100	89	82
176*144	0	62	100	79	62	54	47

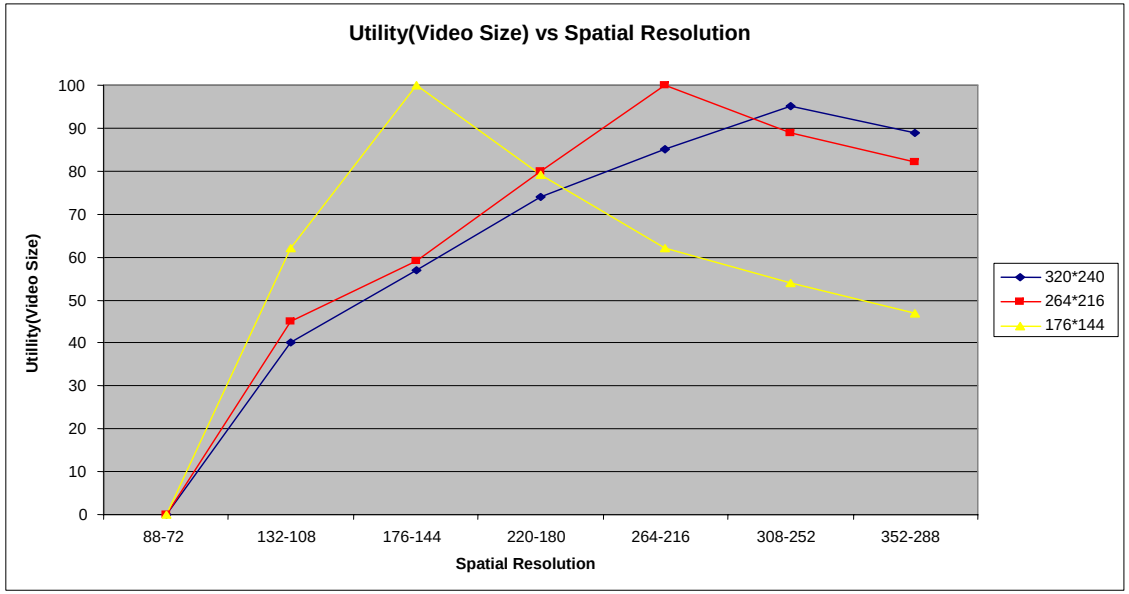

Figure 14 - The utility of the spatial resolution of a video in terms of the spatial resolution for different values of screen size(High CPU case)

The method minimizes *summed square of residuals* between the observed data and the fitted curve. The summed square of residuals is given by the following formula

$$S = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - y_i')^2$$

All the curves, which are fitted to the functions, are selected as exponential fits, with the exception of a_{21} of Eqn. (11), since the exponential fit yields a high residue for that case.

The results of the curve fitting for c_1 of Eqn. (10), performed by the curve fitting toolbox of MATLAB are shown in Fig. 15. The dots indicate the data points, and the curve shows the fitted model by MATLAB. The function which is obtained from the curve fitting is as follows:

$$c_1(CSR) = a\,e^{(b\,CSR)} + c\,e^{(d\,CSR)} \tag{16}$$

where a=1.831, b=1.393 10^{-6}, c=-1.452, d=-3.538 10^{-5}.

Similarly, the data points that are obtained for FR_h of Eqn. (11) are as shown in Table 6. The results of the curve fitting for FR_h of Eqn. (11), are as shown in Fig 16. The function that was obtained from the curve fitting is as follows.

$$FRh(CBR) = a\,e^{(b\,CBR)} \tag{17}$$

where a=48.28 , b=-7.6 10^{-3}

The reason for using a simpler expression for the curve fitting of FRh can be explained as follows: When a curve having a similar form to the fitted curve of c1

is used, the fitted function started increasing after a certain value of CBR. This is not desirable, since the actual FRh is always inversely proportional with CBR.

The data points that are obtained for FR_L are as shown in Table 7.The results of the curve fitting for FR_L of Eqn. (11), are illustrated in Fig. 17. The function which is obtained from the curve fitting is as follows.

$$FR_L(CBR) = a\,e^{(b\,CBR)} + c\,e^{(d\,CBR)} \tag{18}$$

where a=131, b=-0.01098, c=-130.2 and d=-0.01666

Similarly, the data points that are obtained for c2 of Eqn. (11) are as shown in Table 8. The results of the curve fitting for c2 are as shown in Figure 18. The function obtained from the curve fitting is as follows:

$$C_2(CBR) = a\,e^{(b\,CBR)} \tag{19}$$

where a=0.001619 and b=0.01029.

The data points that were obtained for c3 of Eqn. (11) are as shown in Table 9. The results of the curve fitting for c3 are as shown in Fig. 19. The function that was obtained from the curve fitting is as follows:

$$C_3(CBR) = a\,e^{(b\,CBR)} + c\,e^{(d\,CBR)} \tag{20}$$

where a=0.05326, b=-0.004674, c=2.8 10^{-5} and d=0.02913

The constant C_1 in sub_obj2 was found to be "0.238" using MSE methods discussed above.

The data points that are obtained for a_{21} of Eqn. (12) are as shown in Table 10. The results of the curve fitting for a_{21} are as shown in Fig. 20. The function that is obtained from the curve fitting is as follows

$$a_{21}(Screen\ Size) = a\,e^{(b\,Screen\ Size)} \tag{21}$$

where a=16.32 and b=-2.548 10^{-5}

The data points that were obtained for a_{22} of Eqn. (12) are as shown in Table 11. The results of the curve fitting for a_{21} are as shown in Fig. 21. The function that was obtained from the curve fitting is as follows:

$$a_{22}(Screen\ Size) = a\,e^{(b\,Screen\ Size)} \tag{22}$$

where, a=59.28 and b=-7.68 10^{-5}

Table 5 – The data points for c_1 in sub-objective 1

CSR	C1
88*72	0.681
176*144	1.32
264*216	1.78
352*288	2.07

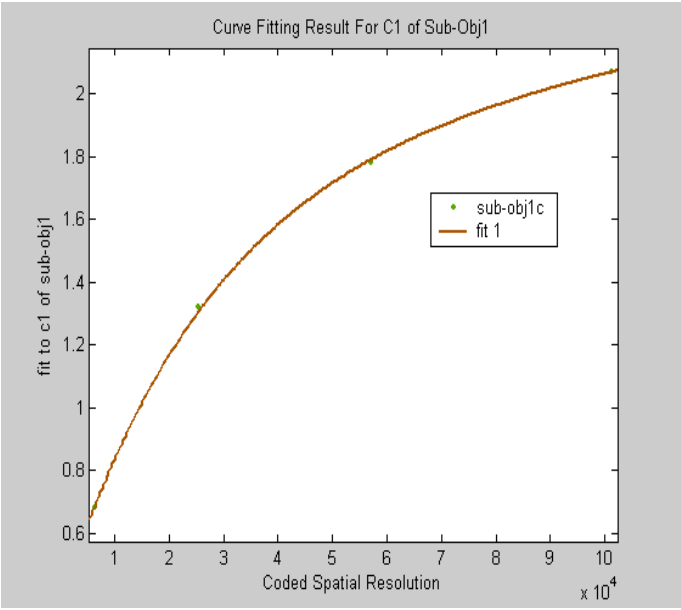

Figure 15 – Curve fitting results for c1 of sub-obj 1

Table 6 – the data points for FRh in sub-objective 2

Coded Bit Rate (Kbits/s)	Frame Rate(fps)
100	23
150	15
200	9
250	9

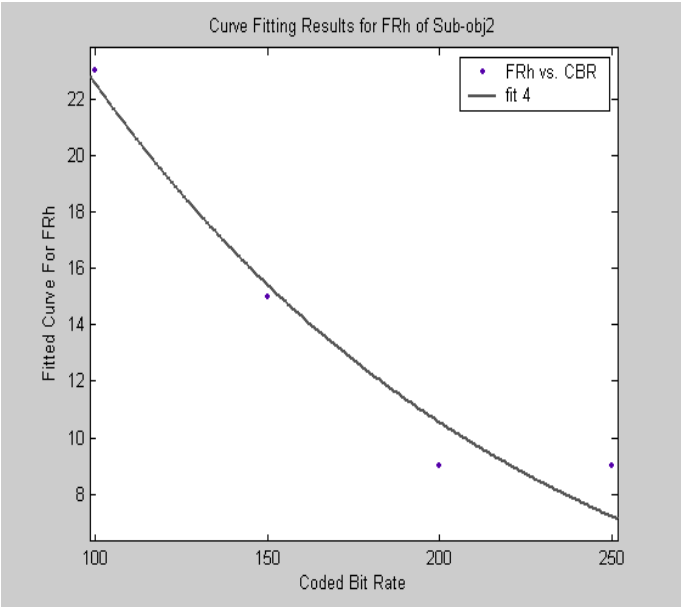

Figure 16 - The Curve fitting results for FRh in sub_obj 2

Table 7 – The data points for FRl in sub-obj 2

Coded Bit Rate (Kbits/s)	Frame Rate(fps)
100	19
150	15
200	9
250	7

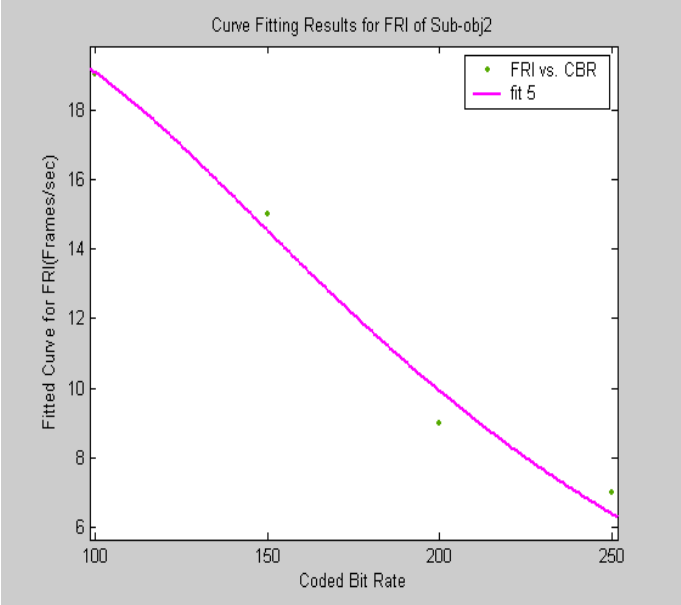

Figure 17 - The Curve fitting results for FRl in sub_obj 2

Table 8 – The data points for c2 in sub-obj 2

Coded Bit Rate (Kbits/s)	C2
100	0.00292
150	0.00884
200	0.01926
250	0.01714

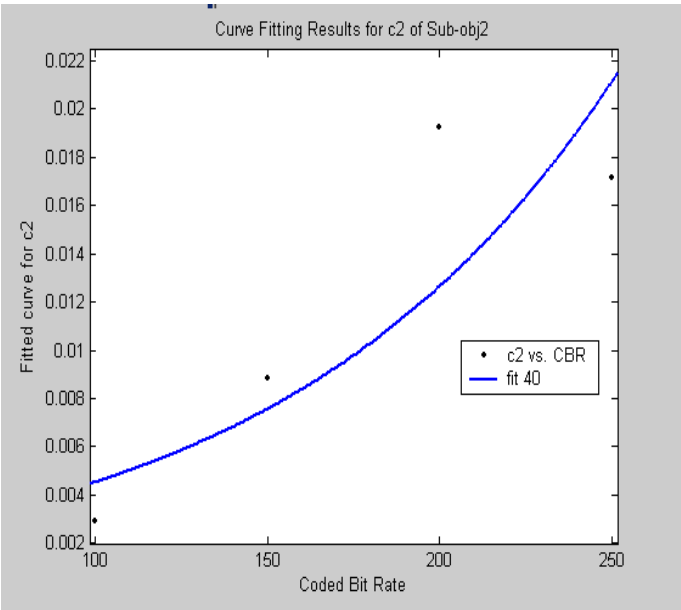

Figure 18 – The curve fitting results for c2 in sub-obj 2

Table 9 – The data points for c3 in sub-obj 2

Coded Bit Rate (Kbits/s)	C3
100	0.03389
150	0.02863
200	0.0304
250	0.05726

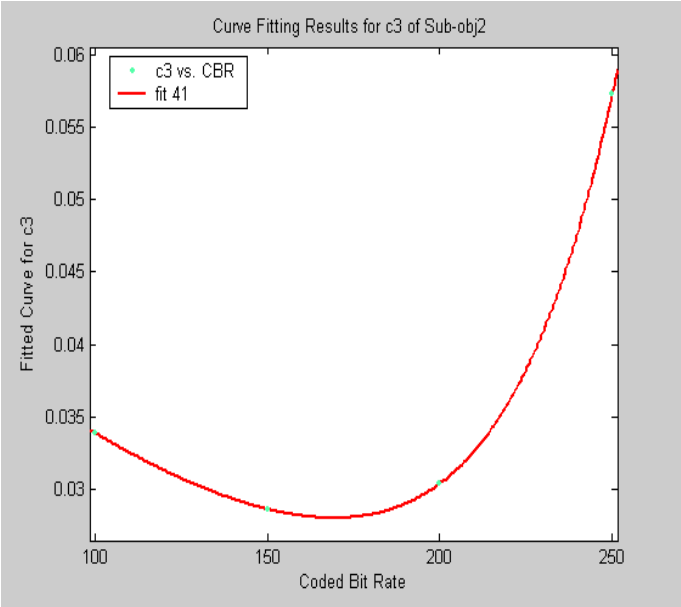

Figure 19 – The curve fitting results for c3 in sub-obj 2

Table 10 – The data points for a_{21} in sub-obj3

Screen Size	a_{21}
176*144	8.32
264*216	3.97
320*240	2.917

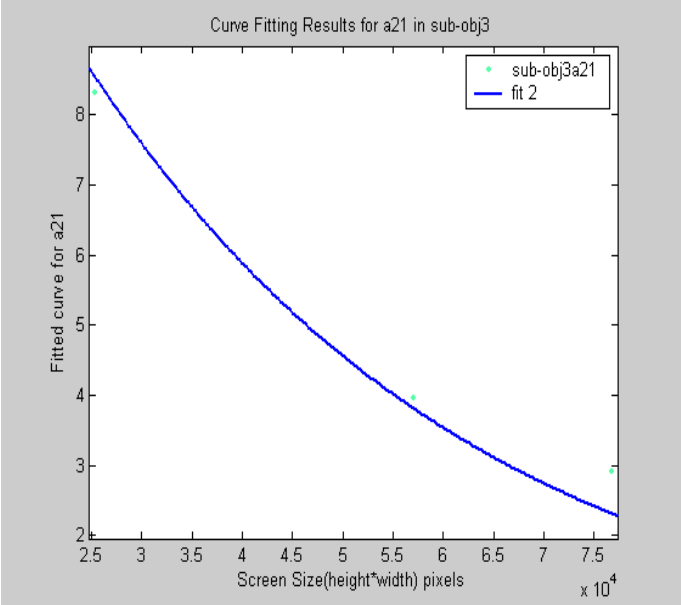

Figure 20 – The curve fitting results for a_{21} in sub-obj 3

Table 11 – The data points for a_{22} in sub-obj3

Screen Size	a22
176*144	8.427
264*216	1.162
320*240	0.1825

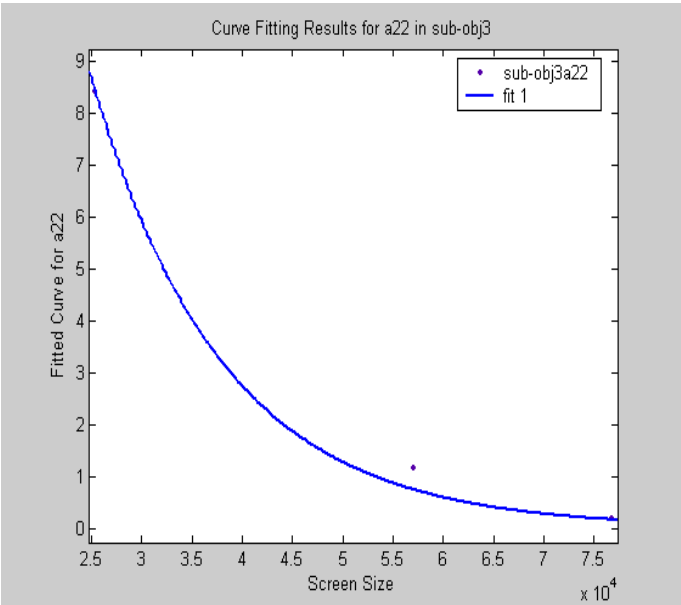

Figure 21 – The curve fitting results for a_{22} in sub-obj 3

5.5 Utility Maximization

For maximizing the obtained utility function, a stochastic optimization technique, known as *simulated annealing* [16] is used. The advantage of stochastic methods over their deterministic counterparts is having a lower probability of getting stuck in local maxima/minima.

5.5.1 Simulated Annealing

Actually, it can be guaranteed that stochastic algorithms will converge to global minima/maxima, if optimization parameters are chosen appropriately. However, ensuring global convergence significantly increases computational complexity of these algorithms.

Simulated annealing is actually inherited from metallurgy. When a metal is heated to high temperatures near its melting point, and then cooled down very slowly to room temperature, dislocations and other crystal lattice irregularities are eliminated [15].

In the simulated annealing algorithm, the objective function to be minimized is evaluated at a randomly chosen initial point $\vec{x}_0$. Then, one of the variables of the function is changed randomly and the function is evaluated at this new point or state $\vec{x}_i$. X's are vectors, since the objective function can be a function of many variables. If the new value of the function is lower than the initial one, the current point, $f(\vec{x}_i)$, is used as a starting point for the next iteration of the algorithm. However, even if the new value is higher than the old value, it is accepted as the new state of the system with a probability [16] :

$$P = \exp(-(f(\vec{x}_i) - f(\vec{x}_0))/T) \tag{18}$$

This is the main reason for this minimization procedure being different from the other approaches. This algorithm can move in a direction which is less favorable (in the sense that the value of the function is higher), with a certain probability, which means that it can jump out of a local minima, even if moving out means going in a direction that increases the function.

'T' is a control variable (temperature); for larger T, the probability that a state change increasing the value of the function will be expected is larger. T, here is analogus to the temparature. The algorithm starts with a large value of T, then T is slowly decreased. The simulted annealing optimization algoritm can be summarized as follows [17]:

1. Choose a random starting point $\vec{x}_0$, Set T=T_{max} and i=0.
2. Choose a new state $\vec{x}_{i+1}$ randomly.
3. Compute $\Delta f = (f(\vec{x}_{i+1}) - f(\vec{x}_i))$.
4. Compute P from

$$P = \begin{cases} \exp(-\Delta f / T) & if \quad \Delta f > 0 \\ 1 & if \quad \Delta f \le 0 \end{cases}$$

5. if P=1, accept $\vec{x}_{i+1}$ as the new state(i.e. $x_i = x_{i+1}$); otherwise generate a random number having a value in the range [0,1]. If the number is less

than P, accept $\vec{x}_{i+1}$ as the new state. If the number is greater than P, do not accept the new state (i.e. $x_{i+1} = x_i$)

6. Set i = i + 1; if i<I_{max}, where I_{max} is predetermined, go to step 2
7. Set i=0, $\vec{x}_0 = \vec{x}_{I\max}$, decrease T, if T>$T_{min}$ go to step 2, otherwise terminate.

The sequence of temperatures, and the number I_{max} is called as the annealing schedule. The temperature is decreased in accordance with a temperature scale function, which has the form $T(i+1) = c\,T(i)$, where c is a constant.

The cooling process must be slow enough, so that the system can move to any part of the function before being trapped in a local minimum; i.e. that is the system needs to spend enough time in the high energy state, so that, it can test all the regions of the objective function. Values of 'c' in the range $0.8 \le c \le 0.99$ have been found to work well in many real life problems [15].

The value of initial temperature needs to be high enough, so that, all configurations of the system are roughly equally probable (i.e. initial T must be larger than the largest possible Δf). This is necessary, since the random initial starting point can be far from optimal.

A final requirement for the convergence of the algorithm to the global minimum, is necessity of the final temperature to be low enough, so as not to allow the system to climb out of the global minimum.

The source code of this algorithm can be found in the Appendix A. The parameter values utilized in this implementation are equal to C = 0.95, T_{max} =10, T_{min} = 0.0001 and I_{max}=1000.

5.5.2 Determining Weights of the Utility Function

The final stage in maximizing the utility function is obtaining the values of the trade-off weights, w_1,w_2,w_3, used in Eqn. (9), at which the utility function has the maximum value.

At this point, the question is whether the values of trade-off weights, at which the utility function is maximum, change for different values of screen size and CPU; i.e. for different user terminals. A series of experiments are performed to answer this question by obtaining the optimum values of the trade-off weights, for four different user terminals having CPU clock speed and screen size values of (400MHz, 176x144), (400MHz, 352x288), (200 MHz, 176x144) and (200MHz, 352x288). The trade off weights are varied between 0.1 and 0.5, in 6 discrete levels (0.1,0.2,0.3,0.33,0.4,0.5) in such a way that, the sum ($w_1 + w_2 + w_3$) always amounted to unity. The results are shown in Table 12.

It can be seen that the value of the weights that maximize the utility function do not change for different user terminals. Although the terminals, on which the experiments, are performed do not span the entire range of terminals, the results are expected to be approximately the same. In the light of these experiments, the trade-off weights are determined as W_1=0.1, W_2=0.4, W_3=0.5.

At this point, the utility function is uniquely determined. The optimal values of the video encoding parameters, CBR,CFR and CSR, can be found for any given terminal device. The system needs only the CPU and the screen size of the terminal device to compute the optimal values of the video coding parameters,

using the utility function with the weights, as determined above. A user can also specify different values for the weights according to his/her personal preferences.

5.6 Video Adaptation Simulations

The simulated annealing algorithm is implemented in MS Visual C++ 6.0, as already stated. The execution time of the algorithm takes 19 seconds on a P4 2.4 GHz PC having 512 MB of RAM. A basic user interface has been designed, through which a user can enter the CPU and the screen size of the end terminal, and also optionally specify trade-off weights other than the default values obtained in the previous section. In a fully automated system, it is expected to receive this information after handshaking between the mobile client and the server. A screen shot of the interface can be seen in Figure 22.

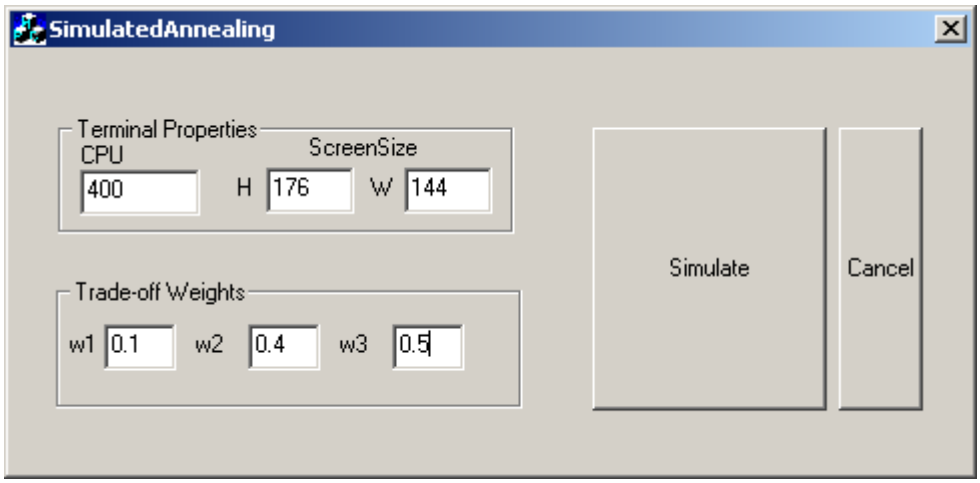

Figure 22 – The User Interface of the Implementation of the Simulated Annealing Algorithm

Table 12 – Experiments Performed By Varying The Weights Of The Components Of The Utility Function For Specific User Terminal Configurations

CPU-ss	w1	w2	w3	utility
200-176144	0,33	0,33	0,33	0,65732
200-176144	0,3	0,3	0,4	0,697564
200-176144	0,3	0,4	0,3	0,696707
200-176144	0,4	0,3	0,3	0,597608
200-176144	0,2	0,4	0,4	0,796663
200-176144	0,2	0,3	0,5	0,797519
200-176144	0,2	0,5	0,3	0,795806
200-176144	0,3	0,2	0,5	0,69842
200-176144	0,5	0,2	0,3	0,498509
200-176144	0,4	0,2	0,4	0,598465
200-176144	0,5	0,3	0,2	0,497652
200-176144	0,4	0,4	0,2	0,596751
200-176144	0,3	0,5	0,2	0,69585
200-176144	0,1	0,4	0,5	0,896618
200-176144	0,1	0,5	0,4	0,895762
200-176144	0,5	0,1	0,4	0,499201
200-176144	0,4	0,1	0,5	0,599321
200-176144	0,4	0,5	0,1	0,595895
200-176144	0,5	0,4	0,1	0,496796
MAX	0,1	0,4	0,5	0,896618

CPU-ss	w1	w2	w3	utility
200-352288	0,33	0,33	0,33	0,65732
200-352288	0,3	0,3	0,4	0,69756
200-352288	0,3	0,4	0,3	0,69671
200-352288	0,4	0,3	0,3	0,59761
200-352288	0,2	0,4	0,4	0,79666
200-352288	0,2	0,3	0,5	0,79752
200-352288	0,2	0,5	0,3	0,79581
200-352288	0,3	0,2	0,5	0,69842
200-352288	0,5	0,2	0,3	0,49851
200-352288	0,4	0,2	0,4	0,59847
200-352288	0,5	0,3	0,2	0,49765
200-352288	0,4	0,4	0,2	0,59675
200-352288	0,3	0,5	0,2	0,69585
200-352288	0,1	0,4	0,5	0,89662
200-352288	0,1	0,5	0,4	0,89576
200-352288	0,5	0,1	0,4	0,4992
200-352288	0,4	0,1	0,5	0,59932
200-352288	0,4	0,5	0,1	0,5959
200-352288	0,5	0,4	0,1	0,4968
MAX	0,1	0,4	0,5	0,896618

CPU-ss	w1	w2	w3	utility
400-176144	0,33	0,33	0,33	0,659044
400-176144	0,3	0,3	0,4	0,699131
400-176144	0,3	0,4	0,3	0,698801
400-176144	0,4	0,3	0,3	0,599172
400-176144	0,2	0,4	0,4	0,79876
400-176144	0,2	0,3	0,5	0,79909
400-176144	0,2	0,5	0,3	0,798429
400-176144	0,3	0,2	0,5	0,699462
400-176144	0,5	0,2	0,3	0,494921
400-176144	0,4	0,2	0,4	0,599503
400-176144	0,5	0,3	0,2	0,499213
400-176144	0,4	0,4	0,2	0,598842
400-176144	0,3	0,5	0,2	0,69847
400-176144	0,1	0,4	0,5	0,898719
400-176144	0,1	0,5	0,4	0,898388
400-176144	0,5	0,1	0,4	0,498687
400-176144	0,4	0,1	0,5	0,597626
400-176144	0,4	0,5	0,1	0,598511
400-176144	0,5	0,4	0,1	0,498883
MAX	0,1	0,4	0,5	0,898719

CPU-ss	w1	w2	w3	utility
400-352288	0,33	0,33	0,33	0,65904
400-352288	0,3	0,3	0,4	0,69888
400-352288	0,3	0,4	0,3	0,6988
400-352288	0,4	0,3	0,3	0,59917
400-352288	0,2	0,4	0,4	0,79876
400-352288	0,2	0,3	0,5	0,79909
400-352288	0,2	0,5	0,3	0,79843
400-352288	0,3	0,2	0,5	0,69946
400-352288	0,5	0,2	0,3	0,49492
400-352288	0,4	0,2	0,4	0,5995
400-352288	0,5	0,3	0,2	0,49921
400-352288	0,4	0,4	0,2	0,59884
400-352288	0,3	0,5	0,2	0,69847
400-352288	0,1	0,4	0,5	0,89872
400-352288	0,1	0,5	0,4	0,89839
400-352288	0,5	0,1	0,4	0,49869
400-352288	0,4	0,1	0,5	0,59763
400-352288	0,4	0,5	0,1	0,59851
400-352288	0,5	0,4	0,1	0,49888
MAX	0,1	0,4	0,5	0,898719

In order to assess the performance of the system, the algorithm is executed for different weights and terminal properties. The values of the parameters for which the executions are performed and the significance of the results are as presented below:

When the parameters are selected to represent a typical PDA with a CPU of 400MHz and a screen resolution of 320x240 (full-screen mode) with weights having the default values, after optimization, the resulting encoding values of the system is turned out to be spatial resolution 321x241, frame rate 12 fps and bit-rate 182Kbits/s. A typical instant for this result is illustrated in Figure 23.

When the same system parameters and trade-off weights, except the screen resolution are used, with a screen size specification of 176x144 (normal mode) the output of the system is obtained as Spatial Resolution 176x144, Frame rate18 fps and bit-rate 129Kbits/s. A typical instant for this result is illustrated in Figure 24.

The first two results are highly satisfactory. It can be seen that, while the system chooses a relatively high bit rate of 182 Kbits/s when the spatial resolution is high, it lowers the bit rate and increase the frame rate when the spatial resolution is reduced to 176x144. The reason for this behavior is to have an acceptable crispness at resolution 176x144, a smaller bit rate will suffice and the resources of the system can be used to increase motion smoothness.

However, when the spatial resolution is 320x240, higher bit-rates are necessary to have an acceptably crisp video. Since the system resources are allocated to the bit rate, the frame rate cannot be increased beyond a certain value.

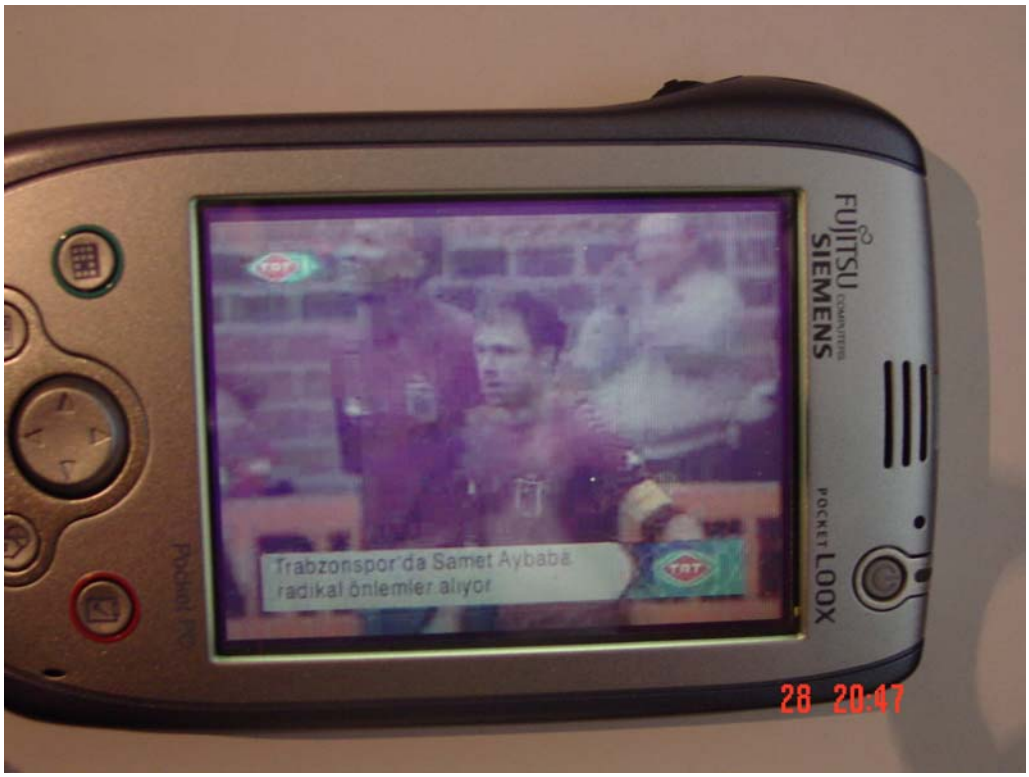

Figure 23 – Optimal encoding parameters for a device with CPU=400MHz and screen resolution 320x240: Spatial Resolution 321x241, Frame Rate 12 fps, Bit rate 182Kbits/s

On the other hand, the frame rate of 12 fps is still sufficient to code a video with satisfactory motion smoothness when the CPU value is entered as 240 MHz, with full screen mode and default weights, the output of the system is equal to spatial resolution 320x240, frame rate 16fps and bit-rate 135Kbits/s.

This result indicates that, when the CPU is lowered, the system reduces the recommended bit rate. This result is reasonable, since bit rate is the resource that requires the most CPU power.

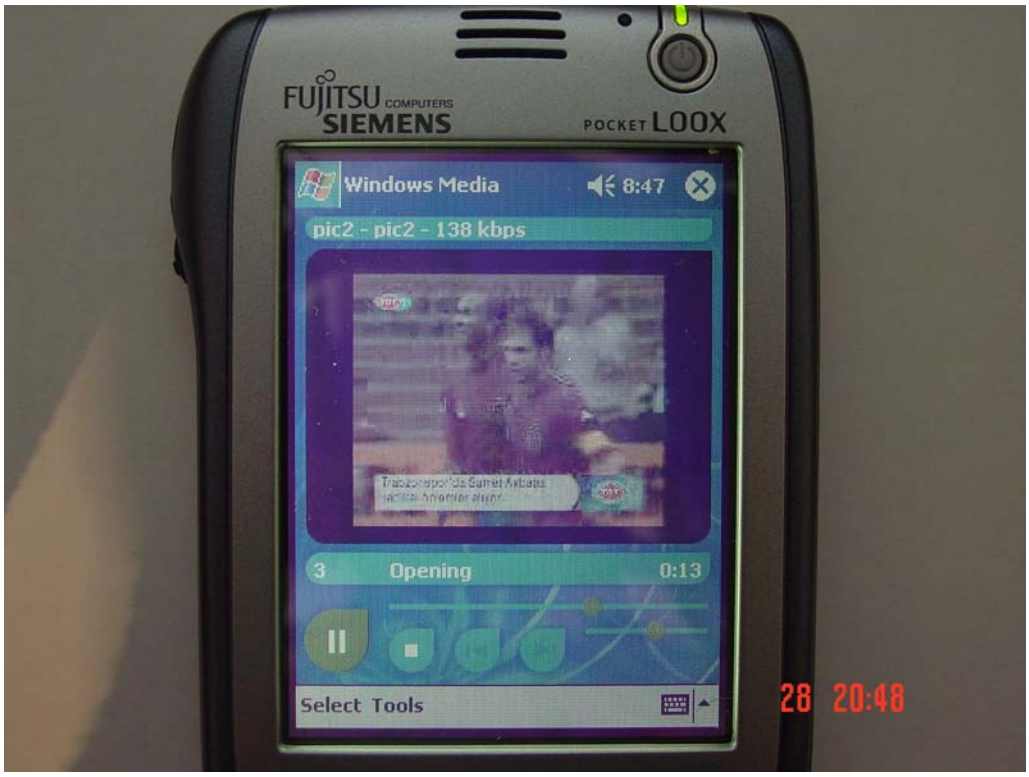

Figure 24 Optimal encoding parameters for a device with CPU=400MHz and screen resolution 176x144: Spatial Resolution 176x144, Frame Rate 18 fps and Bit rate 129Kbits/s

On the other hand, the frame rate of 12 fps is still sufficient to code a video with satisfactory motion smoothness when the CPU value is entered as 240 MHz, with full screen mode and default weights, the output of the system is equal to spatial resolution 320x240, frame rate 16fps and bit-rate 135Kbits/s.

This result indicates that, when the CPU is lowered, the system reduces the recommended bit rate. This result is reasonable, since bit rate is the resource that requires the most CPU power.

When a user wants to watch a very crisp video, the weights can be set as w1=1, w2=0,w3=0, with CPU selected as 400 MHz and the spatial resolution set to 320x240, the output of the system is spatial resolution 88x72, Frame Rate 1 fps and bit-rate 350Kbits/s. This result is illustrated in Figure 25

Figure 25 - User defined weights, as w1=1, w2=0,w3=0 : spatial resolution 88x72, frame rate 1 fps and bit-rate 350Kbits/s

The output has a very high bit rate and relatively small spatial resolution and frame rate values; this means the coded video will indeed have a very high crispness. When the above results are examined, it can be observed that the results

are consistent in a wide range of situations and the system can be used reliably for resource constrained devices.

CHAPTER 6.

Conclusions

The main contribution of this thesis is the decomposition of the satisfaction a user gets from watching a video into three conceptually independent components, as the satisfaction resulting from the crispness of a video, the satisfaction resulting from the motion smoothness of a video and the satisfaction resulting from the spatial resolution of a video. It has been observed that such decomposition enables, more accurate subjective evaluation of the user satisfaction. This in turn makes possible, precise modeling of the user satisfaction in terms of the video coding parameters. In summary, a novel methodology, for accurately modeling user satisfaction, using utility theory is proposed and implemented.

The system implemented in the thesis work successfully determines the representation of a video that will result in maximum user satisfaction, for a specific user terminal with given CPU and screen size. It also enables users to bias the system according to their personal preferences, by specifying trade-off weights for the components of the utility function. As demonstrated in the Section 5.6, the system produces valid and consistent results for the terminals and user preferences that the experiments were performed with.

There is still room for significant improvements in the system. While determining the optimal video attributes, the network parameters like the available bandwidth, end-to-end delay, and error probabilities should be taken into consideration. Also for future compatibility, the system should be implemented in a fully MPEG-21 compatible manner. Finally, the system would be more complete with the added capability of transcoding a bit-stream to the determined video bit rate, frame rate and spatial resolution.

REFERENCES

[1] Y. Neuvo , J. Yrjanainen, "Wireless Meets Multimedia – New Products and Services," IEEE ICIP 2002, pp. I1-I4

[2] W. Zeng , J. Wen, "3G Wireless Multimedia: Technologies and Practical Issues," IEEE ICIP 2002, pp. I5-I8

[3] R. Mohan, John R. Smith, Chung-Sheng Li, "Adapting Multimedia Internet Content for Universal Access ," IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 1, NO. 1, MARCH 1999

[4] M. R. Naphade, R. Mehrotra, A. M. Ferman, J. Warnick, T. S. Huang, A. M. Tekalp, "A High Performance Algorithm for Shot Boundary Detection Using Multiple Cues"

[5] H. J. Zhang, A. Kankanhalli, S. W. Smoliar, "Automatic Partitioning of Full Motion Video," Multimedia Systems (1993) 1:10-28

[6] I. Koprinska, S. Caratto, "Temporal Video Segmentation : A Survey"

[7] S. F. Chang, "Optimal Video Adaptation and Skimming Using a Utility-Based Framework," Tyrrhenian International Workshop on Digital Communications (IWDC-2002) Capri Island, Italy September 2002

[8] B. G. Haskell, "Digital Video: An Introduction to MPEG-2, " Kluwer Academic Publishing, 2000

[9] P. N. Tudor, O. H. Werner, "Real-Time Transcoding of Mpeg-2 Video Bit Streams," International Broadcasting Convention (IBC 97) Amsterdam 12-16 September, pp. 286-301. IEE Conference Publication

[10] A. Vetro, C. Christopoulos, H. Sun, "Video Transcoding Architectures and Techniques: An Overview," IEEE Signal Processing Magazine, March 2003 pp. 18-29

[11] G. Keesman, R. Hellinghuizen, F. Hoeksema, G. Heideman, “Transcoding of MPEG Bit streams,” Signal Processing: Image Communication 8 (1996) 481-500

[12] J. Bormans, J. Gelissen, A. Perkis, “MPEG-21: The 21st Century Multimedia Framework,” IEEE Signal Processing Magazine, March 2003, pp. 53-62

[13] MPEG-21 Requirements v 1.4, ISO/IEC JTC1/SC29/WG11 N5333, Awaji JAPAN, December 2002

[14] A. L. Golub, “Decision Analysis : An Integrated Approach,” John Wiley and Sons Inc, 1997, pp. 350-360

[15] R. O. Duda, P. E. Hart, D. G. Stork, “Pattern Classification,” Wiley Interscience, 2001

[16] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, “Optimization by Simulated Annealing,” Science 13 MAY 1983, Volume 220, Number 4598

[17] A. M. Tekalp, “Digital Video Processing,” Prentice Hall Signal Processing Series,1995

[18] F. Pereira, T. Ebrahimi(Editors) “The Mpeg-4 Book,” IMSC press, 2002

APPENDIX A

Source Code of the Simulated Annealing Algorithm

```
void CSimulatedAnnealingDlg::SimulatedAnnealing(optim_data *
data2optim,int CPU, int ScreenSize)
{

        const int initial_temperature =1000;
        const int final_temperature   =10;
        const int max_perturbations =10000;
        const double temperature_schedule= 0.95;

        double temperature=initial_temperature;

        optim_data system_state;
        optim_data perturbation;

        long no_of_perturbations=0;

        double system_energy;
        double perturbed_energy;

        double rand_element=0.0;

        //randomly select the initial system state

    system_state.BitRate=(50+300*((double)rand()/RAND_MAX));

    system_state.FrameRate=1+50*((double)rand()/RAND_MAX);

    system_state.SpatialRes=6336+200000*((double)rand()/RAND_MA
X);

        ofstream f("optimization_data.txt");

        /*
```

```
                    system_state.BitRate=350;
                    system_state.FrameRate=41;
                    system_state.SpatialRes=6348;


        system_energy=Calculate_Energy_Function(system_state,CPU,Scr
eenSize);

                */


                while((temperature>final_temperature))
                {


                    no_of_perturbations=0;

                    while(no_of_perturbations<max_perturbations)
                    {
                        //calculate the current energy of the system

        system_energy=Calculate_Energy_Function(system_state,CPU,Scr
eenSize);

                        rand_element=3*((double)rand()/RAND_MAX);

                        if(rand_element<1)
                        {

        perturbation.BitRate=(50+300*((double)rand()/RAND_MAX));

        perturbation.FrameRate=system_state.FrameRate;

        perturbation.SpatialRes=system_state.SpatialRes;

                        }

                        else if(rand_element<2)
                        {

        perturbation.BitRate=system_state.BitRate;

        perturbation.FrameRate=1+50*((double)rand()/RAND_MAX);

        perturbation.SpatialRes=system_state.SpatialRes;
```

```
                    }

                    else
                    {

        perturbation.BitRate=system_state.BitRate;

        perturbation.FrameRate=system_state.FrameRate;

        perturbation.SpatialRes=6336+200000*((double)rand()/RAND_MAX
);


                    }




                    //now calculate the energy of the perturbed
system

        perturbed_energy=Calculate_Energy_Function(perturbation,CPU,S
creenSize);

                    if(perturbed_energy<system_energy)  //if  the
energy is lower accept this as the system state
                    {

        system_state.BitRate=perturbation.BitRate;

        system_state.FrameRate=perturbation.FrameRate;

        system_state.SpatialRes=perturbation.SpatialRes;
                        system_energy=perturbed_energy;
                    }

                    else//if the energy is higher
                    {

                        if(          exp(-1    *
        double(perturbed_energy-system_energy)/temperature
        )>rand()          )//accept the perturbation
                        {
```

```
	system_state.BitRate=perturbation.BitRate;

	system_state.FrameRate=perturbation.FrameRate;

	system_state.SpatialRes=perturbation.SpatialRes;

	system_energy=perturbed_energy;
                                        }
                                }

                                f<<temperature<<"\t";
                                f<<no_of_perturbations<<"\t";

	f<<system_state.BitRate<<"\t"<<system_state.FrameRate<<"\t"<<s
ystem_state.SpatialRes<<"\t";

                                f<<system_energy<<endl;
                                no_of_perturbations++;
                        }

                        //cout

                        temperature=temperature*temperature_schedule;


                }




}


double   CSimulatedAnnealingDlg::Calculate_Energy_Function(optim_data
input_vector,int CPU,int ScreenSize)
{



	double result;



	if(CPU>=CPU_threshold )  //high CPU case
	{
```

```
                if (            (input_vector.FrameRate<=
FRh(input_vector.BitRate))&& (input_vector.SpatialRes<=ScreenSize)
                                )

        result=CPUHso2Lso3L(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);
                else if (               (input_vector.FrameRate>
FRh(input_vector.BitRate)) && (input_vector.SpatialRes<=ScreenSize)
                                )

        result=CPUHso2Hso3L(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);
                else if (               (input_vector.FrameRate>
FRh(input_vector.BitRate))&& (input_vector.SpatialRes>ScreenSize)
                                )

        result=CPUHso2Hso3H(input_vector.BitRate,input_vector.SpatialR
es,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);

                else if (               (input_vector.FrameRate<=
FRh(input_vector.BitRate))&& (input_vector.SpatialRes>ScreenSize)
                                )

        result=CPUHso2Lso3H(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);


        }

        else                                    //low CPU case
        {
                        if (            (input_vector.FrameRate<=
FRl(input_vector.BitRate))&& (input_vector.SpatialRes<=ScreenSize)
                                )

        result=CPULso2Lso3L(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);
                else if (               (input_vector.FrameRate>
FRl(input_vector.BitRate)) && (input_vector.SpatialRes<=ScreenSize)
                                )

        result=CPULso2Hso3L(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);
                else if (               (input_vector.FrameRate>
FRl(input_vector.BitRate))&& (input_vector.SpatialRes>ScreenSize)
                                )
```

```
        result=CPULso2Hso3H(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);

                else if (               (input_vector.FrameRate<=
FRI(input_vector.BitRate))&& (input_vector.SpatialRes>ScreenSize)
                                )

        result=CPULso2Lso3H(input_vector.BitRate,input_vector.SpatialRe
s,input_vector.FrameRate,ScreenSize,m_w1,m_w2,m_w3);


        }

        return result;

}
```

Appendix B

Technical Specifications of the Terminal Used for Subjective Tests

Pocket LOOX 600
The modular handheld

Data Sheet Pocket LOOX 600 Issue May 2003

System architecture
Applications processor Intel® PXA250 400 MHz
based on Intel® XScale™
microarchitecture
System memory 32 MB ROM, 64 MB RAM

Interfaces
1 x built-in microphone,
1 x speaker
1 x headphone (3,5mm)
1 x IrDA (115 KB)
USB 1.1 via sync cable
Serial via sync cable
1 x power jack

Display
240 x 320 color reflective TFT TouchScreen,
65.536 colors, light sensor with autodim
Active area: 53.6(H) x 71.5(V), 0.22 mm pixel pitch

Expansion slots
SD/MM card
Compact Flash slot type II

Bluetooth module
Power class 1 (high range)
Profiles*: serial port, file transfer, object push, Personal
Area Networking, Dial Up, LAN access, Fax, headset
gateway

Active synch uses Personal Area Networking
Point to Multipoint
Bluetooth hardware on / off switch

Battery
Included in the device, backup portion included in the battery
Lithium Polymer, rechargeable;
Capacity: 1520 mAh

Power supply
Rated voltage 100 to 240 AC (50 to 60 Hz)
to 5 V DC 2,4 Amp
(switched)
Power cord with EU cable, UK cable additionally available,
standard in UK versions

Operating time
Standby up to 100 h
Working up to 10 h (depending on usage)

Dimensions / weight
Dimensions (W x D x H) 132 x 82 x 17 mm
Weight 175 g

Import and usage of wireless technologies
According to country-specific regulations

Operating system
Microsoft Pocket PC 2002

Applications
Fujitsu Siemens Computers SpeedMenu
Plugfree for Pocket PC (Bluetooth software)
Pocket Outlook, Pocket Word, Pocket Excel
Calculator
Terminal Services Client
MSN Messenger
Pocket Internet Explorer, Microsoft Reader 2.0
Windows MediaTM Player

Transcriber
Voice recorder
Microsoft Active Sync 3.5
SyncSoftware support (MS Outlook 98, 2000 and 2002)

Additional software
KSE Truefax – Tool for sending and receiving fax documents
Web Information Solutions, Inc. Pocket Informant - Personal and Business Information Manager
F-Secure FileCrypto for Pocket PC Personal Edition – Security Software for encryption of folders and storage cards
Nyditot Virtual Display – Software for switching the display to landscape modus
Westtek ClearVue Office – Document, Presentation Worksheet File Viewers

Standard delivery package
Handheld device
USB docking station
Protection bag
USB sync cable
Additional stylus
AC adapter with cable (country specific)
Companion CD
Pocket LOOX AddOn & Doc (CD)
Easy Guide (printed manual)
Getting Started (printed manual)
Safety instructions and warranty booklet

Options / accessories
WLAN CF card,
IBM microdrive
SD/MM cards
GPRS CF card (as of January 2003)
Second battery plug-on module (Li-Ion)
GPS navigator kit
VGA CF card
Foldable keyboard
Leather cases
Leather combo case

USB docking station
Stylus / CF slot cover pack
AC adapter
Car adapter
USB sync cable
Serial sync cable
Additional accessories such as GPS card, LAN card and much more will be available on a project bases