Università degli studi di Padova

Anno accademico 2012-2013 (791°)

Dipartimento di Ingegneria dell'Informazione

Tesi di laurea triennale in Ingegneria dell'Informazione

A Study of Path Selection Algorithms for Mobile Video Streaming QoE

Relatore: Leonardo Badia

Correlatore: Marco Zanforlin

Laureando: Federico Chiariotti

Engineers like to solve problems. When there are no problems, they create their own.

Scott Adams (*Dilbert*)

To Benoît, who shares part of the guilt for unleashing another engineer on the unsuspecting public

To Art, who couldn't escape the long-winded nighttime technical conversations even by emigrating

To Elena and Irene, who had to live with me for three years and didn't run away even once

Contents

Abstract

This thesis discusses the problem of path selection for video streaming over 4G mobile networks; its final goal is to devise path selection strategies and test them with a pre-existing network simulator. The objective of path selection algorithms is to optimize both the use of network resources and the Quality of Experience of end users, quantified by various objective metrics.

We will propose several path selection algorithms for LTE video transmission, testing them with a simulation in both idealized conditions with perfect knowledge of network parameters and a realistic situation in which the algorithms must use delayed or imperfect information. We will demonstrate that the use of path selection algorithms can improve the Quality of Experience even when taking into account the negative effect of handovers (the switches from one server to another).

1 Introduction

In the last few years, mobile networks have dramatically improved their data rate and pervasiveness; along with the rise of smartphones, tablets and other mobile multimedia devices, this phenomenon is radically changing the mobile networking landscape.

In 2012, Internet videos took up 54% of total consumer traffic, with a predicted yearly growth rate of 34% until 2016; the yearly growth rate of mobile video streaming is an impressive 90% [1]. This extremely fast increase in demand presents service providers with new and unexpected challenges: as mobile networks were not designed with video in mind, the optimization work to provide high-quality video streaming without a steep increase in traffic is an ongoing concern.

The effort to establish improved architectures and protocols for video transmission over mobile networks is ongoing on both the academic and the industrial sides. To this end, several research projects have been tackling the related challenges at different layers of the protocol stack. For example, MultiMEDia transport for mobIlE Video AppLications (MEDIEVAL) [2] is a European project that aims at creating a complete cross-layer framework over multiple wireless technologies, with the collaboration of various universities and telecommunication industries from all over Europe.

MEDIEVAL is mostly concerned with Long Term Evolution (LTE) [3], a standard based on the Orthogonal Frequency Modulation Division Access (OFDMA) medium access scheme [4]. The LTE network exclusively uses Internet Protocol (IP) packets in order to provide full compatibility with existing technologies; its large-scale deployment began in North America in 2010. It was born as a direct evolution of the Universal Mobile Telecommunication System (UMTS) [5], a mobile cellular system developed by the Third Generation Partnership Project (3GPP). LTE is often classified as a "3.9G" system, as it does not meet the International Telecommunications Union (ITU) requirements to be a 4G standard (though its later evolution LTE-Advanced (LTE-A) does [6]).

The recent literature dealing with multimedia content often employs an-

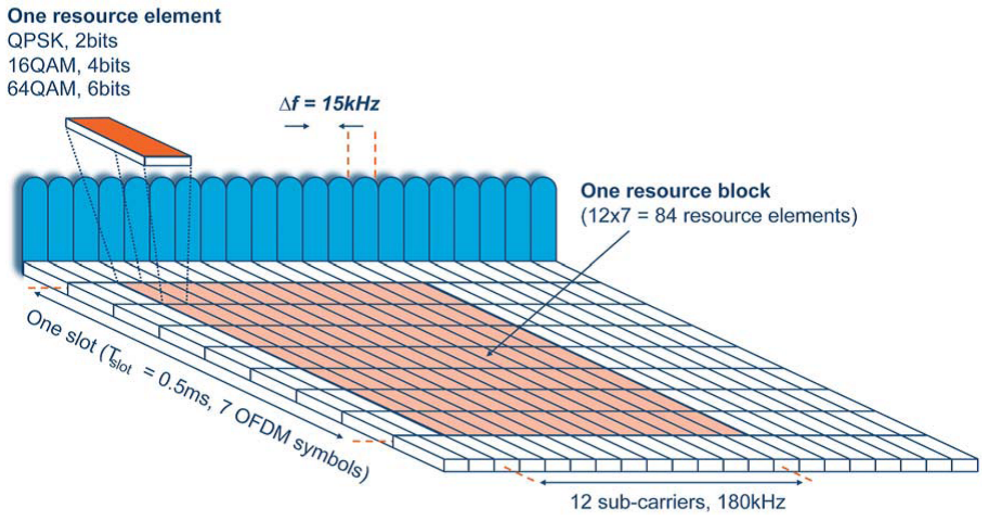

Figure 1: OFDMA resource block scheme

other important concept: Quality of Experience (QoE). QoE is a measure of a user's satisfaction when enjoying a video, image or audio service of any kind; multimedia service quality is often measured with various objective QoE metrics. 4G mobile network operators have to trade-off between reducing overall network resource usage and mantaining a target QoE; as videos sent over different paths through the network may result in different QoE values, one of the ways service providers can solve the problem is through an efficient path selection algorithm [7]. The algorithm needs a cross-layer approach, as QoE is only meaningful at the application layer and network optimization has to take place at the network and transport layers. The importance of path selection increases when mobile users can stream videos from either wireless or cellular local area networks, and from different video caches in the core network.

The MEDIEVAL PathSelection algorithm [8] is an algorithm that allows an LTE mobile user to stream videos with constant quality in a rapidly changing network situation. It uses the Application Layer Transport Optimization (ALTO) framework [9], currently in the development phase, to measure network costs; as both user movements and traffic load of the local transmitting cell may significantly change the available bitrate, the algorithm tries to optimize both user experience and overall network load by choosing

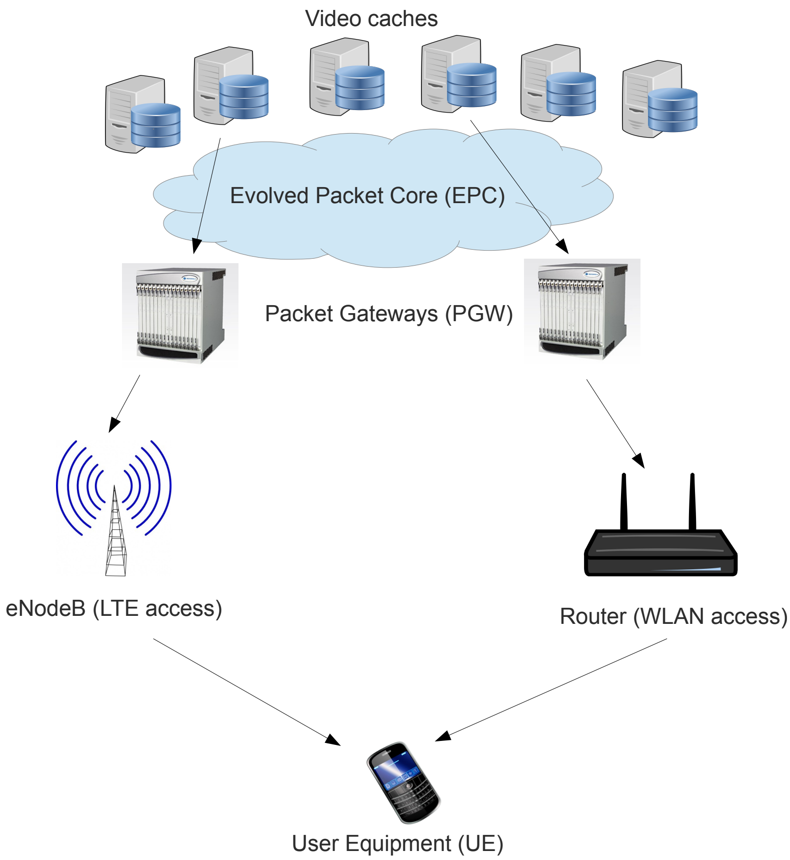

Figure 2: Structure of a hybrid LTE-WLAN mobile network

the best source and path for video download from both the user's and the service provider's perspectives.

The usual approach to deal with this kind of problems is a study through detailed simulation of the system. In this work, we used Network Simulator 3 (ns-3) [10], an open-source discrete event network simulator released under a GNU General Public License, version 2 (GPLv2) [11], to simulate an LTE network. In the simulation, we considered a user application with a primitive version of the PathSelection algorithm downloading a video and choosing between different video caches in rapidly shifting network conditions. Then, we calculated the QoE with a video evaluation tool [12] called QoE-Monitor, which also allowed us to use real videos in the simulation by dividing them in IP-compliant packets and rebuilding the received video.

The rest of this thesis is organized as follows: **chapter 2** consists of a review of the current state of the art of the field, providing an outlook on the LTE architecture and the MEDIEVAL project, as well as other relevant references. In **chapter 3**, we will present the original contribution of the thesis; the development of the path selection algorithms, along with their ns-3 implementation, and the chosen simulation scenarios are described in detail. The results of the simulation will be discussed in **chapter 4**, while in **chapter 5** we will make our final remarks and discuss possible future developments of this work.

2 State of the art

2.1 Overview of 4G network architecture

LTE [3] is one of the newest standards for mobile networks: it was designed by the 3GPP as an evolution of UMTS [5] and Enhanced Datarates for the Global System for Mobile communication Evolution (GSM/EDGE) [13]. LTE-Advanced [6], an enhanced version of LTE, was approved as a 4G system in 2011. LTE is entirely packet-switched, and traditionally circuit-switched traffic such as voice or Instant Messaging (IM) has to be processed with the IP Multimedia Subsystem (IMS) or transmitted through traditional 2G/3G. It uses an adaptive modulation scheme, transmitting with a Quadrature Amplitude Modulation with 4, 16 or 64 symbols depending on wireless channel conditions.

The radio access network connects User Equipments (UEs) such as mobile phones to the base transmitting stations, which are in turn connected to the core network. The LTE radio access network is called Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) [14]; it uses an OFDMA access method [4] for the downlink from the base station to the user, with peak data rates of 300Mb/s (1Gb/s in LTE-Advanced). For uplink connections, E-UTRAN uses a simpler Single-Carrier FDMA [15], which requires less power from the battery-powered mobile devices. LTE-A can also improve communication performance by using the Multiple Input Multiple Output (MIMO) system, if the terminals have the required multiple antenna setup.

Its infrastructure units, equivalent to older networks' base stations, are called eNodeBs (eNBs); eNBs are the interfaces between the users and the Evolved Packet Core (EPC), the main LTE network, both for user traffic and control plane data. eNBs are scalable for deployment both in densely populated urban areas and large, sparsely settled rural areas; while standard eNB transmitting radii are usually tens of kilometers, the network architecture supports both larger cells (macrocells) that can transmit at a distance of over 100 km and smaller cells (nanocells, picocells, femtocells) with smaller power requirements and transmitting radii. Femtocells, or Home

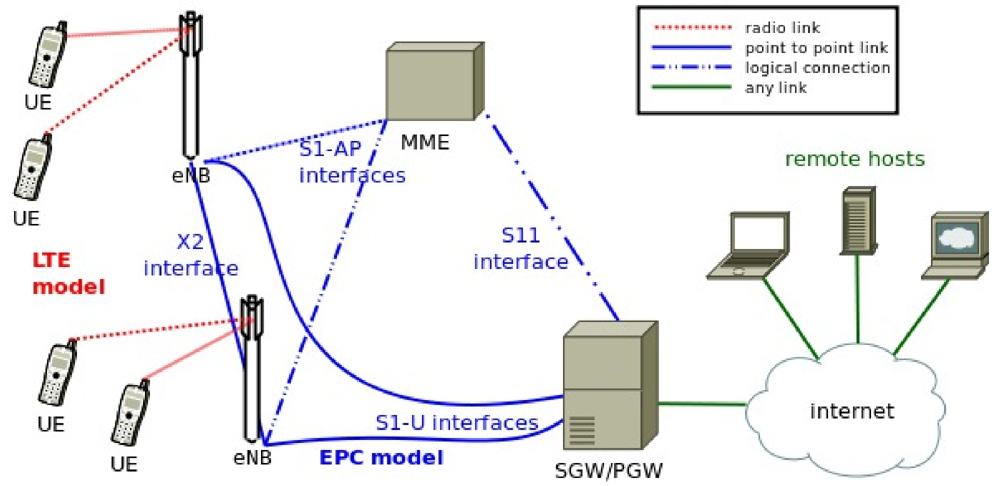

Figure 3: LTE architecture model ([16])

eNBs, are the smallest possible cells, with a transmitting radius of under 100 m. The smaller cells use higher frequency bands and support high speed mobile broadband; a major difference between eNodeBs and NodeBs [17], their UMTS ancestors, is that control functionality is embedded in the eNodeBs, requiring no radio network controllers and simplifying network architecture.

The three lower layers of the Open Systems Interconnection (OSI) protocol stack are handled by the E-UTRAN [18]; their most interesting feature is the division between the user plane and the control plane, which are handled separately by the network layer. While user traffic uses the Internet Protocol (IP) exclusively, control plane data are handled by the Non-Access Stratum (NAS), an LTE-specific set of protocols which handle signaling and all aspects of the radio connection, as well as user mobility [19]. The NAS provides a continous link by establishing a communication session and verifying the user connection.

On the EPC side, the control nodes that manage UE mobility and tracking are the Mobility Management Entities (MMEs). MMEs are the endpoints of NAS signaling, as well as controlling intra-LTE handovers [20] and interactions with 2G/3G networks. The data MMEs receive and process is stored in the Home Subscriber Server (HSS), a central database that is also necessary for IMS to function.

On the transport layer, the Serving Gateways (SGWs) connect the EPC to the Internet, routing and forwarding all packets to and from the UEs; they also serve as mobility anchors during handovers between eNBs. The Packet Data Network Gateways (PGWs) provide connectivity with external packet data networks: they allocate IP addresses to the UEs and filter packets through deep packet inspection. PGWs are the anchors for mobility between LTE and non-3GPP technologies such as Worldwide Interoperability for Microwave Access (WiMAX) [21].

2.2 The MEDIEVAL project

The MEDIEVAL project [2] is structured as a series of algorithms and protocols to increase video transmission efficiency that can be deployed independently on different networks. The MEDIEVAL architecture is divided in four subsystems: Wireless Access, Transport Optimization, Mobility and Video Service [22].

The Video Service subsystem uses a cross-layer approach to provide a link between the application layer and the transport layer; video applications can thus interface directly with the transport optimization module, adapting video attributes such as frame rate and quantization to maximize QoE as well as coordinating packet scheduling and Forward Error Correction (FEC) channel coding in the vulnerable User Datagram Protocol (UDP) packets.

The Wireless Access subsystem uses an abstraction layer between the data link layer and the network layer to optimize video delivery through WLAN or LTE radio access networks, both in unicast and multicast mode. The abstraction layer also provides an Abstract QoS Mapper that allows higher layers to translate their QoE requirements into lower-level parameters.

The Mobility subsystem is based on the Distributed Mobility Management concept, an approach entirely different from existing mobile IP standards. It controls handovers and IP address continuity, as well as managing session and bearer setup and handover candidate selection.

The Transport Optimization [23] subsystem is aimed at optimizing video traffic in the core network, reducing network load without affecting users'

QoE. It creates a mobile Content Delivery Network (CDN) [7] by setting up video caches and providing optimal node selection. The CDN nodes are constantly updated, and videos are uploaded, deleted and copied basing on video popularity and network conditions.

The four MEDIEVAL subsystems interact to provide a complete mobile video architecture; in this thesis, we will focus on the PathSelection [24] algorithm, a part of the Transport Optimization subsystem. The PathSelection algorithm, when installed on the mobile terminal, solves the optimization problem of maximizing a normalized function representing the quality of the video path. The solution to the problem accounts for both the network operator and the user. While the optimum solution for the former maximizes the sum of the proximity values of the steps in the path (*max-sum* criterion), the solution for the latter maximizes the minimum proximity value in order to avoid bottlenecks and guarantee a minimum performance (*max-min* criterion). The Core Network (CN) and wireless Access Network (wAN) metrics are calculated separately; in the proposed implementation, the two CN metrics are the distance to the End Point (EP) and the EP storage occupation, communicated by the ALTO servers, while the only wAN metric is Signal to Noise Ratio (SNR). While the choice of SNR may be considered imprecise when assessing network quality, it can be measured directly by the mobile terminal without complicated signaling. The algorithm's computational complexity is relatively low, as it grows linearly both with the number of network metrics it considers when choosing the path and the number of possible paths.

2.3 Related work

Video QoE in LTE networks has become an object of intensive study since the large-scale deployment of the standard started, leading to the creation of various methods to improve performance. These methods often involve cross-layer algorithms that use QoE, a concept that is evaluated on the application layer, to configure resource allocation on lower layers of the protocol stack.

The use of Scalable Video Coding (SVC) [25] to provide a minimum level of QoE in difficult network conditions can be included in the downlink architecture. SVC is also the concept behind an algorithm [26] proposed by Vergados *et al.* that aims at reducing the sudden and noticeable drops in video quality due to packet loss by providing a constant, lower quality.

QoE-aware resource optimization in LTE networks is another cross-layer optimization method that uses application layer data to efficiently configure data link layer scheduling and resource allocation: with reference to the MEDIEVAL architecture, this optimization is performed by the Transport Optimization subsystem. Shehada *et al.* propose a similar resource allocation scheme [27]; they use a greedy search algorithm to maximize overall QoE, in order to achieve a target mean value.

Singh *et al.* propose another cross-layer resource management scheme based on the rebuffering concept [28]. Rebuffering is defined in the article as the state of streaming in which the video is stalled while the empty playback buffer is being filled. The article poses a trade-off problem between traditional QoE and rebuffering time, as high quality videos use more resources and suffer more from rebuffering; the use of adaptive streaming techniques is proposed as a possible solution.

While the study of path selection problems dates back to the first days of the Internet [29], albeit with different aims and technological constraints, Content Delivery Networks (CDNs) have only been deployed in multimedia services since the early 2000s. Simulation is a valuable tool when studying CDNs, as a rigorous mathematical approach may require simplifications and additional assumptions to reduce the complexity of the networks.

The benefits and design problems of CDNs and alternate path selection in video transmission, as well as the possible use of overlay networks (virtual networks built over a physical substrate), are described by Venkataraman and Chatterjee in [30]: the framework they propose selects the paths that result in the highest QoE in a generic IP network, trying to transmit the most important frames without errors.

Ma *et al.* propose a path selection algorithm [31] for video transmission in cooperative overlay networks; the algorithm takes place entirely on the application layer. The algorithm works with a single video cache, as the overlay nodes are only intermediate nodes in the paths; its complexity depends on the optimization level required, as a perfect solution to the multi-path multi-constraint problem has been proven to be NP-hard.

Jain and Dovrolis compare various path selection techniques for video streaming in [32]; their results show that path selection algorithms based on an estimate of available bandwitdh fare much better than algorithms based on jitter or packet loss in terms of video QoE. They also show that their path selection algorithm outperforms standard FEC methods, as it can avoid network congestion while FEC schemes suffer from the congestion-based concentrated error bursts.

A study by Apostolopoulos and Trott [33] explores the benefits and implementation details of path diversity in video streaming: using SVC, the different layers of a video stream can be sent over more than one path, and from more than one cache in the CDN, improving user QoE by reducing latency and packet loss. The usefulness of path diversity also lies in the possibility of creating a video transmission protocol without the need for feedback. In [34], Guo *et al.* describe a CDN system that fully exploits path diversity. Although path diversity algorithms have been studied extensively, their application in mobile networks such as LTE has never been investigated in the literature.

The MEDIEVAL path selection framework is explained by Munaretto *et al.* in [24]; its structure is based on the CDN concept, adapted to the LTE network. The path selection algorithm and its supporting framework have been described in detail in **section 2.1**.

2.4 Support material

Quality of Experience (QoE) is a measure of a user's experiences with a video, image or audio service of any kind. A number of QoE metrics have been developed, but the most common are Mean Opinion Score (MOS) [35], Peak Signal to Noise Ratio (PSNR) [36] and Structural SIMilarity (SSIM) [37]. While MOS is a no-reference metric, requiring only the received video, both PSNR and SSIM are full-reference metrics, i.e., they require a complete knowledge of the original video to compare it to the received one. While full-reference metrics need fewer assumptions about measuring video quality, no-reference metrics are a subject of ongoing research as they allow online distributed optimization of video streaming networks [38].

MOS [35] is the most direct QoE metric, as it is generated by averaging the rating a group of people gives in a series of standard tests. Its subjective nature makes it inherently impossible to implement algorithmically, but it is often used as a reference to rate the performance of objective metrics.

PSNR [36], usually expressed in decibels, is the signal to noise ratio between the highest pixel value (MAXi) and the noise power, represented by the mean squared error (MSE).

$$\mathrm{PSNR_{dB}} = 20\log_{10}(\mathrm{MAXi}) - 10\log_{10}(\mathrm{MSE}) \tag{1}$$

PSNR is almost universally used as an objective video quality metric, due to its consistency and low complexity, but its simplicity can also be a limit. In some cases, correlation between PSNR values and perceived quality (as measured with MOS) is dubious due to non-linear elements of the human visual system [39]. Another issue is the low impact of PSNR of localized errors: while they have little effect on MSE, if errors concentrate in a small area or a small number of frames the human perception of the video may be deeply altered; due to this effect, PSNR is not a reliable metric when comparing different contents and codecs [40].

SSIM [37] is an objective QoE metric based on perceived change of structural information in an image or video; it is more complex than PSNR but does not fall prey to some of its pitfalls. The local nature of the calculation means that SSIM avoids PSNR's underestimation of local errors and is a better fit for the human visual system. In the following formula for SSIM, μ_j stands for the average of j, σ_j stands for the variance of j and σ_{ij} stands for the covariance of i and j. The two constants c_1 and c_2 depend on the dynamic range of the pixel values and are needed to stabilize the division. SSIM is calculated locally; x and y are usually 8×8 pixel sized windows.

SSIM values have a dynamic range that goes from -1 to 1; an SSIM value of 1 is reached only if the two images or videos are exactly identical.

$$\text{SSIM} = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2}$$

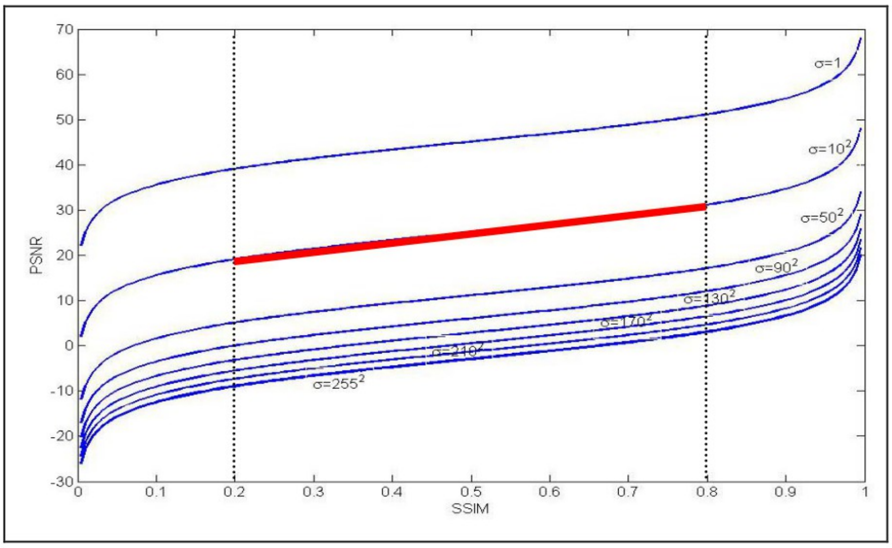

Figure 4: Graph of PSNR and SSIM values ([41])

The correlation of PSNR and SSIM to actual QoE as measured with MOS is an object of debate: studies with different conditions and models have arrived to conflicting conclusions as to their validity. Horé and Diou argue instead [41] that there is a simple analytical link between the two metrics, and the values of one can be predicted from those of the other, as seen in Figure 4.

ns-3 is a network simulator based on the discrete-event paradigm [42]; in a discrete-event simulation, the state of the system only changes in response to events that happen at a discrete point in time. It was written in C++ with a modular structure that encourages customization and support for multiple types of real-world networks and protocols. Its node structure is based on the Linux networking architecture, allowing ns-3 to interact with real network interfaces and simulate entire Linux machines. The simulation engine is triggered by C++ or Python scripts; the simulator is controlled through a set of helper interfaces or its core functions. ns-3 is released under a GNU GPLv2 software license that makes it easy to redistribute and modify, as well as completely open-source. We used the LTE-EPC Network Simulator (LENA) LTE module in the simulation; the model structure is described in detail in [16]. It was developed as an independent simulator by the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC) and later merged with the official ns-3 releases.

One of the largest differences between a real LTE network and the LENA model is that the latter has no independent MME entities: as the code only supports data plane communications, the control role of the MME is taken over by the *EpcHelper* element.

QoE-Monitor [12] is an external module for the ns-3 network simulator that provides a framework for sending real videos over a simulated network and evaluating the QoE of the received video. It relies on the Ffmpeg [44] library to open and manipulate encoded video files. QoE is calculated with the PSNR and SSIM video quality metrics; the 0.1 version we used in the simulation only supports the H264 [43] video encoding format.

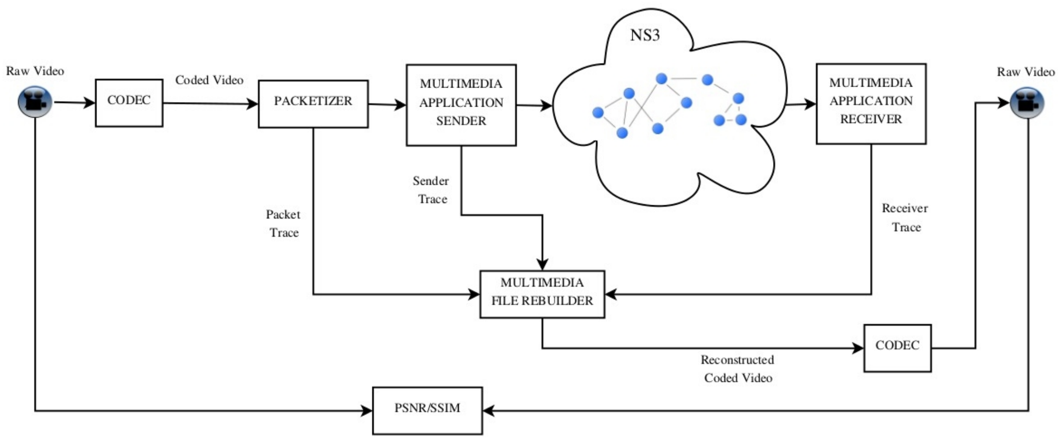

Figure 5: Structure of the QoE-Monitor module [12]

The module works by encoding the original H264 raw video and transmitting it over the simulated network. The *H264Packetizer* class then divides the video into sendable packets, while hiding the implementation details from the application through the *Packetizer* virtual class. The *MultimediaApplicationSender* and *MultimediaApplicationReceiver* classes handle the actual transmission; as they they are derived from the ns-3 *Application* class, they can transmit the video and trace data through any type of network. The video is then rebuilt by the *MultimediaFileRebuilder* class using the control information from the application classes. The *PsnrMetric* and *SsimMetric* classes can then calculate the QoE by decoding the received video with Ffmpeg and comparing it to the raw original. See Figure 5 for a visual depiction of this structure.

The implementation of the *MultimediaApplicationSender* and *MultimediaApplicationReceiver* classes was not flexible enough for the purposes of our simulation: the proposed scenario requires the possibility to switch between CDN nodes in response to shifts in network conditions such as eNB handovers, but QoE-Monitor only supports sending the video without interruptions from a single Sender application to a single Receiver. We modified the two classes extensively, creating a new *SmartApplicationSender* class that can send the video in short packet bursts instead of a single continuous stream and a *SmartApplicationReceiver* class with the ability to periodically choose the sender with the best network parameters from a list. These changes are described in detail in **section 3.2**.

3 Original work

3.1 Build environment

The simulation was set up with ns-3.14 on a Linux Mint Debian Edition (LMDE) laptop with the 3.2.0-4 amd64 Linux kernel. Ns-3 was compiled with gcc 4.7.2, and the QoE-Monitor 0.1 module ran with Ffmpeg 1.0.6. The results of the simulation should be reproducible on any recent GNU/Linux system with ns-3.14 or higher and Ffmpeg 1.0 or 1.2.

3.2 Coding and scripting

The main coding problem in the simulation was adapting the QoE-Monitor module to the proposed scenario. The native *MultimediaApplicationSender* and *MultimediaApplicationReceiver* only allowed sending the whole video from a single sender to a single receiver without interruptions; in order to simulate the path selection problem in shifting network conditions, we had to replace them with more versatile implementations. However, some inherent limitations in the implementation of the module had to be considered and avoided in the proposed scenarios, reducing the scope of the study.

The new *SmartApplicationSender* class we developed sends video packets in the same way as the original, but stops after a 1 second burst instead of sending the whole video; the other necessary change was the implementation of the possibility to discard a number of packets, as in the simulated scenario they may have been already sent by other senders. The tests we ran using a modified version of the QoE-Monitor example scripts and the *MultimediaApplicationReceiver* packet trace show that the new SendPackets method makes the handover between two sender applications with little or no packet loss. The *SmartApplicationReceiver* class uses the completely new *SenderList* class to choose the sender with the best network parameters from a list and schedule a packet burst from that sender; this way, a suboptimal sender choice affects the system for at most 1 second. The *SenderList* class contains a list of senders and the related network parameters; it can update any of the sender parameters, as well as choose the best sender in the list and pass it to the *SmartApplicationReceiver* along with the number of packets to discard. We consider the following algorithms, implemented as methods of the *SenderList* class:

The **Dumb** path selection algorithm simply picks one of the video caches at random and downloads the whole video from it, disregarding network conditions. We used it as a default comparison for the more advanced algorithms.

The **Delay** algorithm chooses the video cache with the lowest delay; for the algorithm to be effective, the received video QoE should be at least as high as the Dumb, with significantly lower delay values.

The **Error** algorithm chooses the video cache with the lowest error rate, disregarding delay values. The resulting video should have the best possible QoE in the given network conditions, but the algorithm has no control over the delay.

The **Smart** algorithm considers a linear combination of error rate and delay, with proper scaling of the two contributions; as the error rate is a probability and the ns-3 *Time* class uses nanoseconds, the delay is scaled down by two orders of magnitude; different network conditions would require a different scaling. Theoretically, its QoE should be slightly worse than with Error, but it should keep delay values significantly lower.

The final script is modeled on the LENA example scripts: it creates an eNB and a PGW, which it links to a number of remote hosts that run the *SmartApplicationSender* application through point-to-point channels. The UE running the *SmartApplicationReceiver* application is then created and connected to the eNB. The random variations in the network parameters described in **section 3.3** were implemented as uniform random variables through the standard library *rand* class.

3.3 Simulated scenarios

The simulations we ran involved the transmission of a video from a server to a mobile user through the LTE network. The reference videos we used are the “Highway” and “Bridge (close)” videos from the EvalVid reference video library [45]. The "Highway" video was also used in the testing of the QoE-Monitor module [12].

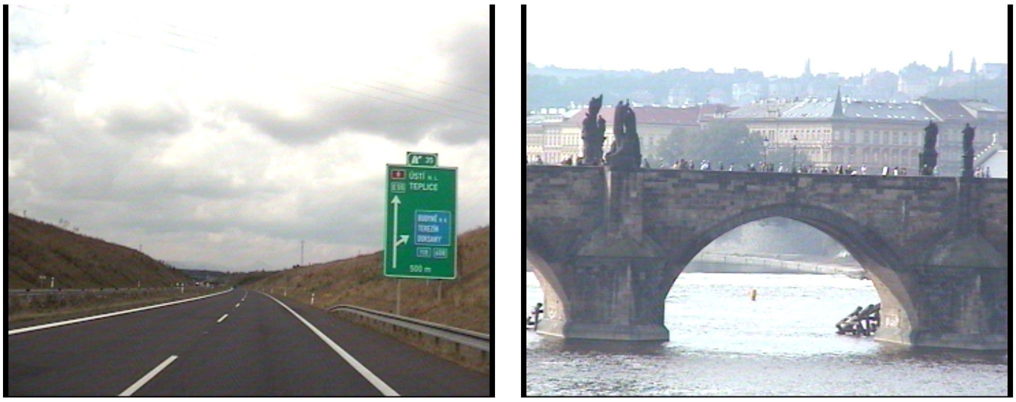

Figure 6: Stills from the reference videos: "Highway" on the left, "Bridge (close)" on the right

The scenarios we considered involved a single UE running the *SmartApplicationReceiver* application and connected to the internet through one eNB. Four CDN nodes are connected to the PGW, and the *SmartApplicationSender* application is installed on each. The connection between the video cache locations and the PGW is represented by a *PointToPointChannel* whose error rate and delay vary randomly over time. The topology of the simulated network is represented in Figure 7.

The initial delay varies between 0 and 50 milliseconds, and the initial error rate can vary from 10^{-5} to 10^{-3}. Error rate variations have a range of $\pm 2 \times 10^{-4}$, while delay can vary up to ± 40 ms; both the initial parameter values and the variations have a uniform distribution, mainly for simplicity of implementation. The variations occur at random times; the average time between variations for each link is 1 second. In the considered scenarios we consciously kept delay variation low in order to reduce the number of out of order packets, as the *MultimediaFileRebuilder* cannot handle them without discarding a whole frame. The total loss of out of order packets, while completely unrealistic for systems that decode the complete video at the end of the transmission, may be a sensible assumption for real-time streaming services.

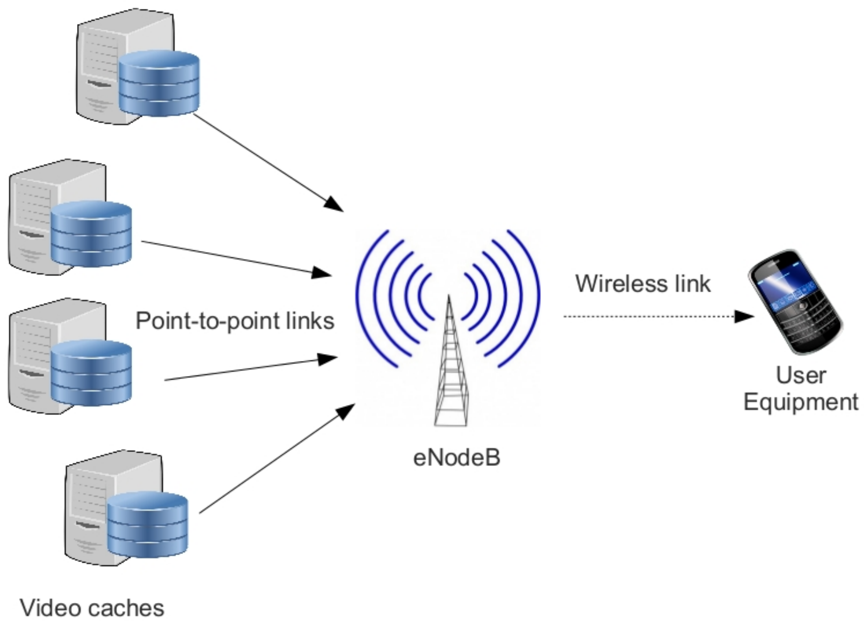

Figure 7: A simple representation of the topology of the simulated scenarios

In the perfect knowledge scenario the information about delay and error rate is perfect and instantaneous. When the *SmartApplicationReceiver* makes its choice, the parameters it considers are always the actual values present in the network at the time.

The delayed knowledge scenario introduces a 0.1 second delay in the recognition of parameter changes, so that the information the *SmartApplicationReceiver* uses is always outdated by 0.1 seconds. This scenario is more realistic than the first one, but it still makes idealistic assumptions about the measurement of network parameters. As the average time between variations is 1 second, a 0.1 second delay in parameter change reception should have a limited effect on the results.

The imperfect knowledge scenario has no delay in transmitting parameter changes to the *SmartApplicationReceiver*, but it generates a random error so that the path selection algorithm has to choose basing on partially incorrect information. In a real network, error rate can only be estimated from packet history and physical layer parameters; such estimates are not always correct, particularly in rapidly shifting network conditions such as the ones in the simulated scenarios. Noise is a uniformly distributed variable with a maximum of ±25% of the original value.

The delayed imperfect knowledge scenario is the most realistic one, with both a 0.1 second delay and a random error in the transmission of network parameters.

We ran 10 simulations for each scenario and algorithm, averaging the results to approximate the normal behavior of the system. The Dumb algorithm was run only once, as the differences between the four scenarios do not affect its performance. We measured the delay and error rate of the chosen sender when the algorithms made their choices, i.e., once every second.

After the simulations, we calculated the resulting PSNR and SSIM values with the QoE-Monitor *PsnrMetric* and *SsimMetric* classes. PSNR was calculated separately on the three video components of the raw YUV [46] video: the Y component represents the luminance of the frame, while the U and V components are used to decode color information (chrominance).

4 Results

4.1 General remarks

In the following sections, we present the simulation data of the "Highway" video; although we also ran the simulations with the "Bridge (close)" video, we only present their results when there are relevant differences with the "Highway" results; in all other cases, the conclusions that can be drawn from the two videos are the same.

All the data we used in the graphs is the average of 10 simulations with the same video and the same network parameters; the data should approximate the average behavior of the system in the given network conditions.

4.2 Perfect knowledge

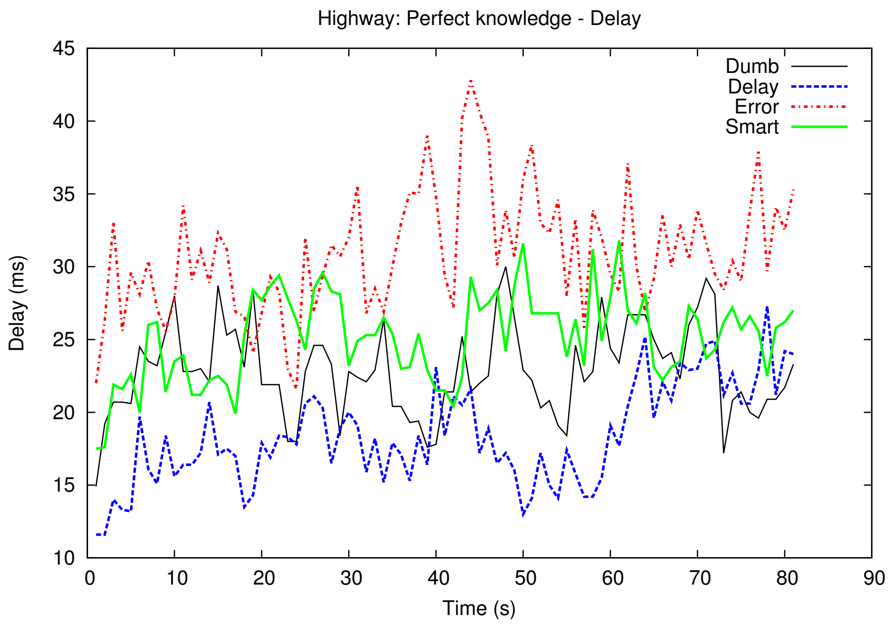

Figure 8

The physical parameters that result from the simulation seem to confirm the validity of the three algorithms. As Figure 8 shows, the Delay algorithm significantly reduces delay values without increasing error rates; its

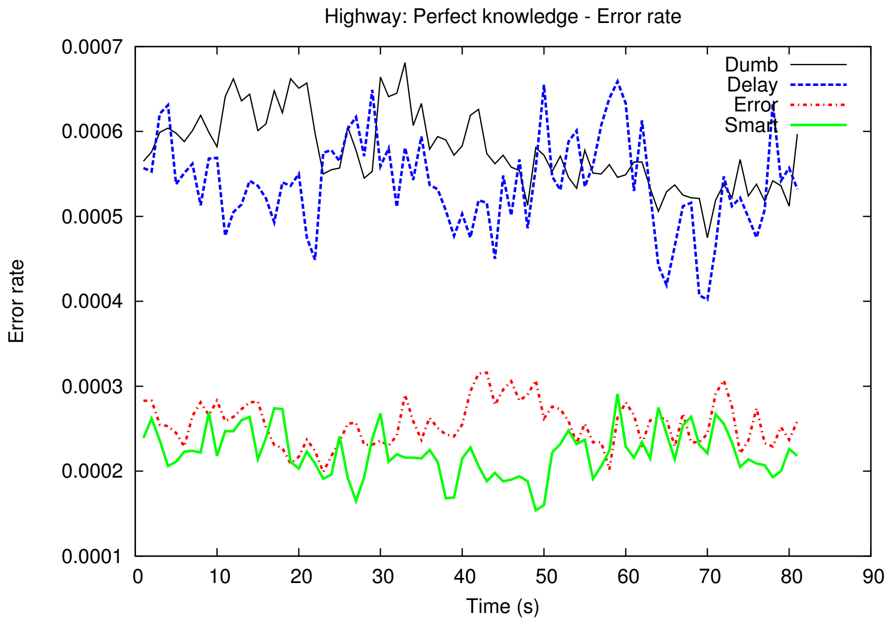

Figure 9

average delay is 20% less than the value of the reference Dumb algorithm, while its error rate is almost the same. The parameters of the Error and Smart algorithms show that an improved error rate comes at at the cost of higher delays. The Error algorithm's delay is higher by 36% than the Dumb algorithm's, while the Smart algorithm only has a 10% increase; however, as figure 9 show, the Error algorithm's error rate is significantly lower. On average, the Smart algorithm's error rate is half the error rate of the Dumb algorithm, while the Error algorithm gets as low as 40%.

Figures 10, 11 and 12 show that the Error and Smart algorithm have significantly higher PSNR values in all three components. The Error algorithm gains, on average, 16 dB over the reference Dumb algorithm on the Y component, 24.7 dB on the U component and 24.8 dB on the V component. The Smart component results in slightly lower quality, gaining 10 dB, 10.8 dB and 11.3 dB over the Dumb algorithm on the three video components. The Delay algorithm also results in slightly higher qulity than the Dumb algorithm, but the average gain is less than 10 dB on all the components.

The SSIM results for the four algorithms are slightly different: while the

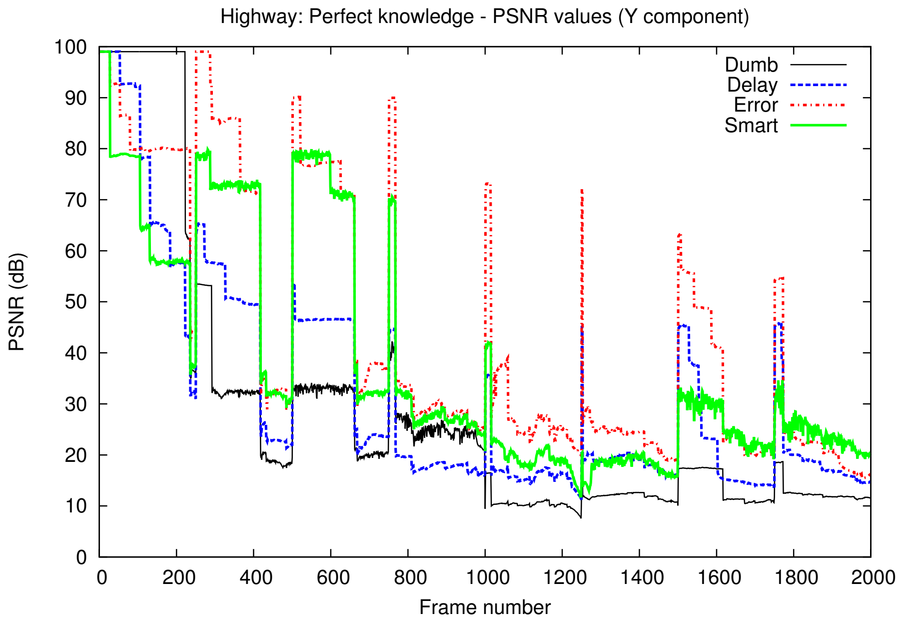

Figure 10

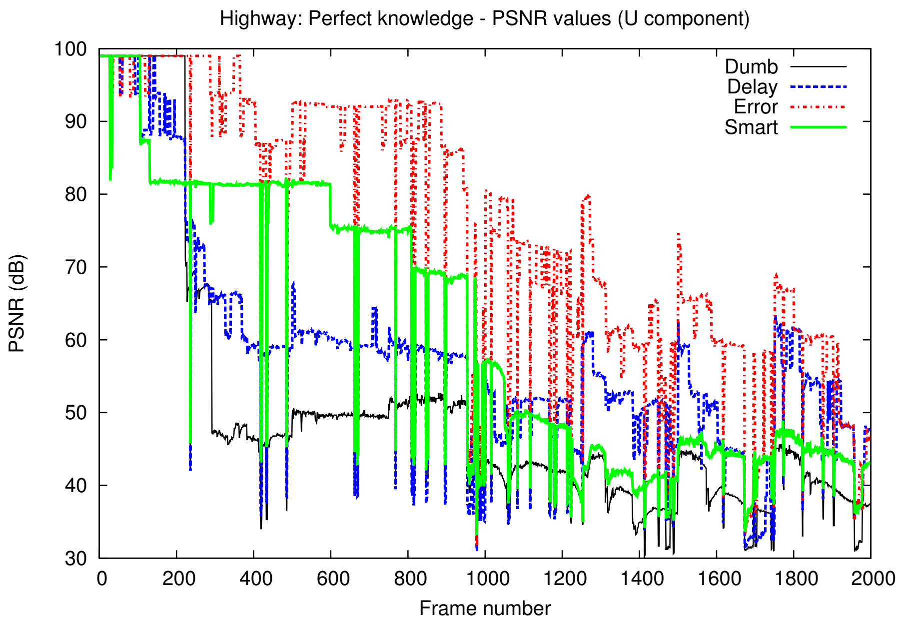

Figure 11

Delay algorithm QoE, with an average SSIM of 0.73, is very similar to the reference Dumb algorithm QoE, with an average of 0.71, the Error and Smart

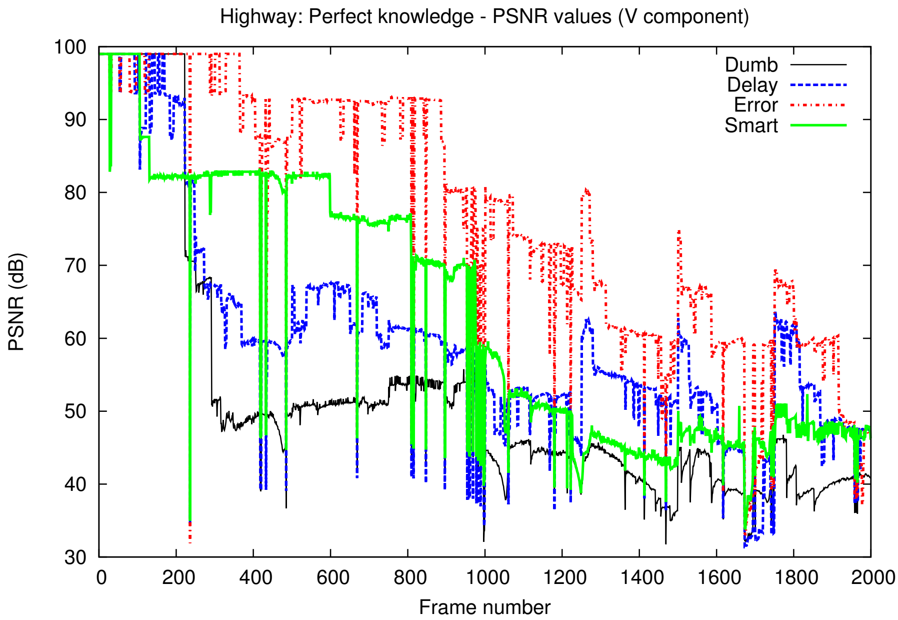

Figure 12

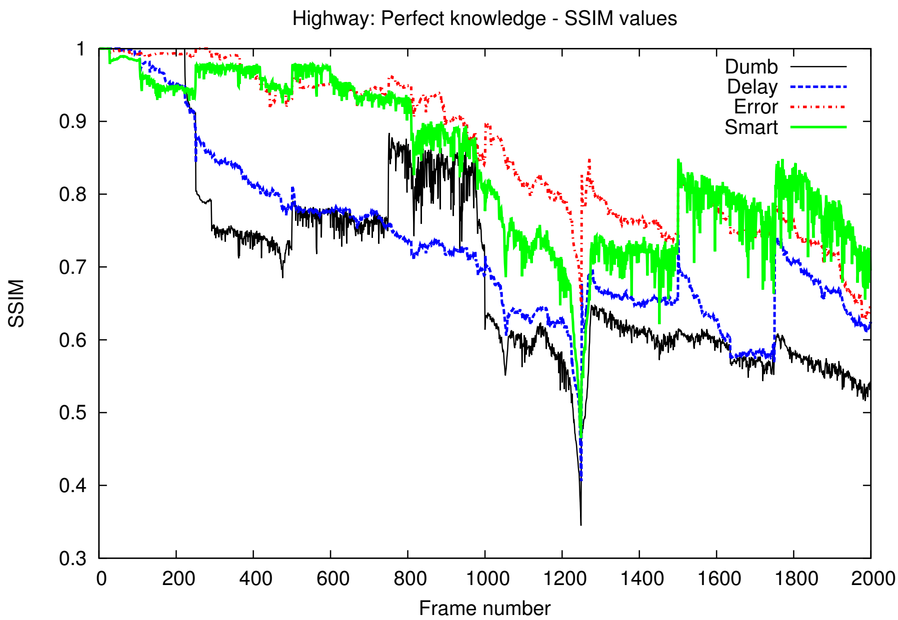

Figure 13

algorithms show a significant improvement in SSIM values, with averages of 0.86 and 0.84 respectively.

Both the chosen channel parameters and QoE measurements seem to confirm the validity of the path selection algorithms; the limits they show are also within the predicted limits; it also seems clear that improvements in error rate, and consequently in PSNR and SSIM, come at the cost of higher delay values.

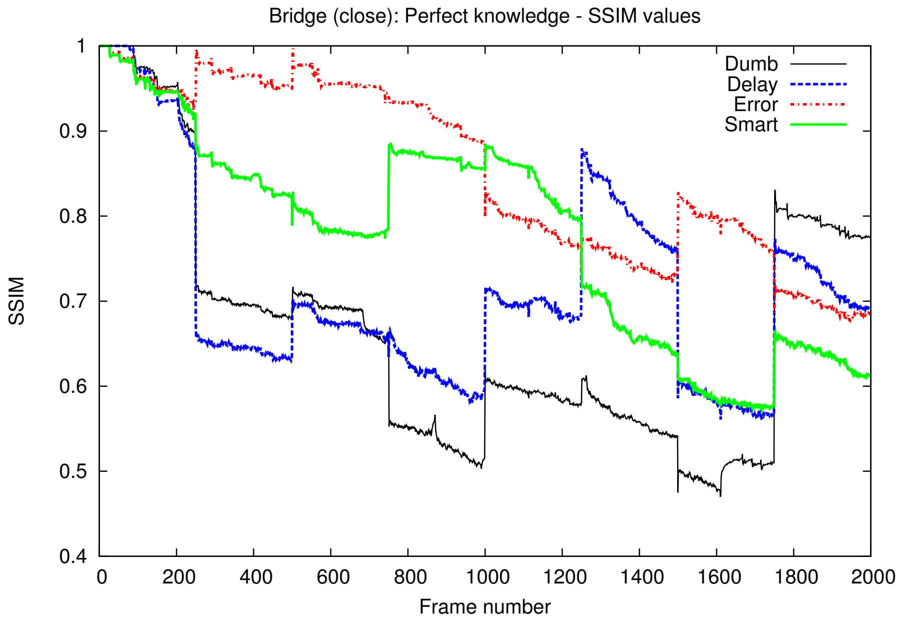

Figure 14

It can be easily noticed that, after about 1200 frames, the SSIM values decrease suddenly for all algorithms. This is due to the mp4 differential compression and to the structure of the video itself: about 45 seconds into the "Highway" video, the car passes under a highway overpass. The content of those frames is thus extremely dynamic, with significant variation between one frame and the next, and any error results in significant quality loss. As the frames with the most dynamic content are also the biggest when encoded with the H264 codec [43], the error is also localized; this is why the quality loss is less noticeable when using the PSNR metric, which underestimates localized errors. This argument is supported by the "Bridge (close)" SSIM values, as Figure 14 shows: the less dynamic nature of the "Bridge (close)" video causes the reference frames the codec uses to calculate differences to be extremely important, as the sudden quality changes show, but at no point in the video do the four algorithms experience the same difficulties.

4.3 Delayed knowledge

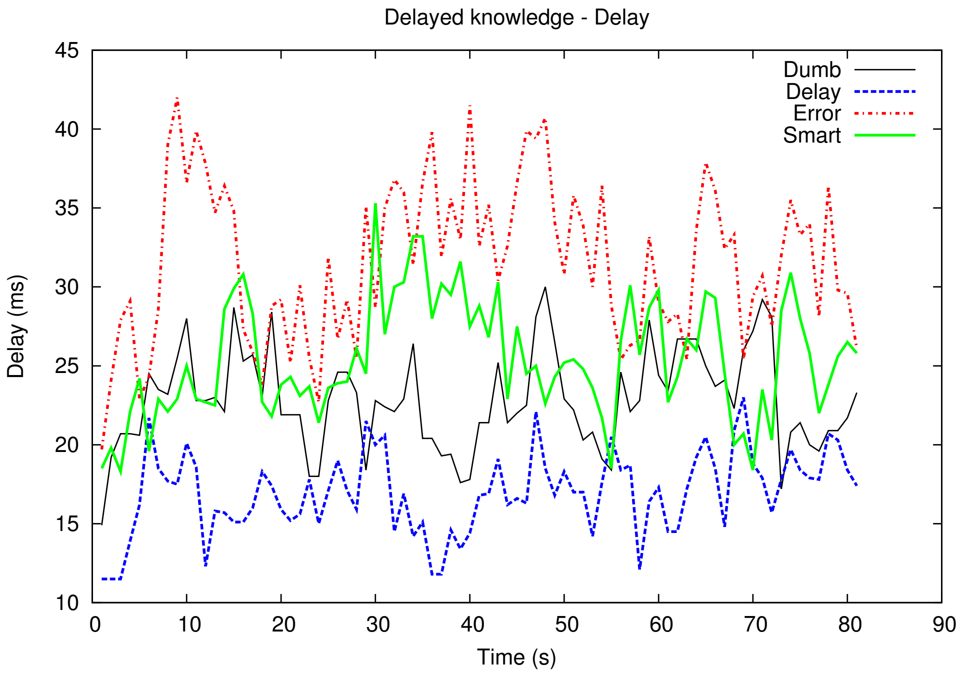

Figure 15

The channel parameters in the second scenario are very similar to the ones in the first one; as the average time between variations in network parameters is 1 second, a 0.1 second delay in their reception should not affect the resulting choices in a significant way. In the delayed knowledge scenario, the Delay algorithm's average delay is 25% lower than the Dumb algorithm's, while its error rate is 15% lower, as figure 15 confirms. The Error algorithm's error rate is 35% the Dumb algorithm's, but its average delay is almost 40% higher. The Smart algorithm still does not quite match the error rate of the Error algorithm, as its average error rate is 45% the Dumb algorithm's, but the increase in delay is only 10%.

As Figures 17, 18 and 19 show, the Delay algorithm still results in a

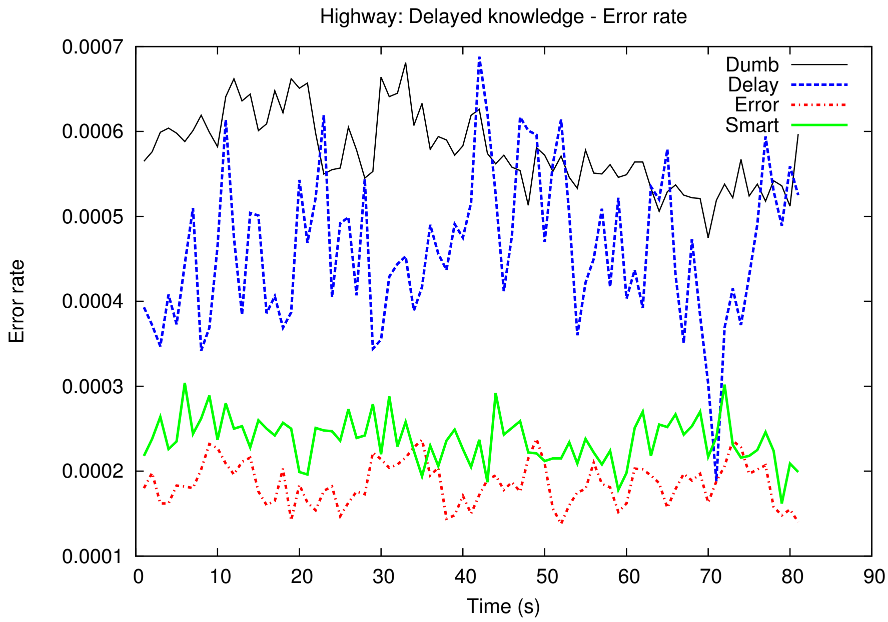

Figure 16

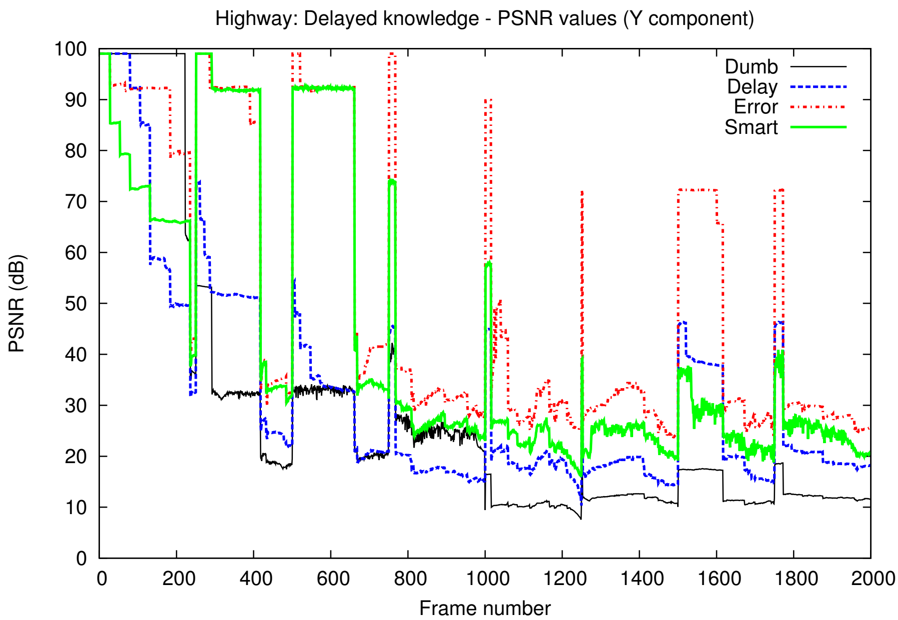

Figure 17

slightly better quality than the reference Dumb algorithm, but the difference between the two algorithms is still under 10 dB. The Smart algorithm gains

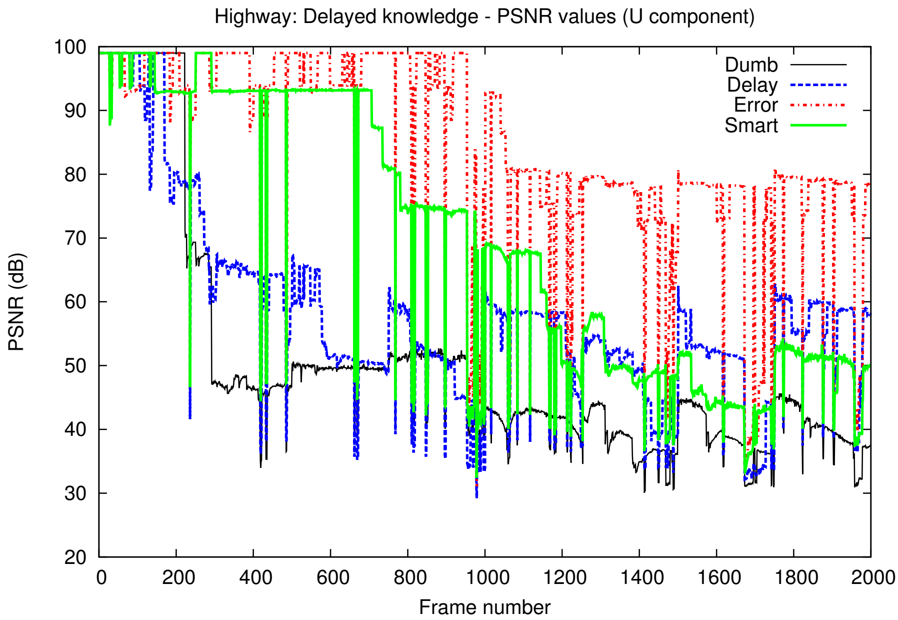

Figure 18

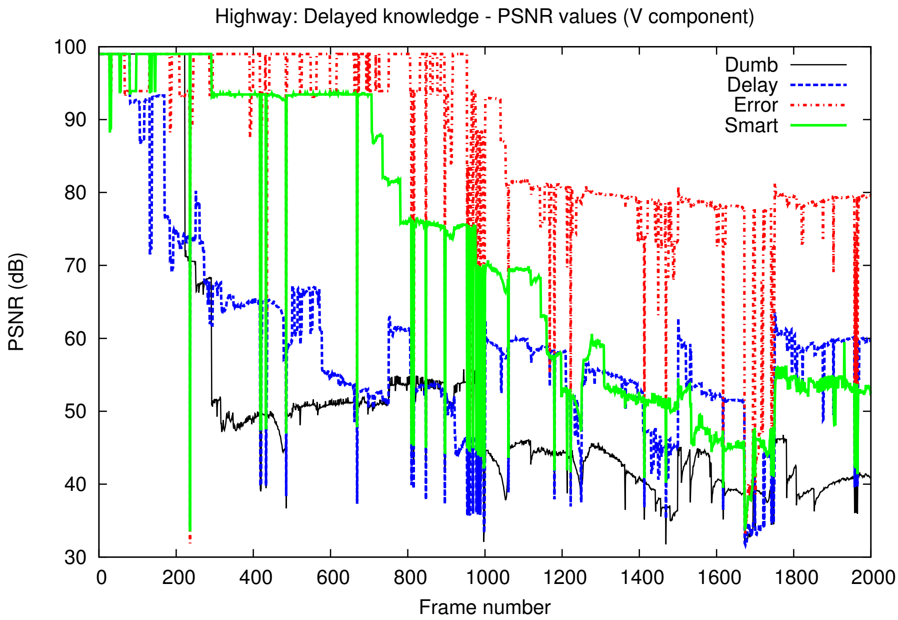

Figure 19

15.1 dB on the Y component, 18.7 dB on the U component and 19.5 dB on the V component over the Dumb algorithm, but the Error algorithm still

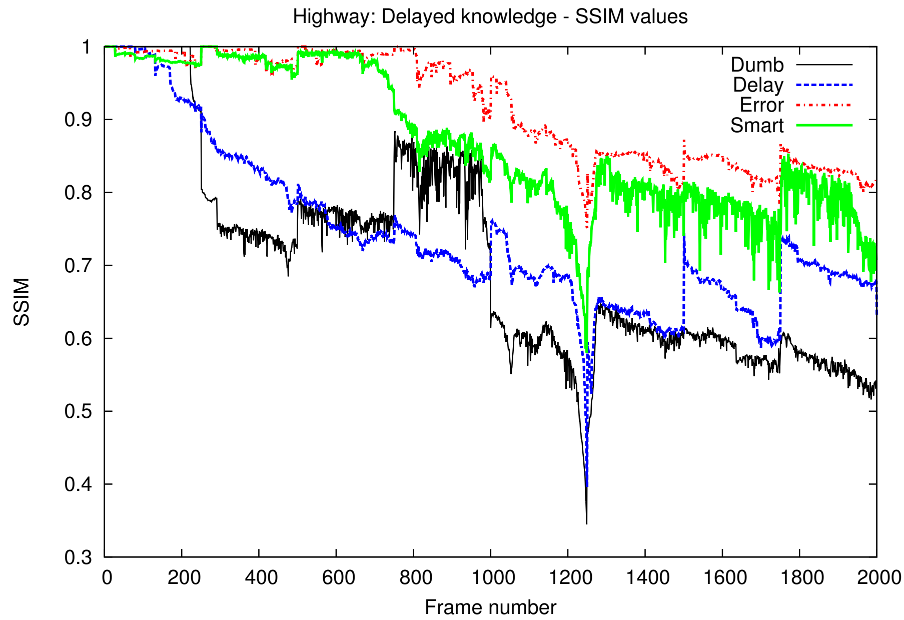

Figure 20

outperforms it by about 15 dB on every component.

In this scenario, the Delay algorithm results in an average SSIM of 0.74; the difference between the Error and Smart algorithms is more significant, with average values of 0.92 and 0.87 respectively. This difference is also noticeable in Figure 20, as in the second part of the video the Error algorithm results in significantly higher SSIM values.

4.4 Imperfect knowledge

In the third scenario we introduced a random noise in error rate and delay measurements; while the noise could change the perceived parameters by up to 25%, the physical parameter results do not degrade significantly. The Delay and Smart average delay values are lower (60% and 90% of the Dumb delay, respectively), and while Figure 21 shows that its delay is higher than in the previous scenarios, with a 60% increase over the Dumb delay, its error rate is, on average, less than 30% the reference value. The Smart algorithm's error rate is about 50% the Dumb algorithm's, while the Delay algorithm

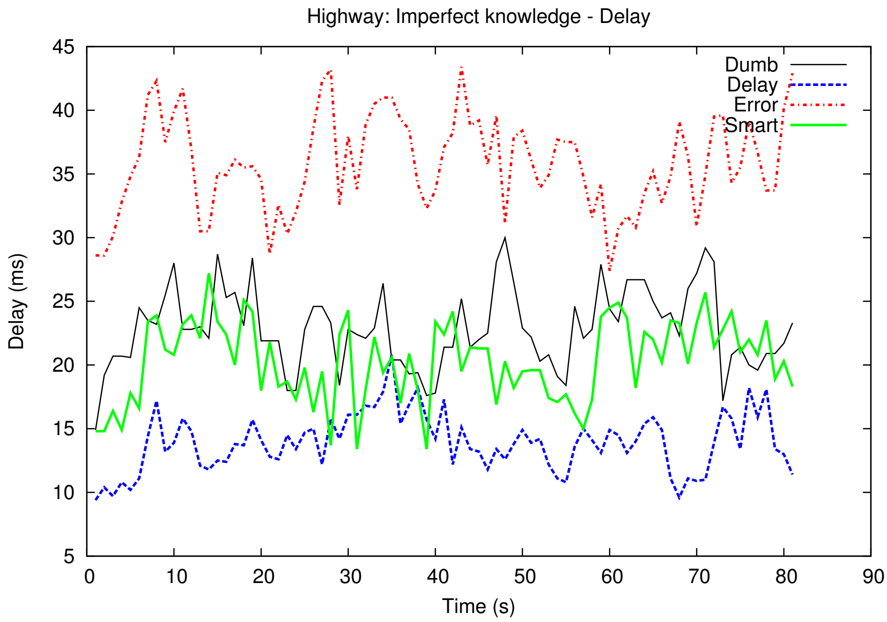

Figure 21

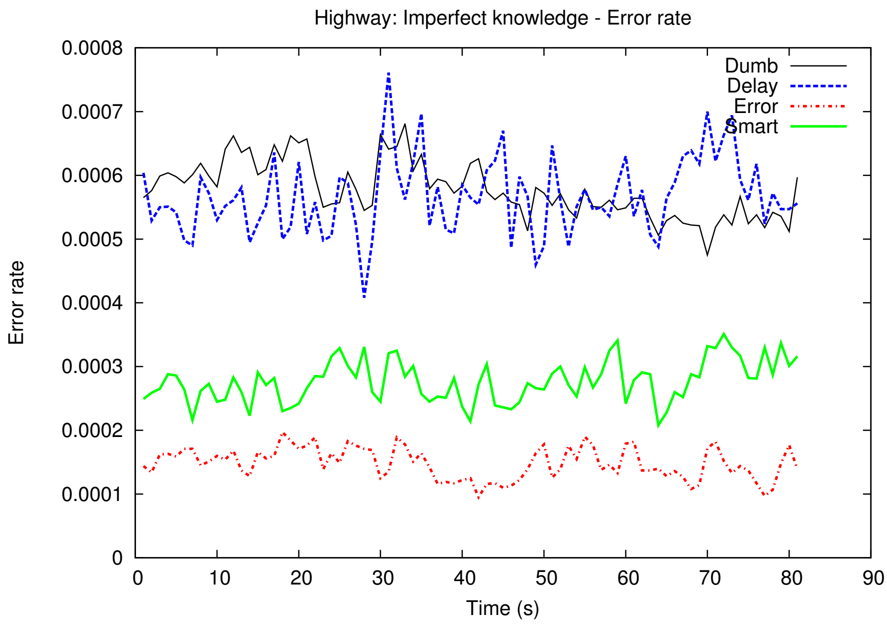

Figure 22

still has an error rate close to the reference value.

In the third scenario, the Error algorithm's PSNR values are higher than

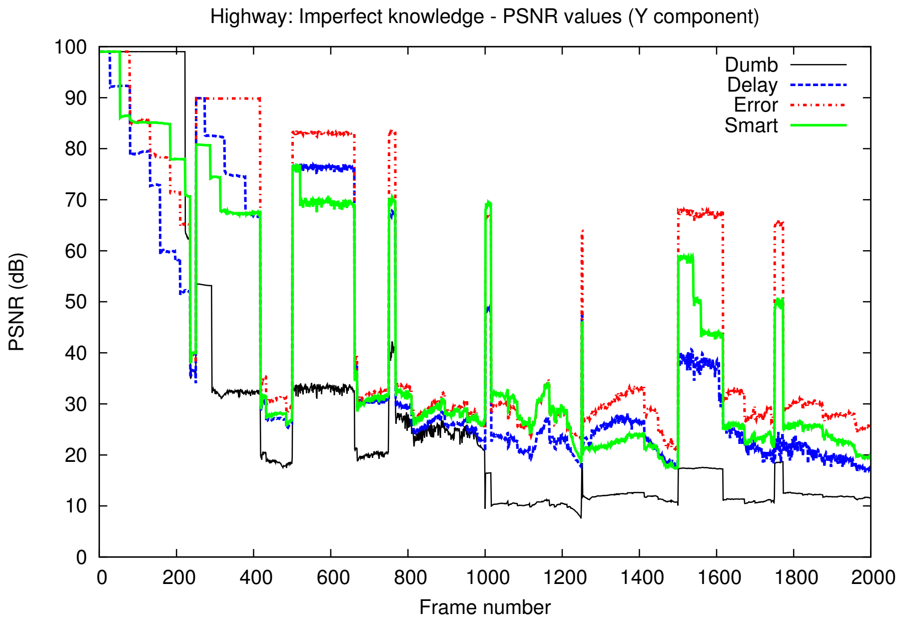

Figure 23

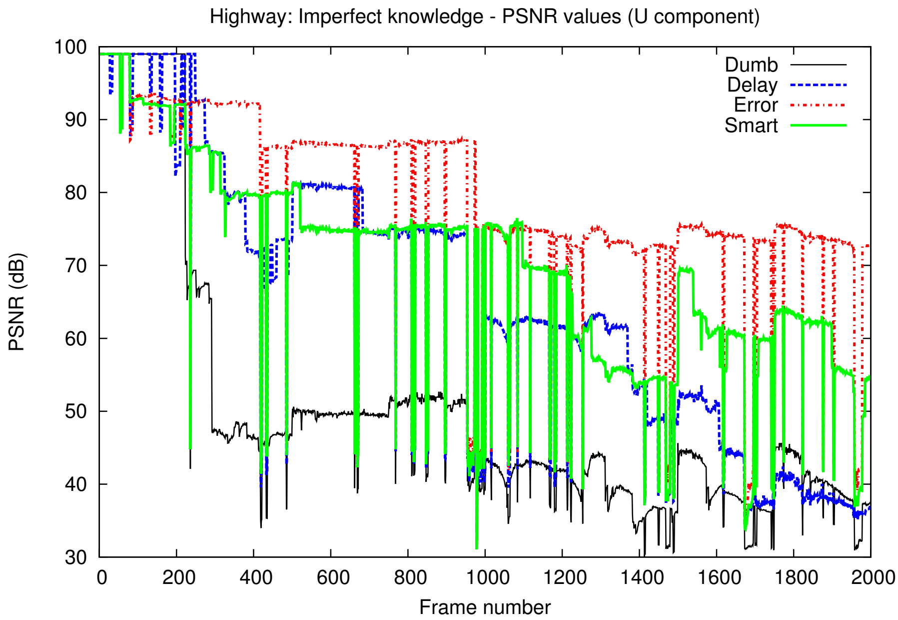

Figure 24

the Dumb algorithm's by 20 dB on the Y component and by almost 30 dB on the other two. The Smart algorithm results in a PSNR higher than the Dumb

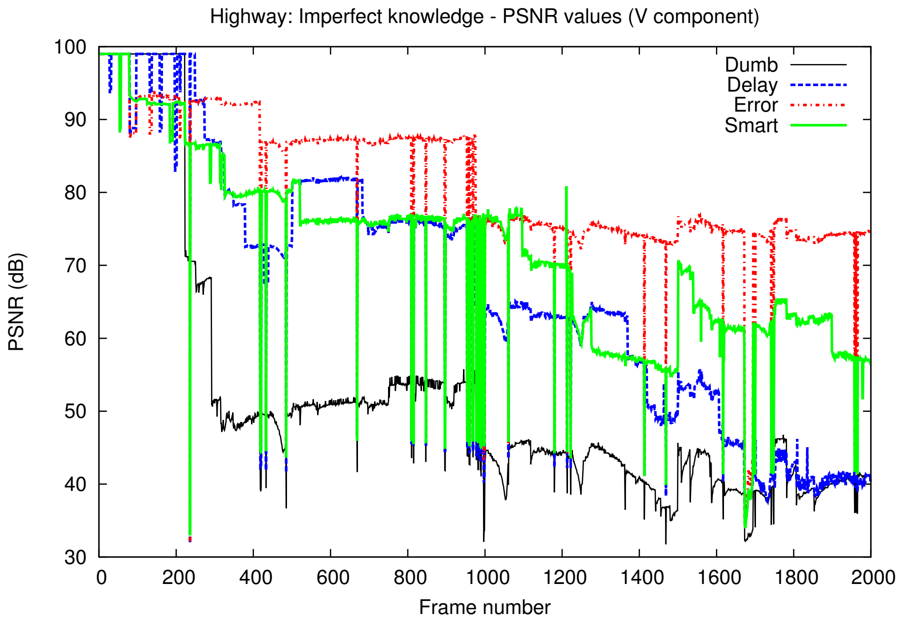

Figure 25

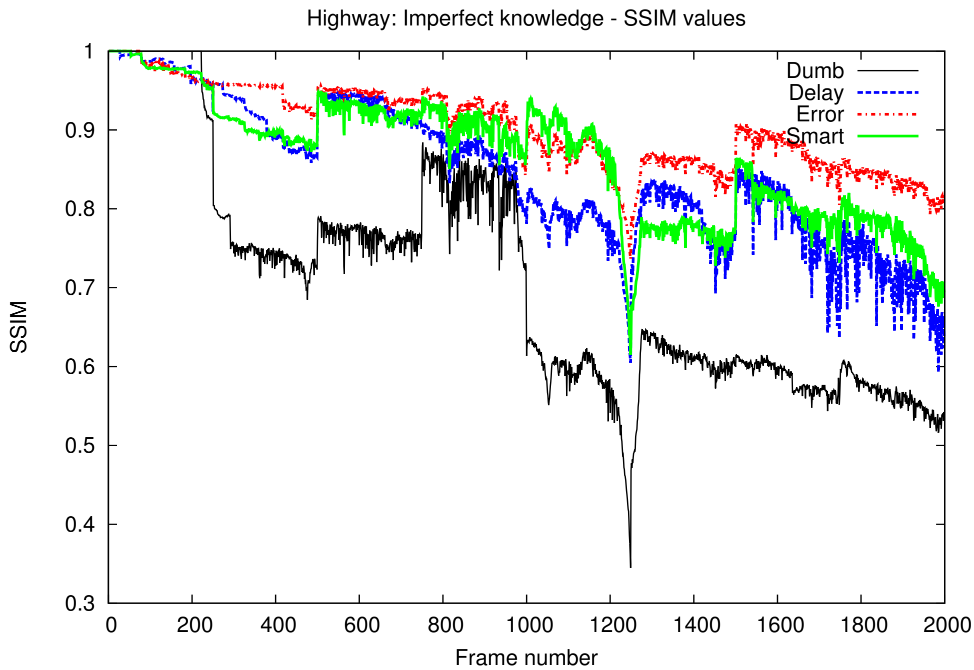

Figure 26

algorithm's by 14.4 dB on the Y component, 19.1 dB on the U component and 19.8 dB on the V component, and while the Delay algorithm has a similar performance on the Y component, its U and V component PSNR values are lower by about 5 dB.

SSIM values confirm the significant QoE difference between the Error and Smart algorithms, with average SSIM values of 0.90 and 0.86 respectively. This was to be expected, as the Smart algorithm's choices are based on two parameters and thus more sensible to noise. The Delay algorithm performed unexpectedly well, with an average SSIM of 0.85, as Figure 26 shows.

4.5 Delayed imperfect knowledge

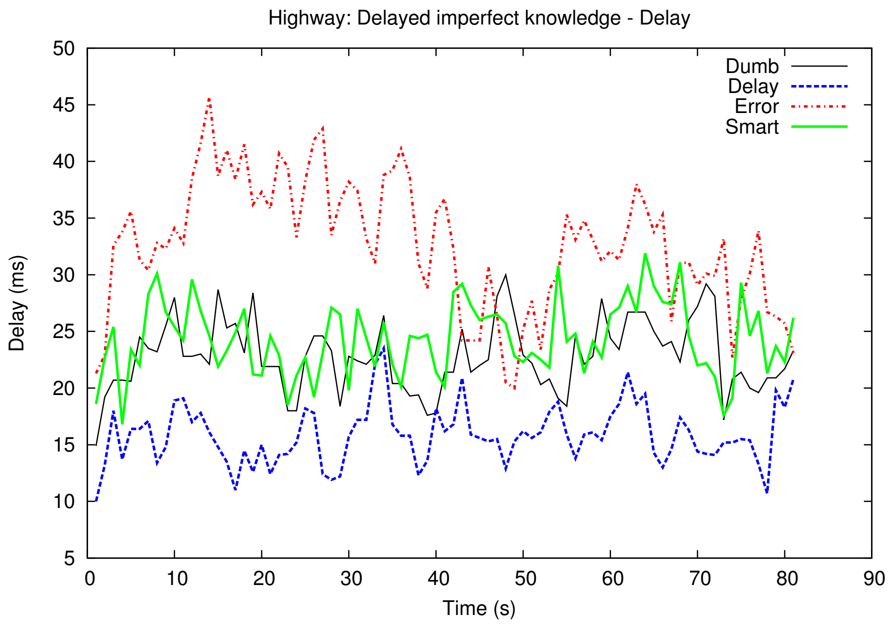

Figure 27

In the fourth and final scenario, the channel parameters still confirm the validity of the three path selection algorithms; the average delay of the Delay algorithm is 70% of the Dumb algorithm's, while the Error algorithm's is 140% and the Smart algorithm's is 105%. The Delay algorithm has an average error rate similar to the dumb algorithm's, while the Error and Smart

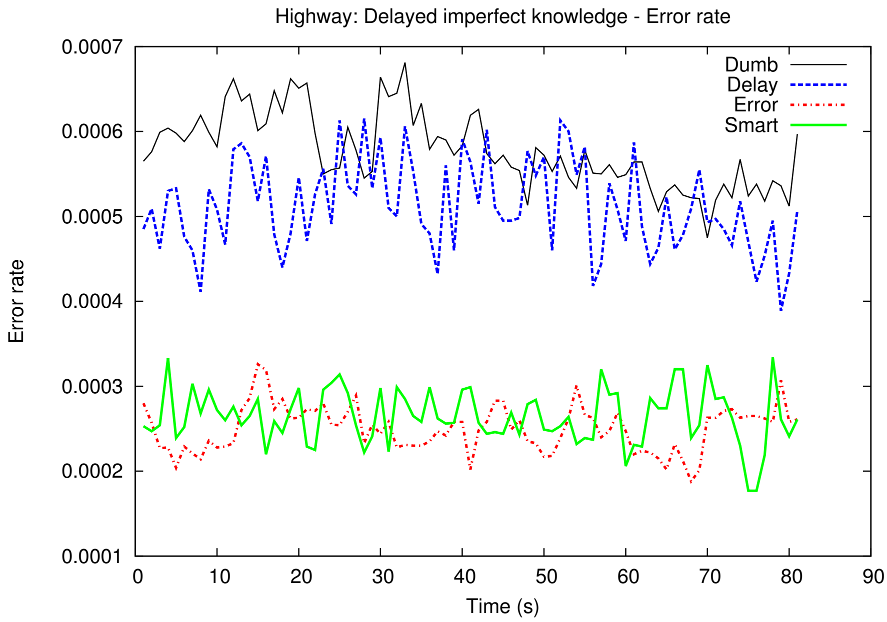

Figure 28

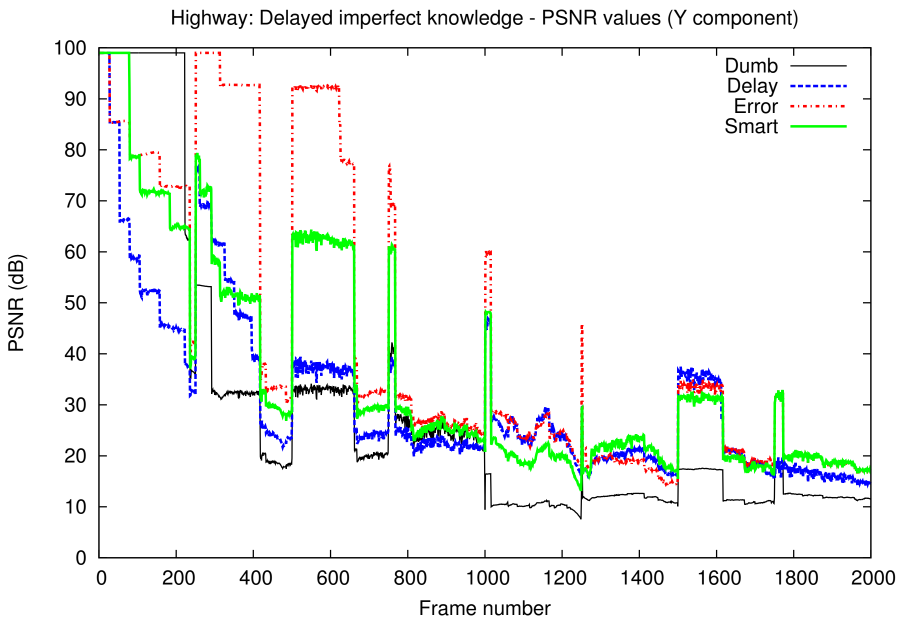

Figure 29

algorithms' average error rate are about half that value.

In the fourth and last scenario, the PSNR values drop significantly, but

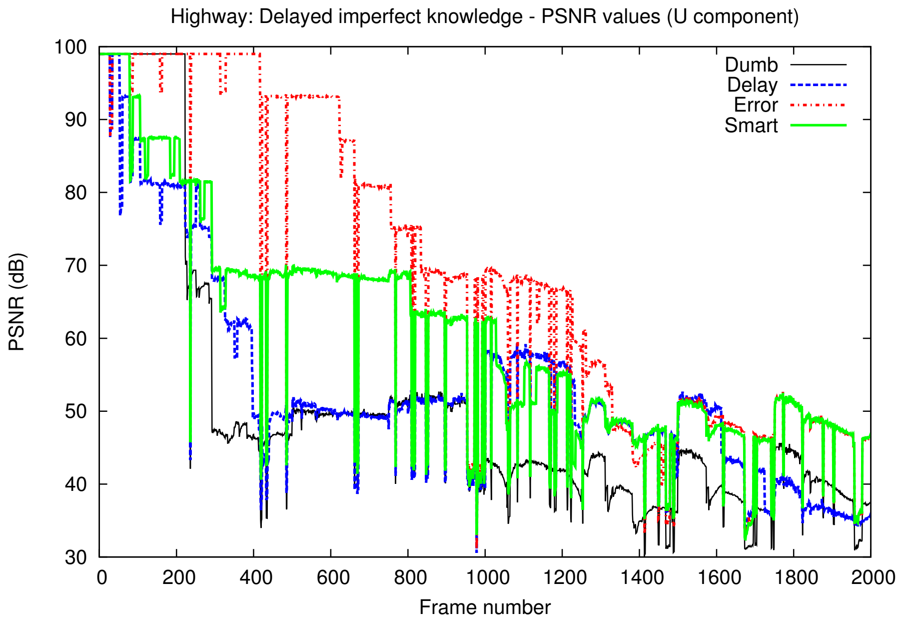

Figure 30

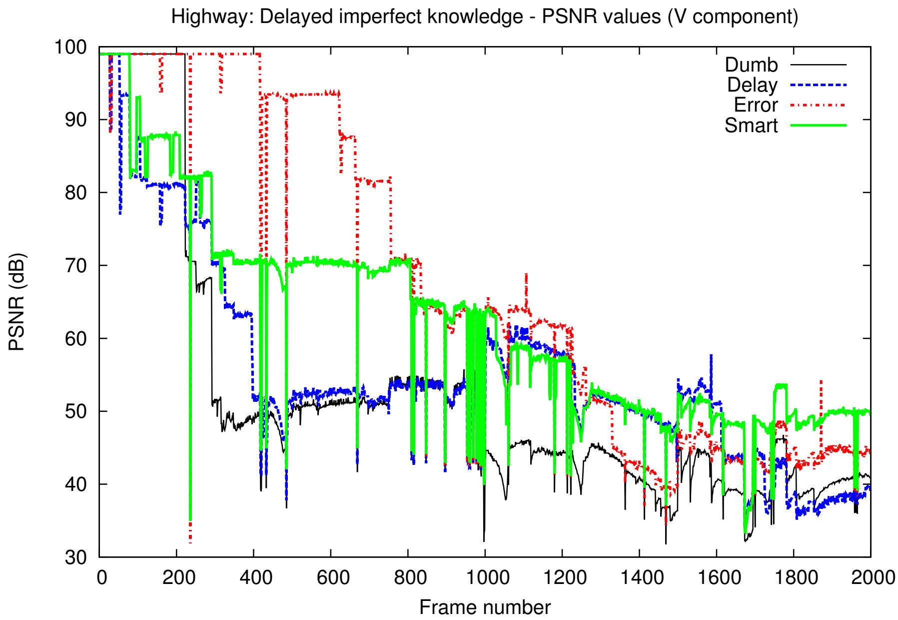

Figure 31

the difference between the three algorithms is the same. The Delay algorithm gains about 5 dB over the Dumb algorithm on the three components, while

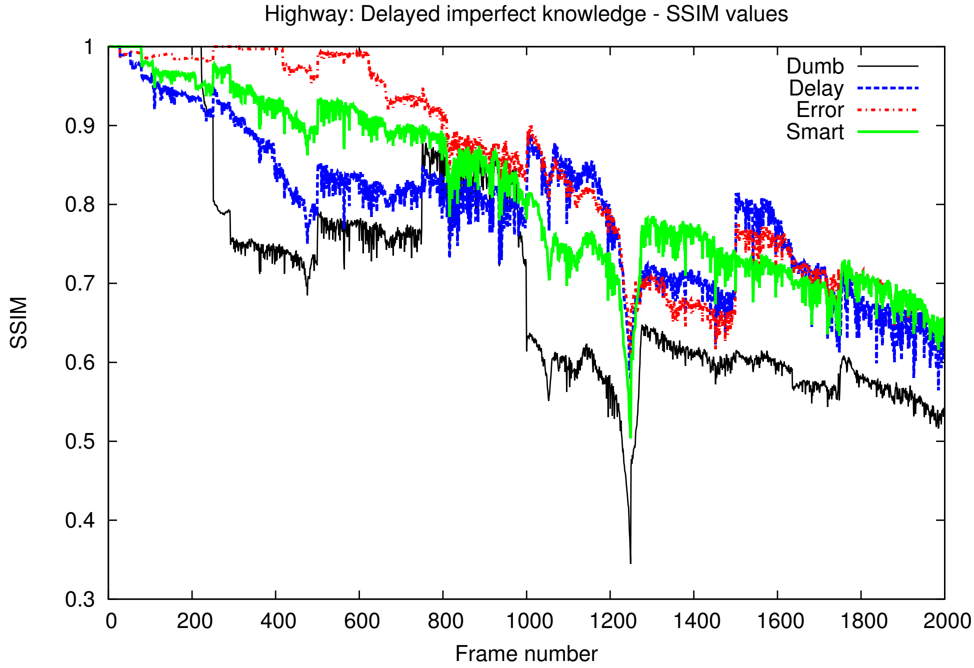

Figure 32

the Smart algorithm's gain is about 10 dB. As in the other scenarios, the Error algorithm has the highest PSNR values, with an increase of 15 dB to 20 dB over the reference Dumb value.

In this scenario, the Delay, Error and Smart algorithms have average SSIM values of 0.79, 0.84 and 0.82 respectively; Figure 32 also shows clearly the similarity of the three algorithms' QoE values. This effect is not present in the PSNR values, as Figures 29, 30 and 31 show; this may be due to the distribution of the errors in the simulation results.

5 Conclusions and future work

5.1 Conclusions

We implemented and tested 3 different path selection algorithms, along with the reference Dumb algorithm to compare them to. As Figure 33 shows, the Error algorithm successfully maximizes QoE but suffers in terms of delay, while the Delay algorithm does the opposite. The Smart algorithm seems to find a successful trade-off between the two, with SSIM values about 0.15 higher than Dumb, without a correspondent increase in delay.

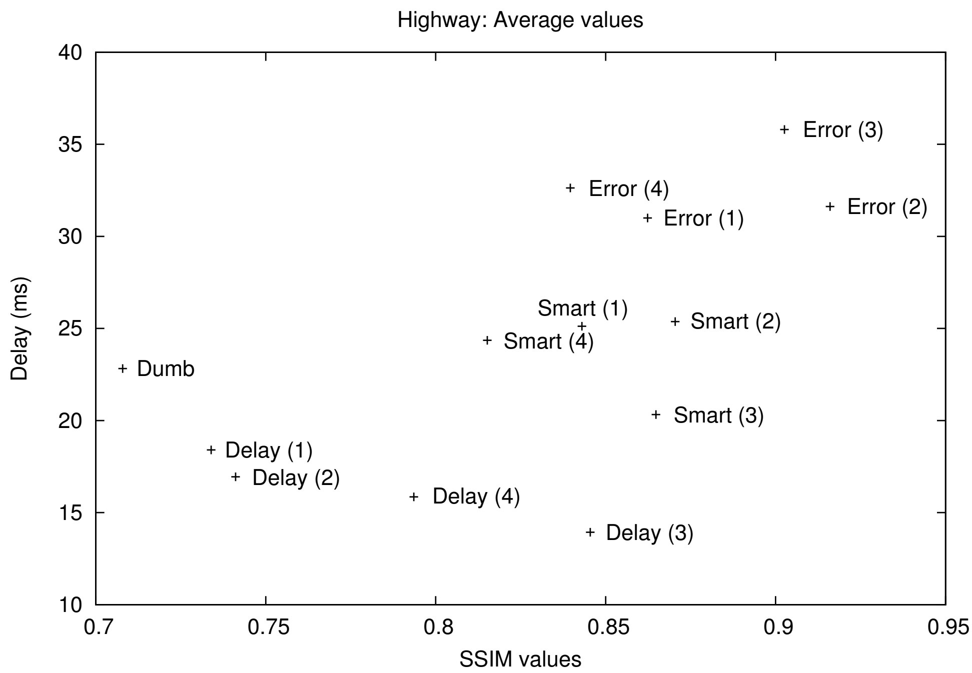

Figure 33: Comparison of the algorithms in the four scenarios in terms of SSIM and average delay

The Delay and Error algorithms represent the two extremes in the trade-off between QoE and low delay; however, their limitations do not mean that there are no use cases in which they might prove beneficial. Live video streaming applications, used for sporting or social event broadcasts, generally have low quality requirements but extremely low latency tolerance, and might be one of the possible applications of a purely delay-based algorithm. High quality video applications such as movie streaming services might, instead, benefit from using the Error application, as the video content is usually already stored n the video cache and has no simultaneity requirements.

The Smart algorithm is a compromise between the two extremes: while the PSNR gain over the Dumb algorithm is about 15 dB to 20 dB in the first three scenario and about 10 dB in the last, and the SSIM gain is about 0.15, its delay values are the same as the Dumb algorithm's within a $\pm 10\%$ tolerance. The results on QoE are not as impressive as the Error algorithm's, but the delay cost is reduced to almost zero. By tuning the parameters in the Smart algorithm formula, the algorithm can be adjusted to the needs of the application and the characteristics of the network; non-linear additions such as a maximum acceptable delay can also be easily implemented. Figure 33 shows a clear correlation between SSIM and delay when using path selection algorithms; the Smart algorithm achieves an efficient balance between the Delay and Error algorithms, as its SSIM values are significantly and consistently higher than the Dumb and Delay algorithms', but there is no correspondent delay cost.

5.2 Future work

While the path selection algorithms we discussed in the earlier chapters have demonstrated their efficiency in reducing delay and increasing QoE values in variable network conditions, there are several possible future developments that might be studied in future works.

The first and simplest development of the algorithms might be the addition of the wireless link parameters; allowing UEs to choose the best wireless link when more than one is available might prove to be a simple and efficient improvement. An interesting problem for an algorithm that switched between different wireless link is handover management; handovers between eNBs and between E-UTRAN and other RAN technologies might have a significant impact on video QoE. A possible solution is the introduction of a "switching cost" so that the UE might avoid handovers unless there is a significant advantage to changing the wireless link.

As we discussed in **section 5.1**, another possible future development is a finer tuning of the Smart algorithm, providing users with a variety of choices and options to optimize its performance in specific applications and network conditions. Although the linear formula's performance is already satisfactory, the use of non-linear formulae in Smart-type algorithms might prove to be even more efficient in increasing video QoE while keeping delay values as low as possible.

Another possible problem is the effect of a multi-receiver scenario: if several UEs that are close to each other stream videos over the LTE network, traffic conditions might become the dominant variable in path selection. However, simulating this scenario might require major changes in the QoE-Monitor module, as at the moment the *MultimediaApplicationReceiver* class has no way of distinguishing its own video packets from the ones requested by other users. Once the technical difficulties are solved, the optimization problem becomes simiar to the ones in the works described in **section 2.3**. As the use of greedy deterministic algorithms such as the ones presented in this thesis might not be efficient in this kind of scenarios, cooperative or adaptive algorithms [47] should be considered and tested.

The effect of different video contents on QoE might also be an interesting subject: as the results show, different videos can have significant QoE variations in the same network conditions, and the more dynamic parts of a video can cause sudden drops in QoE. An optimization framework that takes into account this phenomenon might be developed by studying the codecs and the effect of errors on different packets.

6 References

[1] *Cisco Visual Networking Index: Forecast and Methodology, 2011-2016*, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html, Cisco, 2012

[2] *MEDIEVAL deliverable 1.1, Preliminary Architecture Design*, http://ict-medieval.eu/10.html, 2011

[3] *LTE specification, Release 8*, http://www.3gpp.org/Release-8, 3GPP, 2009

[4] Y. Hujun, S. Alamouti, *OFDMA: A Broadband Wireless Access Technology*, Proceedings of 2006 IEEE Sarnoff Symposium, pp. 1-4, 2006

[5] *UMTS service aspects specification*, http://www.3gpp.org/ftp/Specs/ html-info/22-series.htm, 3GPP, 1999

[6] *LTE-Advanced specification, Release 12*, http://www.3gpp.org/Release-12, 3GPP, 2012

[7] N. Amram, B. Fu, G. Kunzmann, T. Melia, D. Munaretto, S. Randriamasy, B. Sayadi, J. Widmer, M. Zorzi, *QoE-based Transport Optimization for Video Delivery over Next Generation Cellular Networks*, Proceedings of 2011 IEEE Symposium on Computers and Communications (ISCC), pp. 19-24, 2011

[8] *MEDIEVAL deliverable 5.2, Final specification for transport optimisation components and interfaces*, http://ict-medieval.eu/10.html, 2012

[9] *ALTO Working Group Charter*, http://datatracker.ietf.org/wg/alto/ charter/, IETF ALTO Working Group, 2011

[10] *Ns-3 Overview*, https://www.nsnam.org/overview/what-is-ns-3/, Ns-3 Consortium, 2011

[11] *GNU General Public License, v. 2*, http://www.gnu.org/licenses/gpl-2.0.html, Free Software Foundation, 1991

[12] D. Saladino, A. Paganelli, M. Casoni, *A Tool for Multimedia Quality Assessment in NS3: QoE Monitor*, Simulation Modelling Practice and Theory, vol. 32, pp. 30-41, 2013

[13] *GSM/EDGE specification*, http://www.3gpp.org/ftp/Specs/html-info/TSG-WG–GP.htm, 3GPP, 2003

[14] *Long term Evolution (LTE): a technical overview*, Motorola Technical White Paper, 2012

[15] H. G. Myung, J. Lim, D. Goodman, *Single carrier FDMA for uplink wireless transmission*, IEEE Vehicular Technology Magazine, vol. 1, no. 3, pp. 30-38, 2006

[16] *LTE-EPC Simulator Documentation*, http://lena.cttc.es/manual/, Centre Tecnològic de Telecomunicacions de Catalunya (CTTC)

[17] L. Wenjing, W. Dezheng, L. Wei, R. Lanlan Q. Xuesong, *Research on management architecture for Home NodeB access network*, Proceedings of 2nd IEEE International Conference on Broadband Network & Multimedia Technology, pp. 247-251, 2009

[18] A. Larmo, M. Lindström, M. Meyer, G. Pelletier, J. Torsner, H. Wiemann, *The LTE link-layer design*, IEEE Communications Magazine, vol. 47, no. 4, pp. 52-59, 2009

[19] I. Ali, A. Casati, K. Chowdhury, K. Nishida, E. Parsons, S. Schmid, R. Vaidya, *Network-based mobility management in the evolved 3GPP core network*, IEEE Communications Magazine, vol. 47, no. 2, pp. 58-66, 2009

[20] H. Jihai, W. Bingyang, *Handover in the 3GPP long term evolution (LTE) systems*, Proceedings of 2010 Global Mobile Congress (GMC), pp. 1-6, 2010

[21] *802.16-2012 - IEEE Standard for Air Interface for Broadband Wireless Access Systems*, http://standards.ieee.org/findstds/standard/802.16-2012.html, 2012

[22] L. Badia, R. L. Aguiar, A. Banchs, T. Melia, M. Wetterwald, M. Zorzi, *Wireless access architectures for video applications: the approach proposed in the MEDIEVAL project*, Proceedings of 2010 IEEE Symposium on Computers and Communications (ISCC), pp. 991-996, 2010

[23] *MEDIEVAL deliverable 5.1, Transport Optimisation: Initial Architecture*, http://ict-medieval.eu/10.html, 2011

[24] D. Munaretto, T. Melia, S. Randriamasy, M. Zorzi, *Online path selection for video delivery over cellular networks*, Proceedings of IEEE Globecom 2012, pp. 1367-1372, 2012

[25] H. Schwarz, D. Marpe, T. Wiegand, *Overview of the Scalable Video Coding Extension of the H.264/AVC Standard*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 9, pp. 1103-1120, 2007

[26] D. J. Vergados, A. Michalas, A. Sgora, D. D. Vergados, *An Adaptive Video Transmission Algorithm to Improve Quality of Experience over LTE Access Networks*, Proceedings of 16th Panhellenic Conference on Informatics (PCI), pp. 240-244, 2012

[27] M. Shehada, S. Thakolsri, Z. Despotovic, W. Kellerer, *QoE-based Cross-Layer Optimization for video delivery in Long Term Evolution mobile networks*, Proceedings of 14th International Symposium on Wireless Personal Multimedia Communications (WPMC), pp. 1-5, 2011

[28] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, J. G. Andrews, *Video capacity and QoE enhancements over LTE*, Proceedings of 2012 IEEE International Conference on Communications (ICC), pp. 7071-7076, 2012

[29] L. Badia, M. Miozzo, M. Rossi, M. Zorzi, *Routing schemes in heterogeneous wireless networks based on access advertisement and backward utilities for QoS support*, IEEE Communications Magazine, vol. 45, no. 2, pp. 67-73, 2007

[30] M. Venkataraman, M. Chatterjee, *Effects of Internet Path Selection on Video-QoE: Analysis and Improvements*, IEEE/ACM Transactions on Networking, no. 99, pp 1-14, 2013

[31] Z. Ma, H. R. Shao, C. Shen, *A new multi-path selection scheme for video streaming on overlay networks*, Proceedings of 2004 IEEE International Conference on Communications (ICC), pp. 1330-1334, 2004

[32] M. Jain, C. Dovrolis, *Path selection using available bandwidth estimation in overlay-based video streaming*, Computer Networks, vol. 52, no. 12, pp. 2411-2418, 2008

[33] J. G. Apostolopoulos, M. D. Trott, *Path diversity for enhanced media streaming*,IEEE Communications Magazine, vol. 42, no. 8, pp. 80-87, 2004

[34] M. Guo, Q. Zhang, W. Zhu, *Selecting path-diversified servers in content distribution networks*, Proceedings of 2003 IEEE Global Telecommunications Conference (GLOBECOM), vol. 6, pp. 3181-3185, 2003

[35] *ITU-T Recommendation P.912: Subjective video quality assessment methods for recognition tasks*, http://www.itu.int/rec/T-REC-P.912-200808-I, 2008

[36] *ITU-T Recommendation J.340: Reference algorithm for computing peak signal to noise ratio of a processed video sequence*, http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=10551&lang=en, 2010

[37] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, *Image Quality Assessment: From Error Visibility to Structural Similarity*, IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, 2004

[38] T. Melia, D. Munaretto, L. Badia, M. Zorzi, *Online QoE Computation for Efficient Video Delivery over Cellular Networks*, IEEE COMSOC MMTC E-letter, vol. 7, no. 3, pp. 13-16, 2012

[39] B. Girod, *What's wrong with Mean-Square Error?*, Digital Images and Human Vision, pp. 207-220, MIT Press, 1993

[40] Q Huyn-Thu, M. Ghanbari, *Scope of Validity of PSNR in Image/Video Quality Assessment*, Electronics Letters, vol. 44, no. 13, pp. 800-801, 2008

[41] A. Horé, D. Ziou, *Image quality metrics: PSNR vs. SSIM*, Proceedings of 20th International Conference on Pattern Recognition (IPCR), pp. 2366-2369, 2010

[42] M. Djadja, A. Naamane, N. Giambiasi, *Approach for discrete event simulation*, Electronics Letters, vol. 34, no. 16, pp. 1615-1616, 1998

[43] *ITU-T Recommendation H.264: Advanced video coding for generic audiovisual services*, http://www.itu.int/rec/T-REC-H.264, 2013

[44] *Ffmpeg documentation*, http://www.ffmpeg.org/ffmpeg.html, Ffmpeg Project, 2013

[45] *H264 YUV CIF EvalVid Reference videos*, EvalVid Project, http://www2.tkn.tu-berlin.de/research/evalvid/cif.html, Technische Universität Berlin, 2011

[46] G. Sullivan, S. Estrop, *Recommended 8-Bit YUV Formats for Video Rendering in Windows*, http://msdn.microsoft.com/en-us/library/windows/desktop/dd206750%28v=vs.85%29.aspx, Microsoft Corporation, 2008

[47] L. Badia, A. Botta, L. Lenzini, *A genetic approach to joint routing and link scheduling for wireless mesh networks*, Journal of Ad Hoc Networks, vol. 7, no. 4, pp. 654-664, 2009