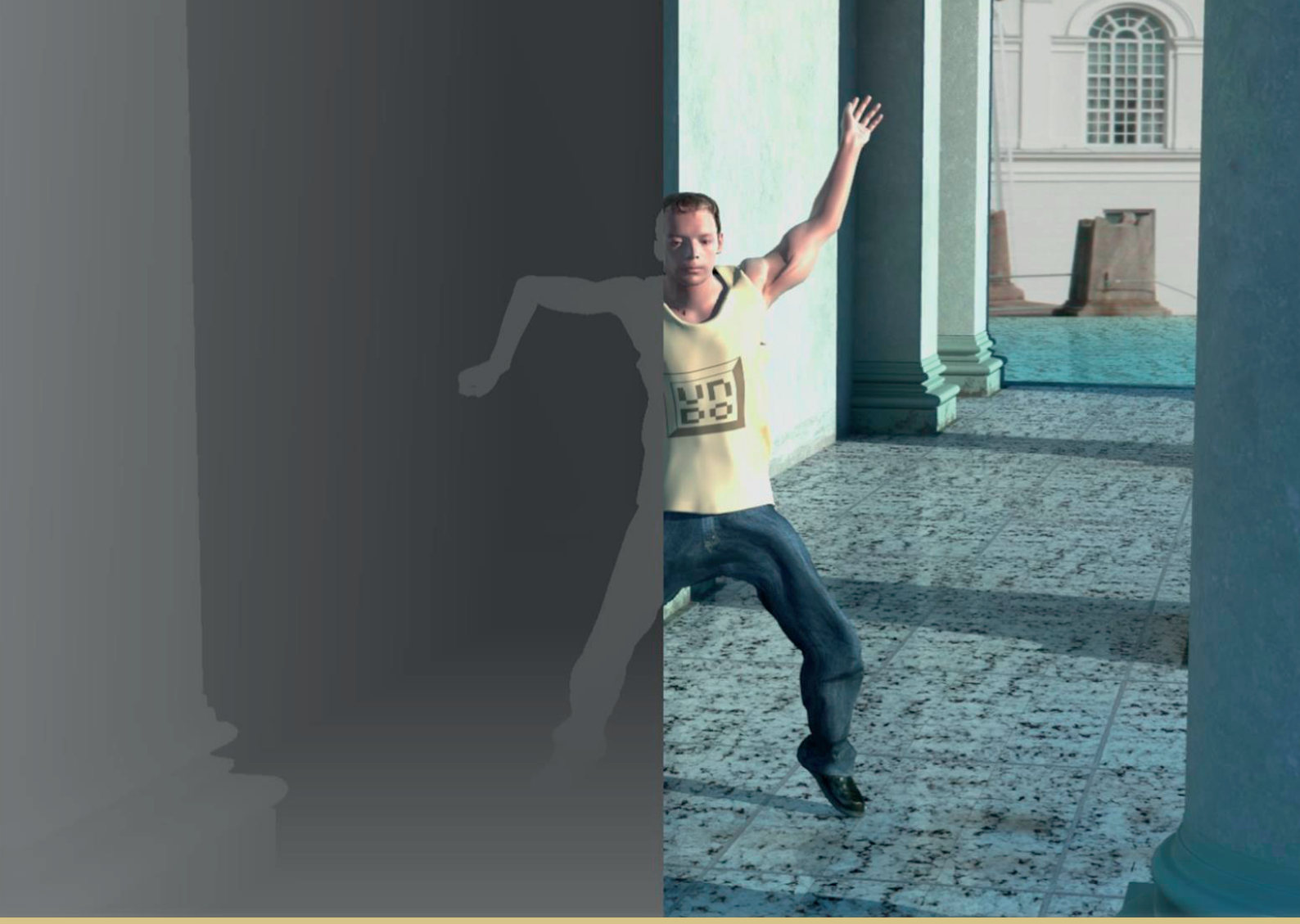


Pedro Miguel Regalo Rocha

NO-REFERENCE MODELS FOR 3D VIDEO QUALITY ASSESSMENT

Master Thesis In Electrical and Computer Engineering

September 2015

· U C ·

Universidade de Coimbra

Objective Quality Method for 3D Video

Author:
Pedro Rocha

Supervisor:
Prof. Dr. Luis Cruz

Master Thesis in Electrical and Computer Engineering

Jury:
Prof. Dr. Vitor Manuel Mendes da Silva
Prof. Dr. Nuno Miguel Mendonça da Silva Gonçalves
Prof. Dr. Luís Alberto da Silva Cruz

Setembro 2015

Acknowledgements

The following lines are dedicated to those who accompanied me during this research work and helped me achieve this goal.

I would like to begin by expressing my sincere gratitude to my supervisor, Professor Dr. Luís Cruz, who have always believed in my working capabilities and was always available to discuss all technical issues and new ideas for the work. He encouraged me to be part of a research team, in which I learned different skills and met interesting people. I am also grateful for the effort he put on reviewing this essay and the patience demonstrated during the last year. For all his support, I am truly grateful.

I would like to thank Instituto de Telecomunicações de Coimbra for the support and laboratory facilities that gave me conditions to accomplish this work.

To my lab colleagues, João Sousa, Nuno Almeida, Nuno Carvalho, Ricardo Rocha, Guilherme Corrêa, Rui Peliteiro and Marlene Machado, thank you for the time well spent, and specially, thank you for being good listeners. Without you, I am sure this work wouldn't be possible.

I would like to mention and thank the volunteers who participated in the subjective session: Dr. Luis Cruz, Osama, Marlene Machado, Nuno Gonçalves, João Sousa, Ricardo Rocha, Rui Peliteiro, Dr. Marcelo, João Seabra, Vasco Mota, Filipe Silva, Joana Figueiredo, João Gante, António Simões, João Maria, Nuno Vicente, Sofia Ferreira, Pedro Bento, Luís Raposo, Guilherme Medeiros, Ricardo Ralha, Andreia Pereira, Ricardo Cabrita, Diana Lopes, André Pinto, José Francisco, João Soares, Pedro Pinto, João Amaro, João Ferreira, João Suzana Ferreira and Francisco Tavares. I hope you all enjoyed watching 97 sequences of 3D video without special glasses. It meant a lot for my work.

Of course, to all my friends that accompanied me during these last years. You were the ones who were there when things looked bad and always cheered me up. If these were the best years of my life, I owe them to you. Thank you.

Finally, to my family: my father, my mother and my sister. I owe everything to you, even when you were the ones that suffered the consequences of my bad humor and absence, you always had a kind word to motivate me. My deepest thank you "crazy family".

Abstract

This thesis presents a research work on no-reference quality assessment models for use in future 3D video broadcasts applications over packet loss prone channels, such as Internet Protocols networks. The objective is to study the most recent quality measures for 3D video available in the scientific literature, and to propose new no-reference quality assessment models oriented towards packet loss effects on the 3D quality of experience (QoE).

Two methodologies, with the same mathematical background, are proposed. The models receive two types of input parameters: packet loss rate (PLR) and size of lost packet (SLP). Each parameter is obtained from the texture and depth stream, being further divided according to the type of frame: I, P or B, which totals 12 input parameters. The proposed models output two different scores: the Structural Similarity Index (SSIM) and 3D Synthesized View Image Metric (3DSwIM) of the DIBR synthesized view. The developed models are based on neural networks (NN), which can efficiently process large numbers of inputs and provide a functional relationship between inputs and outputs. To train the proposed models with a considerable degree of generalization, hundreds of simulations with different packet loss rates and mean burst lengths were performed. Most of the proposed models achieve high accuracy, as the Person Linear Correlation Coefficient (PLCC) of 0.90-0.97 between the estimated and reference objective measures indicates. To verify the correlation between the proposed models outputs and the corresponding subjective scores (measured in differential mean opinion score (DMOS)), a set of subjective tests involving 34 volunteers was conducted. The results show that DMOS correlates well with estimated DMOS from the SSIM of the synthesized view (PLCC of 0.8624) and correlate slightly worst with estimated DMOS from the 3DSwIM (PLCC of 0.8137).

The proposed models can be used in an industrial environment, either for service or network providers, where real time systems need to be monitored in order to identify packet loss events and quantify the impact these losses have on the user's QoE.

Keywords

3D video, texture-plus-depth, Quality of Experience, no-reference quality assessment, neural networks, packet-losses, H.265/HEVC

Resumo

Esta dissertação apresenta um trabalho de investigação no âmbito de modelos sem referência para avaliação de qualidade de vídeo 3D, no formato textura-mais-profundidade. Futuramente, com a maior difusão do vídeo 3D neste formato em redes de pacotes sujeitas a perdas, como por exemplo redes IP, modelos como o apresentado neste trabalho podem calcular o efeito das perdas na qualidade de experiência do utilizador. O objectivo da dissertação é estudar os métodos de qualidade de vídeo 3D mais recentes publicados na literatura científica, e propor novos modelos de avaliação sem referência específicos para degradações ocorridas por perda de pacotes.

Neste trabalho são propostas duas metodologias com a mesma base matemática. Os modelos recebem dois tipos de parâmetros de entrada: a taxa da perda de pacotes e o tamanho médio dos pacotes perdidos. Cada parâmetro é obtido para os *streams* de textura e profundidade, sendo posteriormente divididos consoante o tipo de trama, I, P ou B, perfazendo no total 12 parâmetros de entrada. Os modelos propostos resultam em duas saídas: o SSIM e o 3DSwIM da vista sintetizada a ser avaliada. Para a modelação do prolema, foram usadas redes neuronais, pois estas conseguem de forma eficiente processar várias entradas e produzir uma relação matemática entre as saídas e as entradas. Para treinar os modelos propostos com um certo grau de generalização, foram efectuadas centenas de simulações com diferentes taxas de perda de pacotes e comprimentos médios de rajadas. A maioria dos modelos revelou uma correlação elevada entre as saídas dos modelos e a qualidade estimada pelas métricas objectivas conforme o índice linear de correlação de Pearson (PLCC de 0.90-0.97) indica. Para verificar a correlação entre os modelos desenvolvidos e a opinião subjetiva de diferentes pessoas (DMOS), foi organizada uma sessão de testes subjectivos que envolveu 34 voluntários. Os resultados mostram que o DMOS tem uma boa correlação com o DMOS estimado a partir do SSIM (PLCC de 0.8624) e uma correlação relativamente boa entre o DMOS estimado a partir do 3DSwIM (PLCC de 0.8137).

Os métodos propostos podem ser implementados num ambiente industrial, quer para fornecedores de serviços ou redes, onde sistemas que funcionam em tempo real precisam de ser monitorizados com o objectivo de identificar perdas de pacotes

durante a transmissão e quantificar o efeito que essas perdas têm na qualidade de experiência final do utilizador/cliente.

Keywords

Vídeo 3D, textura-mais-profundidade, qualidade de experiência, avaliação de qualidade sem referência, redes neuronais, perda de pacotes, H.265/HEVC

Contents

List of Tables

List of Figures

Acronyms

3D Three dimensional

2D Two dimensional

3DSwIM 3D Synthesized View Image Metric

3DTV Three Dimensional Television

ACR Absolute Category Rating

ACR-HR Absolute Category Rating with Hidden Reference

AVC Advanced Video Coding

CRA Clean Random Access

CTB Coding Tree Block

CTU Coding Tree Unit

CU Coding Unit

DMOS Differential Mean Opinion Score

DIBR Depth Image Base Rendering

FR Full-Reference

GOP Group of Pictures

HD High Definition

HVS Human Visual System

HEVC High Efficiency Video Coding

IDR Instantaneous Decoder Refresh

IPTV Internet Protocol Television

LCU Large Coding Unit

MBL Mean Burst Length

MOS Mean Opinion Score

MTU Maximum Transmission Unit

NAL Network Abstraction Layer

NN Neural Networks

NR No-Reference

OQM Objective Quality Metric

PLR Packet Loss Rate

PLP Packet Layer Parameter

P2P Peer-to-Peer

PLCC Pearson Linear Correlation Coefficient

PSNR Peak-to-Signal-Noise Rate

POC Picture Order Count

PU Prediction Unit

QP Quantization Parameter

QoE Quality of Experience

QoS Quality of Service

RR Reduced-Reference

RTP Real Time Protocol

SS Single Stimulus

SSIM Structural Similarity Index

SAO Sample Adaptive Offset

SLP Size of Lost Packets

UHD Ultra High Definition

VCL Video Coding Layer

VOD Video-on-Demand

VQM Video Quality Metric

Chapter 1

Introduction

The wide market of 3D video is yet to be fully explored. As the demand for digital 3D video is increasing, particularly in the 3D cinema area where immersive movies such as *Avatar* turned out very profitable, it is expected in a few years a gradual replacement of 2D television broadcasts with 3D Television broadcasts (3DTV), becoming part of our daily life. Some companies have already invested in the broadcast of 3D television, with dedicated channels being broadcast in 2D-frame-compatible side-by-side stereo format. Most current 3D video solutions are based on the rendering and displaying of multiplexed left and right views with depth perception being induced by the stereo parallax effect through special glasses that channel each view to the corresponding human eye. As an alternative and more comfortable solution, autostereoscopic or even holographic displays allow viewers watch 3D videos without the need of wearing special glasses.

In terms of network requirements, the fast growth of the television broadcast (and 3DTV in the future) over the Internet (IPTV) and Video-on-Demand (VOD) services will greatly increase the amount of data traffic exchanged in the communications network. A recent forecast published by Cisco [1] states that "annual global IP traffic has increased fivefold over the past five years, and will increase threefold over the next five years", surpassing the zettabyte (10^{21} bytes) threshold by the end of 2016, and will reach 2 zettabytes per year by 2019. Figure 1.1 shows the Global IP traffic (P2P traffic not included) divided into application categories including Internet video traffic which will account for 80% of global traffic by 2019, an increase of 13% compared to 2014's 67%. If P2P video traffic is taken into account this ratio might be close to 90%. It is clear that this traffic growth will require modifications and upgrades in the existent network infrastructures. Transmission protocols will be revised to support higher bandwidths, specially in real-time protocols (such as real time protocol - RTP) where retransmission is not allowed; at an application level, coding and compression efficiency take an important role in transmission and therefore need to be improved. If these conditions are not met, network Quality of

Service (QoS) will decrease due to the packet loss occurred in congested routers as a result of increased traffic.

Figure 1.1: Global consumer IP traffic forecast by Cisco [1]

The goal of any multimedia delivery system is to make sure the content is delivered to the end client with the best quality possible. In [2] it is described how a digital video, in an IPTV or VOD service, might go through different stages before reaching the end user: it can pass from content provider to service provider to network provider. At each of these stages it is imperative to keep the video quality as high as possible and ensure the next stage won't degrade it significantly. By monitoring the network in specific points (or nodes), i.e., from time to time retrieve information about the state of the data being transmitted, it is possible to detect network failures and errors. The monitoring system may be deployed at the set-top boxes, or at some node in the transport network, updating the service provider about the quality of the data being received. With this information the streaming services can adjust dynamically some of the transmission (and perhaps coding) parameters in order to maintain a minimum quality level [3], [4]. In wireless channels which are prone to high bit-error rates and long bursts of transmission errors, video can be transcoded just before wireless transmission (i.e, at the end of the wired channel) to reduce the coding bitrate. Although this would lower the video quality, it would save more bits that could be used in better error protection schemes to reduce the effect of packet losses and transmission errors. An alternative to this approach would be giving priority to different types of data, with more powerful error protection schemes applied to more important data, to reduce the impact of bit errors and packet losses on decoded video quality [5], [6]. This principle is applicable in 3D video in the texture-plus-depth format (see figure 1.2), where packets transporting texture information are more important than those transporting depth information [7], because if depth is degraded or lost but texture information is re-

ceived, at least 2D video can be reconstructed and presented to the viewer. If reliable transmission is guaranteed it is important to consider the jitter phenomena, which consists of temporal variations of the propagation delay of consecutive packets, and leads to events similar to packet loss as a result of the uselessness of the video data received outside a usability time horizon. Measuring the jitter allows the receiver to adjust the buffer size and buffering times and to request the retransmission of the most important lost packets.

Figure 1.2: Example of a frame extracted from a 3D video in the texture-plus-depth format. The texture frame (left) has a correspondent per-pixel depth frame with depth information in the form of a depth map (right).

Digital video quality monitoring is of extreme importance to multimedia service providers, specially for those dealing with 3D delivery systems. However, due to the subjective factor involved, creating a model that objectively predicts the perceived quality of a visual stimulus by humans is not an easy task [2], [8]. To address different opinions from the viewers it is important to average the mean opinion score (MOS) of the perceived quality of a given visual stimulus from, at least, 15 human observers which are shown the stimulus and are asked to grade it on an opinion grading scale [9], [10]. The obtained MOS must then correlate well with the objective method outputs to prove its consistency. It is also important noticing that the typical 2D video quality assessments are not well suited to fully assess 3D video. Even though 3D video may present the same artifacts as a 2D video, like blurring, ringing, blocking or freezing (see figure 1.3) which are caused by the compression and transmission of the 3D video signal, other factors have to be considered for 3D video, particularly in the texture-plus-depth format, such as distortions induced by the rendering of virtual views using depth-image-base rendering (DIBR). Even if the 2D texture quality is high, a low quality depth map will increase rendering distortions, which may affect the viewer in several different ways: fatigue, nausea, headache and severe mental confusion that result in low 3D quality of experience (QoE) for the viewer.

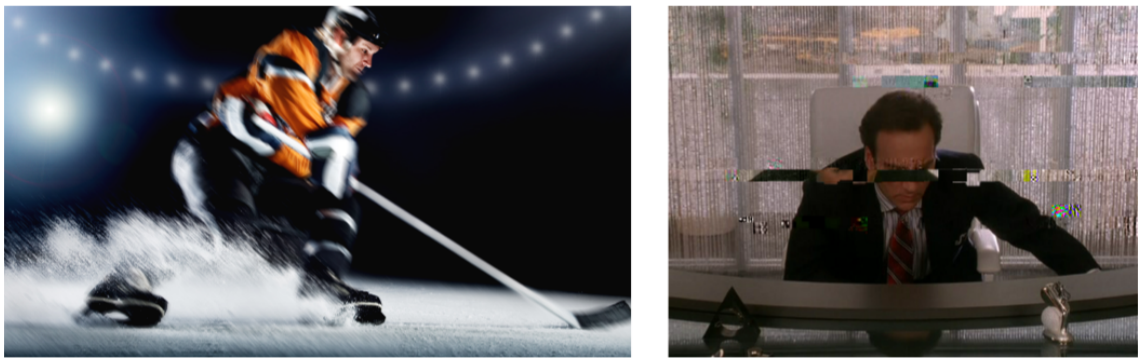


Figure 1.3: Examples of possible artifacts present in videos: motion blur (left) and decoded frame affected by data loss during transmission (right).

All these factors make MOS obtained from the subjective assessment an extremely useful asset because it represents the opinion of the observer already integrating all aspects of the perceptual experience, like visual comfort. The viewer's score reflects a combination of all these factors, with different weights from subject to subject. And these details make the problem of estimating 3D video quality a lot more complex than the same problem for 2D video.

1.1 Main Contributions

In this thesis an objective 3D video quality method that is able to predict the impact of packet losses on the quality of the 3D video (in the texture-plus-depth format) is presented. The study considers a transmission scenario where packets containing texture and depth data are lost during transmission and only these losses are considered when estimating the final 3D video quality degradation. Therefore, the only impairments affecting synthesized views are induced by the packet losses.

During the preparation of this thesis the following tasks were carried out:

- A bibliographic search was conducted in order to identify the most recent models and video quality metrics published in scientific journals or conference proceedings on the topic of objective 3D video quality assessment. A comparison and analysis of the methods found was also considered.

- Develop a new approach for objective quality assessment of synthesized 3D video subject to texture and depth packet losses, based on no-reference low-complexity models.

- Perform a subjective assessment study in order to measure the correlation between the quality scores estimated by the objective metric and the MOS values.

During the development of the work of this thesis, one article related with crowd-sourced methods for assessing 3D video quality [11] was published and presented in QoMEX 2015. One other paper describing this work is currently being prepared for submission.

1.2 Thesis Overview

In chapter 2, a detailed explanation and classification of the different multimedia quality assessment methodologies is given. The most recent methods for video and image quality assessment are presented, as well as an explanation of the reference methods used in this work.

Chapter 3 provides information about the most recent video coding standard, High Efficiency Video Coding (HEVC). The fundamentals and reference software are described. The Gilbert-Elliot model for packet-loss event generation is also explained in chapter 3.

The experimental procedures and results are presented in chapter 4. The proposed method is described in detail, with an explanation over the choice of this methodology.

Chapter 5 describes the procedures and results of the subjective quality grade collection campaigns conducted.

A practical implementation of a similar model proposed in this thesis is briefly explained in chapter 6.

Finally, chapter 7 concludes this thesis with an overview of the results obtained and the work involved. Suggestions for future work on the topic are also given.

Chapter 2

Objective Quality Assessment of 3D Video

The variety of available methods for assessing video or image quality makes it necessary to classify them according to the type of application or information used to predict the quality. This chapter provides an overview of the current standard classification of the quality assessment methods, a study on the most recent published methods and a detailed explanation of the methods used as a reference for predicting video quality.

2.1 Classification

A standard classification was proposed in [12]. The proposal is oriented towards objective quality measurement methods for multimedia transmitted over packet-switch networks, such as the Internet, which uses the input information for quality assessment and the primary application as distinguishing characteristics. Five different models/layers are then identified: packet-layer models, media-layer models, parametric planning models, bitstream layer models and hybrid models. Table 2.1 summarizes these five models.

	Media-layer	**Bitstream-Layer**	**Packet-Layer**	**Planning**	**Hybrid**
Input information	Pixel-domain	Packet-layer and payload information	Packet headers and codec information	Quality design parameters	Combination of any
Primary Application	Quality benchmarking	In-service nonintrusive monitoring	In-service nonintrusive monitoring (e.g. network probe)	Network planning, terminal/application designing	In-service nonintrusive monitoring

Table 2.1: Classification of objective quality measurements methods according to [12].

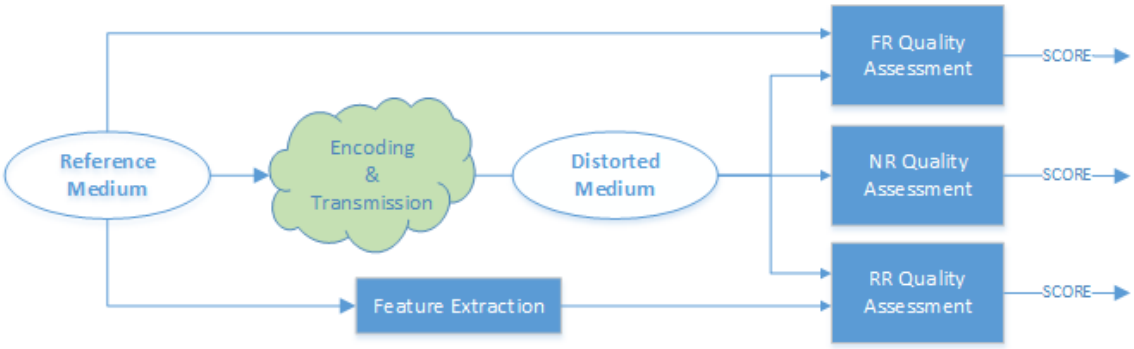


Figure 2.1: The availability or lack of the reference is used to categorize media-layer models.

Media-layer quality models utilize knowledge of the human visual system (HVS) to estimate subjective quality of video. Depending on the degree of information available from the original video as a reference in the quality assessment, the objective methods are further divided into three categories: full reference (FR), reduced reference (RR) and no-reference (NR), as shown in figure 2.1. FR methods extract information from the source video (usually high quality or non-impaired) and its processed counterpart where the reference signal acts as a consistent baseline for comparison. In RR methods, information is extracted from the reference signal, packaged and transmitted alongside the processed video, assuming that a side channel will be available to send the reference signal parameter data. This immediately poses a drawback, since the richness of information describing the properties of the reference signal is dependent on the existence of a side channel and its capacity. NR methods operate solely on information extracted from the processed signal. They search for artifacts with respect to the pixel domain of a video, use information available in the bitstream of the video, or perform quality assessment as a hybrid model based on pixel and bitstream data.

For both FR and RR methods to operate effectively, the reference and processed video sequences must be closely aligned. This spatio-temporal alignment (or registration) requirement represents a major obstacle to practical and real time applications of these models. They are adequate to implement on the server/sender side of the network, predicting the quality of encoders and transmission conditions at an early stage. Despite struggling with obtaining high accuracy without any content or quality benchmark, NR models do not require any kind of registration and as such, represent the most efficient means of measuring quality in a practical environment. They can easily be implemented on the receiver/client side (or other network location where the reference media is not available) and are typically faster than FR and RR methods.

An interesting approach is the combination of two different models. Through

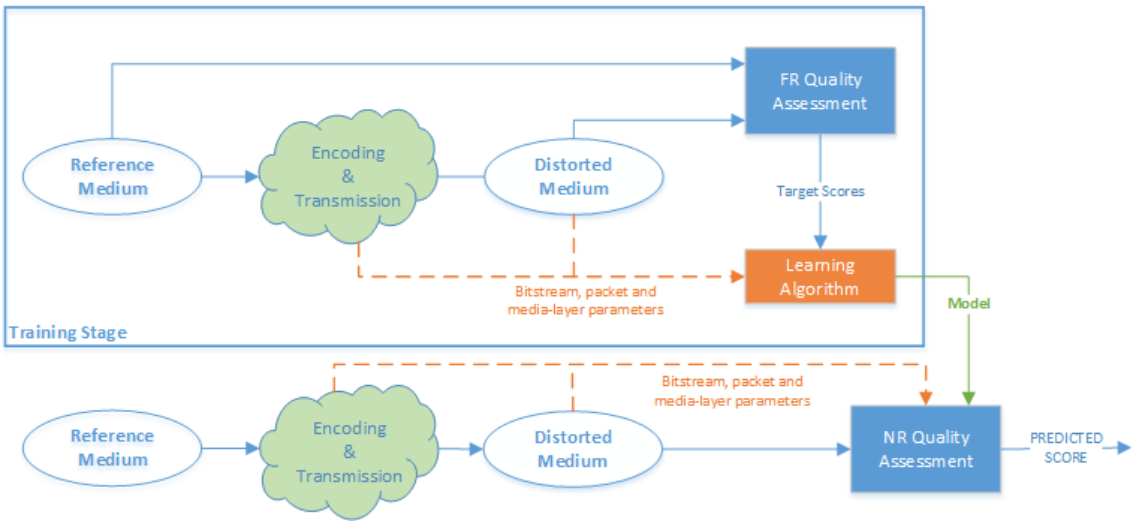


Figure 2.2: No-reference quality method that uses a FR model as a target model.

machine learning algorithms for fitting empirical parametric models such as neural networks, support vector machines or curve fitting, packet-layer, bitstream-layer and no-reference media-layer models can be used to estimate full-reference media quality scores. Depending on their layer of operation, these models use some input parameters to train a model expressing a mathematical relationship between the inputs and estimated quality scores. Typically these models are developed to cover a specific application or particular type of impairment. Figure 2.2 shows a no-reference model architecture using a full-reference model as target. The need for an effective and versatile lightweight NR method for real-time applications motivated the choice of this methodology. Streaming and distribution services are some of the vast business areas interested in having a similar monitoring system implemented, which demonstrates there is a market to explore.

So far, the variety of objective quality models have been discussed. But is important not to forget what is their goal: to estimate the opinion score a viewer would give to a video that might have been subjected to different types of impairments. Thus, the ground-truth score is the viewer's opinion and it is with it objective results must be compared with. What defines a good objective quality method is its correlation with subjective results obtained in a subjective assessment study. Pearson Linear Correlation Coefficient (PLCC or R) is the most used metric to evaluate the performance of an objective video quality model. It measures the correlation between the subjective MOS values x_i and the MOS values y_i predicted from the model. For N data pairs (x_i, y_i), with $\bar{x}$ and $\bar{y}$ being the means of the respective

datasets, the PLCC (or R) is given by:

$$PLCC = R = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}} \quad \in [-1, 1]. \tag{2.1}$$

To map the objective quality metric (OQM) scores and predicted MOS values y_i, ITU-R BT.500-13 [9] recommends the use of a logistic function, defined as:

$$y = \frac{a_1}{1 + e^{a_2(OQM + a_3)}} \quad , \quad a_1, a_2, a_3 \text{ being fitting coefficients} \tag{2.2}$$

As mentioned before, each objective quality metric has its own characteristics: type of application, type of method used and which parameters considers. Moreover, available 2D and 3D video/images databases differ from work to work and become unavailable after a while difficulting the comparison between metrics [13–15]. Table 2.2 shows the discrepancy between algorithms applied to different datasets. The authors in [16], after a comparison between some of the most recent 3D video or image metrics, recognize that it is very difficult to compare the performance of two different 3D quality methods due to different features evaluated, testing conditions, hardware and datasets used. In particular, NR methods are usually tuned to a specific type of artifact, making them inaccurate to evaluate other type of degradations. As each type of impairment usually affects the perceived quality in different ways, it is difficult to confront MOS of different works, due to the subjectiveness of each viewer's score [17].

Database	**VQEG**	**IRCCyN**	**LIVE**
PSNR	0.7683	0.4160	0.5621
SSIM [18]	0.8215	0.5012	0.5444
VQM [19]	0.8170	0.4850	0.7236
MOVIE [20]	0.8210	0.4850	0.8116
3D-SSIM [14]	0.8403	0.8194	0.8353
Yu et al. [21]	0.8170	0.7680	0.8450

Table 2.2: PLCC performance comparison of some of the state-of-the-art image and 2D video quality algorithms [16].

Thus, it is important to compare what is comparable and remember that every quality metric has its advantages and disadvantages, objectives and application scope.

2.2 Newest 3D Video/Image Quality Metrics

In this section, some of the state-of-the-art metrics are briefly presented[1]. Following the classification suggested in section 2.1, the algorithms will be divided in full-reference, reduced-reference and no-reference.

2.2.1 Full-Reference

Having full access to original content, FR metrics usually provide an accurate quality estimation. Based on pixel and feature analysis, these methods often involve complex algorithms which require a lot of time and resources to process.

In 2010, the Perceptual Quality Metric was proposed by Joveluro et al. [22]. The authors stated that metrics with good representation of the HVS will provide a more accurate evaluation, proposing a 2D based metric which measures distortion in the brightness and contrast distortion using an approximation weighted by the mean of each pixel block. Using texture-plus-depth with scalable encoding (Joint Scalable Video Model - JSVM) at different quantization parameter (QP) and applying 2D metrics for evaluating DIBR synthesized views, they achieved a PLCC between MOS and PQM of 0.988 (average).

A year after, the View Synthesis Quality Assessment metric [23] is presented and similarly to the previous method, it is considered an extension of any existent 2D image quality assessment metric. The authors consider view synthesis through disparity estimation between left and right images and then interpolation (or extrapolation) of the virtual view through disparity compensation. The proposed method aims at detecting artifacts in synthesized views and to handle areas where disparity estimation may fail (object borders, thin objects, transparency ...). The main feature of this metric is the use of three visibility maps which characterize complexity in terms of texture, diversity of gradient orientations and presence of high contrast.

During the same year, Solh et al. [24] introduced the 3D Video Quality Measure. This method analyzes the quality of the depth map compared to an ideal depth map (distortion-free image given the same reference image and some DIBR parameters). The estimated ideal depth map is then used to derive three different distortion measures to estimate the quality. These consist of temporal outliers (TO), temporal inconsistencies (TI) and spatial outliers (SO) and when combined, a final quality value is calculated. A PLCC of 0.8942 was achieved after performing subjective tests.

Yasakehu et al. [25] proposed an adapted Video Quality Metric (VQM) [19] that

[1]The large number of methods reported in the scientific literature, makes it impossible to cover them all. An effort was made to cover the most relevant ones.

measures the impact of packet loss on 3D video. It combines 2D color and depth information quality: VQM is used for estimating color quality whereas depth quality measurement is based on the analysis of the depth planes distortion. Three different values are measured and then combined: distortion of the relative distance within each depth plane; the consistency of each depth plane; and the structural error of the depth.

More recently, Fezza et al. [26] introduced a new method which handles effectively asymmetric distortions of stereoscopic images by incorporating HVS characteristics in the algorithm. Asymmetric distortions are caused by the use of different coding settings in one (usually, the auxiliary view) - or more - of the two - or more - views available, also known as asymmetric coding. To measure asymmetric distortion, the authors state that 3D perception places more emphasis on the view containing more information. They define weighting factors for the quality of each view according to the local information content to find out which view contains more information. Furthermore, quality score of each region is modulated based on the Binocular Just Noticeable Difference (BJND) [27] to take into account the sensitivity of HVS.

In [28], Wang et al. presented a quality assessment index for stereoscopic images based on 3D gradient magnitude. A 3D volume/data, constructed from stereoscopic image pair to account for depth perception under different disparity spaces, is used to compute the intensity differences over the spatial positions and the disparity ranges. Using three different kernels (horizontal, vertical and viewpoint directions), they calculate the 3D gradient magnitude of the distorted and original 3D volumes created. Combining both gradients, they obtain the 3D gradient magnitude similarity for each volume point, with the final quality score being an average of the 3D-GMS scores of all points in the 3D volume.

In 2015, F. Battisti et al. proposed a new metric to assess 3D video, the 3D Synthesized View Image Quality Metric (3DSwIM) [29]. It is dedicated to artifact detection in DIBR-synthesized view-points and compares statistical features of wavelet subbands for two input images: the original image and the DIBR-synthesized image. The authors included a registration step before the comparison so that best matching blocks are always compared which prevents shifting blocks from degrading the overall quality score of the image. This means that depending on the warping strategy, objects may be shifted in the synthesized frame and still guarantee a good visual quality. Also, considering humans are more sensitive to impairments affecting humans in a video, a skin-detector was introduced. This step weighs the final quality score so that distorted blocks containing "skin-samples/pixels" are penalized. This method was compared to [14], [21] and [22] and outperformed them, achieving a PLCC of 76.17%, compared to 0.49, 0.54 and 0.48, respectively. This method will be reviewed in detail in section 2.3 because it was used in this work.

2.2.2 Reduced-Reference

Reduced-reference methods often analyse features extracted from the original media. These methods, despite not requiring the availability of the reference data, need an auxiliary channel to transmit the extracted features/parameters.

Hewage and Martini presented in [30] and [31] similar methods. They evaluate the quality of a 3D video using the extracted edge information of color plus depth maps. Edges or contours of the depth can represent different depth levels and thus can be used for measuring structural degradations. Plus, they are also coincident with the corresponding color image object boundaries and both can be compared to obtain a quality index (structural degradation) for the corresponding color image sequence. The algorithm was tested on a lossy network with different packet loss rates (PLR), with sequences encoded in the H.264/AVC texture-plus-depth format.

In 2012, Nur and Akar proposed a metric [32] which compares the bilateral-filtered original depth map and the bilateral-filtered compressed depth map since depth levels of the depth map sequences have great influence on the depth perception of users. VQM [19] is used for comparing the depth maps because it correlates well with the HVS.

In [33], Malekmohamadi et al. proposed a method that measures contrast from gray level co-occurrence matrices (GLCM) [34, 35] for both color and depth which correspond to the spatial information. The metric also extract information from edge properties of the 3D reference video and send it through an auxiliary channel. Other feature present in this algorithm is that color and depth sections have different weights which can maximize the performance in some cases (for specific values).

RR are the least explored methods, even though they could be a good alternative to FR methods by saving resources, and provide better results than NR methods that usually have less information to predict a quality score.

2.2.3 No-Reference

Solh et al. proposed in [36] the NR version of [21]. Similarly to the FR method, the authors derive a no-reference ideal depth map estimated from the available colored images information. From this map, temporal outliers, spatial outliers and temporal inconsistencies are calculated to be combined and provide an objective score. The proposed algorithm achieves a PLCC of 0.916 when correlating subjective differential mean opinion score (DMOS) with the method's score. The algorithm is close in performance to its FR version [21].

In [37], Mittal et al. presented an algorithm that assesses the comfort associated with viewing stereoscopic image and video. The proposed metric extracts statistical

features from disparity and disparity gradient maps as well as indicators of spatial activity from images. In particular for videos, the measure utilizes these spatial features along with motion compensated disparity differences to predict quality.

Feitor et al. proposed in [38] a packet-layer quality assessment of stereoscopic video subject to packet loss. The presented metric estimates the size of the lost frames, frame type (I, P or B) and PLR (based on packet headers information), which is used as a model parameter to predict their objective quality, measured as the Structural Similarity Index Metric (SSIM) [18].

In [39] Soares et al. proposed a no-reference based on an artificial neural networks (ANN) approach to estimate the objective quality of video-plus-depth streams subject to packet loss in depth data. The algorithm parses the compressed bitstreams and extracts a maximum of seven packet-layer parameters from packet headers up to the network abstraction layer (NAL). These parameters are then processed over a pre-defined time window to train the ANN and predict objective quality given as a prediction of the SSIM. The authors also aimed at a low complexity model to reduce overhead and facilitate practical implementations.

Han et al. proposed in [40] the No-reference objective Video Quality Metric. This method was developed for real-time 3D video quality assessment since it has no need of processing video details. The algorithm considers the correlation between network packet loss and perceptual video quality (relying on encoding settings, if available) for different bit-rate side-by-side stereoscopic video sequences. The results showed increases up to 23% in terms of accuracy when compared to SSIM and Video Quality Metric.

The work described in this thesis is limited to the study of NR quality measures. The low complexity and good results obtained in similar works motivated this choice, as well as the need of an adaptable and *evolutionary* (meaning the developed model can be updated) metric.

This section provided a short survey of the most recent methods for video quality assessment in the literature. As demonstrated, each methodology has its own specificity and scope of application, which makes some methods more suitable for a particular situation. Having this in mind, the author chose two FR methods, 3DSwIM and SSIM, to serve as a reference against which the NR methods to be developed will be compared, according to the requirements of this work. The next section provides more information about 3DSwIM and SSIM.

2.3 Reference Media-layer Metrics

The previous section strengthens the idea that the large number of methods reported in the literature makes the task of comparing two different 3D quality metrics hard. Furthermore, it is tough to identify a certain method as a *main* reference due to the variety of factors involved. Still, there are some methods that are typically used as references. One of the most well known and used method is the Structural Similarity Index (SSIM) [18]. The HVS is capable of extracting structural information from visual scenes. By measuring the similarity between a reference stimulus and the stimulus which quality one wants to measure it is possible to derive a good approximation of the perceptual image quality. The SSIM compares the reference and distorted image to evaluate the image similarity by computing three components: luminance $l(x, y)$, contrast $c(x, y)$and structure $s(x, y)$, defined as:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \quad c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \quad s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{2.3}$$

where x and y are the reference and the distorted image luminance pixel values, respectively. μ, σ and σ_{xy} represent their mean, standard deviation and covariance and C_1, C_2 and C_3 are constants. Combining these three components, it is possible to obtain an overall similarity measure:

$$SSIM(x,y) = [l(x,y)]^{\alpha}.[c(x,y)]^{\beta}.[s(x,y)]^{\gamma}, \quad \{\alpha,\beta,\gamma\} > 0 \tag{2.4}$$

$$\begin{cases} \alpha = \beta = \gamma = 1 \\ C_3 = C_2/2 \end{cases} \Rightarrow SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2.5}$$

The SSIM assumes values between 0 and 1 with 1 being the maximum quality (i.e, there is a full similarity) of the distorted image. This method averages local scores obtained from local Gaussian circular-symmetric windows that move pixel-by-pixel over the entire image which results in an overall score. In this thesis, the obtained SSIM score is the average score of each luminance frame of the sequence.

More recently, F. Battisti et al. proposed the 3DSwIM [29]. Despite being a new method and not truly yet tested and compared with other quality metrics, it showed promising results. 3DSwIM algorithm compares statistical features of wavelet subbands of the original and the synthesized image/sequence. Firstly, the image of size m x n pixels is divided into $B = B_n \text{x} B_m$ non-overlapping blocks,

with B_m and B_n being set as metric parameters. To account for the often disparity errors during synthesis process, a registration step is performed to guarantee the best block matching between the two images. An Exhaustive-Search-like algorithm [41] (in the horizontal direction only) is used with a search window of size W pixels. This parameter is changeable and has a great impact on computational cost: larger windows lead to an increased computational cost. Furthermore, the authors believe human beings are more sensitive to artifacts affecting areas with human skin (i.e., face, hands, legs,..) [42] and assume these affected areas have a higher impact on the perceived quality. For that reason, they include a skin-detection feature, which lowers the score in case artifacts affect skin-pixels. 3DSwIM begins with block distortion d_b calculation:

$$d_b = \max(|F_{O_b} - F_{S_b}|) \tag{2.6}$$

where F_{O_b} and F_{S_b} represent the distribution function of original and synthesized view, respectively. The overall normalized image distortion is then obtained:

$$d = \frac{1}{D_0} \sum_{b=1}^{B} w_{skin} \cdot d_b \tag{2.7}$$

where D_o is a normalization constant and w_{skin} is the weight of distortions present in skin-pixels. Finally, the image quality score s is computed:

$$s = \frac{1}{1+d} \tag{2.8}$$

Like SSIM, 3DSwIM metric score s ranges from 0 to 1 with 1 meaning no distortions ($s = 1$ and distortion $d = 0$) and 0 the minimum quality ($s = 0$ and distortion $d \to \infty$). This method obtained very good results when comparing to some of the available image quality metrics, according to the study in [29].

The referred metrics above were used in this thesis as targets for predicting 3D video quality. SSIM is simple, fast, with wide range of applications and it is still very used among the scientific community, which makes it a good choice to compare results with other works. 3DSwIM is a recent method dedicated to 3D synthesized video and, just as SSIM, it is a FR method but with higher computational cost.

Chapter 3

H.265/HEVC standard and Gilbert-Elliot model

The new coding standard High Efficiency Video Coding (HEVC) was chosen as it is the most recent and best performing encoder available, with very high encoding efficiency, doubling the compression efficiency compared to its predecessor H.264/AVC [43] at the same level of video quality. The first version was released in January 2013, being standardized in April 2013. The obtained compression efficiency was seen by experts as a great opportunity for 3D video, with new possibilities for 3D video transmission. This chapter provides information on general HEVC concepts about the reference software. An explanation of the Gilbert-Elliot model is also given in this chapter.

3.1 Frame Types and GOP Structure

A sequence of images (frames) processed electronically into an analog or digital format and displayed on a screen with sufficient rapidity (frame-rate) creates the illusion of motion and continuity, i.e., video. Typical frame-rates are 25 frames-per-second (fps) and 30fps. Encoders like HEVC explore the spatial redundancy of a frame or temporal redundancy of consecutive frames as well as the visual irrelevancy of large amounts of the original video data to compress the video in order to obtain an equivalent reconstructed video quality but with smaller size. Spatial redundancy is the repetition of information which exists within the same frame, i.e., a frame contains pixels which have near similar values to their adjacent neighbors. In the literature, it is called intra-frame redundancy. The DCT-based plus quantization scheme used in JPEG encoder [44] is an intra-frame compression technique that reduces redundancy and discards visually irrelevant data to achieve large compression ratios. Moreover, contiguous frames often have information in common that

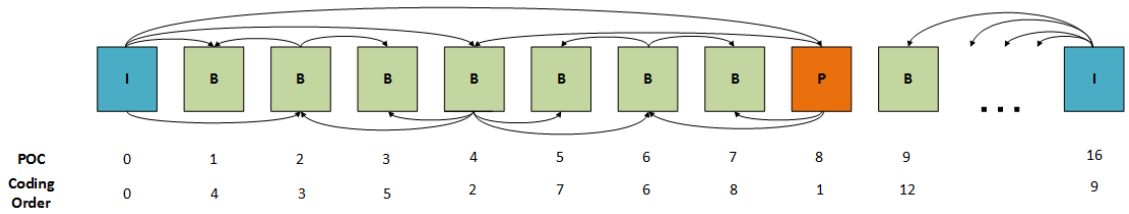


Figure 3.1: Example of a GOP with GOP size 8 and intra-period 16.

can be used to predict parts of frames. When consecutive frames are correlated, a reference frame is signaled and motion vectors (which indicate the displacement of a pixel-block between frames) are computed. Depending on a frame's reference to other frames, they can be classified into 3 types: I-frames when they are coded without reference to other frames and can be decoded without decoding any previous pictures; P-frames are coded with at least one reference to a preceding reference frame; and finally B-frames when a frame is coded using information from past and future reference frames. I-frames are typically the least compressed frames and the reference frames to most of the future frames. P-frames combine information of a reference frame and independent coding of prediction residues. B-frames are typically the most compressed ones and the most dependent on information from other reference frames.

A set of specific consecutive frames is called group of pictures (GOP). The typical definition of GOP consists on the distance between two I-frames. However, HEVC defines GOP as the number of B frames plus one: $GOP = B_{frames} + 1$ and the period of I-frames is defined as the intra-period. As an example, consider a GOP size of 8 and intra-period length 16, illustrated in figure 3.1. The encoder begins with the coding of the first frame, with picture order count (POC) 0 - which is the order of appearance in the video - and jumps to frame with POC 8, being this a P-frame with reference to the I-frame previously coded. The between frames (B-frames) are then coded until the frame with POC 7 is reached. The next frame to encode is the one with POC 16, which is the next I-frame according to the intra-period parameter. B-frames with POC 9-15 are coded with reference to frame with POC 16, followed by the coding of B-frames with POC 17-24 and jumping again to the I-frame with POC 32.

HEVC allows for two types of GOP structure depending on the type of I-frame: if the I-frames are set to *Instantaneous Decoder Refresh* (IDR), no subsequent picture in the bitstream will require reference to pictures prior to the picture that it contains in order to be decoded, i.e., it is strictly confined inside the GOP; if the I-frames are set to *Clean Random Access* (CRA), the GOP structure is classified as *open GOP*, where the I-frame can be a reference for the last B-frames of the previous GOP. The structure and length of a GOP plays an important role in balancing error

propagation/prevention and compression. At the cost of compression efficiency, which can be softened by fixing a lower quantization parameter (QP) or bitrate, the closed GOP offers a better resilience to error propagation. On the other hand, an open GOP improves the compression ratio but it is more vulnerable to error propagation. In this thesis, closed GOP structure was chosen to prevent inter-GOP error propagation.

3.2 Slices and Coding Tree Blocks

In the previous standard H.264/AVC, macroblocks of fixed size 16x16 were the elementary image division unit. In HEVC the coding tree unit (CTU), which can be larger than a traditional macroblock, replaces the macroblocks. The CTU consists of a luma coding tree block (CTB) of size LxL samples and the corresponding chroma CTBs, with $L/2$x$L/2$ samples of each of the two chroma components. The value of L may be equal to 16, 32, or 64 samples. Each CTB can be split recursively in a quad-tree structure, down to 8x8, as illustrated in figure 3.2. The "end" (or *leaf*) of each CTB is called coding unit (CU) and is the basic unit of coding in HEVC.

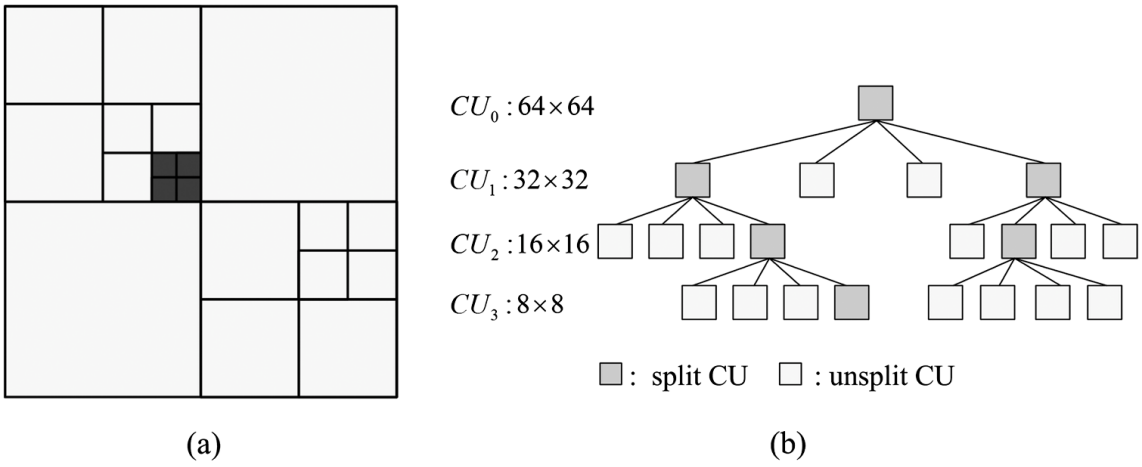


Figure 3.2: Example of a quad-tree coding structure in HEVC [45]. (a) The quad-tree based coding structure of a large coding unit (LCU) of size 64x64 (CU_0). The black 8x8 CU is encoded as four 4x4 prediction units (PU) type. (b) The corresponding quad-tree representation of the LCU. A gray square indicates when a CU is split into smaller CUs whether a white square indicates the CU is not further divided.

CTBs are then arranged into groups forming a slice. A picture can be split up into any number of slices, or the whole picture can be just one slice. Slices play an important role in error resilience since they are data structures that can be

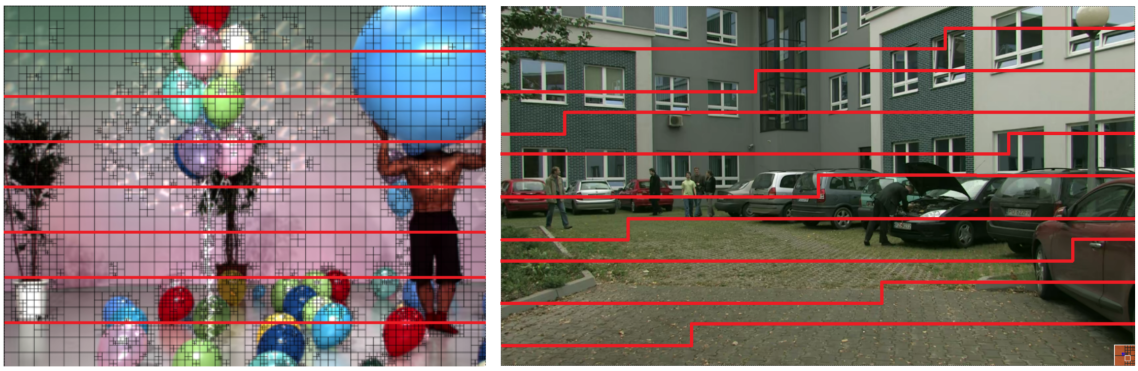


Figure 3.3: Two frames divided into slices (red lines). On the left, not only the the slices are highlighted, but also the CU division is also represented. The Elecard HEVC Analyzer free software was used to extract the images.

decoded independently from other slices of the same picture, in terms of entropy coding, signal prediction, and residual signal reconstruction meaning the effects of an error are restricted to that slice area. Moreover, slices have the purpose of resynchronization and concealment in the event of information loss. The more slices an image is split, the more resilient to error prone channels that frame is improving the overall quality of the video at the cost of compression efficiency (inter-frame prediction is now mostly limited to the slice area) and more overhead during packetization. Figure 3.3 shows an example of two frames divided into equally sized slices. In the case of packetized transmission, which is the scope of this thesis, it is important to set these parameters according to the network characteristics in order to minimize the loss of information. Each slice is then packetized into a network abstraction layer (NAL) unit. For further information, [46–49] provide more information of the H.265/HEVC syntax and available tools.

3.3 HM Software and Packet Scheme

The reference software for HEVC - called HM (**H**EVC Test **M**odel)-, from Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T Video Coding Experts Group (ITU-T Q.6/SG 16) and ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11), was used in this thesis as the H.265/HEVC codec. It is very popular among the scientific multimedia community and was designed mainly for research purposes. Despite the existence of an HEVC reference software for 3D Video (Multiview (MV)- and 3D-HEVC), the aim of this thesis is to analyze the effects of packet loss in the texture and depth bitstreams, leading to the encoding of texture and depth streams separately, i.e., as two independent 2D video streams.

Similarly to H.264/AVC, HEVC uses a NAL unit based bitstream. The NAL was designed to address the need for flexibility and customizability to handle efficiently

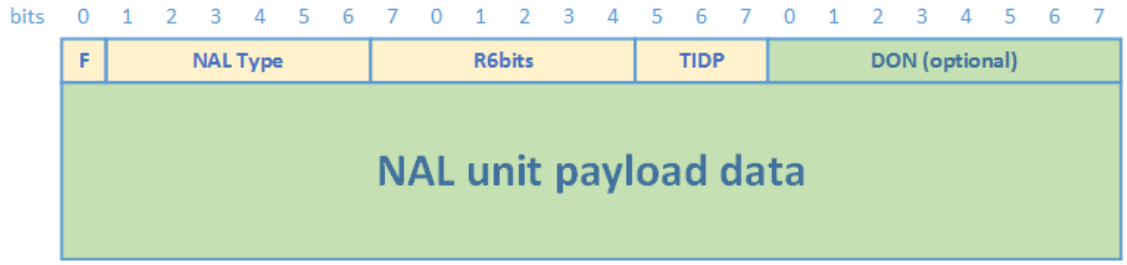


Figure 3.4: Single NAL unit structure of HEVC, according to draft in the footnote and [49]. The region in yellow represents the NAL unit header and the green region is the NAL unit payload.

the variety of existent (and future) applications and networks. The NAL facilitates the ability to map HEVC video coding layer (VCL) data, i.e., the video payload like coded slices, to transport layers formats by providing header information appropriate for communications by these transport layers such as RTP (for any kind of real-time wire-line and wireless Internet services) or MPEG-2 systems for broadcasting services. A coded bitstream is partitioned into NAL units that consist of a NAL unit header followed by the NAL unit payload. Figure 3.4 shows the format of a HEVC NAL unit.

The NAL unit header was extended to 2 bytes instead of 1 byte used in H.264, with the anticipation that this design is sufficient to support the HEVC scalable and 3D video coding extensions. The NAL unit header was designed to co-serve as part of the packet header in RTP based packet networks, such as the one used in this work. The first bit of the NAL unit header is the $forbiddenzero$ and must always be zero. The following six bits determine the NAL unit type (there are 64 types of NAL units [49], divided into VCL and non-VCL, which carry metadata typically belonging to more than one coded picture), followed by 6 bits (R6bits) for the element $reservedzero6bits$ (it is expected this element carries some form of layer identification information in future extensions) and 3 bits for $temporalidplus1$ (TIDP), which allows temporal scalability. In the interleaved packetization mode, the transmission order of NAL units is allowed to differ from the decoding order of the NAL units. Decoding order number (DON) is a field, that may be present or not, in the payload structure or a derived variable that indicates the NAL unit decoding order.

The encoder was set so that each VCL NAL unit contains all the information of just one slice. In order to simulate realistic transmission schemes over IP-networks, each NAL unit is packetized into a variable-length RTP packet with a maximum-transmission-unit (MTU) size of 1500 bytes (the current draft[1] defines the RTP payload format as the packetizing format to use with this video codec). If a NAL

[1]https://tools.ietf.org/pdf/draft-ietf-payload-rtp-h265-13.pdf

unit has over 1500 bytes in payload, it is segmented into IP-datagrams and losing at least one of them results in the loss of the entire slice.

3.4 Error Concealment in the HM decoder

The purpose of a decoder is to translate information present in a stream into its original signal. When a decoder starts decoding a stream, it expects the data to be arranged in a pre-determined way, established by protocols or norms. In the case of errors in the bitstream during transmission, the decoder might not be able to correctly decode the stream. To minimize the risk of failing while decoding a stream, the decoder must be robust, with the ability to recover some bitstream errors, and be able to hide effects from these errors, i.e. be provided with error concealment functions. The most common and simple error concealment technique is copying the slice from the previously corrected received and decoded frame. This technique is called frame-copy, but more complex and robust algorithms can be found in the literature [50–54]. The implementation of these algorithms increases the complexity of the decoder and a trade-off between the algorithm complexity and concealment quality needs to be considered. Moreover, error concealment techniques are not standardized, which means different decoders may adopt its own error concealment method. This means that the effect of a packet loss or error during transmission in video quality depends on the concealment algorithm present in the decoder. The author of this thesis, due to the lack of reference decoders with error concealment techniques adequate to the context of this work, worked on an error concealment technique in the reference software HTM for multiview 3D video (3D-HEVC). The algorithm was based on the frame copy with some modifications: after detecting the area affected by the error, temporal information from previous and leading frames of the same view as well as spatial information from other views were used to estimate an average pixel value for the affected area. During this work, a full-operational and robust decoder for HM was released [55] and was adopted for the rest of this work.

The error concealment technique applied in this decoder adopts intra-spatial concealment for IDR-frames, whereas B- and P-frames are concealed with temporal concealment and motion compensation. Figure 3.5 shows an example of a correctly decoded frame and the correspondent concealed frame using the mentioned decoder, with 20% packet loss rate in both texture and depth streams.

IDR-frames lost slices are extrapolated from neighbour received and correctly decoded slices. This method fails when frames are coded with large slices or several slices of the same frame are lost. If the entire frame is lost, a copy of the previously corrected frame is used before the decoder buffer is refreshed. For P- and B-frames, temporal information is extracted and used to conceal the lost slices. The method

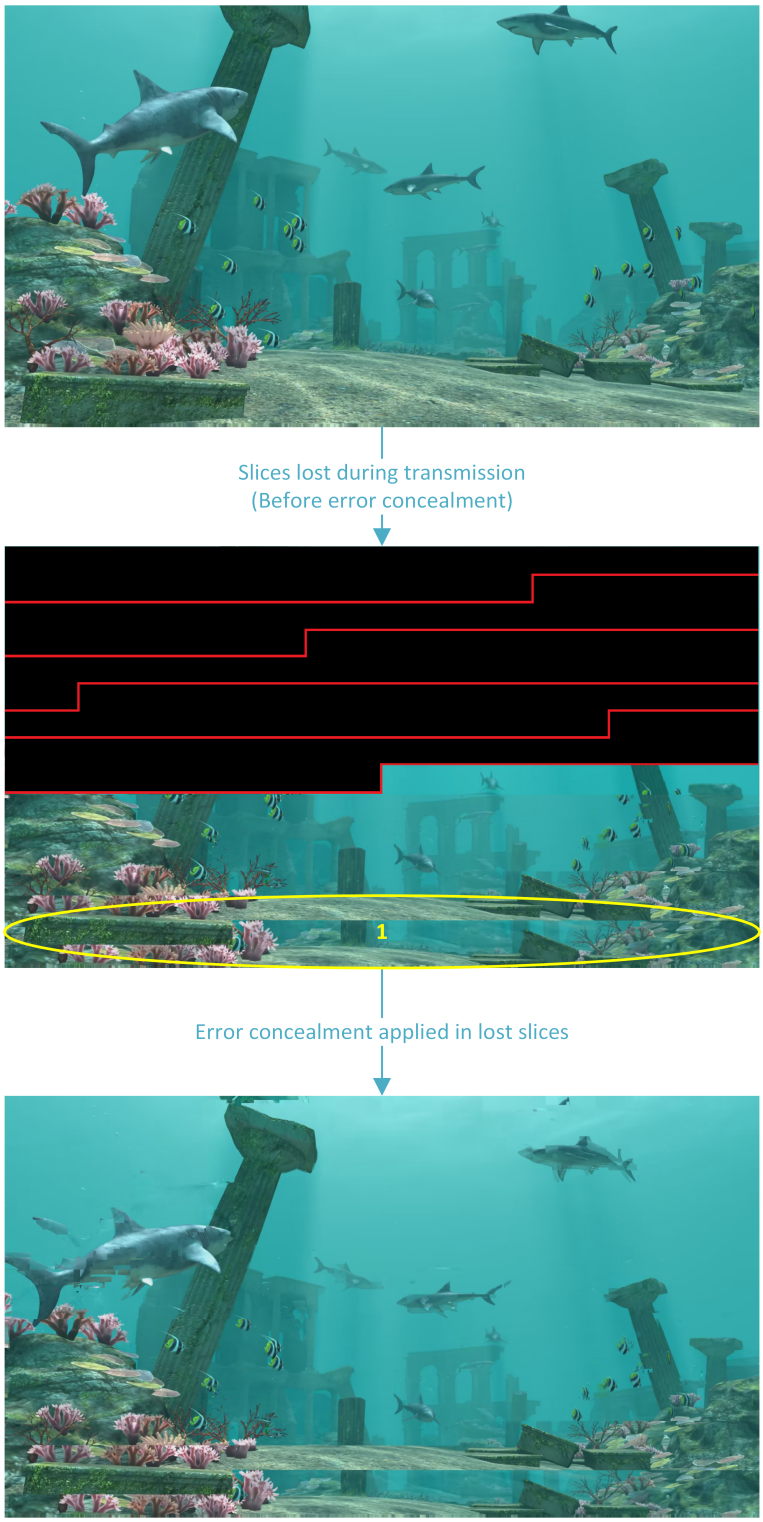


Figure 3.5: Example of a frame concealed by the decoder. On the top, the original frame. The second frame shows the lost slices (black blocks limited by the red lines) and the slices affected by error propagation (region 1, in yellow). The last picture is the concealed frame.

evaluates the overall temporal activity of the corrected received slices. If the activity is below a threshold, then a direct copy of a co-located slice from the closest reference frame is used. Otherwise a motion vector of the missing CUs is estimated from available motion information (either spatial or temporal neighbors) to find out the corresponding CUs in the nearest reference frame, concealing the missing slice. Once again, when frames are split into higher numbers of slices, the better the performance of the concealment technique is. It is worth to mention this method does not prevent error propagation in GOPs: when reference frames are affected, the blockiness effect becomes more and more visible.

Since this work is oriented towards the analyzes of packet loss, a *Matlab* script described in Annex A associates each NAL unit with a RTP packet so that a ratio of packet loss is computed. Another feature present in this decoder, is the option of considering losses in the first frame by flagging $--firstIsLost$ variable. This is a useful tool to define if the first frame to be decoded (an I-frame) is lossless or not. For simplicity reasons, this option was ignored and the first frame is always error free.

3.5 Gilbert-Elliot Model

This section describes the model used to simulate losses in the bitstreams. The impact of packet loss on real-time video streaming services can be modeled mathematically, using real measurements and traces of traffic and loss patterns. Stochastic models such as discrete-time Markov chain models can be used to generate error patterns similar to those measured previously in real time scenarios, resulting in a good approximation for offline simulations.

This work adopted the Gilbert-Elliot model [56] to generate packet loss events. It is a stochastic packet loss model based on a two-state Markov process (figure 3.6). It is a simple model, characterized by a good state ($X = 0$) and a bad state ($X = 1$), with transition probabilities p and q between the two states as a response to one of two possible events: (a) a successful arrival of a packet, making the system change to or remain in the good state; and (b) where a packet loss is detected and the system responds moving to bad state, if it was in the good state, or remaining in bad state.

By saving only the previous state, the probability that the next expected packet will be lost ($P(X_{i+1} = 1)$) depends only on the current state of the system, X_i. The Gilbert-Elliot model has an interesting feature that is the ability to verify the dependence between consecutive losses, making it a suitable model for network transmission scenarios where errors usually occur in bursts. This model is only dependent on two variables to characterize the transmission network: *Packet Loss*

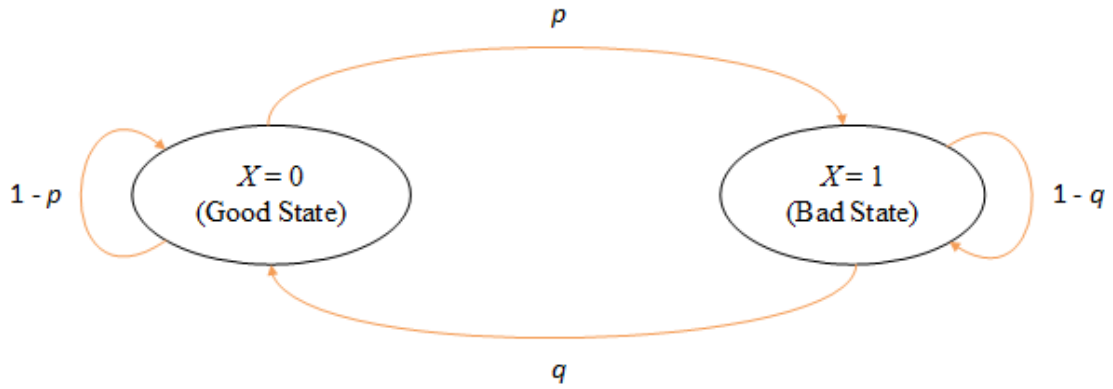


Figure 3.6: Two-state Markov process for the Gilbert-Elliot model.

Rate (PLR) and *Mean Burst Length* (MBL). From them and [57], it is possible to derive the conditional transition probabilities:

$$p = P(X_{i+1} = 1 | X_i = 0) = \left[MBL \cdot \left(\frac{1}{PLR} - 1 \right) \right]^{-1} \tag{3.1}$$

$$q = P(X_{i+1} = 0 | X_i = 1) = \frac{1}{MBL} \tag{3.2}$$

knowing that:

$$\frac{1}{PLR} > 1 + \frac{1}{MBL} \quad , \quad 0 < PLR < 1 \quad , \quad MBL \geq 1 \tag{3.3}$$

This model is implemented as a *Matlab* script listed in annex B. Several trace files were generated, with different combinations of PLR and MBL. Typical values of PLR range from 1% to 20% in non-linear steps [in this work, 1%, 5%, 10%, 15% and 20%]: if provided error concealment is used, losses under 1% are almost undetectable and 20% loss leads to severe degradation affecting video quality in a way it is difficult to distinguish anything. For this reason and the rarity of PLR above 20% our study considered only PLR values below or equal to 20%. MBL depends on the type of network considered: for wired connections usually MBL is lower than a wireless network. As mentioned before, to prevent the task of concealing an entire frame if all slices are lost, a *maximum burst length* of 7 or 9, depending if it is a low resolution sequence or HD, was set. Thus, MBL values range from 3 to 5 for low resolution sequences and 3 to 6 for HD sequences.

Chapter 4

Packet-layer models for synthesized view quality assessment and Results

This chapter describes the proposed method for assessing 3D video quality, in the texture-plus-depth format, based on neural networks (NN) and statistical packet transmission network parameters, with the respective obtained results. Two different methodologies were adopted considering the type of validation used in the NN.

4.1 Objectives and Procedures

The objective of the proposed method is to estimate the overall quality of a synthesized view averaged over a temporal window using an objective quality metric as a reference (e.g. SSIM, 3DSwIM, PSNR...). The quality estimation is computed according to a set of parameters extracted from packet headers. The learning process is relatively fast, which allows the re-training of the model with new samples, increasing the amount of information used to estimate video quality. And in this type of algorithms, the more *valid* [1] input samples are available, the better results will be. As soon as this model is obtained, it is ready to be deployed somewhere in the network channel. Network probes with the task of *packet sniffing* provide a good option for such methods. Figure 4.1 describes an example of probes deployed in three different nodes: in a switch, between the service provider and the client, and in two routers at client's home. Once the service or network providers obtain the estimated quality of the video being delivered to the client, they are capable of detecting failures, vulnerabilities or congested network which degrade the users'

[1]The word valid is stressed here to highlight the importance of using trustworthy samples: having huge amounts of samples does not imply having better results if the samples are not coherent.

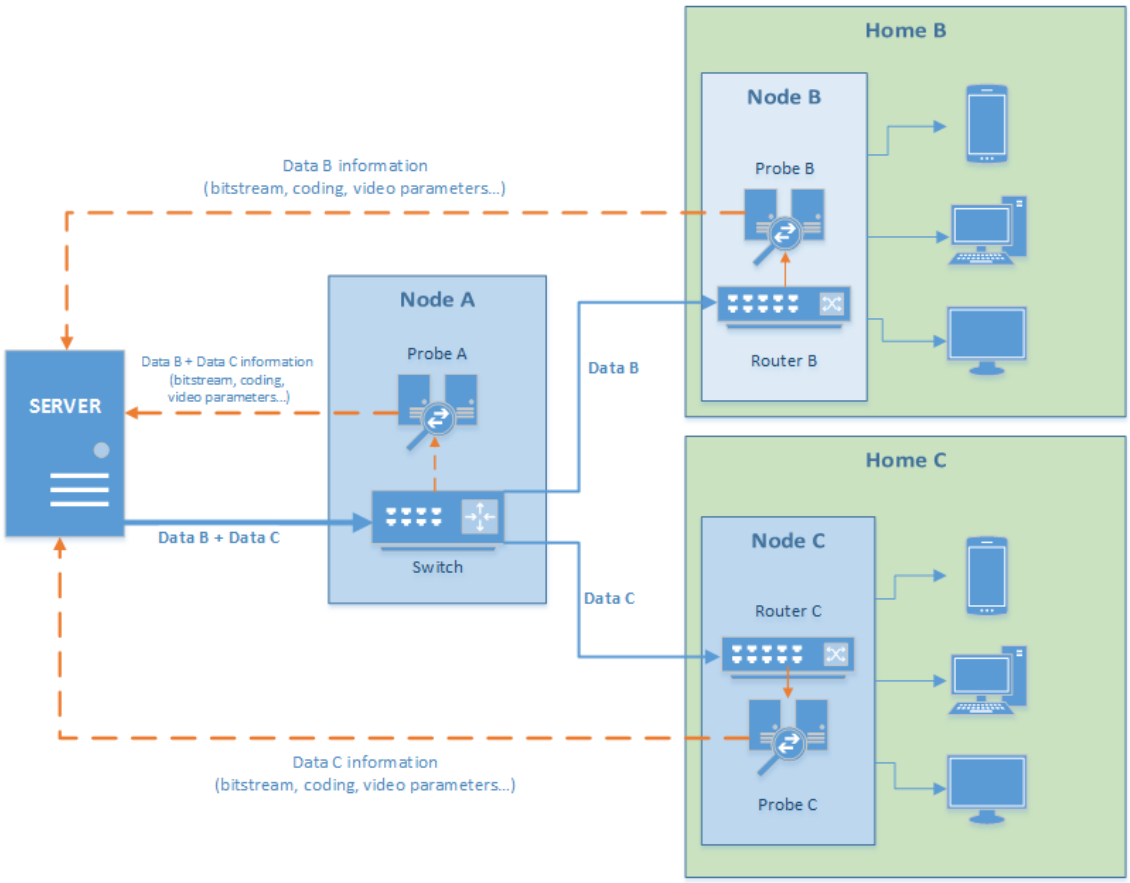


Figure 4.1: Example of a monitoring system with 3 probes *sniffing* packets to report to the service provider.

QoE. To prevent this from happening, they are now in a position where they can act according to circumstances: either by changing video encoding configuration or adjusting network and transport conditions. A well known example is YouTube way of preventing their videos from stopping during loading: if the connection is slow or facing any type of issue that causes the loading to slow down, YouTube streaming service automatically lowers the videos' resolution in order to download it without interruptions. This is only possible because the company providing the service (YouTube in this case) know their client is facing a problem that can be attenuated without costs and worries for the client. Furthermore, with these packet-layer indicators stored and compiled, it is possible to have a record on that specific point of the network, allowing the network engineers to compare results and identify abnormal activities in the transport network.

For 3D video, when texture and depth are encoded separately and packed into different NAL units, it is desirable to transmit them in different channels. This way, if a packet is lost, only one of the video components, i.e. texture or depth, is affected. Although, there are other possible network configurations: (a) both streams can be transmitted through the same network with different degree of importance;

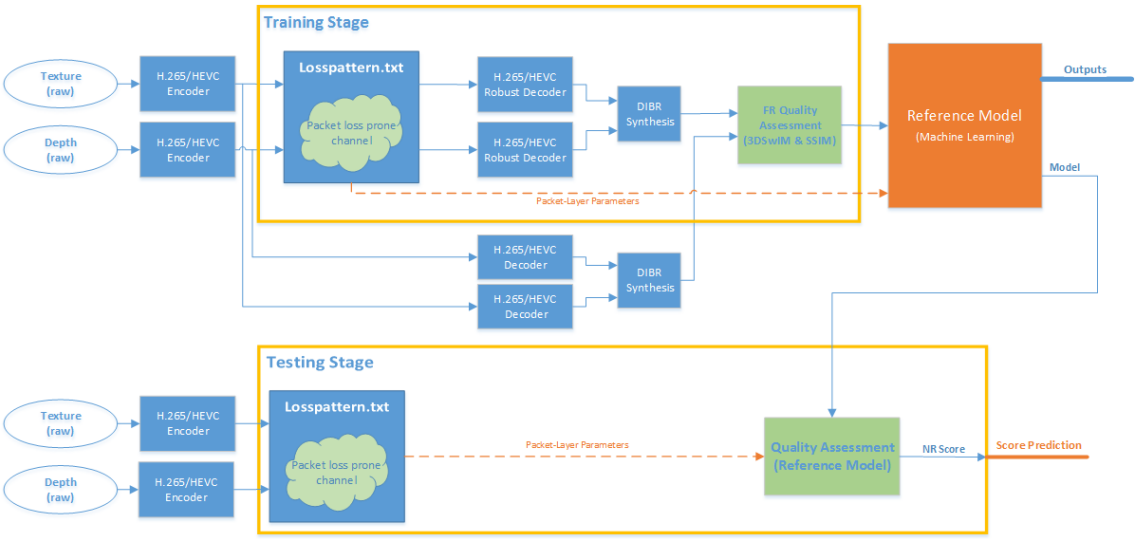


Figure 4.2: Experimental setup for texture and depth loss approach.

(b) one stream (e.g. texture) might have a higher priority network than an auxiliary channel that transmits the depth data to avoid losing 2D color video. In reality, both streams are subject to packet and data loss, despite their importance or configuration, which highlights the importance of monitoring both streams - to increase as much as possible the model's accuracy.

The experiment setup is shown in figure 4.2. It assumes an independent encoding and transmission of texture and depth maps where both streams are subject to packet loss using error events generated by the Gilbert Elliot model. The bitstreams are then decoded with the error concealment and used to synthesize a view with an appropriate baseline. In order to understand the impact of different packet loss rates in texture and depth, multiple combinations were considered. Finally, the SSIM and 3DSwIM scores of the distorted synthesized view are computed with respect to the reference view. These objective FR metrics are then used as ground-truth (*target*) values for the training and validation of the proposed model. The inputs are a set of parameters extracted from packet headers. The use of NN requires hundreds of simulations in order to obtain an accurate and generalized model.

Most of the mentioned tasks are very time-consuming and require substantial processing resources. To accelerate the decoding and synthesis process, a clustered computer was necessary. To synthesize views, the reference software VSRS 3.5 [58] was used. All videos were encoded with HM version 16.0 and decoded with an altered version of HM v.12.1, described in section 3.4.

4.2 Video Dataset and Encoding parameters

The 3D video sequences used in this experiment setup are entitled: Balloons, Kendo, Newspaper, PoznanCarPark and Champagne Tower. Each video has different characteristics which influence the choice of encoding parameters. Even the texture and depth map of a sequence should be encoded with different setting parameters, either to meet network or quality requirements. For the three videos with resolution of 1024x768: texture was encoded with fixed GOP size of 8 frames, intra-period of size 16 frames with 8 slices per frame with fixed number of CTBs and a maximum CU size of 32x32 with a maximum limit of three levels of partition; depth is coded with fixed GOP size of 8 frames and intra-period of length 32 frames. Champagne and PoznanCarPark have higher resolution leading to small changes in the configuration file: number of slices per frame was set to 10, maximum CU size was increased to 64x64 and maximum partition depth was also increased to 4 for both texture and depth. The rest of the parameters is the same depending if its depth or texture. Adding to these, QP was set to 28 and 30 for texture and depth, respectively, a Z-search mode with 64 pixels of range was used and sample adaptive offset (SAO) was activated for all videos. Table 4.1 summarizes the most relevant settings applied.

3D Video		**GOP Size**	**GOP structure**	**Intra Period**	**LCU size**	**QP**	**Slices per frame**	**V+D Bitrate (% Depth)**
Balloons (1024x768)	Texture	8	B-B-B-B- -B-B-B-P-B- ...	16	32	28	8	1.1 Mb/s(26%)
	Depth			32		30	8	
Kendo (1024x768)	Texture	8	B-B-B-B- -B-B-B-P-B-...	16	32	28	8	1.1 Mb/s(19%)
	Depth			32		30	8	
Newspaper (1024x768)	Texture	8	B-B-B-B- -B-B-B-P-B- ...	16	32	28	8	1.1 Mb/s(18%)
	Depth			32		30	8	
Champagne (1280x960)	Texture	8	B-B-B-B- -B-B-B-P-B- ...	16	64	28	10	1.1 Mb/s(13%)
	Depth			32		30	10	
PoznanCarPark (1920x1088)	Texture	8	B-B-B-B- -B-B-B-P-B- ...	16	64	28	10	3.2 Mb/s(29%)
	Depth			32		30	10	

Table 4.1: Encoder setting parameters of the 5 videos used.

4.3 Neural Network based models

Neural Networks appear as a first candidate to address engineering tasks such as the one discussed in this thesis. A NN is a model that, according to a mathematical function, computes an output provided a set of inputs. The advantage of this model is that it is trainable, i.e., the more it is trained, the better the model will fit the problem. [39, 59–64] adopted similar approaches with the goal of predicting video quality scores.

The proposed model is based on a two-layer feedforward network with sigmoid hidden neurons and linear output neurons. This methodology can fit multi-dimensional mapping problems with good performance indicators, as long as the data provided is consistent and the number of hidden neurons is sufficient. Figure 4.3 shows an example of this type of NN with 3 input parameters, 2 hidden nodes and a single output.

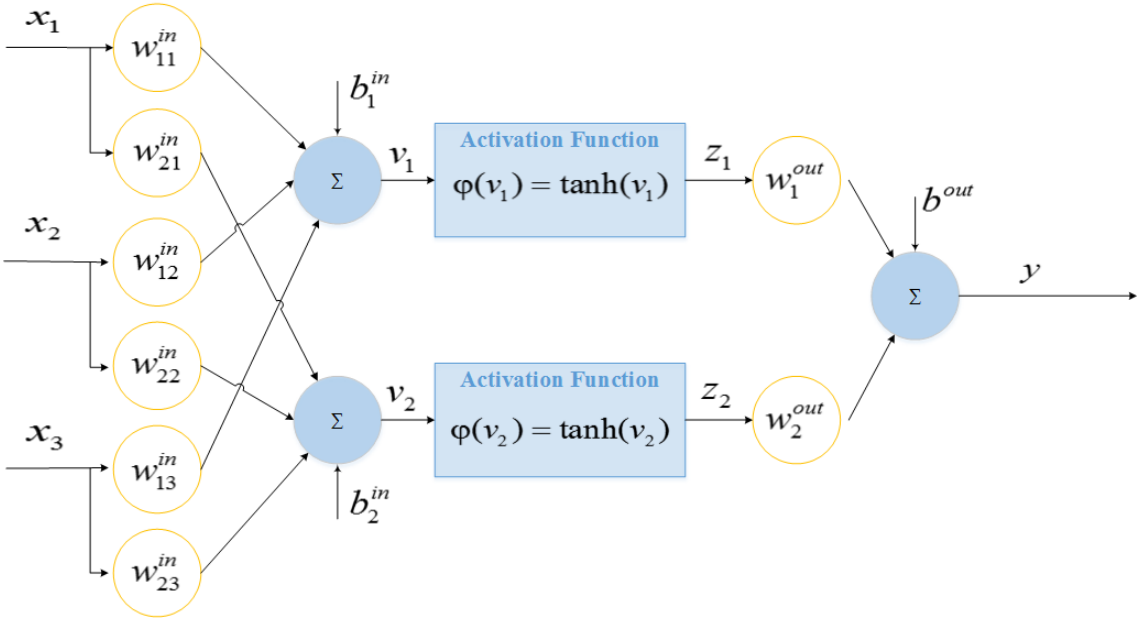


Figure 4.3: Two-layer ANN with N=3 inputs and H=2 hidden nodes.

The activation functions of the first (hidden) layer and the second (output) layer are respectively the hyperbolic-tangent (sigmoid function) and the identity function. The output of the model is described by equation 4.1 and equation 4.2:

$$y(x) = \sum_{j=1}^{H} (w_j^{out} \cdot z_j) + b^{out} \tag{4.1}$$

with

$$z_j = tanh\left(\sum_{i=1}^{N} (w_{ji}^{in} \cdot x_i) + b_j^{in}\right) \tag{4.2}$$

where N is the number of input parameters, H is the number of hidden neurons and w and b are the weights and biases tuned during training session, with w^{in} and b^{in} being the weights and bias of the first layer and w^{out} and b^{out} the weight and bias of the second layer.

Matlab® *nftool* provides an intuitive tool that helps implementing the desired method. It uses the Levenberg-Marquardt [65] back-propagation algorithm for training the neural network. Despite *nftool* is a good auxiliary tool to make some experiments, it is somehow limited when choosing some configuration parameters which

play an important role in the models' performance. In order to tune the NN with the desired settings, *nftool* allows the user to generate a script after performing a full simulation. Thus, *nftool* was used only as an initial experiment and to generate the scripts used in this work.

One more aspect worth to mention is the process of training, validation and testing. *nftool* randomly divides the set of inputs into three groups: the training set, validation set and testing set. Training set uses input samples to iteratively train the network, the validation set is used to measure network generalization and to signal the network when generalization stops improving and testing set takes samples to independently test the performance and accuracy of the model. In this thesis, two different scripts with different validation and testing methods, explained in detail in 4.3.1 and 4.3.2, were generated.

To train the NN a maximum of twelve parameters are extracted from the parsing of texture and depth bitstreams. Not only the packet loss rate is considered but also the type of slice and data size of the lost packet which lead to the following processed packet layer parameters (PLPs):

Packet Loss Rate (PLR): Ratio of texture and depth slices lost during a time interval (10 seconds in this work) of the transmission. As mentioned before, encoding configuration was set specifically so that each NAL unit contained only one slice (each frame was divided into equal number of slices). The RTP payload format allows for packetization of one or more NAL units in each RTP packet payload, which means each RTP packet contained one slice, i.e if a packet was lost, the correspondent slice was lost. In the scenario of an IP wired network, maximum transfer unit (MTU) sizes are set to roughly 1500 bytes, value considered in this work.

Size of Lost Packet (SLP): The size of the lost packet plays an important role on the video quality. If a packet with 1000 bytes of video data is lost it will have a higher impact in the video quality's degradation than a packet containing 500 bytes.

As not all packets are of equal importance, it is important to know the slice type of the lost packet. Even if the size of the packet is known, which is somehow related with the slice type of the packet, the effect of error propagation is different depending on the type of slice. Thus, each PLP is derived for each slice type, I, P or B and for texture and depth, totalling the 12 possible inputs of the NN: PLR_I^t, PLR_P^t, PLR_B^t, SLP_I^t, SLP_P^t, SLP_B^t, PLR_I^d, PLR_P^d, PLR_B^d, SLP_I^d, SLP_P^d, SLP_B^d,

To find the NN configuration that offers the best trade off between performance and computational cost, it is necessary to understand the impact of the number of hidden neurons H and inputs N on the NN results. Table 4.2 lists the packet parameters used for each group of inputs considered. To limit the influence of

the random initial values for each training session in the final results, a set of 100 iterations with H hidden nodes varying from 1 to 10, were performed for both topologies with different number of inputs. The average PLCC of each number of hidden nodes is computed as it is a good indicator of the networks' accuracy. Figure 4.4 shows the relation between the two mentioned variables: the number of inputs considered and the number of hidden neurons. For a better analysis, a close up of the results with 12 inputs and 6 texture inputs is also provided.

N	Packet Layer Parameters (PLPs)
3	$PLR_I^t \cdot PLR_P^t \cdot PLR_B^t$
6	$PLR_I^d \cdot PLR_P^d \cdot PLR_B^d \cdot SLP_I^d \cdot SLP_P^d \cdot SLP_B^d$
6	$PLR_I^t \cdot PLR_P^t \cdot PLR_B^t \cdot SLP_I^d \cdot SLP_P^t \cdot SLP_B^t$
12	$PLR_I^t \cdot PLR_P^t \cdot PLR_B^t \cdot SLP_I^t \cdot SLP_P^t \cdot SLP_B^t$ $PLR_I^d \cdot PLR_P^d \cdot PLR_B^d \cdot SLP_I^d \cdot SLP_P^d \cdot SLP_B^d$

Table 4.2: PLPs used for ANN training

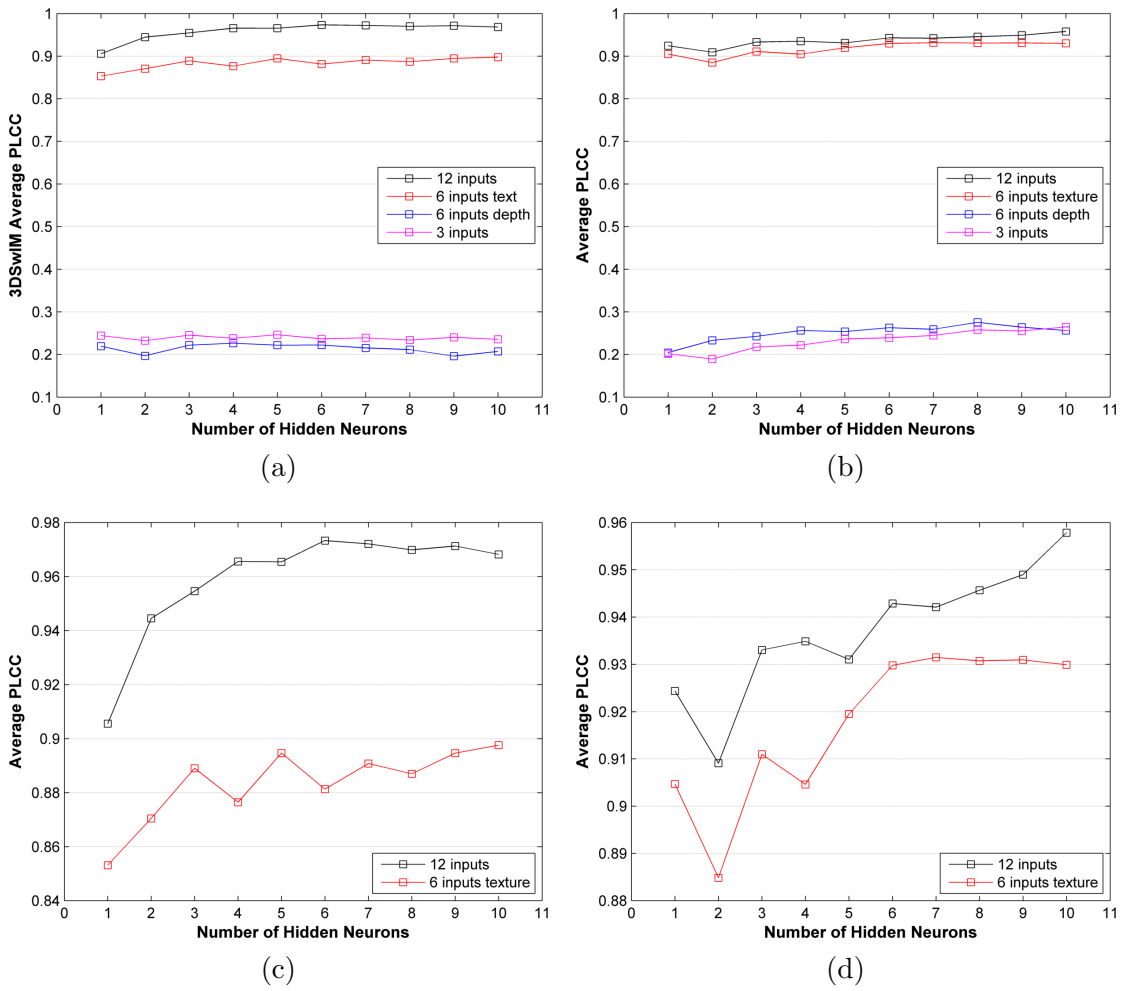


Figure 4.4: PLCC between estimated scores and real scores. 3DSwIM: all inputs in Figure 4.4a and close-up in Figure 4.4c; SSIM: all inputs in Figure 4.4b and close-up in Figure 4.4d.

For both metrics, the plots show an increase in the PLCC with the number of hidden nodes up to 6, where it stabilizes, particularly for the 3DSwIM algorithm. It is also clear the impact the type and number of inputs have in the NN performance.

When the input set is only three indicators of the packet loss rate of the texture component, the algorithm has a poor performance, with a PLCC ~ 0.24 . Even though they carry texture information, which contributes significantly to the quality degradation and might present good data to the NN, the lack of information about the size of the packet lost is enough to justify the low correlation values. With only these three inputs, it is as if the network considers every packet loss has the same impact on the video's quality, which is not true. The difference between the curve of "6 inputs texture" and "3 inputs texture" supports this idea, being, however, this difference larger than expected. Furthermore, the performance with 3 texture inputs is almost the same as the performance with 6 depth inputs. As expected, the NN accuracy is low when using only depth related inputs when assessing texture-loss videos. 3D video, in the texture-plus-depth format, quality is mostly dependent on texture, which leads to a higher impact on the overall quality if texture is affected by any kind of impairment than if the same impairment occurred on the depth component. Finally, the best performance is achieved with the use of all 12 inputs, followed relatively close by the 6 texture inputs.

The plots in figure 4.4 also give information about the number of hidden nodes needed to achieve good performance results. To avoid wasting time and processing resources, H needs to be chosen carefully. The close up plots (figure 4.4c and 4.4d) show that the more hidden nodes are used, the better the performance usually is, but the gains in performance tend to stabilize as the hidden nodes increase. To the author of this work, 6 hidden neurons and 12 input parameters should be used since it proved to be accurate enough without being too much computational costly. Although, for a complete study, for the first methodology, all the simulations were performed using all the different combinations of PLPs, whereas in the second, only the set of 6 texture and 12 inputs were used.

4.3.1 First Methodology

The first approach considers a *nftool* configuration of the neural network, with all the data samples available being provided to the network and dividing them according to the following proportions: 50% of the input samples were used to train the network; 20% of the samples were used to validate the model and the other 30% were used for testing the network's performance. This configuration is expected to be quite accurate given sufficient hidden neurons and inputs, with high PLCC values as the samples used to train and validate the NN might be similar to those used in

the testing phase.

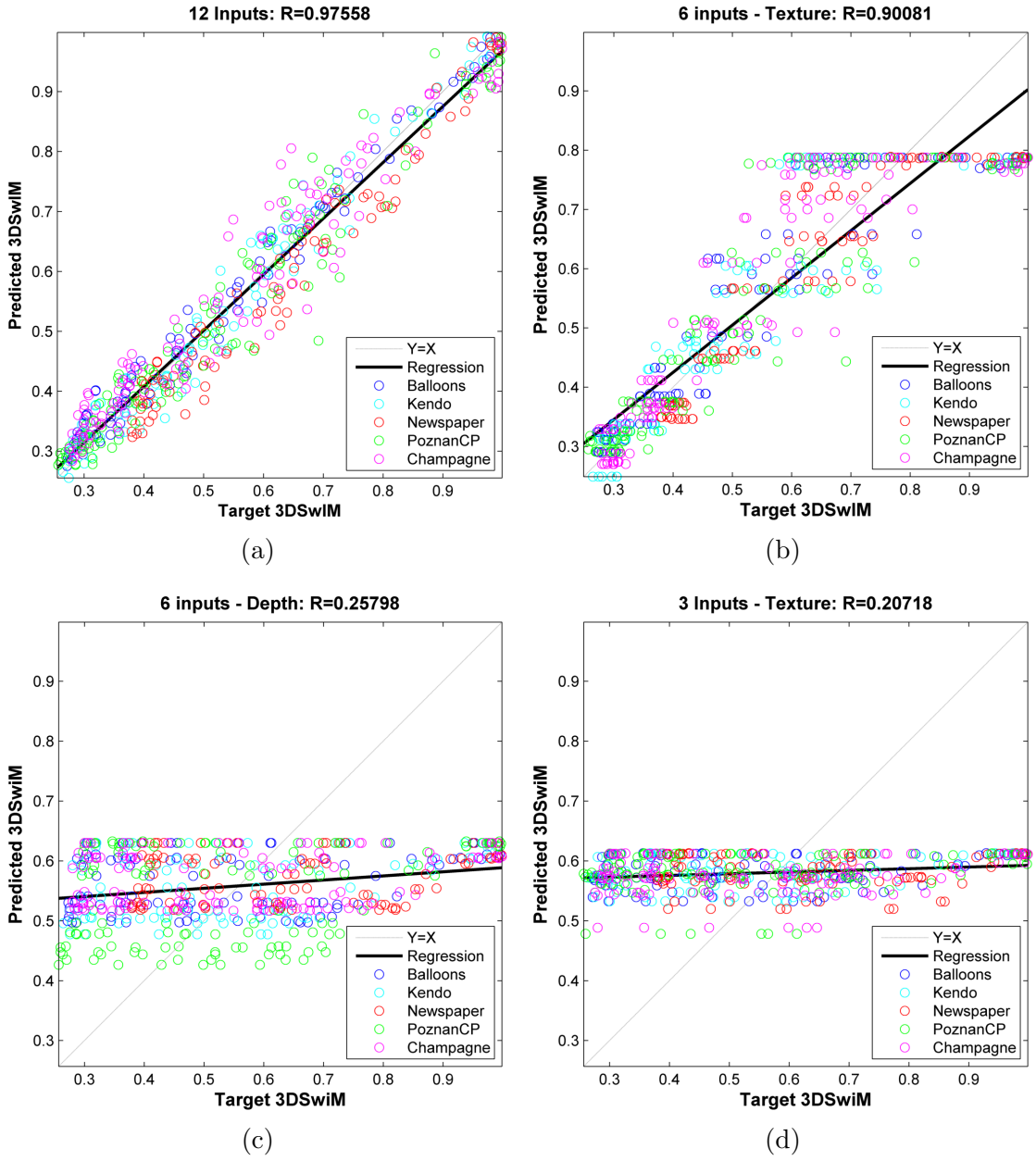


Figure 4.5: $3DSwIM_p$ vs $3DSwIM$: Figure 4.5a - 12 inputs; Figure 4.5b - 6 texture inputs; Figure 4.5c - 6 depth inputs; Figure 4.5d - 3 texture inputs.

All results in the plots are extracted from simulations using 6 hidden nodes. According to figure 4.5 and figure 4.6, the use of only 3 texture inputs proved once again its extremely low accuracy. With a correlation coefficient R of only 0.2, the number of inputs is not enough to estimate accurately the quality. An analysis of figure 4.5c shows that in addition to the lack of input parameters, the inputs only provide depth information, which is not sufficient for this model, where videos are impaired in texture and depth. The correlation obtained using six texture inputs might be accurate enough for some cases. In scenarios where the depth stream is not available for parsing or a technical issue prevents the server receiving the depth

inputs, this model can be an alternative to the twelve input model, which has the best accuracy achieving a correlation of 0.98. Table 4.3 shows the difference in terms of PLCC between methods and the number of inputs used.

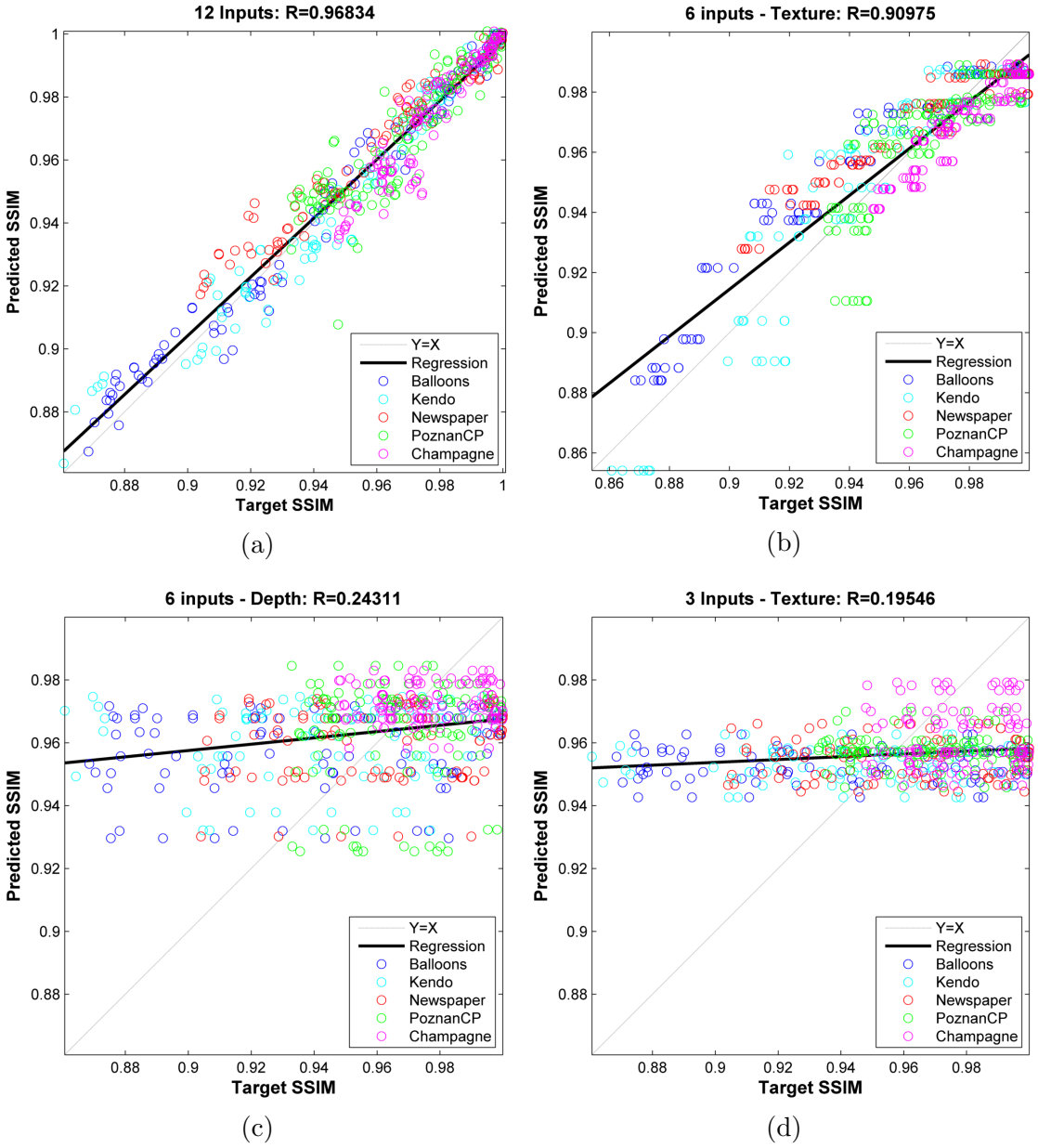


Figure 4.6: $SSIM_p$ vs $SSIM$: Figure 4.6a - 12 inputs; Figure 4.6b - 6 texture inputs; Figure 4.6c - 6 depth inputs; Figure 4.6d - 3 texture inputs.

Number of Inputs	3DSwIM	SSIM
12 Inputs	0.97558	0.96834
6 Texture Inputs	0.90081	0.90975
6 Depth Inputs	0.25798	0.24311
3 Inputs	0.20718	0.19546

Table 4.3: PLCC of the simulated models for the two reference metrics.

Comparing both figure 4.5 and figure 4.6, which use different objective metrics but have the same range ([0,1]), it is obvious 3DSwIM has a wider range of scores. Even though the correlation coefficients are very similar, the most degraded sequences are given a quality indicator under 0.3 when using 3DSwIM, whereas the SSIM algorithm scores are still on the 0.85-0.0.88 interval, which is very high considering the final quality of the sequences.

4.3.2 Second Approach

The second methodology has a different and more *aggressive* validation philosophy. The objective of this approach is to infer the performance of the neural network using different sequences input samples to test the NN performance. Therefore, a *leave-one-out* scheme is implemented. Figure 4.7 shows an example of the mentioned scheme with 4 sequences. The diagram shows that for simulation 1 (where simulation means the full training and testing of the NN), after the training stage with three different sequences, the *left-out* sequence (#4) samples are used to test the model. Simulation 2 is then performed with sequence #4 trading place with sequence #1, being this one the *left-out* sequence. With this approach, the model is being tested with samples of a sequence that were not present in the training stage, which means the model might perform differently due to unknown characteristics of the sequence used for testing. If the characteristics of the sequence being tested are similar to the ones used for training, then a high correlation value is obtained. On the other hand, if the testing sequence is very different from the ones used in training, then the correlation values won't be as good. Thus, with this approach, it is expected to improve generalizability at the cost of the models' accuracy. For the sake of simplicity, and considering what has been said about the use of 3 texture and 6 depth inputs, only the results for 6 texture and 12 inputs are presented.

Plots in figures 4.8, 4.9, 4.10 and 4.11 show the obtained results and confirm the predictions made before. Despite the high correlation values obtained, the second approach does not achieve the values obtained in the first method, particularly for the PoznanCarPark and Champagne sequence. This is explained by the fact both sequences have different resolutions from the other three (the first is an HD sequence and the second has an intermediate resolution) used for training. In addition, the different coding parameters used may also influence the results. As previously said in chapter 3.4, the number of slices that a frame is divided has high impact on error resilience. Since both Champagne and PoznanCarPark have 10 slices/frame instead of the 8 slices/frame of the Balloons, Kendo and Newspaper sequences, the effects of packet loss in the first two sequences are different. Finally, the use of LCU of different sizes also affects the concealment process, which might have different

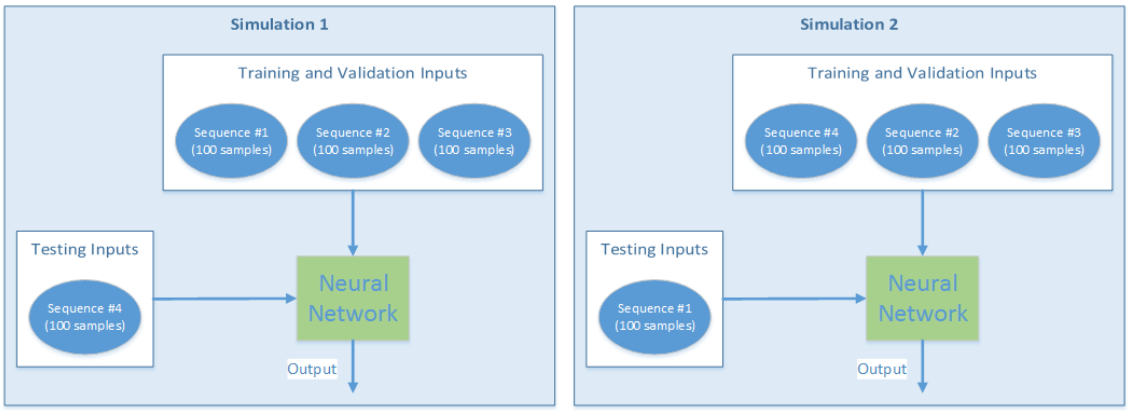


Figure 4.7: Example of the *leave-one-out* scheme. First simulation uses samples from sequences #1, #2 and #3 to train the model whereas sequence #4 (*left-out*) is only used for testing the NN. In the second simulation, sequence #4 (as well as sequences #2 and #3) is now used for training and sequence #1 is now the *left-out*, i.e., is used only for testing.

performances depending on the size of the LCU. In order to increase the PLCC and consequently, the model's performance, in identical situations, the training process should be updated with a larger set of inputs, with different characteristics and different coding parameters.

To conclude this chapter, a brief comparison between the two approaches is given. Even though the testing conditions were different in both simulated methodologies, results showed good correlation values. The obtained results confirmed the predictions made. Methodology 1 is more accurate, achieving higher PLCCs when using 6 texture inputs and 12 inputs. The distribution of samples between the three stages was balanced, which contributed to good results. As a drawback, this method might fail when assessing video or sequences with characteristics different from those used in training and testing stages. Methodology 2 proved to be more general, for situations where each video has its own properties. Testing a NN with samples containing different features from the ones used in training, enables the model to handle more efficiently with *unknown* inputs, giving this model the advantage of being more adaptable. All in all, it is difficult to decide which has the better performance because they depend on where and what they are going to be used for. Once again, each metric/method has its own singularities and utilities, being the choice of the method dependent on the answer of the question "What is the application scope of the desired method?".

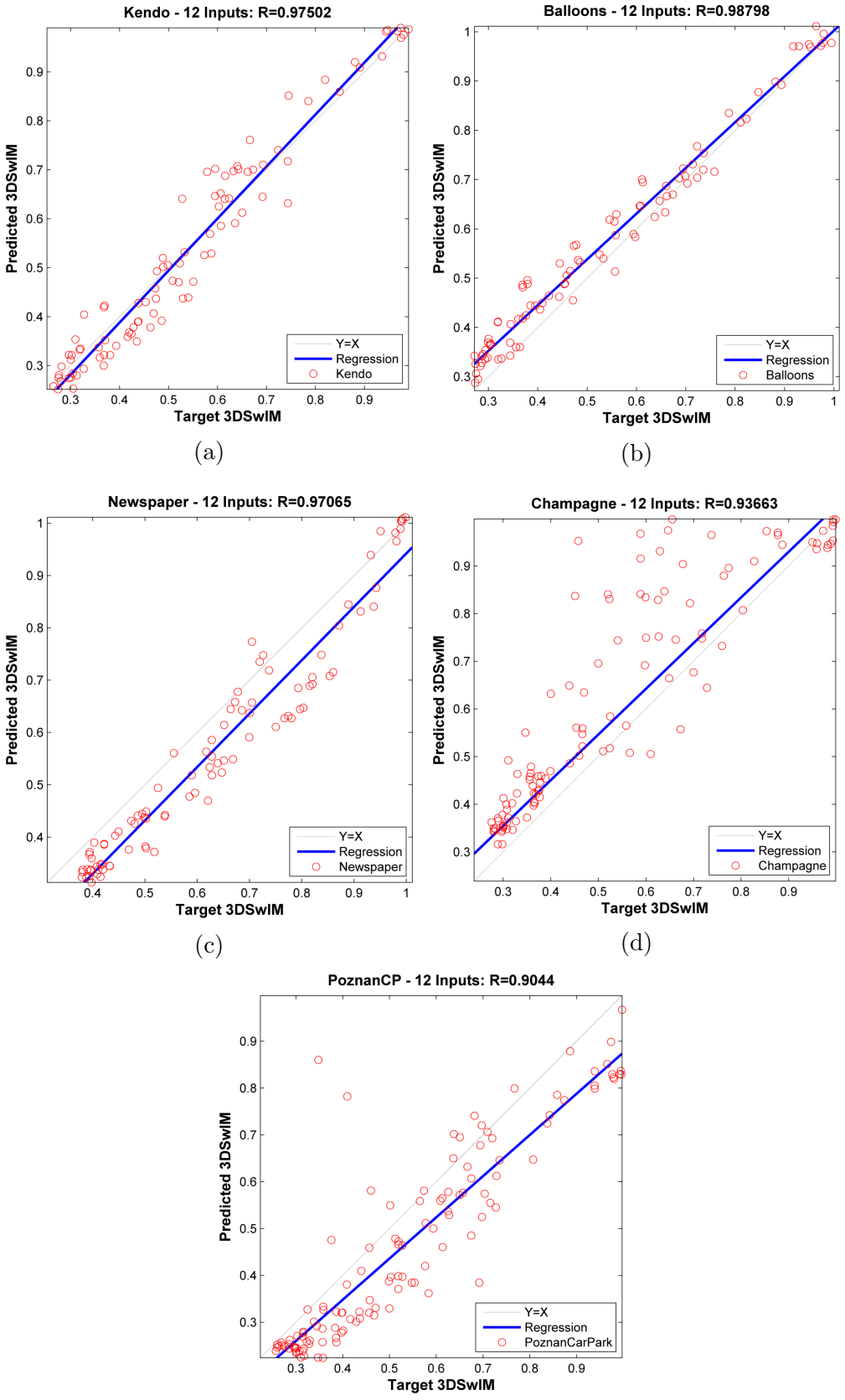
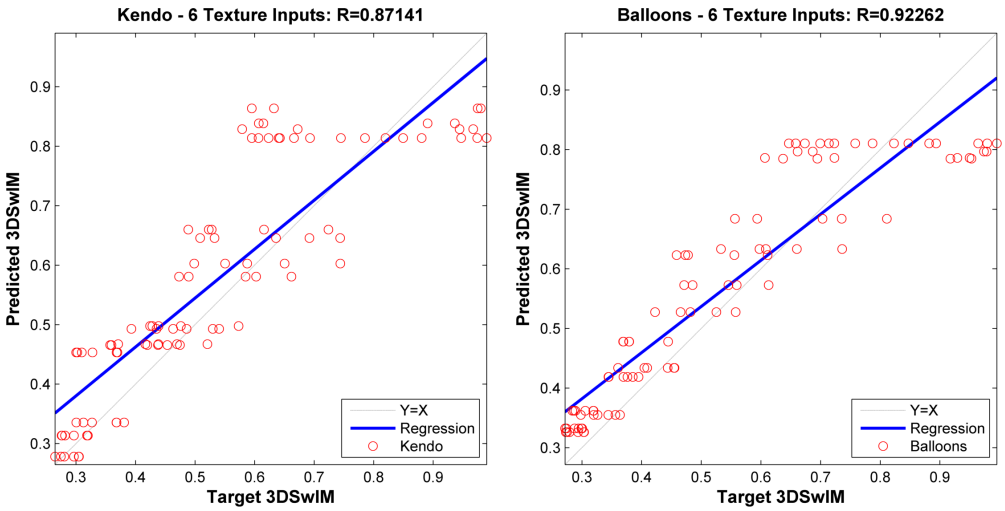
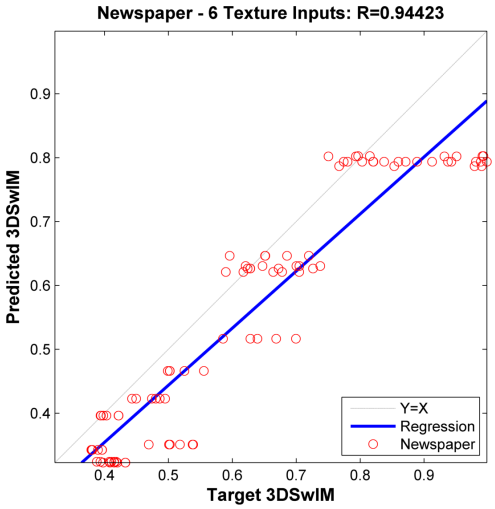


Figure 4.8: 12 inputs with $3DSwIM_p$ vs $3DSwIM$: 4.8a - Kendo; 4.8b - Balloons; 4.8c - Newspaper; 4.8d - Champagne 4.8e - PoznanCarPark

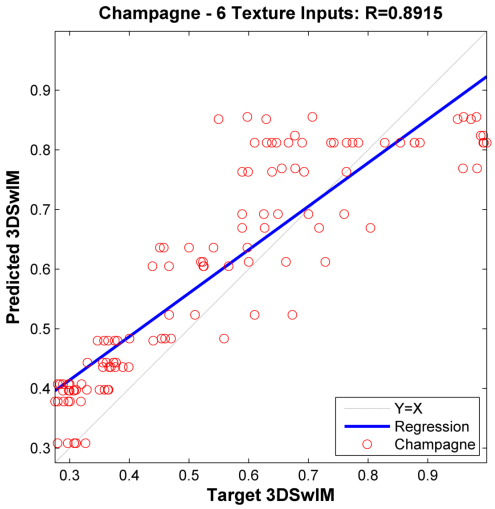


(a)

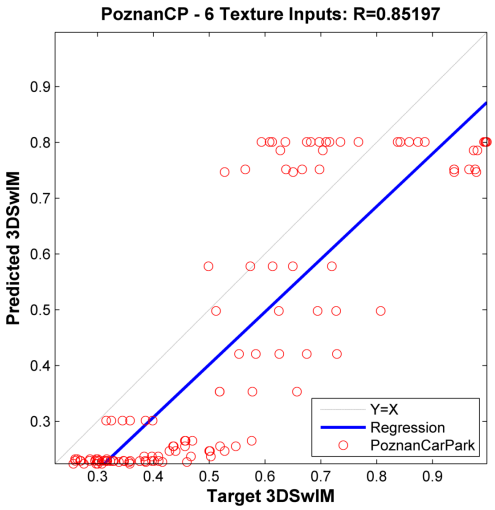
(b)



(c)



(d)



(e)

Figure 4.9: 6 texture inputs with $3DSwIM_p$ vs $3DSwIM$: 4.9a - Kendo; 4.9b - Balloons; 4.9c - Newspaper; 4.9d - Champagne 4.9e - PoznanCarPark

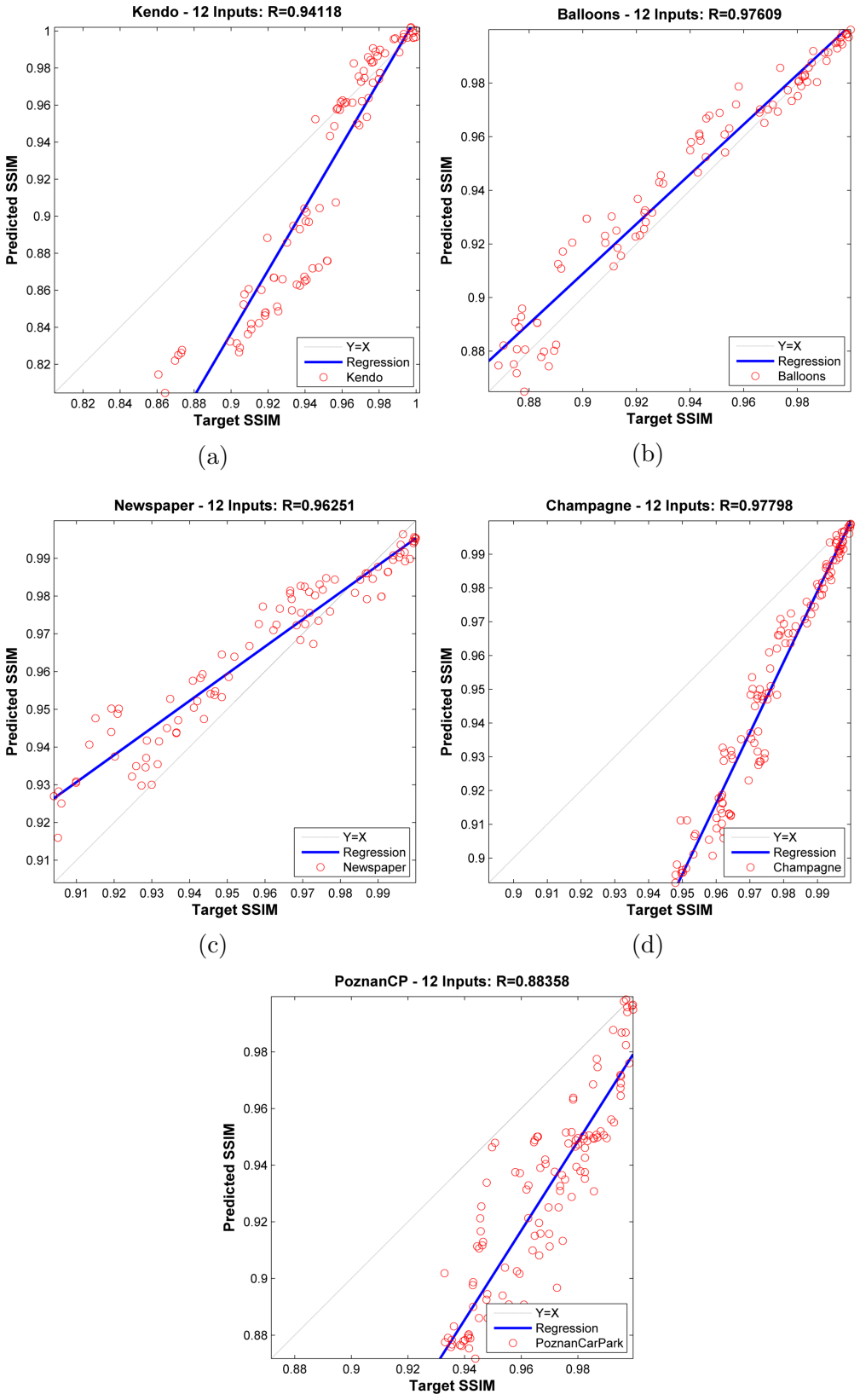


Figure 4.10: 12 inputs with $SSIM_p$ vs $SSIM$: 4.10a - Balloons; 4.10b - Kendo; 4.10c - Newspaper; 4.10d - Champagne 4.10e - PoznanCarPark

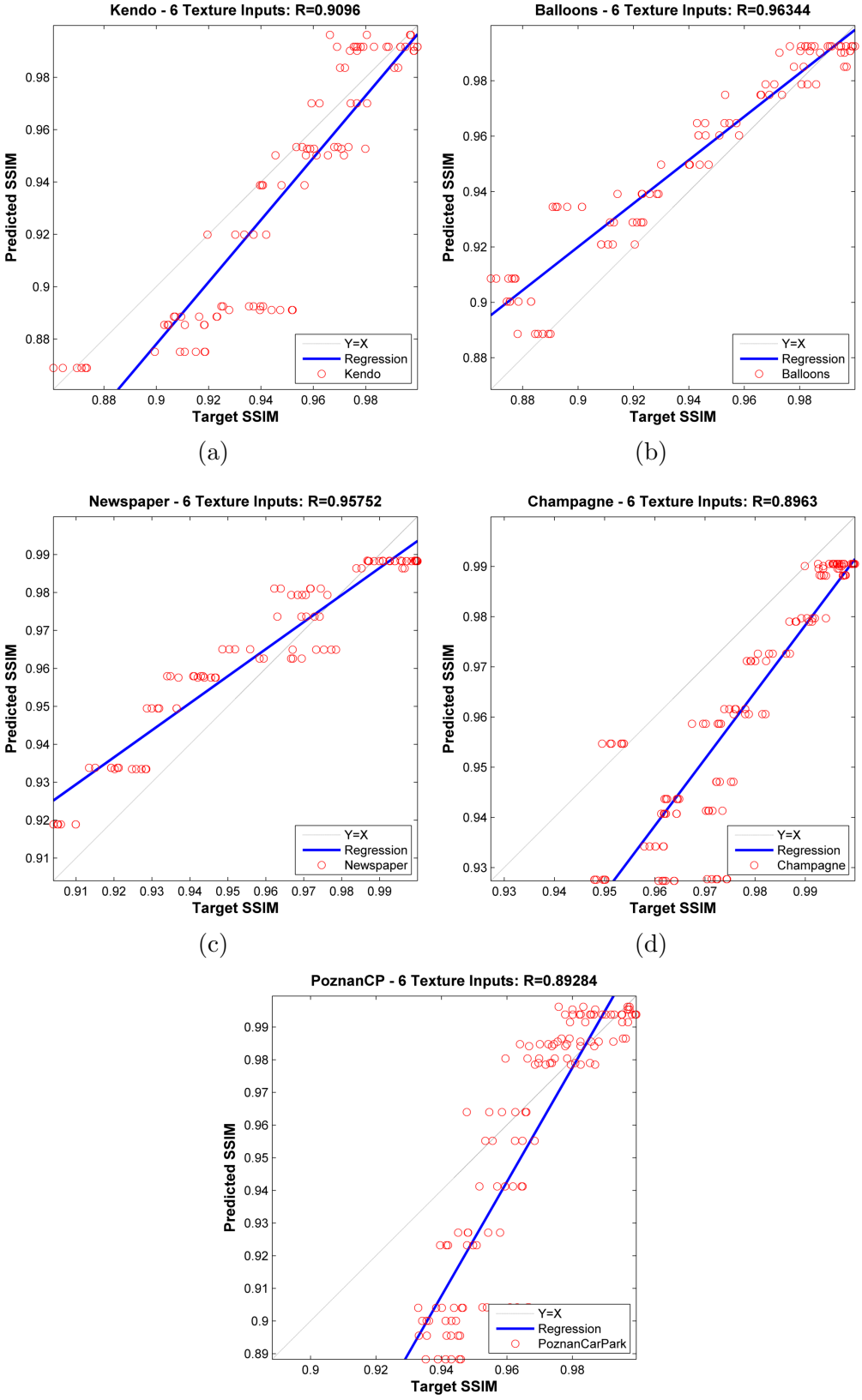


Figure 4.11: 6 texture inputs with $SSIM_p$ vs $SSIM$: 4.11a - Kendo; 4.11b - Balloons; 4.11c - Newspaper; 4.11d - Champagne 4.11e - PoznanCarPark

Chapter 5

Subjective Quality Assessment of 3D Video

As mentioned earlier, assessing 3D video quality poses some challenges to engineers and researchers. Quantifying the quality of 3D video is extremely subjective due to the dependence on each viewer's opinion. The true indicator of a quality assessment model's performance is the correlation between the objective metric quality scores obtained and the perceived quality in terms of mean opinion score (MOS), which is the *real QoE indicator*. It is important to mention that subjective sessions are used in almost every study that involves assessing quality, which includes impairments originated by coding, decoding with concealment, random events during transmission or view synthesis algorithms, due to the relevance of a subject's opinion. Thus, it is necessary to conduct a subjective assessment session to collect scores indicating the quality perceived by the subjects. In fact, the main goal of the quality assessment methods is to develop an objective method capable of outputting a score as close as possible to the subjective scores assigned by a group of observers. A computable model is desirable since conducting test sessions as the one performed in this work, are costly, time consuming and require a considering number of resources. This led the scientific community to look for alternatives for subjective studies that are costly. Moreover, some of these subjective sessions are uncomfortable for viewers, which can be reduced if the test is performed in a more *friendly*, but still controlled, environment. One alternative approach is using a crowd-sourced [66–69] based platform where the subjective video quality assessment is conducted over the Internet, allowing for faster and cheaper evaluations, and reaching a larger number of evaluators. The author of this thesis contributed to the construction of a crowd-sourced quality evaluation platform which was used in a group study conducted by the Instituto de Telecomunicações - Coimbra, the Hellenic Open University at Patras and the Department of Wireless Communications of the University of Zagreb, which resulted in the publication of a scientific

paper [11]. Its purpose was to collect a large number of subjective scores from three different research centers in different countries using a self-developed crowd-source platform and make it public for scientific purposes. In order to perform the tests correctly in the laboratory where the study was conducted, several changes had to be made to the platform due to compatibility problems with the existent hardware. Several setups were experimented, using different browsers and software versions. After a period of experiments, a functional setup was chosen and the subjective sessions were ready to begin. With great success, around 40 volunteers participated in this experiment. But despite the effort of crowd-source methods, several issues arise from it: the reliability of user ratings, the influence of incentives, payment schemes and the unknown environmental context of the tests are among the main concerns. Still, it is a valid alternative with a lot of potential to ease the task of conducting subjective tests. In this work, due to the specificity of the impairments, a local subjective study was conducted.

There is no optimal methodology to evaluate the quality perceived by humans of 3D video subjected to impairments with unpredictable effects, such as bit errors or packet loss. However, it is convenient to follow the rules specified in the recommendations [9, 10] for the environment and test conditions, in order to have results that can be compared with other experiments. In this work, the *Single Stimulus* (SS), particularly the Absolute Category Rating with Hidden Reference (ACR-HR) [70], methodology was used, whose procedures and results are presented in section 5.2.

5.1 Test Conditions and Subjects

When conducting a subjective study of this type, certain conditions should be met in order to reduce the influence of surrounding elements in the subject's evaluation session. The tests were conducted in a quiet and daylight-illuminated room. The platform used was designed in HTML and PHP, with highly intuitive graphical interfaces for grading and voting on a tablet-PC, as shown in figure 5.1. The viewers started the test by introducing their age and gender in a tablet for posterior statistical analysis, with the sequences being played after the *Play* button was pressed. A server workstation was used to reproduce all the test sequences in a random order and register in a file the grading votes. A 20-inch Philips WOWvx 9-view autostereoscopic display was used to display the 3D sequences. During the test, the viewer was comfortably seated in front of it at an optimal distance of 80cm. The overall time duration of the evaluation session was about 27 minutes. A total of 6 female and 28 male voluntary viewers, aged from 21 to 48 with average of 25 years, participated in the subjective assessment experiments. Most of the participants were students and only a few were familiar with this kind of procedure. The so-called

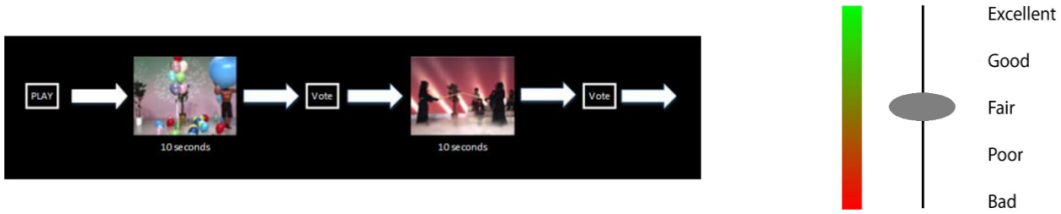


Figure 5.1: Graphic interface used during tests. On the left, the picture shows the instructions before the start of the test. The picture on the right shows the intuitive grading bar.

"fly" depth acuity test was performed to every participant to check if they had good vision acuity and stereo vision. A detailed explanation was given to the subjects before each test session, in order to clarify the objectives, grading procedures and answer possible questions the participants might have had.

5.2 ACR-HR Session

The adopted ACR-HR methodology conforms with the ITU-T P.913 [69] and ITU-R BT.500-13 [9] as a Single Stimulus (SS) method. ACR-HR is based on the ACR method, which is a category judgment where the test stimuli are presented one at a time and are rated independently on a category scale. The subject observes one stimulus of 10 seconds and then has time to rate that stimulus. ACR method uses a five-level rating scale from 1 to 5, with 1 being *very bad* and 5 *excellent*. With ACR-HR, the experiment includes a reference version of each video segment, not as part of a pair, but as a freestanding stimulus for rating like any other. The viewers did not know that there was a reference sequence and the sequences were displayed randomly, changing from test to test.

The video presentation scheme is shown in figure 5.2. 92 impaired versions of each 3D sequence simulated in the previous chapter, with PLR ranging from 1% to 20%, plus the 5 original sequences totals 97 videos to evaluate. Each sequence was only showed once, in order to reduce the test time and avoid fatigue on the subjects. Thus, each test session results in 97 scores, one for each sequence, leading to 34 scores for each sequence. Before each test session, the viewer was presented with a reference demo sequence to familiarize the observer with setup and typical quality of the display device.

During the data analysis the ACR-HR scores are subtracted from the corresponding reference scores to obtain a DMOS, with this procedure being known as "hidden reference removal" [70]. Differential viewer scores (ΔDV) are calculated for each video j. The appropriate hidden reference score (V_{ref}), which is the score of

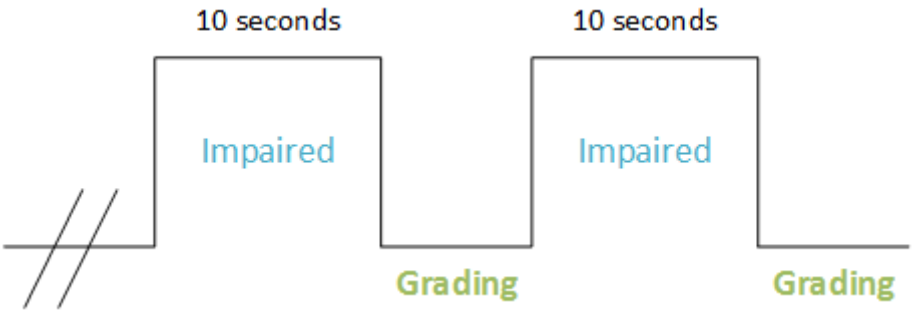


Figure 5.2: Presentation structure.

the reference video j, is used to calculate ΔDV using the following formula:

$$\Delta MOS_j = MOS_j - MOS_{ref} + 5 \tag{5.1}$$

where V_j is the viewer's ACR-HR score for sequence j. In using this formula, a DV of five indicates "Excellent" quality and a DV of one indicates "Bad" quality. Any DV values greater than five (i.e., where the impaired sequence is rated better quality than its associated hidden reference sequence) is considered valid. Letting N be the number of viewers, the DMOS for each video j is then obtained:

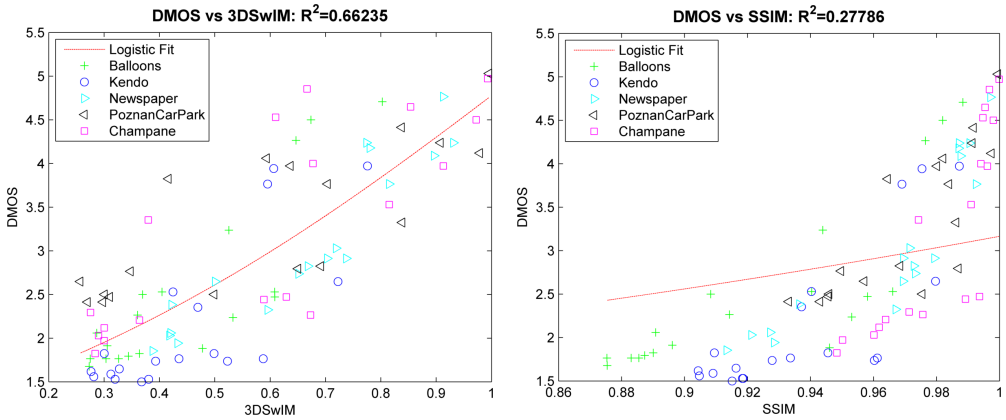
$$DMOS_j = \frac{1}{N}\sum_{i=1}^{N}(\Delta MOS_j)_i \tag{5.2}$$

Figure 5.3 and figure 5.4 show the results of the *DMOS* vs. *SSIM* and *DMOS* vs. *3DSwIM*. Two different regression methods were considered: a logistic function in Equation 2.2 (repeated in equation 5.3 and recommended in [9] for mapping objective quality metric (OQM) scores and predicted MOS y); and using a polynomial fit of second order defined in equation 5.4. As mentioned earlier, a higher value of DMOS means *excellent* quality, which is contradictory with the recommended logistic fit defined in equation 5.3, reason for the use of a second approach. DMOS values are obtained from equation 5.2 and mapped with the correspondent sequence objective quality metric, either 3DSwIM or SSIM. The regression plots are also represented. The dispersed data, which means the DMOS do not correlate very well with the estimated quality score, contribute to the reduction of the PLCC.

$$y = \frac{a_1}{1 + e^{a_2(OQM + a_3)}} \quad , \quad a_1, a_2, a_3 \text{ being fitting coefficients} \tag{5.3}$$

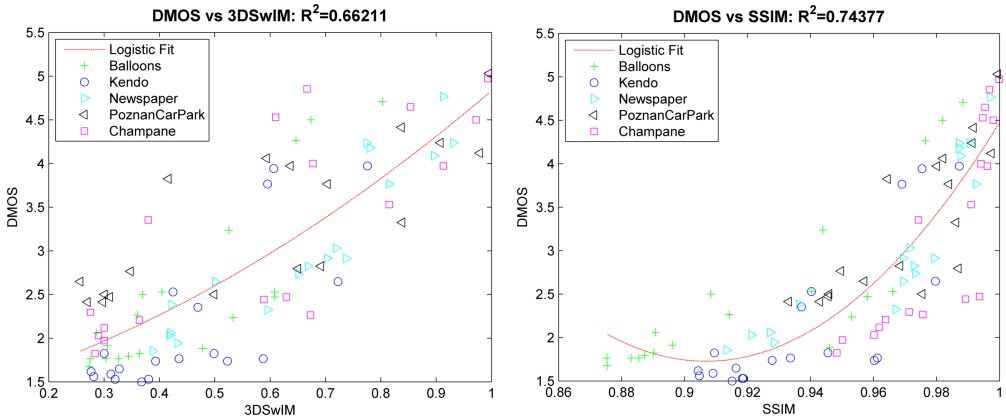
$$y = a_1.OQM^2 + a_2.OQM + a_3 \quad , \quad a_1, a_2, a_3 \text{ being fitting coefficients} \tag{5.4}$$

The results obtained in terms of PLCC are lower than expected, particularly for



(a) (b)

Figure 5.3: DMOS vs. 3DSwIM (left) and DMOS vs. SSIM (right) for the 92 evaluated videos using the logistic fit in Equation 5.3.



(a) (b)

Figure 5.4: DMOS vs. 3DSwIM (left) and DMOS vs. SSIM (right) for the 92 evaluated videos using a polynomial fit.

	a1	**a2**	**a3**	**PLCC between DMOS and $DMOS_{predicted}$**
3DSwIM	10.4916	-1.8528	-1.0969	**0.8139**
SSIM	349.4390	-2.1410	-3.1931	**0.8004**

Table 5.1: Fitting coefficients and PLCC of the plots of Figure 5.3 - logistic fit results.

	a1	**a2**	**a3**	**PLCC between DMOS and $DMOS_{predicted}$**
3DSwIM	1.8678	1.6636	1.2999	**0.8137**
SSIM	316.1511	-573.3892	261.7094	**0.8624**

Table 5.2: Fitting coefficients and PLCC of the plots of Figure 5.4 - polynomial results.

3DSwIM PLCC values. It was expected that SSIM would be more inaccurate when correlating with DMOS values, since despite the severity of the degradations, it still provided high SSIM values, on the opposite of the MOS values obtained, which were very low. The polynomial fit for the SSIM method is the one with better correlation coefficients, thus being a good predictor for video quality. The obtained fitting coefficients for both methods and the PLCC between DMOS and $DMOS_{predicted}$ are shown in table 5.1 and table 5.2.

Chapter 6

3DVQM: a practical implementation

The author of this thesis also had a participation in a project called 3D Video Quality Monitor (3DVQM), supported by the Instituto de Telecomunicações. The developed model was a complete monitoring system that predicted the perceived quality of video streams subject to packet loss in real time. The proposed model adopts the same structure as the one presented in this thesis: using parameters extracted from packets, a simple mathematical model outputted a quality score which was shown in a real time chart. Using the H.264/AVC encoder in the stereo format (left+right view), the streams were transmitted through a lossy channel, with the packet losses being simulated by the Gilbert-Elliot model. Two clients were receiving and decoding two different sequences in real time, using the VLC player as a decoder and player. Between the server/sender and receiver/client, probes could be assigned from a web application to sniff packets and find any discontinuity in the stream. The probes could be assigned to specific nodes, identified by an IP address. If a probe was configured to inspect a certain node of the network, the graphic with the quality estimation would pop up, with 5 to 10 seconds of update interval. In order to constantly update the chart, the web application retrieved the quality scores from a database, in which the scores estimated by the objective metric were sent by the active probes. Figure 6.1 shows the setup working in a telecommunications meeting, in Aveiro, with the authors of the proposal. The model was developed in Python by the main author of the project, Nuno Martins, with the author of this thesis developing the web application, using HTML, PHP and JavaScript, as well as the database in MySQL.

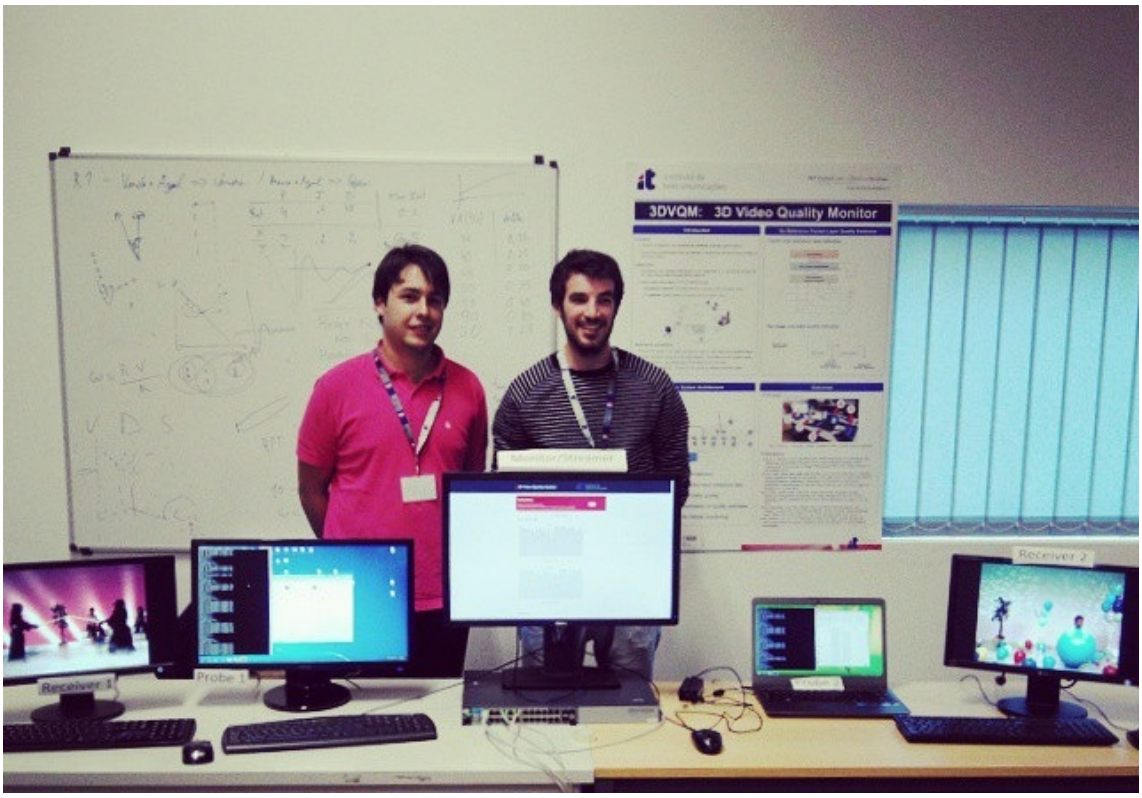


Figure 6.1: Setup experiment of 3DVQM.

This project showed it is possible, with few resources, to build a monitoring system such as the one proposed in this thesis. Furthermore, during the project presentation at the meeting, several people wanted to know more about the idea and what was the purpose of such project. The response was more positive than expected, with this experience being considered a success.

Chapter 7

Conclusion

This thesis aimed to address the problem of assessing 3D video quality transmitted over packet-loss-prone channels. The main objectives proposed in section 1.1 have all been successfully achieved. The quality models discussed and developed in this work yield very good results, regarding the assumptions made and the methodologies followed - the encoding and transmission setup and the definition of the used input parameters and targets. As discussed in chapter 2, it is very difficult to compare different quality assessment methods developed and published in the scientific literature, because almost all of them have different kinds of applications, different goals and even different assumptions regarding, for instance, the encoding and transmission setup. Nevertheless, before closing this essay, a further explanation is provided on this issue.

The first approach to address this work's subject was to try to assemble a large number of recent quality assessment models for 3D video quality available in the scientific literature. The type of impairment applied to test sequences was early defined and restricted to packet losses, since this work also aimed at developing and improve the method proposed in [37]. The use of HEVC as codec and a different set of sequences required a new setup for simulation environment. To accomplish this task, new raw texture-plus-depth sequences were encoded with their bitstreams being packetized, followed by the packet loss simulation generated by the Gilbert-Eliot model. These impaired bitstreams were decoded with a error concealment decoder for quality evaluation, using the chosen reference metrics.

The first task faced some unexpected issues. HEVC offered a wide variety of coding configurations, with new parameters to adjust, and above all, to understand what was their role. In order to discover what configuration should be used, the author read several possible combinations in the literature and experimented them using the reference software. After the choice of the parameters to use and recalling the available models in the literature, the author began to realize methods based only on pixel information lacked something to be even more accurate. If the following

question is asked: is it possible to objectively evaluate 3D video or image - and even 2D video or image - without any reference model, just simply by analyzing the pixel-domain information? To understand better this question, in the perspective of a viewer, how easy would it be to subjectively evaluate a 3D video or image without any common-sense reference regarding how a *poor* and *excellent* quality should look like? In the thesis's author opinion, the answer is no: the errors associated would be enough to label it as *unreliable*. Notice that the mentioned *poor* and *excellent* quality are tremendously subjective and keeps changing all the time: advances in technology together with new and more efficient coding tools turn an *excellent* quality video with low resolution displayed in normal display two years ago into a *not so good* quality if the same video is displayed in a HD or even UHD display. This explains why during the subjective assessment, despite the subjects who participated in the test session in chapter 5 did not know that five of the sequences were the original, a pre-test demo sequence was shown so the viewers could observe the effects of the autostereoscopic display. In fact, this was one of the issues that viewers complained about: the quality of the general 3D provided by the autostereoscopic display was far from ideal.

Having this in mind, it is clear methods based on only pixel-domain information might be highly susceptible to inaccurate prediction of QoE simply because the original content might be already of poor quality, even if the transmission occurs without any problem. These type of methods could benefit from other layers' information, if available, such as coding parameters or the type of display used, which are available at the bitstream or packet-level for coding parameters, whereas the type of display could be inferred at the application layer. It is the author's belief that these extra pieces of information might add value and increase robustness to the quality models, giving them enough arguments to distinguish original bad videos from good videos that were impaired by a random event. This led to the choice of developing a no-reference model, based on NN with reference to media-layer models. In a way, the developed model might be considered hybrid, as it uses different sources of information.

The combination of the media-layer and packet-layer models opened a window of possibilities in encoding configurations. The ones used in this thesis are not the only ones possible and were based on what a "typical" configuration should be. If the proposed method was adopted by the industry, each company would have to adjust and tune the model to the encoding settings of the content provider, adopt new packetization schemes according to the network service provider and perform new training sessions with the new data. And even though this might seem a big concern, in fact it is limited to the number of encoding configurations (GOP, bit-rate, size, depth-level, and so on) that is usually reduced, as some of them are imposed by the

codec manufacturer. And even if a specific set of encoding parameters is required, the use of such models allows a fast and reliable adjustment to the industry needs.

To conclude this thesis, the developed model proved to be accurate enough for solving quality assessment problems in error-prone channels. It is an automatic system and highly adaptable which are two components really appreciated in the video or image quality assessment market. As part of a monitoring system, the model predicts the QoE at the client side, identifying any event that is degrading the video's quality. chapter 6 described a model with these characteristics and it worked very well. The author of this thesis hopes in the next few years that an experimental setup of the proposed method may become real, with the expectation of one day having one system like this installed in our homes.

Appendix A

NAL splitting into RTP packets - Matlab script

```
1
2  %NALU Losspattern generator using a tracefile from G.-E. Model for HM 12.1
       __EM_JEG_v1.2.1
3  %By Pedro Rocha
4
5  %The script loads a data file generated by the decoder containing all
6  %the NALU's idx and size. For each NALU it is verified if there's the
7  %need of segmentation assuming a MTU of 1500 bytes. After calculating the
8  %number of RTP packets needed, trace_file with a certain PLR and MBL is
9  %loaded to compare with the results and generate a NALU LossPattern
10 %according to the trace_file.
11
12 %% data.txt Loading
13 %Generates a table with the number of RTP necessary for the bitstream transmission.
14 %This table will be compared to the trace file and create a NALU
15 %losspattern file for HM-12.1__EM_JEG_v1.2.1 decoder.
16
17 clc
18 clear all
19
20 n_sequences = 10;     %1,2 - Balloons (texture, depth)
21                       %3,4 - Champagne
22                       %5,6 - Kendo
23                       %7,8 - Newspaper
24                       %9,10 - PoznanCarPark
25
26 RTP_packet = [];
27 %simulate all Videos
28 for k = 1 : n_sequences
29     switch k
30         case 1
31             sequence = 'Balloons';
32             nalu_file = 'balloons_color_nalu.txt';
33             trace_path = 'MaxBL7';
34         case 2
35             nalu_file = 'balloons_depth_nalu.txt';
36             trace_path = 'MaxBL7';
37         case 3
38             sequence = 'Champagne';
39             nalu_file = 'champagne_color_nalu.txt';
40             trace_path = 'MaxBL9';
41         case 4
42             nalu_file = 'champagne_depth_nalu.txt';
```

```
43              trace_path = 'MaxBL9';
44          case 5
45              sequence = 'Kendo';
46              nalu_file = 'kendo_color_nalu.txt';
47              trace_path = 'MaxBL7';
48          case 6
49              nalu_file = 'kendo_depth_nalu.txt';
50              trace_path = 'MaxBL7';
51          case 7
52              sequence = 'Newspaper';
53              nalu_file = 'newspaper_color_nalu.txt';
54              trace_path = 'MaxBL7';
55          case 8
56              nalu_file = 'newspaper_depth_nalu.txt';
57              trace_path = 'MaxBL7';
58          case 9
59              sequence = 'PoznanCP';
60              nalu_file = 'poznancp_color_nalu.txt';
61              trace_path = 'MaxBL9';
62          case 10
63              nalu_file = 'poznancp_depth_nalu.txt';
64              trace_path = 'MaxBL9';
65      end
66
67      %NALU information loading
68
69      data = dlmread(['C:\VIDEODATABASE\PRocha_3D-HEVC\Lossy_Videos\Concealed_Videos\
            ' sequence '\NALU\' nalu_file]);
70
71      N = length (data);  %Total number of NALU's
72      n_RTP_packets = 0;  %Total number of RTP packets needed
73          %RTP packet index
74                          %col. 1 -> RTP_packet index
75                          %col. 2 -> NALU index
76                          %col. 3 -> Payload size
77                          %col. 4 -> Loss Flag (packet loss or not)
78                          %col. 5 -> Slice Type
79                          %col. 6 -> POC
80      indicator = 1;
81      for n = 1:N
82          %Number of RTP packets for each NALU
83          rtpp_per_nal = ceil(data(n,2)/1500);
84          %Total number of RTP packets
85          n_RTP_packets = n_RTP_packets +  rtpp_per_nal;
86
87          nalu_size = data(n,2);
88          %In case NALU is segmented in more than one RTP packet
89          if (rtpp_per_nal > 1)
90              for m=1:rtpp_per_nal
91                  %Update RTP packet ID
92                  RTP_packet{k}{indicator}.table((n_RTP_packets-rtpp_per_nal)+m,1) =
                        n_RTP_packets - (rtpp_per_nal-m);
93                  %Update RTP packet NALU idx
94                  RTP_packet{k}{indicator}.table((n_RTP_packets-rtpp_per_nal)+m,2) =
                        data(n,1);
95
96                  %Attribute RTP packet size according to NALU size
97                  if(nalu_size >1500)
98                      RTP_packet{k}{indicator}.table((n_RTP_packets-rtpp_per_nal)+m,
```

```
                        3) = 1500;
99                      nalu_size = nalu_size - 1500;
100                 else
101                     RTP_packet{k}{indicator}.table((n_RTP_packets-rtpp_per_nal)+m
                            ,3) = nalu_size;
102                 end
103             end
104         else %1 NALU per RTP packet
105             RTP_packet{k}{indicator}.table(n_RTP_packets,1) = n_RTP_packets - (
                    rtpp_per_nal-m);
106             RTP_packet{k}{indicator}.table(n_RTP_packets,2) = data(n,1);
107             RTP_packet{k}{indicator}.table(n_RTP_packets,3) = nalu_size;
108         end
109     end
110
111     indicator = 1;
112     % Loading and comparison with trace_file generated by the Gilbert-Elliot Model
113
114     if(strcmp('MaxBL9',trace_path))
115         bl = 6;
116     else
117         bl = 5;
118     end
119
120     for b = 3:bl
121         %Packet Loss Ratio
122         plr = 1;
123         while plr < 21
124             if indicator ~= 1
125                 RTP_packet{k}{indicator}.table = RTP_packet{k}{1}.table;
126             end
127             fid = fopen(['C:\VIDEODATABASE\PRocha_3D-HEVC\Lossy_Videos\
                        Concealed_Videos\trace_files\new\' trace_path '\b' num2str(b) 'plr
                        ' num2str(plr)]);
128             trace_file = textscan(fid, '%c');
129             % Convert into an integer to compare
130             trace_file = str2num(trace_file{1,1});
131             fclose(fid);
132             RTP_packet{k}{indicator}.name = ['b' num2str(b) 'plr' num2str(plr)];
133
134             losspattern = zeros(1,N);       % Losspattern to save
135             packet_corrupted = 0;           % If packet is corrupted
136             last_nal_id = RTP_packet{k}{indicator}.table(1,2); % Auxiliar variable
                    to save previous NALU index
137             saved = false;
138             already_corrupted =false;       % Auxiliar variable to prevent
                    redundancy
139             new_packet = false;             % Indicates if a new packet will be
                    analyzed
140             aux = 1;                        % Auxiliar variable to keep 'j' updated
141
142             for i=1:N
143                 for j=aux:length(RTP_packet{k}{indicator}.table);
144                     % Updates current packet id with corresponding NALU idx
145                     curr_nal_id = RTP_packet{k}{indicator}.table(j,2);
146                     %Condition to check if NALU has changed
147                     if curr_nal_id ~= last_nal_id
148                         %write to losspattern
149                         losspattern(i) = packet_corrupted;
```

```
150                         new_packet = true;
151                         already_corrupted = false;
152                         saved = true;
153                     end
154 
155                     %Save if packet is loss or not
156                     RTP_packet{k}{indicator}.table(j,4) = trace_file(j);
157 
158                     if(already_corrupted == false || new_packet == true)
159                         if (trace_file(j) == 0)
160                             packet_corrupted = 0;
161                         elseif trace_file(j) == 1
162                             packet_corrupted = 1;
163                             already_corrupted = 1;
164                         end
165                         new_packet = false;
166                     end
167 
168                     last_nal_id = curr_nal_id;
169 
170                     if(saved)
171                         %Slice type of packet indicator
172                         %SLICE TYPE: 0 -> B ; 1 -> P ; 2 -> I
173                         RTP_packet{k}{indicator}.table(j,5) = data(i+1, 4);
174 
175                         %Slice POC
176                         RTP_packet{k}{indicator}.table(j,6) = data(i+1, 3);
177                         saved = false;
178                         %Leave the inner for loop
179                         break;
180                     else
181                         %Slice type of packet indicator
182                         %SLICE TYPE: 0 -> B ; 1 -> P ; 2 -> I
183                         RTP_packet{k}{indicator}.table(j,5) = data(i, 4);
184 
185                         %Slice POC
186                         RTP_packet{k}{indicator}.table(j,6) = data(i, 3);
187                     end
188                 end
189                 aux=j+1;
190             end
191 
192             %Save losspattern
193             if(rem(k,2) == 0)
194                 fid_1 = fopen(['C:\VIDEODATABASE\PRocha_3D-HEVC\Lossy_Videos\
                        Concealed_Videos\' sequence '\losspattern\losspattern_b'
                        num2str(b) 'plr' num2str(plr) '_depth.txt'], 'w');
195                 fprintf(fid_1, '%d\n', losspattern);
196                 fclose(fid_1);
197             else
198                 fid_2 = fopen(['C:\VIDEODATABASE\PRocha_3D-HEVC\Lossy_Videos\
                        Concealed_Videos\' sequence '\losspattern\losspattern_b'
                        num2str(b) 'plr' num2str(plr) '_color.txt'], 'w');
199                 fprintf(fid_2, '%d\n', losspattern);
200                 fclose(fid_2);
201             end
202 
203             % PLR and size of lost slices/NALU I,P an B.
204             for k = 1: length(RTP_packet)
```

```
205             for j = 1: length(RTP_packet{k})
206
207                 %PLR_B
208                 RTP_packet{k}{j}.PLR_B = sum (RTP_packet{k}{j}.table(:,4) == 1
                        & RTP_packet{k}{j}.table(:,5) == 0 ) / sum(RTP_packet{k}{j
                        }.table(:,5) == 0 );
209                 %PLR_I
210                 RTP_packet{k}{j}.PLR_I = sum (RTP_packet{k}{j}.table(:,4) == 1
                        & RTP_packet{k}{j}.table(:,5) == 2 ) / sum(RTP_packet{k}{j
                        }.table(:,5) == 2 );
211                 %PLR_P
212                 RTP_packet{k}{j}.PLR_P = sum (RTP_packet{k}{j}.table(:,4) == 1
                        & RTP_packet{k}{j}.table(:,5) == 1 ) / sum(RTP_packet{k}{j
                        }.table(:,5) == 1 );
213                 %PLR
214                 RTP_packet{k}{j}.PLR = sum (RTP_packet{k}{j}.table(:,4) == 1) /
                         length(RTP_packet{k}{j}.table(:,5));
215
216                 %Size of packets lost
217                 %B packets lost
218                 filter_B = RTP_packet{k}{j}.table(:,4) &  RTP_packet{k}{j}.
                        table(:,5) == 0;
219                 [auxiliar_table_B idx] =  sort( filter_B , 'descend') ;
220                 B_affected = accumarray(auxiliar_table_B+1, 1);
221                 if(length(B_affected) == 2)
222                     B_affected = B_affected(2);
223                     RTP_packet{k}{j}.num_lost_B = B_affected;
224                     RTP_packet{k}{j}.size_lost_B = sum(RTP_packet{k}{j}.table(
                            idx(1:B_affected),3));
225                 else
226                     RTP_packet{k}{j}.size_lost_B = 0;
227                     RTP_packet{k}{j}.num_lost_B = 0;
228                 end
229
230                 %P packets lost
231                 filter_P = RTP_packet{k}{j}.table(:,4) &  RTP_packet{k}{j}.
                        table(:,5) == 1;
232                 [auxiliar_table_P idx] =  sort( filter_P , 'descend') ;
233                 P_affected = accumarray(auxiliar_table_P+1, 1);
234                 if(length(P_affected) == 2)
235                     P_affected = P_affected(2);
236                     RTP_packet{k}{j}.num_lost_P = P_affected;
237                     RTP_packet{k}{j}.size_lost_P = sum(RTP_packet{k}{j}.table(
                            idx(1:P_affected),3));
238                 else
239                     RTP_packet{k}{j}.num_lost_P = 0;
240                     RTP_packet{k}{j}.size_lost_P = 0;
241                 end
242
243                 %I packets lost
244                 filter_I = RTP_packet{k}{j}.table(:,4) &  RTP_packet{k}{j}.
                        table(:,5) == 2;
245                 [auxiliar_table_I idx] =  sort( filter_I , 'descend') ;
246                 I_affected = accumarray(auxiliar_table_I+1, 1);
247                 if(length(I_affected) == 2)
248                     I_affected = I_affected(2);
249                     RTP_packet{k}{j}.num_lost_I = I_affected;
250                     RTP_packet{k}{j}.size_lost_I = sum(RTP_packet{k}{j}.table(
                            idx(1:I_affected),3));
```

```
251                         else
252                             RTP_packet{k}{j}.num_lost_I = 0;
253                             RTP_packet{k}{j}.size_lost_I = 0;
254                         end
255 
256                         %All packets
257                         filter_all = RTP_packet{k}{j}.table(:,4);
258                         [auxiliar_table_all idx] =  sort( filter_all , 'descend') ;
259                         all_affected = accumarray(auxiliar_table_all+1, 1);
260 
261                         all_affected = all_affected(2);
262                         RTP_packet{k}{j}.num_lost_all = all_affected;
263                         RTP_packet{k}{j}.size_lost_all = sum(RTP_packet{k}{j}.table(idx
                                (1:all_affected),3));
264                     end
265                 end
266 
267                 if(plr == 1)
268                     plr = 5;
269                 else
270                     plr=plr+5;
271                 end
272                 indicator = indicator + 1;
273             end
274         end
275     end
```

Appendix B

Trace-file Generator - Matlab Script

```
% Trace-file generator
% by Chamitha de Alwis, adapted by Pedro Rocha

%If p is the probability of transferring from Good State to the bad state
%and if r is the probability of transferring from the bad state to the Good
%state, given the p and r values, this code will generate a packet loss
%pattern (with burst losses) and save it to a file named Loss_Pattern.txt.

% p = P(X=1/X=0)
% r = 1 - q = 1 - P(X=1/X=1) = P(X=0/X=1)

%MEAN BURST LENGTH
MBL = [4   6];
%Packet Loss Rate
%        1%   5%  10%  15%  20%
PLR = [0.01  0.05  0.1  0.15  0.2];

%For 8 slice/frame
%maxBL = 7; % maximum burst length

%For 10 slice/frame
maxBL = 9;

for g = 1:length(MBL)
    for h = 1:length(PLR)

        p = 1/(MBL(g)*(1/PLR(h) - 1));
        r = 1/MBL(g);
        total_packs = 10000;

        check = 100; % check the consistency of the trace-file

        while check >= 10

            loss = 0;
            packets = zeros(1, total_packs);

            for i=1:total_packs
                if loss == 0
                    burst = 0;
                    packets(i) = loss;
                    loss = (rand(1) < p); % P(X=1/X=0), if 1, moves to bad state
                elseif loss == 1
                    burst = burst+1;
                    if burst <= maxBL
                        packets(i) = loss;
                        loss = (rand(1) < (1-r)); %  P(X=1/X=1)
```

```
48                      else
49                          packets(i) = 0;
50                          loss = 0; %  forces to get back to the good state is maxBL
                                is reached
51                      end
52                  else
53                      fprintf('error\n');
54                      break;
55                  end
56              end
57
58              received_packs = total_packs - nnz(packets);
59              theo_pack_loss_rate = 1 - r / (p+r);
60              act_pack_loss_rate = 1 - received_packs/total_packs;
61
62              % check the real PLR of the trace-file
63              check = abs(theo_pack_loss_rate - act_pack_loss_rate) /
                    theo_pack_loss_rate * 100;
64
65          end
66
67          fid = fopen(['C:\Users\PRocha\Dropbox\Tese\HM-12.1_EM_JEG_v1.2.1 - Editado
                \bin\vc9\x64\Release\trace_files\new\MBL9\b' num2str(MBL(g)) 'plr'
                num2str(PLR(h)*100)],'w');
68          fprintf(fid, '%d', packets);
69          fclose(fid);
70
71          %packets;
72          %theo_pack_loss_rate = p / (p+r);
73          act_pack_loss_rate = 1 - received_packs/total_packs;
74
75      end
76  end
```

Bibliography

[1] C. V. N. Index, *The Zettabyte Era - Trends and Analysis*. 2015.

[2] J. Apostolopoulos and A. Reibman, "The Challenge of Estimating Video Quality in Video Communication Applications [In the Spotlight]," *Signal Processing Magazine, IEEE*, vol. 29, pp. 160–158, March 2012.

[3] M. Martini, M. Mazzotti, C. Lamy-Bergot, J. Huusko, and P. Amon, "Content adaptive network aware joint optimization of wireless video transmission," *Communications Magazine, IEEE*, vol. 45, pp. 84–90, Jan 2007.

[4] Z. Song, H. Wang, Y. Wen, D. Wu, and K. Lee, "Depth-color based 3D image transmission over wireless networks with QoE provisions," *Computer Communications*, vol. 35, no. 15, pp. 1838 – 1845, 2012. Smart and Interactive Ubiquitous Multimedia Services.

[5] P. Perez and N. Garcia, "Lightweight multimedia packet prioritization model for unequal error protection," *Consumer Electronics, IEEE Transactions on*, vol. 57, pp. 132–138, February 2011.

[6] T.-L. Lin, S. Kanumuri, Y. Zhi, D. Poole, P. Cosman, and A. Reibman, "A Versatile Model for Packet Loss Visibility and its Application to Packet Prioritization," *Image Processing, IEEE Transactions on*, vol. 19, pp. 722–735, March 2010.

[7] C. Hewage, S. Nasir, S. Worrall, and M. Martini, "Prioritized 3D video distribution over IEEE 802.11e," in *Future Network and Mobile Summit, 2010*, pp. 1–9, June 2010.

[8] A. K. Moorthy and A. C. Bovik, "Visual Quality Assessment Algorithms: What does the future hold?," *Multimedia Tools Appl.*, vol. 51, pp. 675–696, Jan. 2011.

[9] ITU, "Recommendation ITU-R BT.500-13: Methodology for the subjective assessment of the quality of television pictures.," pp. 1–46, 2012.

[10] ITU, "Recommendation BT.2021-1: Subjective methods for the assessment of stereoscopic 3DTV systems," pp. 1–31, 2015.

[11] C. Mysirlidis, T. Dagiuklas, D. Kotaranin, S. Gruicic, E. Dumic, P. Rocha, L. da Silva Cruz, and A. Skodras, "STESCAL3D: Subjective evaluation of HD

stereo video streaming using H.264 SVC in diverse laboratory environments," in *Quality of Multimedia Experience (QoMEX), 2015 Seventh International Workshop on*, pp. 1–6, May 2015.

[12] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *Communications Magazine, IEEE*, vol. 46, pp. 78–84, February 2008.

[13] S. R.-D. Mario Vranjes and K. Grgic, "Review of objective video quality metrics and performance comparison using different databases," *Signal Processing: Image Communication*, vol. 28, no. 1, pp. 1 – 19, 2013.

[14] K. Zeng and Z. Wang, "3D-SSIM for video quality assessment," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pp. 621–624, Sept 2012.

[15] S. Chikkerur, V. Sundaram, M. Reisslein, and L. Karam, "Objective video quality assessment methods: A classification, review, and performance comparison," *Broadcasting, IEEE Transactions on*, vol. 57, pp. 165–182, June 2011.

[16] C. Hewage and M. Martini, "Quality of Experience for 3D video streaming," *Communications Magazine, IEEE*, vol. 51, pp. 101–107, May 2013.

[17] M.-J. Chen, D.-K. Kwon, and A. Bovik, "Study of subject agreement on stereoscopic video quality," in *Image Analysis and Interpretation (SSIAI), 2012 IEEE Southwest Symposium on*, pp. 173–176, April 2012.

[18] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, pp. 600–612, April 2004.

[19] M. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *Broadcasting, IEEE Transactions on*, vol. 50, pp. 312–322, Sept 2004.

[20] K. Seshadrinathan and A. C. Bovik, "Motion tuned spatio-temporal quality assessment of natural videos," *Trans. Img. Proc.*, vol. 19, pp. 335–350, Feb. 2010.

[21] J. You, J. Korhonen, A. Perkis, and T. Ebrahimi, "Balancing attended and global stimuli in perceived video quality assessment," *Multimedia, IEEE Transactions on*, vol. 13, pp. 1269–1285, Dec 2011.

[22] P. Joveluro, H. Malekmohamadi, W. Fernando, and A. Kondoz, "Perceptual Video Quality Metric for 3D video quality assessment," in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), 2010*, pp. 1–4, June 2010.

[23] P.-H. Conze, P. Robert, and L. Morin, "Objective View Synthesis Quality Assessment," in *Stereoscopic Displays and Applications* (SPIE, ed.), vol. 8288 of *Proc. SPIE*, (San Francisco, United States), pp. 8288–56, Jan. 2012.

[24] M. Solh, G. Al-Regib, and J. M. Bauza, "3VQM: A vision-based quality measure for DIBR-based 3D videos.," in *ICME*, pp. 1–6, IEEE Computer Society, 2011.

[25] S. Yasakethu, S. Worrall, D. De Silva, W. Fernando, and A. Kondoz, "A compound depth and image quality metric for measuring the effects of packet loss on 3D video," in *Digital Signal Processing (DSP), 2011 17th International Conference on*, pp. 1–7, July 2011.

[26] S. A. Fezza, M. Larabi, and K. M. Faraoun, "Stereoscopic image quality metric based on local entropy and binocular just noticeable difference," in *2014 IEEE International Conference on Image Processing, ICIP 2014, Paris, France, October 27-30, 2014*, pp. 2002–2006, 2014.

[27] Y. Zhao, Z. Chen, C. Zhu, Y.-P. Tan, and L. Yu, "Binocular Just-Noticeable-Difference Model for Stereoscopic Images," *Signal Processing Letters, IEEE*, vol. 18, pp. 19–22, Jan 2011.

[28] S. Wang, F. Shao, F. Li, M. Yu, and G. Jiang, "A Simple Quality Assessment Index for Stereoscopic Images Based on 3D Gradient Magnitude," *The Scientific World Journal*, vol. 2014, 2014.

[29] F. Battisti, E. Bosc, M. Carli, P. L. Callet, and S. Perugia, "Objective image quality assessment of 3D synthesized views," *Signal Processing: Image Communication*, vol. 30, no. 0, pp. 78 – 88, 2015.

[30] C. Hewage and M. Martini, "Reduced-reference quality assessment for 3D video compression and transmission," *Consumer Electronics, IEEE Transactions on*, vol. 57, pp. 1185–1193, August 2011.

[31] C. Hewage and M. Martini, "Edge-Based Reduced-Reference Quality Metric for 3-D Video Compression and Transmission," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 6, pp. 471–482, Sept 2012.

[32] G. Nur and G. Akar, "An abstraction based reduced reference depth perception metric for 3D video," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pp. 625–628, Sept 2012.

[33] H. Malekmohamadi, A. Fernando, and A. Kondoz, "A new reduced reference metric for color plus depth 3D video," *Journal of Visual Communication and Image Representation*, vol. 25, no. 3, pp. 534 – 541, 2014. QoE in 2D/3D Video Systems.

[34] R. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-3, pp. 610–621, Nov 1973.

[35] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st ed., 1992.

[36] M. Solh and G. AlRegib, "A no-reference quality measure for DIBR-based 3D videos," in *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pp. 1–6, July 2011.

[37] A. Mittal, A. Moorthy, J. Ghosh, and A. Bovik, "Algorithmic assessment of 3D quality of experience for images and videos," in *Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE), 2011 IEEE*, pp. 338–343, Jan 2011.

[38] B. Feitor, P. Assuncao, J. Soares, L. Cruz, and R. Marinheiro, "Objective quality prediction model for lost frames in 3D video over TS," in *Communications Workshops (ICC), 2013 IEEE International Conference on*, pp. 622–625, June 2013.

[39] J. Soares, L. da Silva Cruz, P. Assuncao, and R. Marinheiro, "No-reference lightweight estimation of 3D video objective quality," in *Image Processing (ICIP), 2014 IEEE International Conference on*, pp. 763–767, Oct 2014.

[40] Y. Han, Z. Yuan, and G.-M. Muntean, "No reference objective quality metric for stereoscopic 3D video," in *Broadband Multimedia Systems and Broadcasting (BMSB), 2014 IEEE International Symposium on*, pp. 1–6, June 2014.

[41] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: a review and a new contribution," *Proceedings of the IEEE*, vol. 83, pp. 858–876, Jun 1995.

[42] E. Bosc, M. Koppel, R. Pepion, M. Pressigout, L. Morin, P. Ndjiki-Nya, and P. Le Callet, "Can 3D synthesized views be reliably assessed through usual

subjective and objective evaluation protocols?," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pp. 2597–2600, Sept 2011.

[43] "ITU-T H.264: Advanced Video Coding for generic audiovisual services," Feb 2014.

[44] International Organization for Standardization, *ISO/IEC 10918-1:1994: Information technology — Digital compression and coding of continuous-tone still images: Requirements and guidelines*. Geneva, Switzerland: International Organization for Standardization, 1994.

[45] J. Xiong, "Fast coding unit selection method for high efficiency video coding intra prediction," *Optical Engineering*, vol. 52, no. 7, pp. 071504–071504, 2013.

[46] "ITU-T H.265: High Efficiency Video Coding," Sept 2015.

[47] M. Wien, *High Efficiency Video Coding – Coding Tools and Specification*. Berlin, Heidelberg: Springer, Sept. 2014.

[48] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, pp. 1649–1668, Dec 2012.

[49] R. Sjoberg, Y. Chen, A. Fujibayashi, M. Hannuksela, J. Samuelsson, T. K. Tan, Y.-K. Wang, and S. Wenger, "Overview of HEVC High-Level Syntax and Reference Picture Management," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, pp. 1858–1870, Dec 2012.

[50] L. Trudeau, S. Coulombe, and S. Pigeon, "Pixel domain referenceless visual degradation detection and error concealment for mobile video," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pp. 2229–2232, Sept 2011.

[51] H. Hui and C. Tie-Yong, "An efficient error concealment algorithm for intra-frames of H.264," in *Communication Technology (ICCT), 2010 12th IEEE International Conference on*, pp. 576–579, Nov 2010.

[52] C. Hewage, S. Worrall, S. Dogan, and A. Kondoz, "A Novel Frame Concealment Method for Depth Maps Using Corresponding Colour Motion Vectors," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008*, pp. 149–152, May 2008.

[53] B. Yan and J. Zhou, "Efficient Frame Concealment for Depth Image-Based 3D Video Transmission," *Multimedia, IEEE Transactions on*, vol. 14, pp. 936–941, June 2012.

[54] Y. Zhang, X. Xiang, D. Zhao, S. Ma, and W. Gao, "Packet video error concealment with auto regressive model," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, pp. 12–27, Jan 2012.

[55] M. Barkowski, E. Masala, G. V. Wallendael, K. Brunnstrãm, N. Staelens, and P. L. Callet, "Objective Video Quality Assessment —Towards Large Scale Video Database Enhanced Model Development," *IEICE Transactions on Communications*, vol. E98.B, no. 1, pp. 2–11, 2015.

[56] G. Hasslinger and O. Hohlfeld, "The gilbert-elliott model for packet loss in real time services on the internet," in *Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB), 2008 14th GI/ITG Conference* -, pp. 1–15, March 2008.

[57] R. Skupin, C. Hellge, T. Schierl, and T. Wiegand, "Packet level video quality evaluation of extensive H.264/AVC and SVC transmission simulation," *J. Internet Services and Applications*, vol. 2, no. 2, pp. 129–138, 2011.

[58] "View Synthesis Reference Software (VSRS), version 3.5." `http://wg11.sc29.org/svn/repos/MPEG-4/test/tags/3D/view-synthesis/VSRS-3-5`.

[59] P. Le Callet, C. Viard-Gaudin, and D. Barba, "A convolutional neural network approach for objective video quality assessment," *Neural Networks, IEEE Transactions on*, vol. 17, pp. 1316–1327, Sept 2006.

[60] M. Chambah, S. Ouni, M. Herbin, and E. Zagrouba, "Toward an automatic subjective image quality assessment system," *Proc. SPIE*, vol. 7242, pp. 72420E–72420E–12, 2009.

[61] D. Kukolj, D. Dordevic, D. Okolisan, I. Ostojic, D. Sandic-Stankovic, and C. Hewage, "3D image quality estimation (ANN) based on depth/disparity and 2D metrics," in *Computational Intelligence and Informatics (CINTI), 2013 IEEE 14th International Symposium on*, pp. 125–130, Nov 2013.

[62] P. Frank and J. Incera, "A Neural Network Based Test Bed for Evaluating the Quality of Video Streams in IP Networks," in *Electronics, Robotics and Automotive Mechanics Conference, 2006*, vol. 1, pp. 178–183, Sept 2006.

[63] B. Akoa, E. Simeu, and F. Lebowsky, "Using Artificial Neural Network for Automatic Assessment of Video Sequences," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, pp. 285–290, March 2013.

[64] J. Choe, J. Lee, and C. Lee, "No-reference video quality measurement using neural networks," in *Digital Signal Processing, 2009 16th International Conference on*, pp. 1–4, July 2009.

[65] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[66] F. Ribeiro, D. Florencio, and V. Nascimento, "Crowdsourcing subjective image quality evaluation," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pp. 3097–3100, Sept 2011.

[67] C. Keimel, J. Habigt, and K. Diepold, "Challenges in crowd-based video quality assessment," in *Quality of Multimedia Experience (QoMEX), 2012 Fourth International Workshop on*, pp. 13–18, July 2012.

[68] K. Sakic, E. Dumic, and S. Grgic, "Crowdsourced subjective Video Quality Assessment," in *Systems, Signals and Image Processing (IWSSIP), 2014 International Conference on*, pp. 223–226, May 2014.

[69] T. Hossfeld, C. Keimel, M. Hirth, B. Gardlo, J. Habigt, K. Diepold, and P. Tran-Gia, "Best Practices for QoE Crowdtesting: QoE Assessment With Crowdsourcing," *Multimedia, IEEE Transactions on*, vol. 16, pp. 541–558, Feb 2014.

[70] "ITU-T P.913 methods for the subjective assessment of video quality, audio quality and audiovisual quality of internet video and distribution quality television in any environment," Jan 2014.