

SCALABLE UNEQUAL PACKET PRIORITY FOR
REAL-TIME WIRELESS VIDEO STREAMING OVER DDS
BASED MIDDLEWARE

BY
MOHAMMAD F. AL-HAMMOURI

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

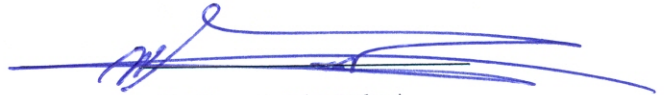
COMPUTER ENGINEERING

November 2014

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN- 31261, SAUDI ARABIA
DEANSHIP OF GRADUATE STUDIES

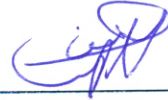
This thesis, written by **Mohammad Fawzi Ahmad Al-Hammouri** under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER ENGINEERING**.

Thesis Committee



Dr. Basem Al-Madani
(Advisor)

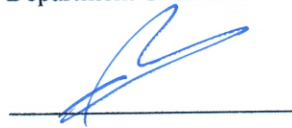
(Co-Advisor)



Dr. Dr. Tarek Sheltami
(Member)

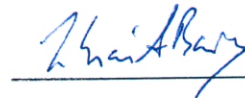


Dr. Ahmad Almulhem
Department Chairman



Dr. Salam A. Zummo
Dean of Graduate Studies

2/12/14



Dr. Zubair Baig
(Member)

Date

© *Mohammad F. Al-Hammouri*

2014

DEDICATION

This work is dedicated to my beloved parents and my wife.

ACKNOWLEDGMENTS

All praise and thanks are due to Almighty Allah. He bestowed upon me health, knowledge and patience to complete this work.

I will take this opportunity to acknowledge, with deep gratitude and appreciation, the encouragement, Valuable time and continuous guidance given to me by my thesis advisor, Dr. Basem Al-Madani. I am highly grateful to my thesis committee members: Dr. Tarek Sheltami and Dr. Zubair Baig for their valuable guidance, suggestions and motivation. Also, I am highly grateful to Dr. CHIH HENG KE (National Kinmen Institute of Technology in Taiwan) for his valuable guidance and suggestions. Thanks to all KFUPM, lab's technicians, engineers for their help and guidance in conducting some of experimental works related the study specially computer engineer department. Acknowledgement is due to KFUPM for the support extended towards my research and for granting me the opportunity to pursue graduate studies. My heartfelt thanks are due to my Family (mother, sisters and brothers) for their prayers, guidance, and moral support throughout my academic life. Finally, special thanks are due to my senior colleagues at KFUPM for their Help and prayers.

TABLE OF CONTENTS

Contents

ACKNOWLEDGMENTS.....	III
TABLE OF CONTENTS.....	IV
LIST OF TABLE.....	VII
LIST OF FIGURES.....	VIII
LIST OF ABBREVIATIONS	XI
ABSTRACT.....	XIII
خلاصة الرسالة.....	XV
CHAPTER 1 INTRODUCTION	1
1.1 Introduction and Problem Description	1
CHAPTER 2 BACKGROUND.....	3
2.1 Video streaming concept	3
2.2 Scalable video coding	5
2.2.1 Temporal Scalability.....	7
2.2.2 Spatial Scalability	7
2.2.3 Quality Scalability (SNR).....	8
2.2.4 Combined Scalability.....	9
2.3 Data Distribution Service (DDS) Middleware	10

CHAPTER 3 LITERATURE REVIEW	12
3.1 Adaptive techniques.....	12
3.1.1 Adaptive video transcoding	12
3.1.2 Rate and Congestion control	13
3.2 Cross-layer technique	14
3.3 Error control and concealment	16
3.4 Application layer solution: middle-ware	16
3.5 Scalable Video Coding	18
CHAPTER 4 SYSTEM DESIGN AND IMPLEMENTATION	22
4.1 System Architecture and Components	22
4.1.1 JSVM Encoder	22
4.1.2 SVEF Streamer	24
4.1.3 DDS Publisher	26
4.1.4 DDS Subscriber.....	27
4.1.5 SFEV Receiver.....	27
4.1.6 JSVM Decoder.....	27
4.2 Quality of Service (QoS).....	28
4.2.1 DEADLINE.....	28
4.2.2 RELIABILITY	29
4.2.3 HISTORY	29
4.2.4 LIFESPAN	30
4.2.5 TIME_BASED_FILTER.....	30
4.2.6 DURABILITY.....	30
4.2.7 LIVELINESS	30
4.2.8 PRESENTATION	31
4.2.9 PARTITION.....	31
4.3 Proposed Methodology and system Behavior	32
CHAPTER 5 EXPERIMENTAL WORK AND EVALUATION	41

5.1	Experiment setting and components	41
5.2	Evaluation Framework and tools	43
5.2.1	Performance metrics	43
5.2.2	Evaluation tools for Scalable Video Coding SVC	45
5.3	Performance Evaluation and Result	46
5.3.1	PSNR Results	49
5.3.2	Frame Delay Results	59
5.3.3	Jitter Results	62
 CHAPTER 6 CONCLUSION AND FUTURE WORK		65
 APPENDIX A VIDEO ENCODING AND INSTRUCTION		67
A.1	Video Encoding	67
A.1.1	Temporal type.....	67
A.1.2	Spatial type:	70
A.1.3	SNR (Quality Scalability).....	72
A.2	Video publishing and receiving	74
A.3	NALU Filtering and Decoding	75
A.4	PSNR , Delay and jitter Calculation	76
 APPENDIX B DDS IMPLEMENTATION AND QOS		78
B.1	XML QoS Profile for DDS_VideoStream	78
B.2	Sample OF DDS Implementation with SVEF	85
 REFERENCES		89
 VITAE		95

LIST OF TABLE

Table 4.1: Summary of the used QoS.....	32
Table 5.1: Source Video Parameter.....	42
Table 5.2: PSNR to MOS Conversion.....	44
Table 5.3: Average frame delay in (ms) for different SVC types	61
Table 5.4 Average jitter in (ms) for different SVC type	64

LIST OF FIGURES

Figure 2.1: An example of GOP, Black frame represents I-Type Frame	4
Figure 2.2: SVC Encoding and decoding to different scalability types. [6].....	6
Figure 2.3: Decoding video to different qualities depends on receiver capabilities..	6
Figure 2.4: Hierarchical Temporal scalability. [28].	7
Figure 2.5: Spatial Scalability. [38].....	8
Figure 2.6: Quality Scalability. [31].....	9
Figure 2.7: DDS environment. [39].....	11
Figure 3.1: Using Transcoder buffer at the sender side after encoding. [5].....	13
Figure 3.2: Proposed architecture Design in [33] for cross-layer approaches.....	15
Figure 3.3: H.264/SVC over Data Distribution Service (DDS). [25].....	18
Figure 3.4: Example of using partition QoS for encoded video frame. [3].	21
Figure 4.1: General System Archctiture	22
Figure 4.2: Example Of Temporal Encoding	23
Figure 4.3: Example Of Spatial Encoding.....	24
Figure 4.4: Example Of Quality Encoding.....	25
Figure 4.5: Data Structure For NALU Packet	25
Figure 4.6: DDS Packet structure (IDL File)	26
Figure 4.7: Packet priority inside each scalability layer using Spatial SVC	33
Figure 4.8: Packet priority inside each scalability layer using Quality SVC	34
Figure 4.9: Example of sending different sub-streams using DDS Middleware.....	36
Figure 4.10: Example 1, Partitioing for temporal scalability	36
Figure 4.11: Example 2, Partitioning for spatial scalability	38
Figure 4.12: Example 3, Partitioning for Quality(SNR) scalability	39
Figure 4.13: DDS with SVEF tool for scalable real-time video streaming.....	40
Figure 5.1: Experiemt components.....	42
Figure 5.2: General scheme for the experimental and evaluation framework.....	43
Figure 5.3: SVEF Framework Structure.....	46
Figure 5.4: Supported scalable layer using tempotral SVC	47

Figure 5.5: Supported scalable layer using Spatial SVC.....	47
Figure 5.6: Supported scalable layer using SNR (Quality) SVC	48
Figure 5.7: Supported scalable layer using Combined SVC	49
Figure 5.8: PSNR, Temporal SVC, Subscriber=1, AVG =36.85 dB	50
Figure 5.9: PSNR, Temporal SVC, Subscriber =4, AVG = 33.5 dB	50
Figure 5.10: PSNR, Temporal SVC, Subscriber =8, AVG = 24.85 dB	51
Figure 5.11: PSNR, Temporal SVC, Sub =12, AVG = 21.8 dB	51
Figure 5.12: Y- PSNR Results, Temporal type	52
Figure 5.13: PSNR, Spatial SVC, Subscriber =1, AVG = 35.73 dB.....	52
Figure 5.14: PSNR, Spatial SVC, Subscriber =4, AVG = 33.06 dB.....	53
Figure 5.15: PSNR, Spatial SVC, Subscriber =8, AVG = 26.29 dB.....	53
Figure 5.16: PSNR, Spatial SVC, Subscriber =12, AVG = 25.22 dB.....	53
Figure 5.17: Y- PSNR Results, Spatial type.....	54
Figure 5.18: PSNR, SNR SVC, Subscriber =1, AVG = 38.68 dB	54
Figure 5.19: PSNR, SNR SVC, Subscriber =4, AVG = 32.88 dB	55
Figure 5.20: PSNR, SNR SVC, Subscriber =8, AVG = 29.97 dB	55
Figure 5.21: PSNR, SNR SVC, Subscriber =12, AVG = 27.07 dB	55
Figure 5.22: Y- PSNR Results, SNR type.....	56
Figure 5.23: PSNR, Combined SVC, Sub=1, AVG = 36.22 dB	56
Figure 5.24: PSNR, Combined SVC, Sub=4, AVG = 33.64 dB	57
Figure 5.25: PSNR, Combined SVC, Sub =8, AVG =30.1 dB	57
Figure 5.26: PSNR, Combined type, Sub=12, AVG = 27.36 dB	57
Figure 5.27: Y- PSNR Results, Combined SVC type	58
Figure 5.28: Video Snapshots for different SVC types and Subscribers	59
Figure 5.29: Frames Delay, Temporal SVC.....	60
Figure 5.30: Frames Delay, Spatial SVC.	60
Figure 5.31: Frames Delay, SNR SVC.....	61
Figure 5.32: Frames Delay, Combined SVC.....	61
Figure 5.33: Frame Jitter, Temporal SVC.	62
Figure 5.34: Frame Jitter, Spatial SVC.	63
Figure 5.35: Frame Jitter, SNR SVC.....	63

Figure 5.36: Frame Jitter, Combined SVC..... 63

LIST OF ABBREVIATIONS

AP: Access Point.

AU: Access Unit.

ARQ: Automatic Repeat Request.

AVC: Advanced Video Coding.

CGS: Coarse Grain Scalability.

CORBA: Common Object Request Broker Architecture.

DDS: Data Distribution Service.

DID: Dependency ID.

DR: Data Reader

DW: Data Writer.

FEC: Forward Error Correction.

FGS: Fine Grain Scalability.

GOP: Group of Pictures.

GOV: Group of Video.

JSVM: Joint Scalable Video Model.

MGS: Medium Grain Scalability.

MOS: Mean Opinion Score.

NAL: Network Abstraction Layer.

NALU: Network Abstraction Layer Unit.

PSNR: Peak Signal to Noise Ratio.

QID: Quality ID.

QoS: Quality of Service.

QuO: Quality Objects.

SVC: Scalable Video Coding.

SVEF: Scalable Video Evaluation Framework.

TID: Temporal ID.

TFRC: TCP Friendly Rate Control.

VLC: Video Layer Coding.

ABSTRACT

Full Name : Mohammad Fawzi Ahmad Al-Hammouri
Thesis Title : SCALABLE UNEQUAL PACKET PRIORITY FOR REAL-TIME WIRELESS VIDEO STREAMING OVER DDS BASED MIDDLEWARE.
Major Field : Computer Engineering
Date of degree : Novemer.2014

The great advances in wireless communications over the past years facilitate the wireless video transmission in real-time basis. However, wireless video transmission still needs high capacity channels and good techniques that mitigate the error-prone wireless channels effect. Recently, different techniques have been proposed for efficient real-time video streaming over wireless and heterogeneous networks. The Scalable Video Coding (SVC) or layer coding is proposed to achieve graceful degradation for video quality in lossy transmission environments by dropping part of the enhancement layers without re-encoding. SVC faces the problem of unequal layer protection, where the base layer, inside sub-stream, may be dropped while the enhancement layer could be dropped first. Furthermore, the whole scalable layer may be dropped while there is a chance to drop packet by packet. These limitations lead to a significant effect on the graceful degradation mechanism. In this paper, an application-layer and middleware-based solution is proposed to implement SVC, which increases network reliability, flexibility and provides Quality of Service (QoS) control. Specifically, due to the real-time and QoS support of Data Distribution Service (DDS) middleware, it is used to implement the four types of SVC scalability, Viz. Temporal, Spatial, Quality and Combined Scalability. Furthermore, an open source evaluation tool called Scalable Video-streaming Evaluation Framework (SVEF) is used to assess the video transmission performance, with performance metrics, Viz. Peak Signal to Noise Ratio (PSNR), Mean Opinion Score (MOS), frames delay and jitter. The experiments' results show a graceful degradation to the video quality when using

the DDS-based SVC, especially when the number of receivers are increased. Furthermore, we notice that the video quality is sensitive to the encoding type, thus, SNR performs better due to the efficient encoding, which leads to the smallest video encoding size and packet size.

خلاصة الرسالة

الاسم الكامل: محمد فوزي احمد الحموري

عنوان الرسالة: تدفق الفيديو المتحجم غير متكافئ الأولوية في الوقت الحقيقي

التخصص: هندسة الحاسب الآلي

تاريخ الدرجة العلمية: نوفمبر 2014

سهل التطور الكبير والمستمر في الشبكات اللاسلكية من عملية نقل الفيديو في الوقت الحقيقي . إن عملية إرسال الفيديو في الوقت الحقيقي عن طريق الشبكات اللاسلكية يحتاج لشبكات ذات سعة عالية وظروف جيدة ، لكن هذا النوع من الشبكات يعرف بالتقلب والتغير بالسعة المتاحة مع مرور الوقت، هذه الأسباب تؤدي إلى فقدان أجزاء من حزم الفيديو. تم إقتراح طرق عديدة في البحوث للحصول على كفاءة وجودة عالية في تدفق الفيديو في الوقت الحقيقي عبر الشبكات اللاسلكية والغير المتجانسة . حديثاً تم طرح طريقة ترميز الفيديو المتحجم SVC (إس في سي) أو ما يسمى بـ ترميز الطبقات للحصول على انحطاط رشيق لجودة الفيديو خاصةً في بيئات النقل التي تكون نسبة ضياع البيانات فيها عالية كشبكات واي فاي، تعتمد هذه الطريقة على حذف جزء من طبقات التعزيز في الفيديو المرسل دون الحاجة إلى إعادة الترميز. تواجه طريقة ال إس في سي بعض الصعوبات تتمثل في عدم تساوي أولوية الحماية لطبقات الفيديو، هناك احتمالية لفقدان طبقة القاعدة في الفيديو بينما يجب إسقاط طبقات التعزيز أولاً في حال هبوط مستوى الشبكة. علاوة على ذلك، يمكن فقدان طبقة كاملة من الفيديو بينما هناك امكانية لإسقاط جزء منها، نتيجة لذلك تم استخدام حلول على مستوى التطبيقات (Application Layer) باستخدام وسيط (Middleware) وذلك لاستخدام وتنفيذ طريقة SVC مما يعطي مرونة اكبر في استخدام الشبكة ويزيد من توفرها بتوفير ما يسمى جودة الخدمة (QoS) الذي يوفرها الوسيط. في هذا العمل تم إختبار عملية تدفق الفيديو في الوقت الحقيقي باستخدام تقنية جديدة لترميز وضغط الفيديو وهي ترميز الفيديو المتحجم (إس في سي) و استخدام طريقة الوسيط لتوزيع ونشر البيانات (دي دي إس). تم إختبار الطريقة المقترحة عبر الشبكة اللاسلكية (واي فاي) رقم IEEE 801.11 وكذلك إختبار أربع أشكال للترميز وهي : المؤقت، الحيزي، إس ان ار، والمتحد. تم إستخدام أداة التقييم (SVEF) في عملية التقييم وإستخراج النتائج بعد أن تم إدراج الوسيط (دي دي إس) وذلك لتشمل طريقة الناشر والمشارك (Publisher/Subscriber) الذي يوفرها هذا الوسيط. تم استخدام مقاييس الأداء لتقييم كفاءة الفيديو مثل: بي إس

إن ار (PSNR) ، ومستوى ال MOS، زمن وصول حزم الفيديو و jitter. أظهرت النتائج العملية إنحطاط رشيق في جودة الفيديو نتيجة لإستخدام طريقة ترميز الفيديو المتحجم خصوصا عند ازدياد نقاط الاستقبال. علاوة على ذلك فإن جودة الفيديو تعتمد على نوع طريقة الترميز المستخدمة. أثبتت النتائج أن طريقة ال إس ان ار تعطي افضل نتائج وذلك نتيجة لجودة عملية الترميز والذي ينتج عنه حجم فيديو مشفر أقل وحجم حزم اقل.

CHAPTER 1

INTRODUCTION

1.1 Introduction and Problem Description

Real-Time video streaming that is needed in our life has a lot of difficulty since it requires a high bandwidth channel and rigorous transport delay. Furthermore, wireless environment is widespread due to flexibility and low cost compares to other wired communication technique. In Live Streaming, we must maintain end-to-end continuous video playback to get a continuous video without discontinuities, which is difficult over wireless networks because of error-prone channel nature, frequent handovers and more delay due to congestion. In fact, continuous live video is more affected by delay than packet losses [1]. High-Quality real-time video streaming over wireless channel is an important and challenging research area nowadays. Multimedia streaming is highly sensitive to delay; it requires a QoS to divide the allocated resources fairly and maintain delivery on time. Throughput and packet losses aren't the only factor affect video streaming over wireless, delay and jitter are also important parameters in multimedia streaming since. In live video, if the packet isn't received in the required time; it becomes useless [2]. Wireless networks have a time varying conditions and instability due to high error rate, collision, contention and attenuation [4].

In video streaming, the content of the video (static or dynamic) and decoding method are important factors in transmission performance, the compression type and adaption technique also affect the process. Video by nature is described as bursty since its consist of frames, video frame can't be decoded or played out at receiver side until most packet for corresponding frame is received successfully on time. In wireless, the allocated bandwidth between any two AP is unknown and changed by the time, thus congestion will occur and packets will be lost when the sender sends faster than available bandwidth, in the contrary the quality will reduced if the sending is slower. The end-to-end packets delay

in the network oscillate from one packet to another which lead to packet jitter, Jitter lead to jerks in the reconstructed video at receiver side [3].

Several problems was associated with streaming real-time video over wireless. Real-time streaming requires an adaptation method to deals with varying network condition and available bandwidth. As A result; different approaches have been proposed to maintain a high quality real-time video streaming over error-prone networks. Adaptive video streaming is used to solve channel capacity and condition variation based on video transcoding and rate control, Crosse layer approache is proposed by exchanging information between different layer to get better decision about network status for efficient channel allocation and congestion avoidance, Error control and concealment is used to decrease errors effect on the receiving video like ARQ and FEC techniques, many solutions depends on the new proposed video coding technique: H.264/SVC Scalable video coding which based on unequal layer protection where video stream is encoded to multiple layer where the upper enhancement layers are dropped one by one in case of limited network capability. SVC or layer coding face a problem of unequal layer protection, base layer may (inside sub-stream) be dropped while enhancement layer must be dropped first. Furthermore, whole scalable layer may dropped while there is chance to drop packet by packet which prevents graceful degradation. Thus, application layer solution using a middleware is used to increase networks reliability and flexibility using QoS.

The work presented by this thesis explore and study the behavior of transmitting video over wireless based on DDS middleware with publisher subscriber model; Scalable video coding is used with its for type: temporal, spatial, SNR (Quality) and combined scalability to achieve the required scalability for real-time video transmission. In this work, I propose unequal packet dropping priority inside each encoded SVC layer where every packet has two dimensional priority based on enhancement layer ID (LID) and temporal ID (TID) inside each layer, dropping priority increases as LID and TID increases. We develop a Prototype based on DDS publisher subscriber with QoS to implement the proposed approach then evaluate Scalable wireless video transmission using different types of SVC scalability.

CHAPTER 2

BACKGROUND

In this chapter I will give an introduction that discuss and explain the main concepts and techniques related to my thesis work

2.1 Video streaming concept

Video is a very important media used for communication and amusement in long times. Digital video make a revolution in streaming video and compression, which open the door for a lot of application. Video represents a heavy load data and its transmission faces a lot of problem like time varying bandwidth, losses and jitter.

There are different types of video depends on its use, Interactive application like video conference or non-interactive like watch alive sport game, these application required a real-time live video without discontinuity or interruptions. On demand streaming is another type of video, where the video content is pre-encoded and stored to be viewed like watching film stored on DVDs On-demand streaming exist also over internet like watch a video on YouTube; the user can wait until video buffered on his PC, these type of video doesn't require a real-time constraint [36].

In real-time live video streaming; there is what called playback time, video packet must be received within playback time to be useful and maintain a live streaming. In addition, the source video in live streaming must be captured, encoded, transmitted then decoded with delay constraints; they need more complex computation and resources. Video compression is an important concept in video streaming by removing or reducing redundant contents from video file so that it will be effectively sent over networks, it's very useful in video streaming by reducing storage requirement and transmission capacity. Video compression takes the advantage of similarities and redundancies in video data, always successive frames have a redundant data since it contains the same objects but with some motion. In Addition, video compression concentrates on reducing the irrelevancy

between frame after compression; this main that it codes the features that is more important and don't waste extra space to code information that less important or irrelevant [36].

Video data consist of consecutive frames each one is coded as a separate image, as we mention; we explore the similarities between adjacent images and get High compression ratio, the most popular approaches for exploiting the similarities between frames are: prediction frame based on previous coded one then coding the error resulting from this prediction. Adjacent and successive frames have same imaginary data but with different location because of motion, frames prediction are done by fist estimate the motion between frames, which is called motion estimation then make suitable prediction to compensate for the motion and this called motion-compensated prediction.

There are three types of pictures (Frames) in the video sequence GOP (Group of pictures) see Figure 2.1. Base Layer in SVC consist of consecutive GOPs, each GOP begins with Intra-Coded Picture (I-Frame), I-Frames is encoded independently of other frames and called reference picture (Other types of frame depend on it) in GOP. Predictive frames (P-Frames) contain compensation difference information and coded based on prior I or P frames. The third type is Bidirectional predictive pictures (B-Frames) which coded based on previous and the following I and P frames. In SVC, the base layer consists of I or P frame and the enhancement layers consist of P and B frame.

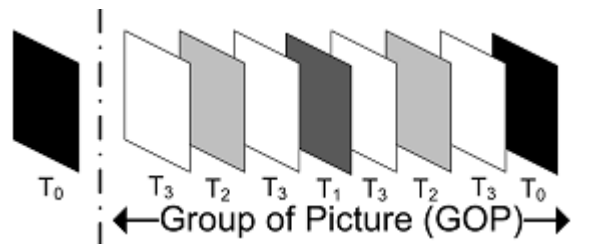


Figure 2.1: An example of GOP, Black frame represents I-Type Frame

Video codec is a hardware device or software solution that enables the encoding (compression) of video then decoding received one to reconstruct original transmitted video. There are a widely used software codecs that used a video compression standard like MPEG 1 and 2, H.261, H.263, MPEG-4, H.264/AVC and the nowadays H.264/SVC extension of AVC standard is the primary used method for video streaming.

Finally, I would like to mention one important concept in video streaming which is the using of Playout buffer [36], it's common in video streaming at receiver side to wait some time before playback begins; this period of time called playback time. A video can be viewed by a sequence of media samples, buffering gives some more time for frames to be received and relaxed. Jitter reduction is one of the main benefits from playout buffer, the variation in network condition make the time for the packet to travel between sender and receiver vary also, playout buffer gives more deadline for packets to be received within constant time; this will eliminate jerks in the reconstructed video and reduces the number of packet that received after playback time. Furthermore, the extended presentation deadline gives the ability to retransmit packets when it was lost.

2.2 Scalable video coding

Different scalability types are already exist in the previous encoding standard MPEG2, MPEG4 and H.263 like temporal scalability in terms of maintain different bit rate. Including different scalability types like spatial or Quality (SNR) increase the complexity of encoder and decoder then decrees coding efficiency. SVC as an extension to H.264/AVC come to solve these issues and provides different scalabilities in term of bitrate, resolution and quality with maintain light increasing in coding complexity and increasing its performance.

SVC is encoding and compression technique for high quality video bit stream proposed in 2005 as encoding standard. SVC is a charming solution for modern video streaming, it's provide a high compression efficiency and more robust with error. The scalable feature means the adaption of the transmitted video to different channel condition by removing part of video bit stream to adapt available varying networks capability [28], the resulting bit stream will represents a another valid bits stream that will be used in another receiver which has lower resources capabilities. IN SVC the bit stream is represented by one base layer and one or more enhancement layers, decoding the maximum number of layers mean achieving the high quality, Figure 2.2 explain SVC encoding process.

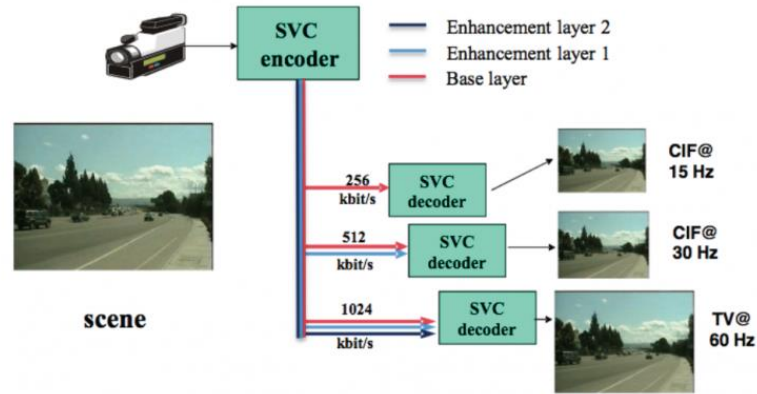


Figure 2.2: SVC Encoding and decoding to different scalability types. [6].

In SVC, the video is encoded once with the highest requirement (Quality and resolution) and then decoded to multiple receivers with different rates and resolutions. The main desire of SVC is the adaption to different receiving capabilities in term of processing, display and varying Channel condition like in heterogeneous environment, see Figure 2.3.

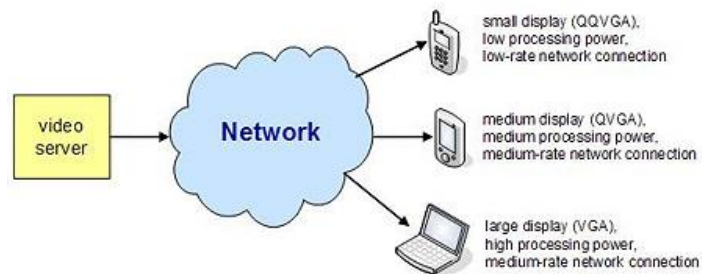


Figure 2.3: Decoding video to different qualities depends on receiver capabilities

Before we discuss scalability type in details, we need to talk about video encoding process. In SVC, the video is encoded to base layer and one or more enhancement layers. The scalability is achieved by decoding the layer depends on receiver capability and discard the other, a full quality is achieved by decoding the whole layers. H.264 defines network abstraction layer (NAL) which format the compressed bit stream produced by Video coding Layer (VCL) into NAL units (NALU). One or more NALU form what's called Access Unit (AU), Every NALU represents VCL type which contains encoded data or Non-VCL which contains additional Information [28], [42].

There are three types of Scalability in SVC standard: Temporal, Spatial and Quality (SNR) scalability in addition to the combination of them. In Temporal scalability, the transmitted bit stream represents the source content with different bit rates while spatial type the bit stream is transmitted with different resolutions. In quality scalability, the bit stream maintains the same bit rate and resolution ratio but with different qualities (fidelity). Thus, the resulting images have the same frame rate and resolution but with different qualities [31].

2.2.1 Temporal Scalability

In the temporal scalability, the AU unit is partitioning into one base layer with $T=0$ and one or more enhancement layers, the enhancement layer get $T=1$ and incremented by one for the above layers. Higher temporal ID mains higher available Bit rate. Every enhancement layer is decoded based on the previous base and enhancement layer(s). The decoder at receiver side is free to select the appropriate T_{ID} and discard and NULUs above it (Discard all layers above). One of the main type of temporal scalability is the hierarchical prediction where the enhancement layers are encoded as B picture, the prediction for each picture depends on the preceding and succeeding picture see Figure 2.4. A set of pictures between two base layer ($T=0$) are called Group of Pictures (GOP) [28], [31].

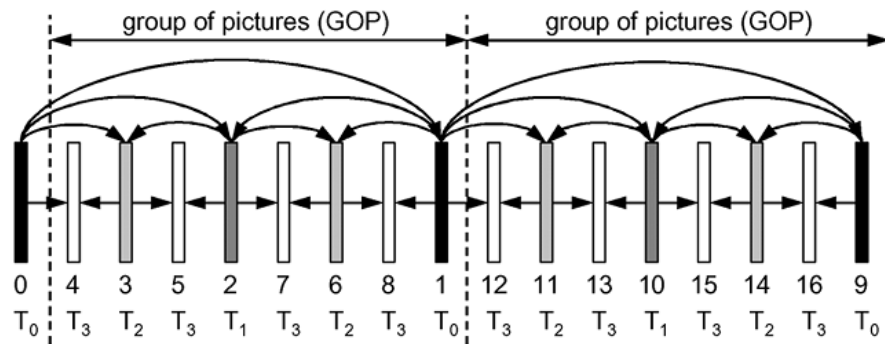


Figure 2.4: Hierarchical Temporal scalability. [28].

2.2.2 Spatial Scalability

In spatial scalability, SVC uses the Multi-layers approach like in MPEG-2, H.263, and MPEG-4. Each layer is responsible to support spatial resolution and referred to it by

dependency ID (DID), see Figure 2.5. Base layer supports the minimum resolution with $D=0$, the first enhancement layer take $D=1$ and D incremented by one for the following layers. In each layer, the motion compensation and intra prediction coding are used. Furthermore, inter-layer prediction was used to improve coding efficiency [28], [38].

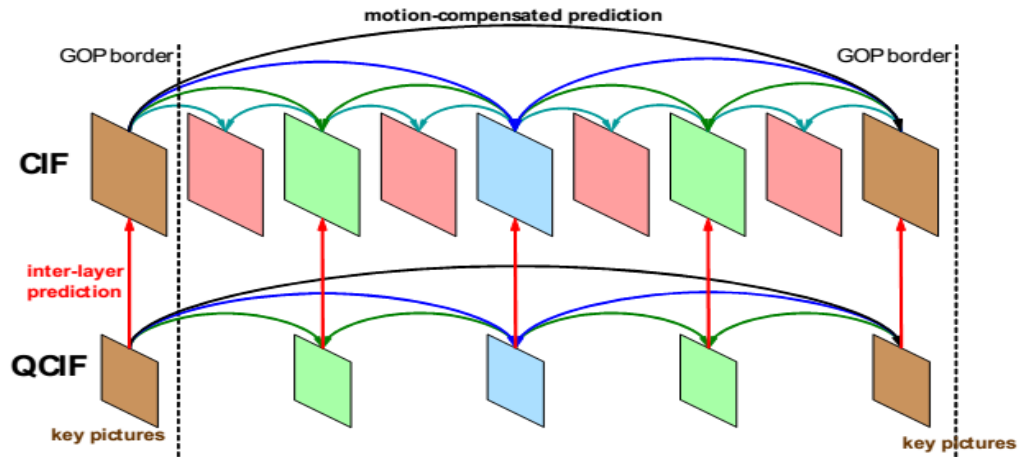


Figure 2.5: Spatial Scalability. [38].

We see from fig. 2.5 that the base layer contains QCIF resolution (176 x 155) with coded frame rate equal 15Hz, the enhancement layer provide CIF (352 x 288) with frame rate of 30Hz. We see that the red arrow in the middle indicates the inter-layer prediction which used between base and enhancement layer. Motion compensation prediction is used inside layer.

2.2.3 Quality Scalability (SNR)

Quality Scalability represents spatial type of Spatial Scalability where the enhancement layers have the same resolution. As in previous type, higher number of enhancement layers means high quality. Every layer has QID, base layer has $QID=0$, the first enhancement layer take $Q=1$ and Q incremented by one for the following layer [28].

There two types of Quality Scalability, Coarse grain (CGS) and Medium Grain Scalability (MGS). In CGS, the video is encoded to set of layers using different quantization parameters, the encoder has the ability to select the required bit rate by

increasing or decreasing the number of layers. MGS is used to increase bit stream flexibility and increase coding efficiency. In MGS each enhancement layer is partitioning to multiple sub-layers, this increases scalability and allow the application to decode the suitable sub-layers to get the required quality [31], see figure 2.6.

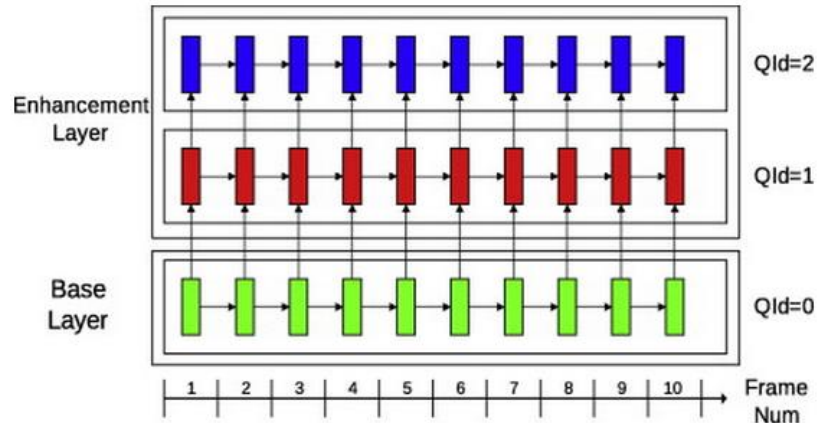


Figure 2.6: Quality Scalability. [31].

Using MGS, the NULUs can be dropped and discarded with still allow frames decoding but with decrees quality level, by this we achieve scalability at packet level. The previous features make MGS more flexible and more robust a gains error compare to CGS scalability. FGS coding supports a rate interval instead of a limited set of rate points, and a sub-stream for each rate (inside the supported interval) can be extracted.

2.2.4 Combined Scalability

The concept of spatial and quality scalability can be combined for supporting different spatial-temporal resolutions and rate points [38]. One important different between dependency layer (Spatial) and quality layer is that switching between different resolutions is only allowed at defined points, but the switching between different qualities is permissible in any access unit [28]. Using combined type increases flexibility by introducing NALUs that represent improvement signal for picture which can be truncated at any points [44].

2.3 Data Distribution Service (DDS) Middleware

Data Distribution service (DDS) is a data space, where the data properties and structure are defined by what's called a topic, it's a set of specifications standardized by Object Management Group (OMG). A topic defines a set of related data that have the same data-structure and property. For example, a topic with name "Pressure" used to store data samples that are taken from distributed pressure sensors. The main objects in DDS are: *Publisher*, which is responsible for writing data in the middleware and *Subscriber* that reads the data-sample. Publisher consists of data writers, where each one writes a data for a specific topic. On the other side, subscriber contains data readers that read data from DDS for a specific topic [39], see figure 2.7.

Publisher subscriber architecture provides a synchronous means of communication for a specific topic, we don't care about data source or where it will go, Subscriber only subscribes to a particular topic to get data. In DDS you can get the data any time regardless of the location (publishers and subscribers can be located anywhere) [24]. This makes DDS more suitable for large scale real-time heterogeneous environments.

Unlike other traditional point-to-point communications like client-server where the server is the data producer and this makes what's called a server bottleneck, if the server becomes down there is no other source of data (single point of failure). In addition, a communication system with a single source (server) will lead to high load especially in many-to-many communication [40]. In the Publisher subscriber model the event is delivered to the subscriber immediately after it's available without the need for a request like a client-server architecture, the communication is asynchronous and there is no need for waiting acknowledgment from the subscriber. As a result, the sender can move to the next receiver in a short period of time without the need for synchronization. One important point of using the Publisher/subscriber model is reliability, the publisher produces one copy of the event and the event broker is responsible for delivering it to many subscribers. The previously mentioned features make DDS middleware more suitable for large scale real-time heterogeneous environments [21].

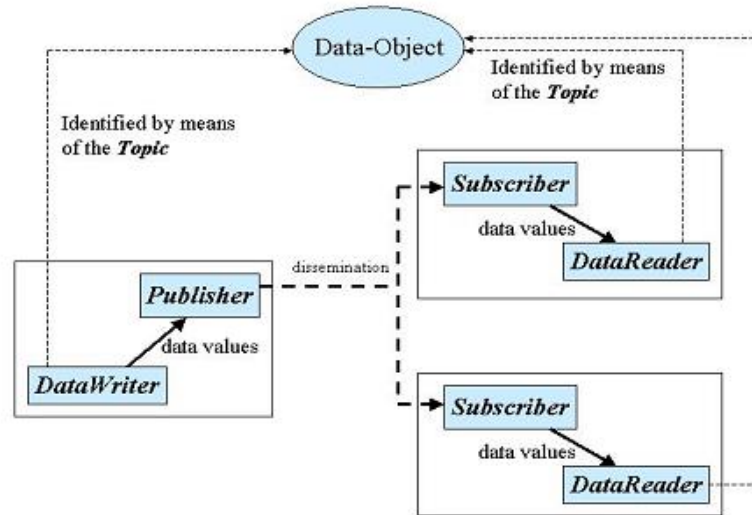


Figure 2.7: DDS environment. [39].

DDS middleware supports variety QoS to achieve real-time requirement and high performance transmission. In publisher/Subscriber model, every topic is characterized by a set of QoS to control data distribution. For example **LIFESPAN** determines the max time that the data can remain in Middleware after it was written, **HISTORY** QoS determines the max data sample that can be stored, new data sample will replace old one when the data sample exceed History defined number [39]. In DDS, when an application wants some data for specific topic, it just feeds DDS with the topic then DDS takes care of configuring all underlying network and required QoS.

CHAPTER 3

LITERATURE REVIEW

Several problems associated with real-time video streaming over wireless. Consequently, different approaches are proposed to maintain a high quality real-time video streaming over error-prone networks. In the next sections, I will summaries the recent solutions for real-time video over wireless that mentioned in the literature.

3.1 Adaptive techniques

Different adaptive techniques exist in the literature for streaming video over wireless network; we summarize it in the following:

3.1.1 Adaptive video transcoding

Due to variation in capability and characteristics in the current wireless network, a lot of algorithms were proposed to adapt this variation especially for video transmission where encoding and compression process must also be adopted. It's very hard for video encoder to produce different videos for different channels that have varying conditions. One of the main techniques for wireless channel adaption is transcoding technique; a Video transcoder can be used in the video transmission source to scale the bit rate, it's convert the previous encoded video to another low quality format. Transcoder adjust some parameters like video quality, bitrate and resolution. For example the video frame rate is reduced if the channel is bad and it will be the same for normal channel. In [5] they proposed a combination of ARQ techniques for error control and video transcoding, they applied content based approach to calculate frame bit rate then they adjust frame rate depending on channel conditions and bandwidth, figure 3.1 shows block diagram for their system. They achieved a perfect results. One of the main drawbacks for this technique that it's reduce the visual quality for the receiving video, also the process of encoding is complex.

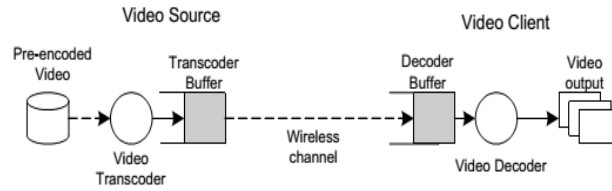


Figure 3.1: Using Transcoder buffer at the sender side after encoding. [5].

3.1.2 Rate and Congestion control

Rate control is an important issue for streaming video over wireless channel, where packets are dropped due to channel error in the physical layer and congestion. Using rate control mainly reduce transmitting frame rate (without change video format) to adapt changing in channel condition. Rate control is not suitable for real-time video streaming since it's requires reinitiating encoder parameter for changing to the new rate.

The popular control scheme in wired transmission is the TCP Friendly Rate Control (TFRC). In wireless network TFRC doesn't distinguish between losses due to channel error or congestion [7]. To solve this problem, an integration between TFRC with another network layer was done in [7] to form a comprehensive design for efficient real-time video over multi-hop wireless network, varying received channel information used to adapt encoding parameters and minimize packet losses. [8] Also proposed TFRC friendly protocol (WMSTFP) over wireless to differentiate between congestion and erroneous losses.

Retransmission lead to further congestion in wireless due to error nature [9], they mentioned a solution by delivering part of corrupted packets to application layer rather than reduced rate immediately, this will decrease retransmission and then congestion but at the cost of delivering error bits in packets payload.

A Multiple of simultaneous TFRC connections are proposed in [10] as a solution for video streaming over wireless, the main pros of their approach: it's end-to-end and doesn't require changes on network protocols or infrastructure, fully utilize the wireless available bandwidth where packets size and number of connections are provided. The one con is the complicated control procedure.

In [11], they study the behavior of MPEG4 VBR video coder after using rate control mechanism (RC-VBR) over ZigBee wireless network. They conclude that using their rate control algorithm a void un-predictable rate variation and remove delay that results from coding. In [12], they used adaptive approach for real time video streaming over wireless. They mentioned that achieving high quality video streaming we must adjust encoding process at sender side based on channel conditions. They proposed an adaptive approach to get network status depends on information from MAC layer since it's hard to predict congestion depends on physical layer only.

3.2 Cross-layer technique

As mention earlier, wireless networks are subjected to packet loses specially for high load data like video. A lot of link adaption method in wireless LAN depends on the point that when network becomes congested this is due channel error rather than collision. In addition, they don't consider the decreasing of the physical data rate on video quality because they don't consider the transmitted video properties (video codec). Thus the idea of Cross Layer Link Adaption (CLLA) was proposed as new link adaption algorithm, where different layers are incorporated and exchange information upon them especially between application layer and lower layer, this information allow more potential to do better decision [13, 14, 16, 33].

In [13], they proposed a Cross-layer link adaption as a solution for wireless real-time video transmission, which doesn't based only on mac level statistic, but also on video coded and quality of the received video such as: Error resilience, Quantization, GOV (group of video) structure and decoding. Furthermore the collision probability measures are incorporated with physical channel condition to determine physical data rate.

In [33], they shows an architecture cross layer based design to improve H.264 video streaming over wireless. The application layer, mac layer and physical layer are incorporated for solving the problems, see figure 3.2. Application layer provide the optimal allocation for bits which will minimize distortion, also it manages packet priority. Network estimator in mac layer determine by estimation the required network like bandwidth, furthermore, it maps the transmitted frame into class service based on its priority.

Several Qualities of service are proposed for real time transmission in different layers such as TCP and RTP, where it received QoS from above layer (Application). QoS is applied in [14] using cross-layer approach on transmission of MPEG video; they apply cross-layer on both application and data link layer. They use priority based QoS depending on frame type (I, P and B frame) and gives each frame type a class number; the frame class is encapsulated in RTP payload. In case of packet dropping an ARQ decision will be taken based on priority Adaptive QoS.

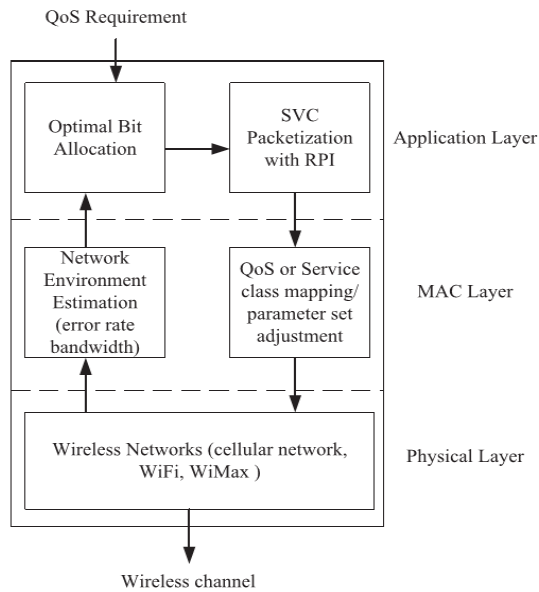


Figure 3.2: Proposed architecture Design in [33] for cross-layer approaches

Based on cross-layer [15] proposed a new scheme for packets in the application layer where it's filled in equal size and integer number of radio link protocol packet, in their method; the unimportant packets are dropped as adaptations to bad channel condition. [16] Mentioned that cross-layer is widely used for efficient resource allocation and maximize data rate especially for wireless environment like LTE. They proposed framework based on cross-layer approach consists of an integrated factor: system throughput, QoS constraint and fairness scheduling, this design used for dynamic radio resource allocation to multiple users. Furthermore, it is used to effectively choose encoding parameters as an adaptation to varying channel condition.

3.3 Error control and concealment

Error control and concealment for video transmission over wireless becomes an important issue because error degrades the quality of received video in this type of channel. Transmission over wireless has two main kinds of error: random bit and erasure error. The former is caused by a fault in physical, where bit inverting, inserting or deleting. The second one is due to packet loses, burst error or system failure. Retransmission based on automatic repeat request (ARQ) was used in [17] to solve the problem packet losses. An Adaptive sub-packet forward error correction (SPFEC) was proposed and used in [18] to improve the quality of received video over wireless. By this, the recovery performance will be enhanced in addition to decreases jitter. Each packet is divided into n virtual sub-packets, checksum was used in each sub-packet for error control.

One of the main challenges in real-time video transmission is that video packets must be received in its playback time to be useful, as a result retransmission techniques some time become useless especially in highly loaded channel with bad conditions [31]. To solve the previous mention problems, [35] proposed unequal layer error protections by using layered multicast approaches which facilitate Applying FEC error correction to individual layers, this leading to achieve graceful degradation in video quality rather than visible error or interruptions. In [19] they reviewed the main issues that affect the reliability of video streaming over wireless channel by providing the required QoS and error control mechanisms. They integrated and compiled error control techniques to achieve required multimedia reliability, Adaptive error technique: FEC was used to supplement the existing MAC mechanism for error correction. They examined how the network best explores packet loses by using FEC feedback adaptations. To decrease the number of retransmission in case of multiple receivers, [20] proposed a network coding scheme, where the sender can combine and retransmit the lost packets to multiple receivers in one transmission.

3.4 Application layer solution: middle-ware

There are few works in the literature are solving real-time video transmission based on application layer solution like middle-ware.

CORPA middle-ware approaches are used in [22], they build an Adaptive distributed multimedia streaming server architecture (ADMS) based on CORPA component, this provide a flexible streaming service over distributed network.

Data distribution service (DDS) middle-ware solution is proposed in variety of works in literature. A high need for dynamic middle-ware real-time continues data stream was mentioned in [23] and [24]. This requirement is achieved by insuring a QoS that govern a reliable connection. If the transport layer doesn't supports the required QoS, The middleware must be flexible to tune connection to better transport layer [23]. They depend on a flexible framework called (FLEXMAT) to address the real-time requirement. They integrate FLEXMAT with OpenDDS which an open-source implementation for DDS middleware. OpenDDS will give more flexibility since it's depending on publisher/subscriber transmission and support different transport protocol (TCP, UDP, and IP) [23]. Efficient distribution for real-video surveillance needs a QoS grantee like in industrial application control [24]. In their work, they depend on middle-ware solution that provides flexibility and efficient deployments. A DDS middle-ware based on publisher subscriber communication is proposed as core backbone with include QoS.

Scalable video streaming H.264/SVC over DDS middleware was evaluated and tested over wireless channels in [25], see figure 3.3. The authors mention two major points regarding using DDS. Firstly, the user is freely to choose the proper data samples (NULU, Frame etc.) because this issue solve by using tunneling data technique in middleware. Secondly, DDS build-in functionality supports rate-control based on the structure of data sample, the video rate is adapted based on channel capacity. Furthermore, the scalable and flexible feature in SVC is very effective in DDS (publisher subscriber communication) where multiple subscribers exist with different capability.

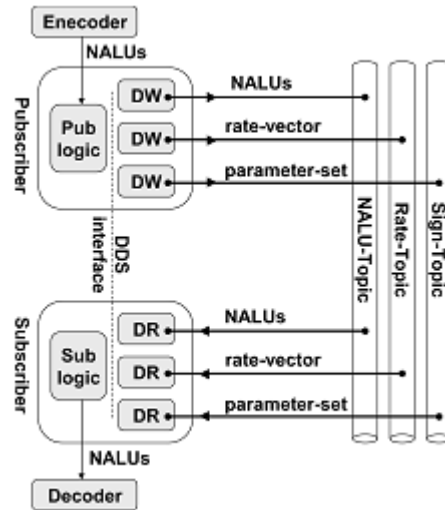


Figure 3.3: H.264/SVC over Data Distribution Service (DDS). [25].

Performance measures throughput and jitter of transmitting video over wireless LAN 802.11g using DDS were computed in [26] and [27]. In [27] Different scenarios are done regarding number of receivers (subscribers) and QoS used. The result shows that DDS depict the superiority of streaming video over wireless LAN that has varying available capacity. Using DDS we need high bandwidth and we get low jitter, its give more flexibly in dealing with video frames since it's providing a rich QoS that govern a reliable connection. Study the suitability of DDS for transmitting real-time video was done in [27]. They mention that DDS provide a flexibility and smooth integration with real-time and mission critical application. They did a Comparison between Video over DDS with streaming video using VLC player in term of bandwidth, required jitter and packet losses. Experimental results showed that DDS is more flexible for media streaming, it need low bandwidth and leads to low jitter and packet losses.

3.5 Scalable Video Coding

SVC is a video encoding technique that recently proposed for high quality video bit stream as an extension of H.264/AVC encoding standard. SVC is a charming solution for modern video streaming, it's provide a high compression efficiency and more robust with errors. Scalable feature means the adaption of the transmitted video to different channel condition by removing part of video bit stream to adapt available varying networks

capability [28], video is encoded once and then decoded to multiple receivers with different rates and resolutions. SVC represents an adaptation type for real-time video streaming like what mentioned in chapter 2, also its type of application layer solution, but I put it in separate section because of its importance and my thesis works are based on this technique. There are a wide works mentioned in the literature using SVC [29, 30, 31, 32, 33, 34, and 37].

In [29], They studied the feasibility of apply adaptive video transmission using H.264/SVC at low network level (router level) .They build a software-based proxy process solution inside router, This application manage multimedia stream request between client and server. They showed that a developed proxy able to apply efficient H.264/SVC adaptation scheme per packet with higher flexibility compared to other packet dropping mechanism. Adaptive network monitoring technique based on Scalable video coding (SVC) over wireless broadband IEEE 802.16 was done by [30], the study concentrated on how SVC in corporation with congestion control algorithms manage video streaming for many subscribers, temporal and FGS quality scalability are used. Performance measures show that system lead to efficient sharing for network resources. Furthermore, the video streaming is smooth with no interruption or jerks. The effect of using the new define video coding SVC over wireless environment like WIMAX is tested in [31], they mentioned that SVC provide a better adaptation in wireless environment where a heterogeneous environment are found and a variable channel condition that will lead to more losses. They mentioned that SVC solve wireless transmission problems by encoded video frames in scalable and adaptive manner. Two types of SVC scalability are used: Temporal and SNR scalability to encode video and send it over WIMAX. They evaluate different transmission scenarios using different SVC types and different number of retransmissions based on PSNR. The results show that quality (SNR) scalability give us better PSNR values but in the same time; it's more sensitive to noise. Furthermore, increasing the number of retransmission is not always do well specially for bad channel condition, discarded delay packet at sender side is better than retransmission.

SVC is stunning approach for streaming video over wireless heterogeneous network [32], [37]. In [32] they concentrated on the effect of inter-layer prediction in scalable video

coding. Their performance analysis of inter-layer prediction in SVC showed that it's more efficient in fast sequence compare with slow one. In addition, for SNR type scalability, medium grain quality scalability provides a better efficiency compare to coarse grain type. [37] Study the impact of using SVC standard in mobile communication delivery method where heterogeneous and different receivers capabilities are exist, they mentioned that video always consume the major of available bandwidth compare to other data types which supports the need for adaptive and scalable techniques, SVC offers the required scalability with reducing complexity and required computation compares to other techniques like transcoding and re-encoding. Furthermore, SVC has an obvious eligibility to be used in mobile environment where the receivers have varying receiving condition and it's very hard for the encoding to be done individually for each receiver. In [34], they applied cross-layer approach for streaming scalable video based on H.264/SVC. They consider the information from application layer and wireless channel together for better adaptive video broadcast to multiple clients.

Very good work is done in [3]; they proposed a scalable streaming method SVC based on Data distribution service (DDS) middle technique for real time video streaming over wireless. In their work, they served different clients (subscribers) with the optimal video quality based on available bandwidth and varying channels condition. Adaption technique was used to estimate link congestion and packets loss based on DDS quality of service. They used two scalability types, single layer Advance video coding AVC, Multi-layer scalable video coding SVC. Unequal protection is used by dropping the less important video packets to maintain real time continues video even with low quality. Using DDS middle-ware partition QoS, the encoded video subs-stream at publisher side is partitioning to one or more partitions (Temporal scalable video coding). See Figure 3.4. As we see from figure, partition 0 contains only I frame (Base Layer), partition 1 contains I and P frame, the last partition contains the full frame (the highest quality enhancement layer). A performance measures were computed to assess the proposed approach, results showed that system based on RTPS (Real time publisher subscriber) gives a better video quality when the number of subscriber (receiver) increased. Furthermore, their approach gives a graceful degradation of video quality as networks become more loaded but with preserving a continuous video without interruptions or errors.

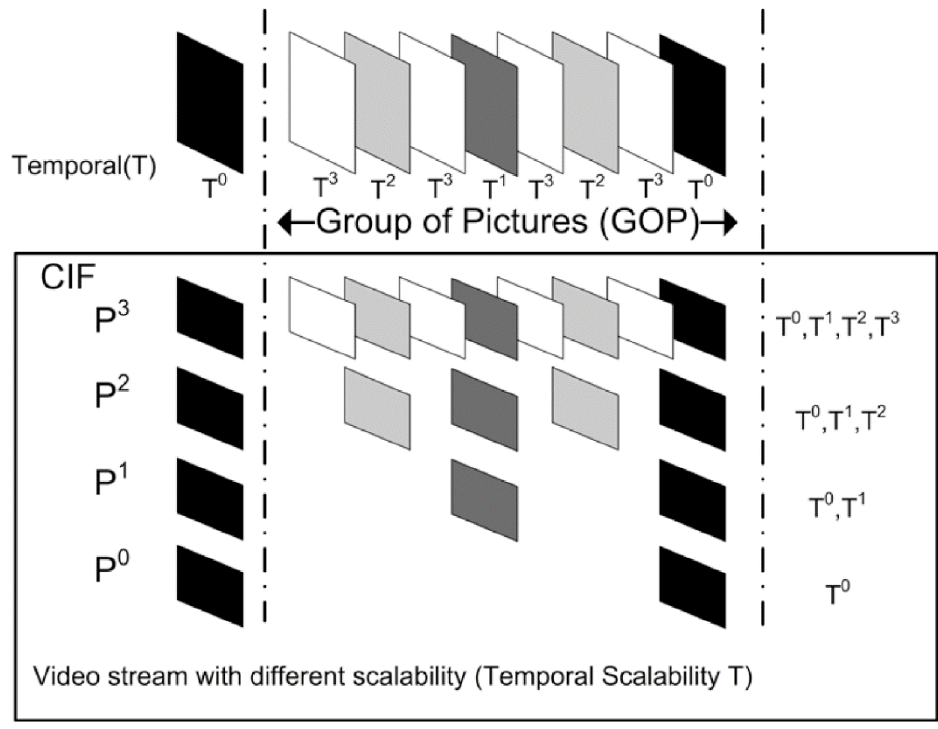


Figure 3.4: Example of using partition QoS for encoded video frame. [3].

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

This chapter will cover system architecture and design of the proposed technique for Real-Time video streaming using Data distribution service Middleware (DDS) and the new video coding technique scalable video coding (SVC). The chapter contains three main parts: main components that the system consists of, required DDS QoS for Real-Time video streaming and methodology used in this thesis to achieve our goals.

4.1 System Architecture and Components

This section shows the general architecture for my thesis work, Figure 4.1 shows the general system overview. System architecture consists of six main components: JSVM Encoder, SVEF Streamer, DDS Publisher, DDS Subscriber, SVEF Receiver and JSVM Decoder.

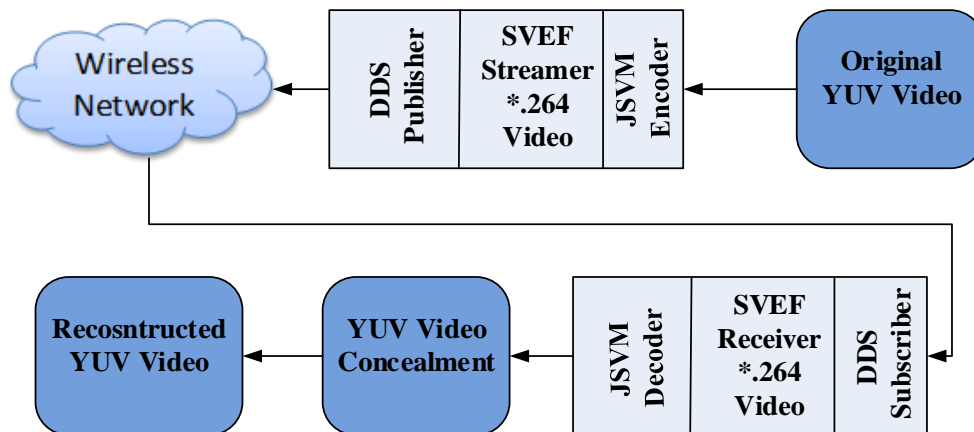


Figure 4.1: General System Architecture

4.1.1 JSVM Encoder

Video encoder was used to compress video and convert it from YUV to 264 format. JSVM encoder will be based on the new proposed multilayer scalable video coding

H.264/SVC standard. Four main scalability types will be used: Temporal, Spatial, Quality (SNR) and Combined scalability.

JSVM is configured by two or more configuration files depends on the required scalability type. By using Encoder, Video frames are encoded to one or more network access layer units (NALU) that are easily managed by network. NALU contains one-byte header which contains information about packet contents and its relevance in decoding process, the remaining bytes contains payload data. In temporal scalability, the video is encoded to multiple sub-streams that are different in Frame rates: one base layer with the lower rate (TID=0) and multiple enhancement layer, the higher enhancement layer has the highest Frame rate. Figure 4.2 shows encoding result after using temporal type.

```
>BitStreamExtractorStatic test.264
```

Contained Layers:
=====

Layer	Resolution	Framerate	Bitrate	MinBitrate	DTQ
0	176x144	1.8750	68.10	68.10	(0,0,0)
1	176x144	3.7500	94.60	94.60	(0,1,0)
2	176x144	7.5000	122.10	122.10	(0,2,0)
3	176x144	15.0000	155.00	155.00	(0,3,0)

Figure 4.2: Example Of Temporal Encoding

In Spatial Scalability video is encoded to multiple streams each one supports certain resolution, the layer that has higher dependency id DID will support higher resolution. Base layer has DID=0, Figure 4.3 shows encoding results after using Spatial type. We see from figure that the video is encoded to two spatial layer, DID=0 for base layer and DID=1 for the enhancement layer. Every spatial layer has different temporal frame rates from TID 0 to 4. Higher quality packet that has DID=1 and TID=4.

```

JSVM 9.19.14 BitStream Extractor

Contained Layers:
=====
Layer  Resolution    Frame rate  Bit rate   MinBitrate  DTQ
0      176x144          3.7500     49.90      49.90       (0,0,0)
1      176x144          7.5000     63.50      63.50       (0,1,0)
2      176x144          15.0000    77.80      77.80       (0,2,0)
3      352x288          3.7500     166.60     166.60      (1,0,0)
4      352x288          7.5000     208.10     208.10      (1,1,0)
5      352x288          15.0000    251.50     251.50      (1,2,0)
6      352x288          30.0000    291.20     291.20      (1,3,0)

```

Figure 4.3: Example Of Spatial Encoding

In Quality (SNR) scalability, video is encoded to different sub-streams each supports certain quality. Every layer has QID value, Higher QID means higher supporting quality, Base layer has QID=0. Figure 4.4 shows encoding results after using Quality type. We see from figure that the video is encoded to two Quality layers, QID=0 for base layer and QID=1 for the enhancement layer. Every Quality layer has different temporal bitrate from TID 0 to 3, Higher quality packet that has QID=1 and TID=4. In this work fine-grain scalability (FGS) quality type was used.

4.1.2 SVEF Streamer

Scalable Video Evaluation Framework (SVEF) is an open source evaluation tool for scalable video coding (SVC). Streamer is a main part of SVEF which has two essential jobs; the first one is to parse video trace file that's resulting from JSVM encoder and load NAL packet information (Frame size, Frame number, Scalability IDs) then load frame payload from *.264 compressed video. Figure 4.5 shows the data structure used to store NALU, from NAL data; we interested in three main parameters: temporal ID, dependency ID and quality ID. The second task is to send NALU as RTP layer-5 format then encapsulate it in UPD packet.

In my work NALU packet will be sent using DDS middle ware, thus the main task of streamer is only to prepare NALU information by reading its associated information from video trace file and the compressed video file.

Contained Layers:				
=====				
Layer	Resolution	Framerate	Bitrate	DTQ
0	352x288	1.8750	269.00	(0,0,0)
1	352x288	1.8750	466.00	(0,0,1)
2	352x288	3.7500	368.00	(0,1,0)
3	352x288	3.7500	659.00	(0,1,1)
4	352x288	7.5000	466.00	(0,2,0)
5	352x288	7.5000	878.00	(0,2,1)
6	352x288	15.0000	586.00	(0,3,0)
7	352x288	15.0000	1159.00	(0,3,1)
8	352x288	30.0000	721.00	(0,4,0)
9	352x288	30.0000	1475.00	(0,4,1)

Figure 4.4: Example Of Quality Encoding

```

Struct ourpacket
{
  Streamer_onebyte_t  lid;

  Streamer_onebyte_t  tid;

  Streamer_onebyte_t  qid;

  Streamer_onebyte_t  flags; // 6 bits are used: last (1 bit),
  // NALU type (2 bits), discardable (1 bit),
  // truncatable (1 bit), two nalus (1 bit)

  Streamer_fourbytes_t  naluid;

  Streamer_twobytes_t  total_size; /* in bytes */

  Streamer_twobytes_t f  rame_number;

  Streamer_onebyte_t  payload [MAX_PAYLOAD]; }

```

Figure 4.5: Data Structure For NALU Packet

4.1.3 DDS Publisher

This component responsible for distributes and publish data to middleware global data space based on specific topic. In this work, two main topics are published; the main topic is the video data topic “V.D” that represents NALUs packets prepared by streamer and resulting from JSMV encoder, the second topic is the configuration topic that contains header information “H.I” which used for decoder initiation at receiver side. “V.D” will contains NAL information like frame size, frame number, frame payload, NAL scalability IDs (TID, DID and QID) and one byte explains packet content.

The “V.D” will be Partitioning to multiple copy sub-streams using PARTITION QoS policy, each video partition will support certain quality depends on scalability layer that’s contain. In Addition, every partition will take string name “T.n.m” where *n* refers to enhancement layer number and *m* refer to TID number inside each layer, Partition name with higher *n* and *m* will refer to the sub-stream with higher quality, the procedure will be explained in details in the coming section. DataWriter (DW) represents the main access for application to purplish data into DDS domain based on specific topic. Every data writer will responsible to publish certain “V.D” topic partition based on resulting sub-stream after encoding. Figure 4.6 shows Data structure for DDS transmitted packet (from DDS_VideoStream.idl file).

```
const long MAX_PAYLOAD_SIZE=196500;

struct VideoStream
{
    sequence<char,MAX_PAYLOAD_SIZE> VideoData;
};
```

Figure 4.6: DDS Packet structure (IDL File)

4.1.4 DDS Subscriber

Subscribers are responsible for receiving published data by publisher DataWriter in specific topic. Subscriber is receiving data through its attached DataReader (DR) by specifying its interest in a topic. For a communication to occur properly, QoS must be compatible between publisher and subscriber. As we mention, the main topic in this work is the video data that is partitioning to multiple copies with different qualities (frame rate, resolution and quality). Firstly, subscribers are configured to subscribe a partition with high quality (*P.n.m*: higher *n* and *m*), but when network effected by bandwidth limitation and congestion; the subscriber can switch to a partition with lower quality and so on. Based on QoS feedback, switching between different partitions will be achieved. DEADLINE QoS is set to the maximum possible value for video frame to be received at subscriber side which is 150ms for interactive video. If frame fails to receive within this period, switching to lower quality sub-stream will be occurs.

4.1.5 SFEV Receiver

NULUS that was received by DDS will be used to build received trace file, this file will contain NALUs after transmission. The output video trace file will has the same format as sender one but with include delay that packet experienced by network. NALU filter in SVEF will filter the received trace by remove NALUs that have excessive delay, discard NALUs that unsatisfied decoding dependency (if NALU **W** depends on NALU **Z** and **Z** was not exist in received trace file, then **W** will discarded). Furthermore, NALU filter will recorde received packet according to sending order, reorder is important since decoded unordered frame will construct inconsistent video. After trace file received and filtered it passed to JSVM decoder.

4.1.6 JSVM Decoder

Video trace file that was received by DDS subscriber and filtered by SVEF will used to construct *.264 encoded video after transmission. BitStreamExtractorStatic will be used to generated *.264 video depending on filtered-trace file and the original encoded video. The result will represent received encoded video that will used as input to JSVM decoder

to produced uncompressed *.YUV video. A concealment will performed on reconstruct *.YUV video to compensate missing frames due to transmission over wireless. Concealment will be done for missing frames by repeating previous ones. Note that decoder must be initiated prior to subscription phase, to know packet information and scalability type.

4.2 Quality of Service (QoS)

DDS supports QoS which is a set of characteristics used to modify and control the DDS service as an adaptation to application requirements. QoS consist of individual QoS policy that may associated with all DDS environment entities: Publisher and subscriber, Data reader and data writer, Topic and domain participant.

Some types of QoS policy are not consistent with other type (Conflict with each other); if the addition of one QoS on the top of other QoS is inconsistent, the resulting operation will fails. For proper communication between publisher and subscriber, The QoS must be compatible and consistent. For example if the publisher defines Reliability QoS as BEST-EFFOR and the subscriber requires a reliable connection, in this case the connection will not happen as requested because of inconsistency [39]. DDS contains property called "RxO", which indicates if QoS policy requires compatibility between publisher and subscriber. Value "YES" required a compatibility between QoS value at publisher and subscriber side , value "NO" indicates that QoS policy can be set at both side but independent of each other, Value "N/A" indicates that the policy can be set either at publisher side or at subscriber side but not in both.

The next subsections discuss the required QoS for efficient video streaming behind the justification of using it. For example; interactive video requires one way latency less than 150 ms and jitter not exceed 30 ms.

4.2.1 DEADLINE

It's indicate that subscriber expect new data sample for each instant at least one every deadline period. In video streaming it's used to indicate when video packet must be received at receiver to ensure real-time continuous video. Video frame must not exceed its

playback time which is 150 ms. This QoS policy used as congestion prediction and an indicator for network status since when packet doesn't received within deadline period, this mean there is limitation in the receiver connection [27].

4.2.2 RELIABILITY

This QoS policy indicates the level of reliability offered by the system. In RELIABLE type, the service try to deliver all samples in the history, missed or drop samples are retransmitted to insure the delivery for almost all sample. BEST_EFFORT type is the fastest and require less resources, where new samples are generated enough so no need for retransmission or acknowledgment [46]. In video streaming; retransmitting of the dropped packets becomes useless especially when packet was received after deadline period. BEST_EFFORT is effective for real-time streaming in error-prone network where the losing of some frames degrade video quality but doesn't corrupt it. Furthermore, sometime packets retransmission (when lost occurs) leads to more congestion which will increase the problem further. In my experiment there are two main topics: Header information Topic "H.I" is critical for receiver and needed for initiating decoder parameter, this topic has high delivering priority; a RELIBLE kind of reliability will be used. The other topic is video data frame (NALUs) "V.D", for this topic BEST_EFFORT reliability will be used.

4.2.3 HISTORY

This policy controls what the service should deliver, the most recent samples or all samples for subscriber. In video streaming, we interest with the most recent frames since old ones are useless. The value of History QoS will be KEEP_LAST, so the publisher will keep the last recent "depth" sample. KEEP_LAST will be used also at subscriber side as a buffer for the most recent GOP (depth value will be more than one), buffer is important to minimize jitter delay between consecutive frames; it must be below 30ms for interactive video [47]. If we used KEEP_LAST history type with "depth" value there is no benefits for using RESOURCE_LIMITS QoS policy.

4.2.4 LIFESPAN

This QoS policy specifies the max validly time for the written data by data writer. In video streaming, we are interested in the data that have short delay [46]. In my proposed work it will be equal to the maximum excepted delay for the frame which is deadline period.

4.2.5 TIME_BASED_FILTER

Used to control data sample receiving rate and minimize application load. This policy indicates that subscriber is interested in receiving only one data sample every minimum separation period regardless of data rate speed. It's Cleary for minimum separation time to be less than or equal deadline period.

4.2.6 DURABILITY

This QoS policy controls the living time (data validity) after it was written by data reader. It determines if the topic samples are saved and how. VOLATILE type means that data reader keeps only the new data sample that is known by data reader and discard previously written one. In TRANSIENT_LOCAL type, the middleware attempt to save previous data sample for late joined data reader.

In video streaming, frames decoding is based on prediction from previous decoded one, for example P-frames (predictively coded) are coded based on previous I and P frame, B-frames (bi-directionally predicted) are encoded based on previous and next I and P frames. As a result, the frame that was already received and decoded, it will be needed for the following successive frame. The durability QoS will be set to TRANSIENT_LOCAL, so the middleware will store some previous data for late joining reader. The durability value is affected by history and resource limit QoS because the middleware keep only a number of frames equals "Depth" value.

4.2.7 LIVELINESS

Used to determine whatever the entity (sender or receiver) is live or down. Used for presence control and it detects if the participant joint or left the domain. The value that will

be used is AUTOMATIC where the infrastructure will automatically send a liveness signal to joining data writer at least once every lease duration period.

4.2.8 PRESENTATION

This QoS policy specify how samples represent changing to data instant are appeared at subscriber side. It effects the system ability to receive coherent and ordered data sample. In video streaming, video frame should be decoded and received in the same order to reconstruct coherent video. Sometime video frames are received out of order or buffered at receiver side for small period to uniform jitter that was produced by varying arrival time, after buffering; video frame must be viewed in order. PRESENTATION will be used to control receiving order and guarantee the coherency within instant, topic and across data writer (different topic).

4.2.9 PARTITION

PARTITION QoS controls the transmission between data writers and data readers. A data writer can only communicate with data reader if the associated publisher and subscriber have the same partition string name in addition to have same topic. In this work; a set of strings are define each for specific partition. Two main topics are used, Header information topic with the partition name "H.I" where it's contain a needed parameters for decoding. Video data NALU topic "V.D", this topic will be used to implement video streaming scalability feature by using partitioning QoS, every partition will refer to one sub-stream with specific quality. Scalable video coding with different four types (temporal, spatial, quality and combined) depends on encodes video to multiple sub-streams, one base layer and one or more enhancement layers. At receiver side, the video are decoded to better quality depends on receivers capability. The partition QoS will be used to configure the transmitted video sub-stream, every sub- stream will be assigned to certain partition, a sub-stream with base layer and all enhancement layer will be assigned to the higher partition number N and M with string name "P.n.m" where n refer to enhancement layer number and m to temporal ID.

As a result the encoded bit-stream will be assigned to different partitions, from higher quality partition (Contains all enhancement layer with all packets) to lower quality one (contains the last packet in base layer). The subscribers (receivers) first will subscribe to high quality partition, then it's automatically switching to the lower quality partition when networks degradation occurs (delayed packet, congestion, more packet loses).

Table 4.1 summarize the used QoS with its value and RxO compatibility between publisher and subscriber.

Table 4.1: Summary of the used QoS

QoS	QoS Value	Compatibility (RxO)
DEADLINE	150 ms , packet playback time	Yes
RELIABILITY	BEST_EFFORT at both side for "V.D" topic. RELIABLE for "H.I" topic	Yes
HISTORY	KEEP_LAST at publisher and with d= two GOP at subscriber	No
LIFESPAN	Equal to Deadline QoS value 150ms	N/A
TIME_BASED_FILTER	"minimum_separation" is set to 30 ms	N/A
DURABILITY	TRANSIENT_LOCAL at both side	Yes
LIVELINESS	AUTOMATIC	Yes
PRESENTATION	As encoded order before publishing	Yes
PARTITION	Partition string name "P.n.m"	No

4.3 Proposed Methodology and system Behavior

The main goal of our system is to distribute scalable video over wireless network by using DDS based middleware. DDS will be used to Supports QoS required for real-time video streaming and work with Publisher-subscriber model. Scalable video coding will be used as latest video coding standard to achieve scalable and graceful degrading in video

quality when it experience bandwidth limitation and varying network condition like in wireless networks.

In this work different types of scalability will be used to support different kinds of video adaption in term of quality, resolution and bit rate as mentioned earlier. Our approach will support different video copies based on DDS Partition QoS, Other QoSs will be configured to alternate between different sub-streams depend on receiver capabilities. As we mentioned, SVC encodes video to multiple sub-streams, one base layer and multiple enhancement layer. High quality video will be achieved by decoding all enhancement layer (of course with base layer), Scalability achieves by dropping one or more enhancement layer to still allowing live video but with lower quality.

In my proposed approach, instead of discards whole enhancement layer, smooth dropping will be achieved depending on unequal packets priority. Inside each enhancement layers, packet with higher temporal ID will has high dropping priority, see Figure 4.7 and 4.8, for example packet 7 in first enhancement layer (DID=1) is dropped firstly when needed. Consequently, switching between different video sub-streams will be achieved at packet level rather than at layer level (packet 7 then 6 and so on rather than dropping packets (7, 6, 5 and 4) once), Packet priority computed by the formula $4DID+TID$. Fig4.7 and 4.8 consider GOP=8 which supports 4 temporal layers. The same procedure is done for SNR type.

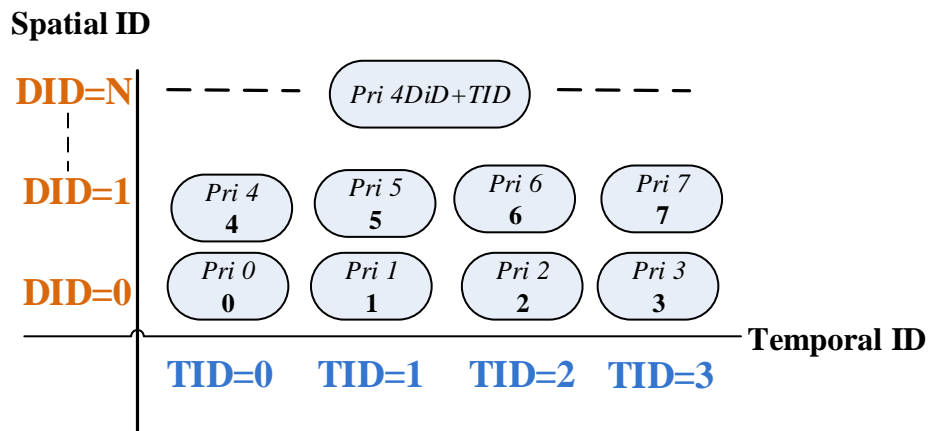


Figure 4.7: Packet priority inside each scalability layer using Spatial SVC

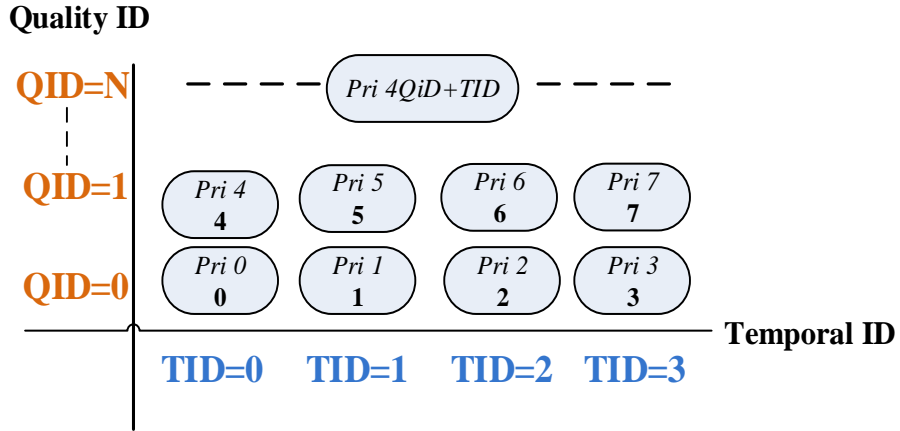


Figure 4.8: Packet priority inside each scalability layer using Quality SVC

After encoding, NALU trace is generated with full information about every NAL packet like: size and corresponding frame number, the main information are the D, T, Q IDs, which assign each packet to its corresponding enhancement layer. For example, for spatial encoding, if the value of (D, T, Q) is 1, 3, and 0 respectively, this means that the packet is located in the first enhancement layer (D=0 is the base layer) with Temporal ID=3. After we know the packet location, it will assign a dropping priority number, higher dropping priority means a higher probability to be dropped, as we see in figure 4.7 and 4.8. Packet dropping is needed when the switching from a sub-stream to another lower one is needed as a result of bad receiver conditions.

Partition QoS will be used to implement video scalability by assigning every sub-stream to a certain partition, each partition is different from the previous one by only one frame. Higher quality partitions will contain all frames for a certain GOP and will be given a partition string name "P.n.m" where n refers to the enhancement layer number and m to the Temporal ID, the next partition is "P.n.m-1" that differs by only one frame. If we apply this idea to figure 4.8, the first partition will contain all packets [0:7] and has "P_{1.3}" name, the next low quality one "P_{1.2}" where packet (7) was dropped and so on.

Every partition will include all other partitions lower than it, for example, the higher quality "P.n.m" will contain all subsequent partitions "P.n.m-1 ... to P_{0.0}". The switching from one partition to another is achieved by dropping packets that have the highest dropping priority because every partition differs from the next one by only one packet type. This

proposed mechanism provides smooth switching where one packet type was dropped when switching was needed in contrast with other method [3] where whole enhancement layer is dropped when switching occur for spatial and quality scalability type .

After encoding at receiver side, DDS will prepare Video partitions for the topic "V.D" by copy each frame to the different partitions depends on its D, Q and T IDs. If we take figure 4.8 as example, Packet (7) will be exist the highest quality $P_{1.3}$ only, Packet (6) will be copy to $P_{1.3}$ and $P_{1.2}$ only..., Packet (1) will be copied to all one except last lowest quality partition $P_{0.0}$, Packet (0) will be copied to all partition and it's exist in all from $P_{0.0}$ to $P_{1.3}$, DDS publisher will publish only the partition where subscriber interest in and subscribe to. Firstly, subscribers are subscribed to the higher quality partition "P.n.m" to receive high quality video, when NULU received at receiver side with more than allowed delay (frame deadline period governed by DDS QoS DEADLINE); subscriber will automatically switch to the next lower quality partition "P.n.m-1". See figure 4.9. Using DDS with publisher subscriber model the subscriber doesn't need to send an acknowledgment to publisher to send the next partition, instead when subscriber subscribes to another partition, publisher will automatically stop sending the first one and switch to the new partition that match subscriber interest. In this design every Data Writer (DW) responsible for publishing specific partition with certain quality, all DWs use the same reliability QoS which is BEST_EFFORT to achieve fast transmission which is needed in real-time video. This design supports number of Data Writers equal to the number of required partitions as we see in figure 4.9. Also From the figure we see a buffer at subscriber side where NULUs are stored before it was read by Data Reader (DR), the buffer has GOP size. In Addition, every subscriber can subscribes and read from certain DW that meet its capabilities.

The question now, How the partitions will look like when using different types of scalability: temporal, spatial, quality and combined. In the next paragraphs I discuss different partitioning examples for GOP of size 8 pictures, the source video (foreman.yuv) with CIF 352x288 resolution. Example 1 in figure 4.10 shows partitioning example for single layer temporal scalability using GOP with the

sequence (IBBPBPBI) frame. Since we have only one layer of scalability with the same resolution, the number of partition will equal the number of temporal layer.

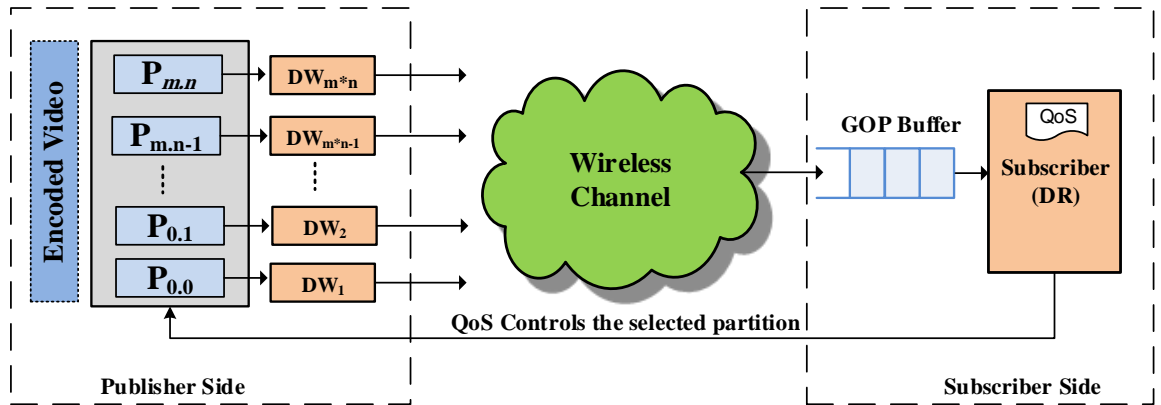


Figure 4.9: Example of sending different sub-streams using DDS Middleware

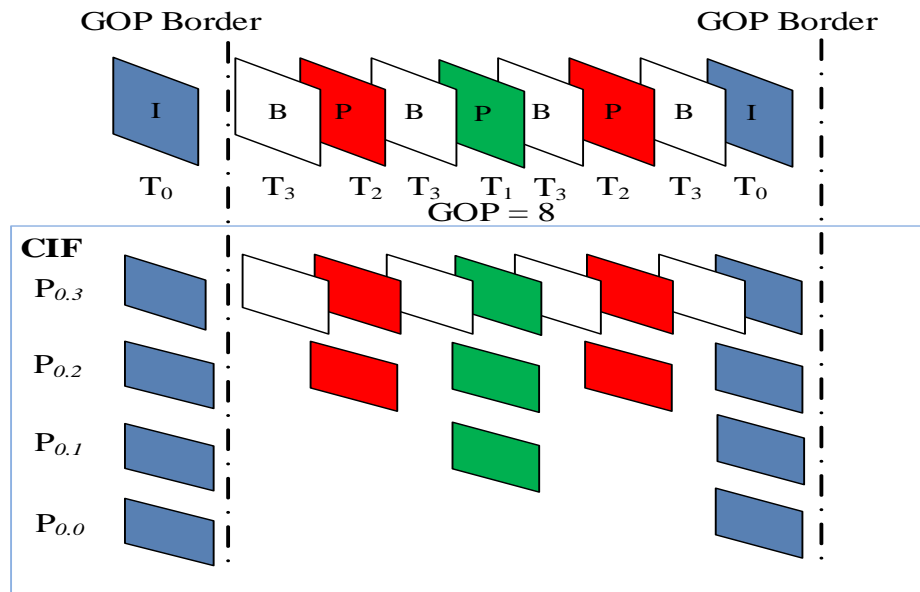


Figure 4.10: Example 1, Partitioning for temporal scalability

In figure 4.10, the higher quality partition $P_{0.3}$ will contains all temporal layers from T_0 to T_3 with all GOP frames. Frames that belong to higher temporal layer T_3 will be dropped in the next partition $P_{0.2}$, $P_{0.0}$ will contain only one temporal layer with I frame type.

For spatial scalability, the video is encoded to two or more resolutions based on encoding parameter. In this work the encoding supports two resolutions: CIF (352x288) and QCIF (176x144) see example 2 in figure 4.11, for every spatial resolution; different temporal rate points are supported from T_3 to T_0 . Figure shows base layer ($D_{id}=0$) supports QCIF resolution and one enhancement layer with ($D_{id}=1$) supports CIF resolution. Using Partition QoS, every layer will be partitioning at temporal rate point to different partitions, every partition different from preceding one by only one frame. D_1T_3 is the highest quality partition with highest resolution that contains complete sub-stream and includes all other partitions, D_0T_0 is the lowest quality partition that's contains only I frame. The number of partitions equal to the number of spatial layer (2) multiplies by number of temporal layers (4) which equal to 8. In this design, smooth switching will be achieved by dropping only one frame type at a time rather than dropping whole CIF layer at once when switching to lower resolution.

Partitioning in Quality scalability is the same idea as in spatial type but with encoding differences, there are two or more quality layers. In Quality type, every layer supports different temporal rate points: T_3 to T_0 , see example 3 in figure 4.12 which shows partitioning example after using Quality scalability with fine-grain-scalability (FGS) type. Figures shows a base layer with $Q_{id}=0$ and one enhancement layer with $Q_{id}=1$, the highest quality will be achieved by receiving base and enhancement layers. As in spatial type, every layer will be partitioning to multiple partitions where everyone will support certain quality. Q_1T_3 is the highest quality partition that contains the entire sub-streams and include all other partitions, Q_0T_0 is the lowest quality partition that's contains only I frames. The number of partitions equal to the number of Quality layers (2) multiplies by number of temporal layers (4) which equal to 8. In this design, smooth switching will be achieved by dropping only one frame type at a time rather than dropping whole enhancement layer when switching to lower quality partition.

Figure 4.13 shows a system architecture of using DDS with SVEF tool to achieve scalable video streaming, this figure explains the publication of video topic using different Data Writers (DW) each one publish video with certain quality with specific partition string

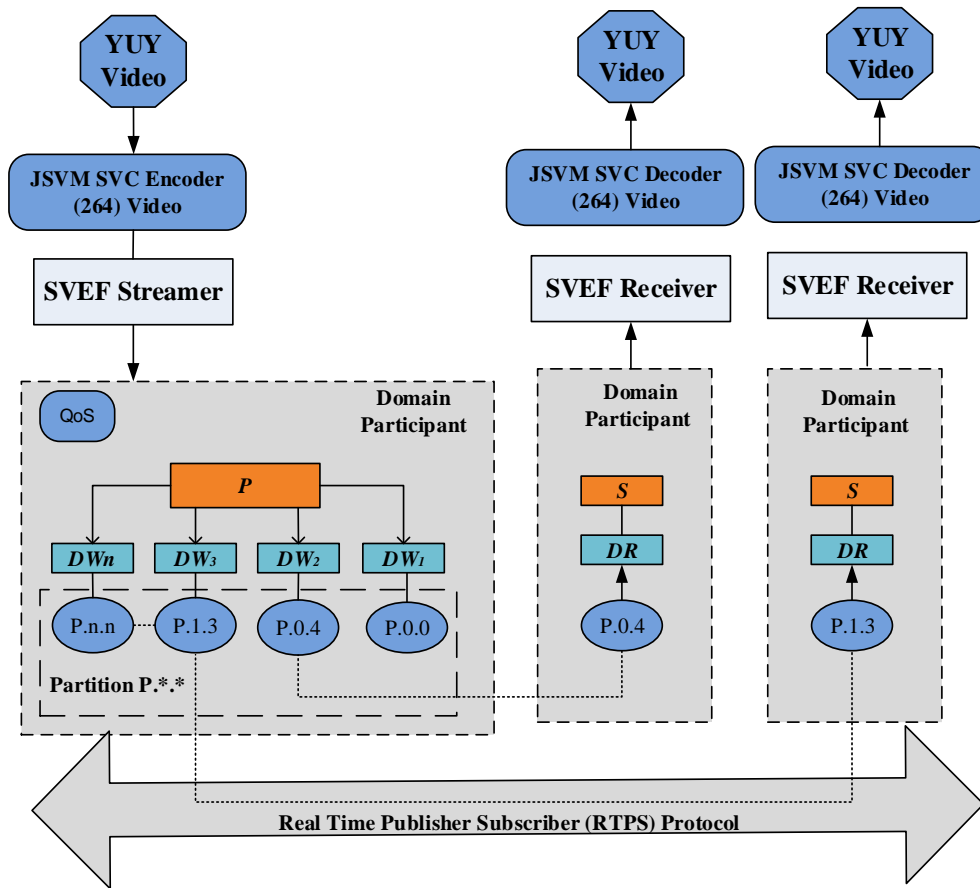


Figure 4.13: DDS with SVEF tool for scalable real-time video streaming

CHAPTER 5

EXPERIMENTAL WORK AND EVALUATION

This chapter will explain the experimental work for real-time video streaming over wireless network based on H.264/SVC standard and using DDS architecture [39] and [45].

5.1 Experiment setting and components

The main goal of experimental work is to test the transmitting of scalable video using publisher/subscriber architecture, where the video are published to multiple receivers (subscriber). The experiment framework contains five host one represents publisher and four for subscribing, these hosts are connected throw wireless access point (54 Mbps) as shown in figure 5.1.

The source video which is in YUV format is encoded via JSVM encoder [38] based on H.264/SVC to four scalability type as mention earlier: temporal, spatial, SNR and combined scalability. The encoded video transmitted over wireless network based on DDS middleware that will be used to implement scalability depends on network condition. To test system scalability with different loads, we run the experiment with different number of subscribers (receivers): 1, 4, 8 and 12. As we see from figure 5.1, Video source at the top of figure represents video publisher to the other hosts (A, B, C and D) that represents subscriber side, all nodes are connected throw wireless access point. The video sequence that was used in this work is the *Foreman* YUV video sequence which was widely used in research work. Two video resolution will be used CIF (352×288) and QCIF (176×144). See table 5.1.

Performance measures will be computed to assess the four scalability types in terms of: PSNR, delay, MOS level and jitter to see which SVC types will be more adaptive for real-time video streaming especially in bad network conditions. Figure 5.2 shows a general scheme for the experimental and evaluation framework.

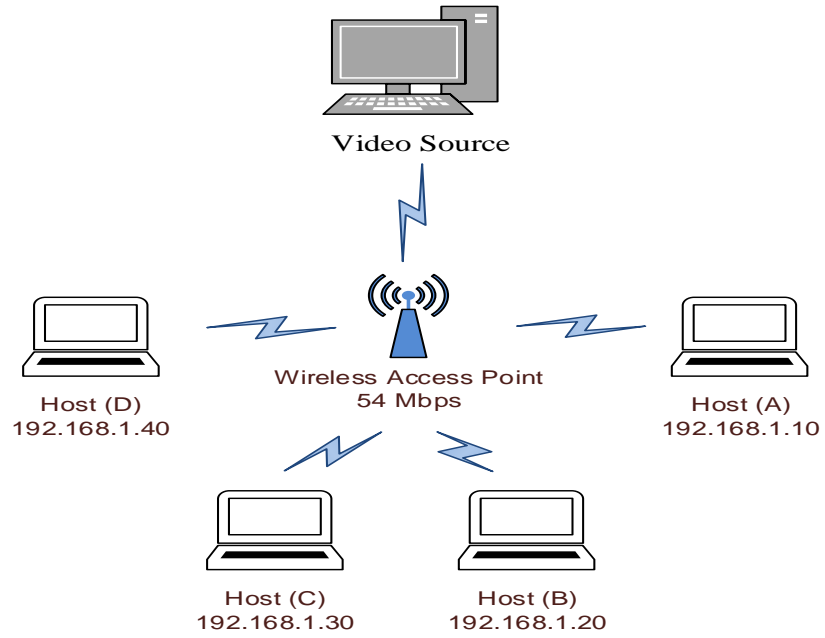


Figure 5.1: Experiment components

Table 5.1: Source Video Parameter

Video Sequence	Resolution	Frame Rate	Number Of Frame	GOP Size
Foreman (CIF)	352×255	30 Hz	300	8
Foreman (QCIF)	176 × 144	15 Hz	300	8

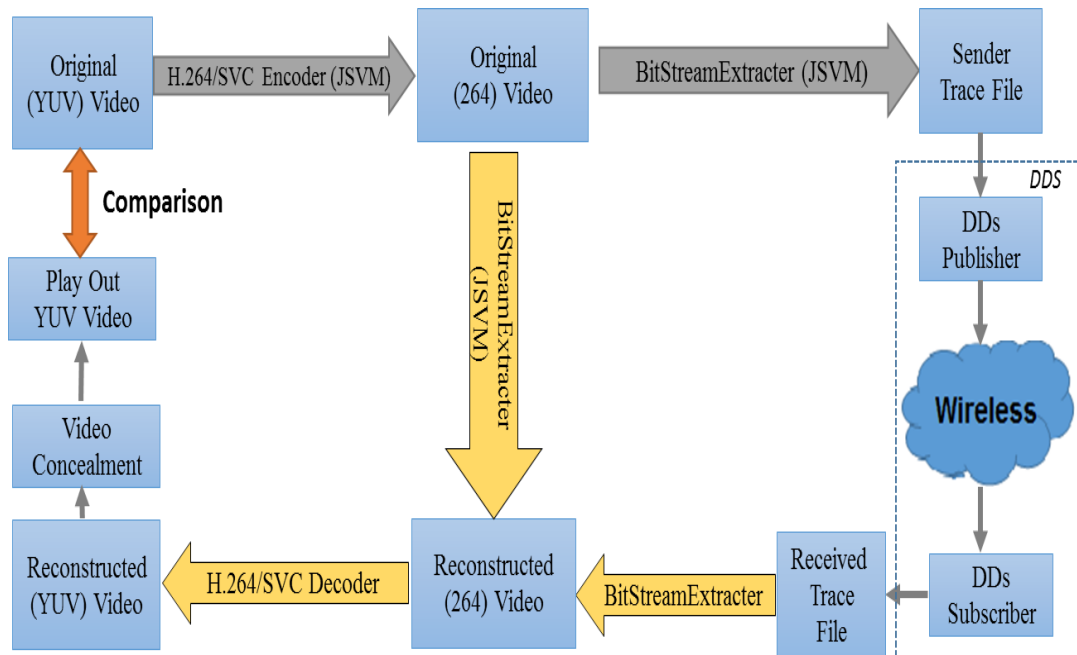


Figure 5.2: General scheme for the experimental and evaluation framework

5.2 Evaluation Framework and tools

5.2.1 Performance metrics

Different number of image and video quality metrics are used in the literature [52]. In this work I will use the most commonly performance metrics which are Peak signal to noise ratio (PSNR), Mean opinion score (MOS) level, Frame Delay and jitter.

5.2.1.1 Peak signal to noise ratio (PSNR)

PSNR is widely used as performance metrics for image and video quality assessment because it's has clear physical meaning and easy to calculate in spite of providing an approximate measure of the quality as subjectively perceived by human observers [53]. PSNR represents the signal to noise ratio in (dB) between original image and reconstructed image after compression and transmission. Higher PSNR value means smaller degradation in image and video quality between original and reconstructed one. PSNR equation is define as follow:

$$PSNR = 20 \log_{10} \left(\frac{MAXP}{\sqrt{MSE}} \right) \quad (5.1)$$

Where

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2 \quad (5.2)$$

MAXP is the maximum possible number of pixel in the image. Mean Square error (MSE) accumulative squared error between original and reconstructed image [31] and [48].

5.2.1.2 Mean Opinion score (MOS)

MOS level represents a numeric value indicates human impression for the video quality. MOS level measurement requires user interpretation which is hard and time consuming. An approximation value for MOS is calculated based on mapping table define in [52], the mapping table depends on PSNR value see table 5.2.

Table 5.2: PSNR to MOS Conversion

PSNR	MOS Level	Quality	Impairment
>37	5	Excellent	Imperceptible
31-37	4	Good	Perceptible
25-31	3	Fair	Slightly annoying
20-25	2	Poor	Annoying
<20	1	Bad	Very annoying

5.2.1.3 End-To-End frame delay

Frame delay represent one-way delay needed for frame to be transmitted from sender to receiver. The main goal is to see how transmission delay over wireless was varying as the number of receiver (subscriber) increased. Processing time for encoding and decoding time will be discarded since our interest only on channel delay.

5.2.1.4 Jitter

Inter arrival jitter J is the mean deviation of the time difference between receiving and sending time for a pair of packets [41]. The jitter is very important in real-time video streaming because it's directly affects the quality of the received video.

5.2.2 Evaluation tools for Scalable Video Coding SVC

Different evaluation tools are implemented to evaluate scalable video coding and compute PSNR; some of them support single layer coding and other support single and multi-layer coding.

Evalvid [48], is an evaluation tool for video quality assessment over network, Evalvid is very good choice to assess the received video by compute PSNR value in the simulated network environments. Evalvid used to evaluate single layer video coding AVC, It doesn't support multi-layer coding scalable video SVC. EvalSVC [49] is an evaluation platform for the scalable video coding standard (SVC), this tool measures video quality in term of PSNR and shows that SVC gives more adaption for network bottleneck specially the SNR scalability type. In [31]; they worked on scalable video coding over WiMAX, An integrated evaluation framework based on [50] and [51] was developed to compute performance measures.

In this work, I use Scalable Video-streaming Evaluation Framework (SVEF) tool [43] and [50] to compute performance measures: PSNR, Delay and Jitter. SVEF supports evaluation for the new video encoding standard H.264/SVC. Figure 5.3 shows SVEF structure from the original YUV video before encoding and transmission to reconstructed YUV video after reception and decoding. Firstly the video is encoded to .264 format by JSVM then trace file extracted from encoded video which include different entry for each

NULU like: size, TID, QID and DID. F-N stamp add new column to trace file which is frame number. Trace File in addition to encoded video will be sent.

At receiver side, the received trace file is passed to Null Filter which will apply the following: reorder NULU according to the sending order, remove NULU that unsatisfied decoding dependency and remove packets (NALU) that was received after play-out buffer deadline.

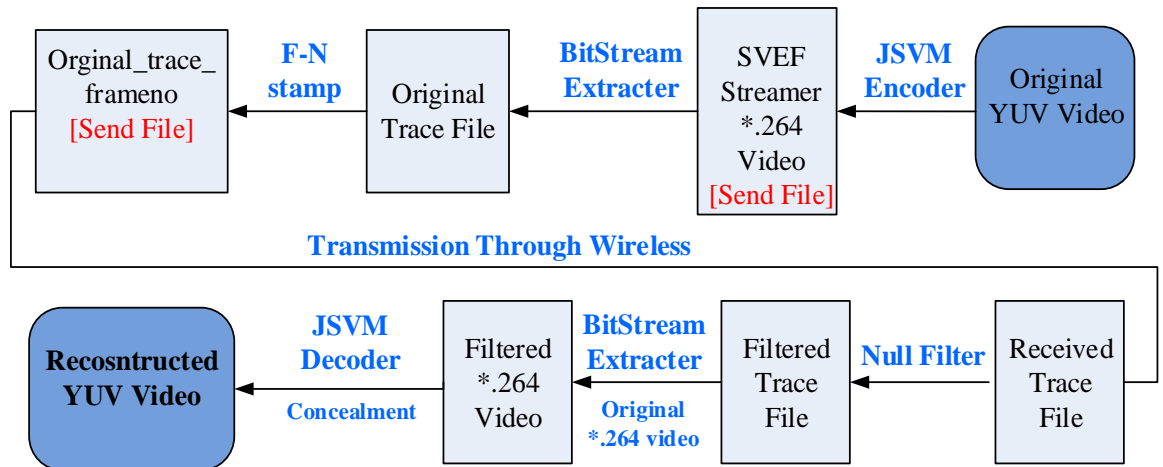


Figure 5.3: SVEF Framework Structure

BitStreamExtractor will take the filtered trace file (after remove NALUs that are delayed or unfulfilled dependency condition) in addition to original encoded video (.264) to generate received *.264 video, JVM decoder will take received video in compressed format (.264) in addition to the original compressed video as input to reconstruct YUV video, concealment will be needed to compensate missing frames due to transmission by copying the previous frames.

5.3 Performance Evaluation and Result

The experimental work has five main steps as I mentioned earlier: Video encoding (compression), video publishing over wireless (Based on DDS), video subscribing, video

Decoding, video reconstruction and evaluation. See figure 5.2 (general scheme for the evaluation framework). The experiment was run four times to simulate the behaviour of SVC types: temporal, spatial, SNR and combined scalability. For every experiment, all performance metrics are computed.

The encoding phase is performed using JSVM tool, where the source video is encoded from YUV to .264 format, encoding parameters are varying depend on SVC type (explained in details in Appendix A). In this experiment the video is encoded based on GOP = 8 and contains 300 frame. Figure 5.4 shows the supported scalable layers by Temporal scalability, we see from figure that this encoding support four layers: base layer with TID= 0 and 3 enhancement layers. Figure 5.5 shows the supported scalable layers by Spatial scalability, we see from the figure that there are two spatial scalable layer; base layer with DID=0 and enhancement layer with DID=1, every spatial layer supports certain resolution (the higher layer supports higher resolution) and contain different temporal rate points from TID equals 0 to 3.

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	352x288	3.7500	182.50	182.50	(0,0,0)
1	352x288	7.5000	230.30	230.30	(0,1,0)
2	352x288	15.0000	276.90	276.90	(0,2,0)
3	352x288	30.0000	327.10	327.10	(0,3,0)

Figure 5.4: Supported scalable layer using temporal SVC

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	176x144	3.7500	49.90	49.90	(0,0,0)
1	176x144	7.5000	63.50	63.50	(0,1,0)
2	176x144	15.0000	77.80	77.80	(0,2,0)
3	352x288	3.7500	166.60	166.60	(1,0,0)
4	352x288	7.5000	208.10	208.10	(1,1,0)
5	352x288	15.0000	251.50	251.50	(1,2,0)
6	352x288	30.0000	291.20	291.20	(1,3,0)

Figure 5.5: Supported scalable layer using Spatial SVC

Figure 5.6 shows the supported scalable layers by SNR(Quality) scalability, we see from the figure that there is two quality scalable layer; base layer with QID=0 and enhancement layer with QID=1, every quality layer supports certain quality (the higher layer support higher quality) and contain different temporal rate points from TID equals 0 to 3.

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	352x288	3.7500	153.40	153.40	(0,0,0)
1	352x288	7.5000	195.60	195.60	(0,1,0)
2	352x288	15.0000	236.80	236.80	(0,2,0)
3	352x288	30.0000	283.60	283.60	(0,3,0)
4	352x288	3.7500	273.30		(0,0,1)
5	352x288	7.5000	343.30		(0,1,1)
7	352x288	15.0000	406.80		(0,2,1)
7	352x288	30.0000	477.10		(0,3,1)

Figure 5.6: Supported scalable layer using SNR (Quality) SVC

Figure 5.7 shows the supported scalable layers by Combined scalability where the three mentioned SVC types are combined, we see from figure that there is two spatial scalable layer every one support certain resolution, each layer support two quality (QID 0 and 1) and four temporal rate points.

PSNR values are calculated using *PSNRStatic* tool (supported by JSVM) which compare the original un-encoded YUV video with the reconstructed video after transmission and decoding (to account for the distortion due to losses during transmission) "Main PSNR", Also we get "Reference PSNR" which was computed by comparing original un-encoded video with YUV video after encoding and decoding but before transmission (to account for distortion due to compression process). PSNR tool gives three PSNR value, we interested in Y-PSNR which the luminance component

Jitter values are computed using "ComputeJitter.py" tool and Frames delay are computed by "ComputeDelay.py" tool (supported by SVEF too) Performance evaluation calculation is explained in (Appendix A).

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	176x144	3.7500	49.70	49.70	(0,0,0)
1	176x144	7.5000	63.10	63.10	(0,1,0)
2	176x144	15.0000	76.80	76.80	(0,2,0)
3	176x144	3.7500	81.20		(0,0,1)
4	176x144	7.5000	98.20		(0,1,1)
5	176x144	15.0000	115.80		(0,2,1)
6	352x288	3.7500	130.30	98.80	(1,0,0)
7	352x288	7.5000	161.90	126.80	(1,1,0)
8	352x288	15.0000	196.00	157.00	(1,2,0)
9	352x288	30.0000	227.20	188.20	(1,3,0)
10	352x288	3.7500	336.00		(1,0,1)
11	352x288	7.5000	393.80		(1,1,1)
12	352x288	15.0000	449.80		(1,2,1)
13	352x288	30.0000	498.40		(1,3,1)

Figure 5.7: Supported scalable layer using Combined SVC

5.3.1 PSNR Results

Figures 5.8 to 5.27 report the Y-PSNR per frame of Foreman video after transmission over wireless using DDS publisher subscriber model. In all previous mentioned figures H.264/SVC is used. For PSNR, two values were computed: main PSNR and reference PSNR. Furthermore, MOS level is indicated on right side for each figure based on table 5.2. For every SVC type, 4 scenario are tested: 1, 4, 8 and 12 subscribers.

Figures 5.8 to 5.12 report PSNR and MOS level for temporal SVC. We see from figure 5.8 that PSNR value approximately around reference at low load (subscriber = 1), Also in figure 5.9 the PSNR value around reference PSNR when the number of receiver increase to 4 except short negative peak between frame 200 and 240. In the contrary, we see a persistent degradation in PSNR when the number of subscribers increased to 8 and 12 as we see in figure 5.10 and 5.11 respectively, we see a stable low PSNR value compare to reference one especially at 12 subscribers, See figure 5.11. Also we notice that the average PSNR dropped from (36.85 dB) in case of 1 subscriber to (21.7955 dB) in case of

12. Number at right side indicate MOS level (per frame) that related directly to PSNR values, for example in figure 5.8 all frame located in in region of 5 (Excellent) and 4 (Good) MOS Region, but when the subscribers increased to 8 and 12, some of frames have 1 (Bad) MOS value.

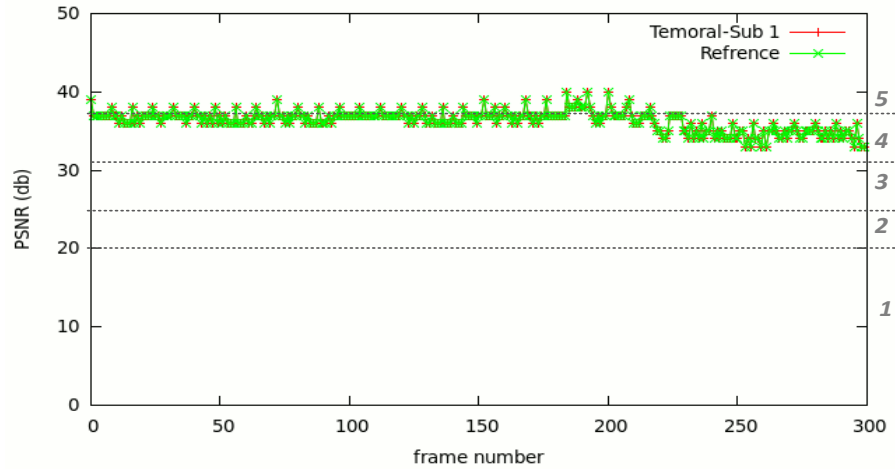


Figure 5.8: PSNR, Temporal SVC, Subscriber=1, AVG =36.85 dB

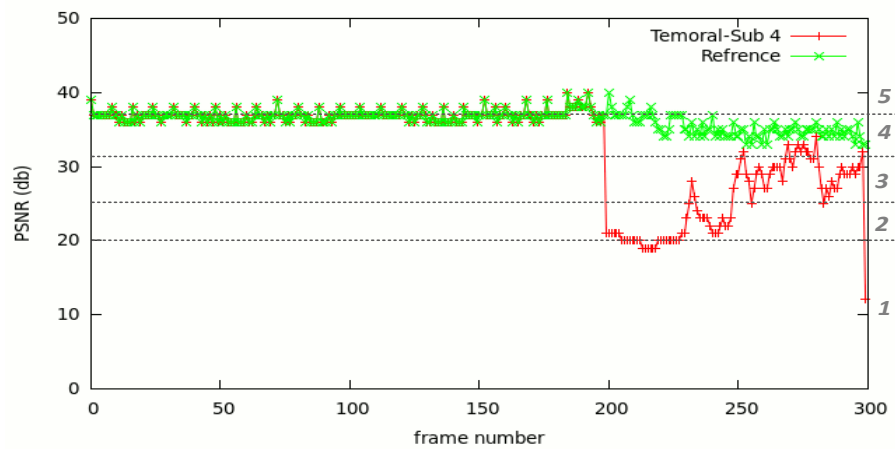


Figure 5.9: PSNR, Temporal SVC, Subscriber =4, AVG = 33.5 dB

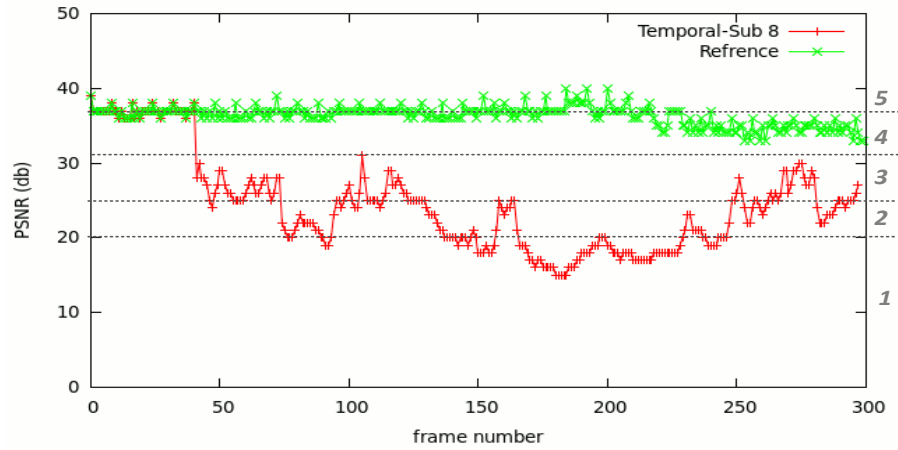


Figure 5.10: PSNR, Temporal SVC, Subscriber =8, AVG = 24.85 dB

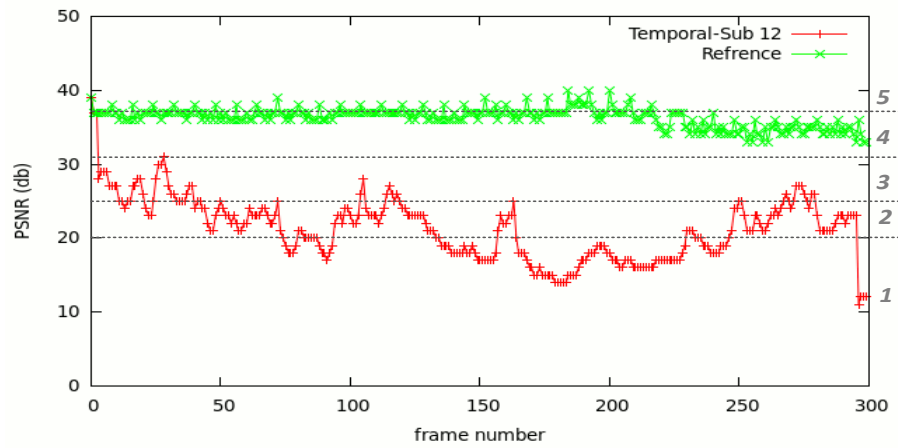


Figure 5.11: PSNR, Temporal SVC, Sub =12, AVG = 21.8 dB

Figure 5.12 shows PSNR values using temporal SVC for different number of subscribers together

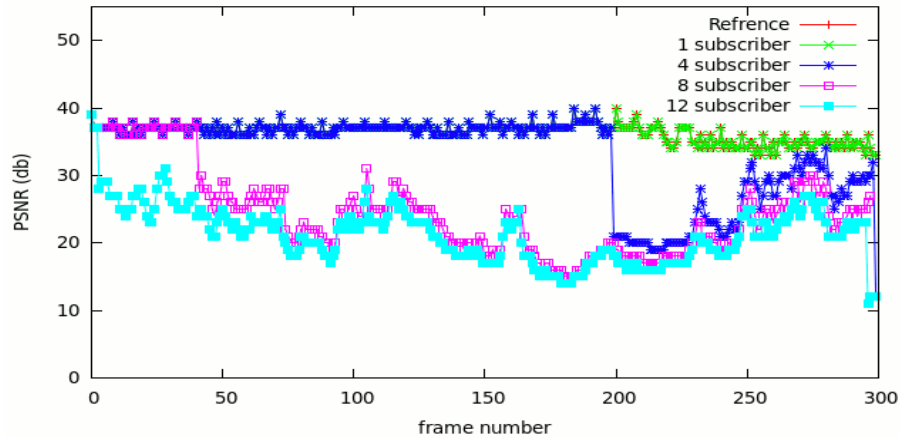


Figure 5.12: Y- PSNR Results, Temporal type

Figures 5.13 to 5.17 report PSNR and MOS level for Spatial SVC. We see from figure 5.13 that PSNR value approximately around reference at low load (subscriber=1) as in temporal type. In figure 5.14 at medium load (subscriber=4), Frames PSNR remain around reference one for the first 200 frames, negative sharp peak occurred to the PSNR but with short duration, PSNR tends to return it's reference curve , the average PSNR equal to 33.06 dB. For 8 and 12 subscribers in figures 5.15 and 5.16 respectively; we see long degradation in the PSNR especially in the case of 12. The average PSNR is 26.29 dB and 25.22 dB. Same as in temporal, the perfect MOS level at low load. The MOS level decrease to level 3 and 2 and some time to 1 but still better than temporal scalability.

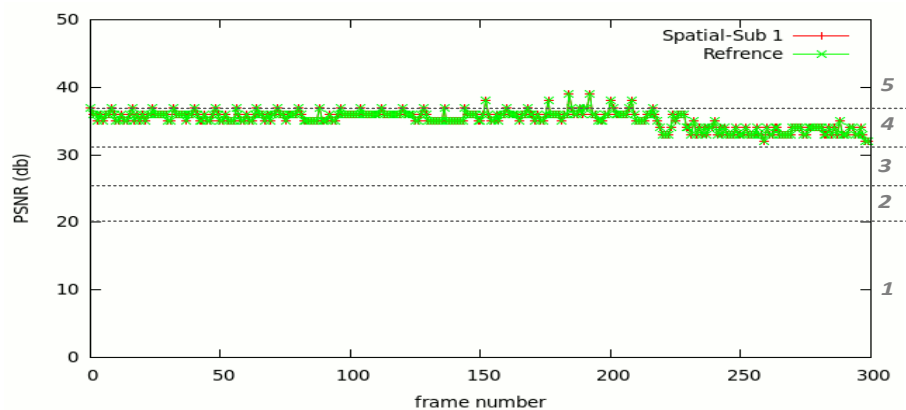


Figure 5.13: PSNR, Spatial SVC, Subscriber =1, AVG = 35.73 dB

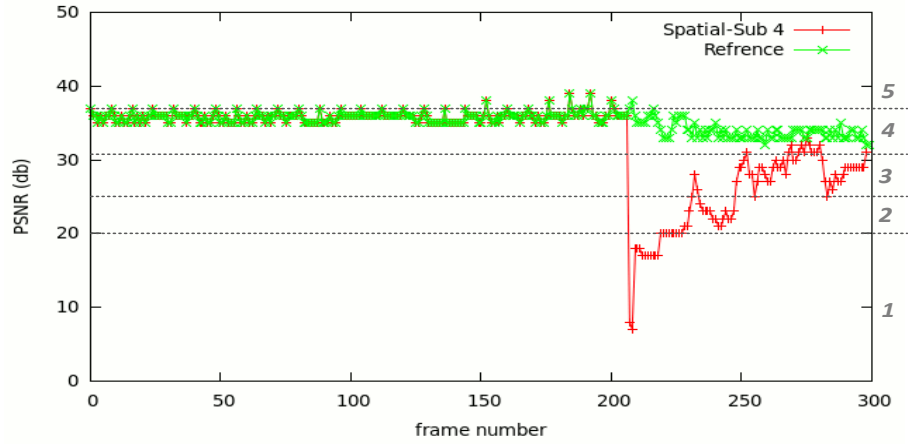


Figure 5.14: PSNR, Spatial SVC, Subscriber =4, AVG = 33.06 dB

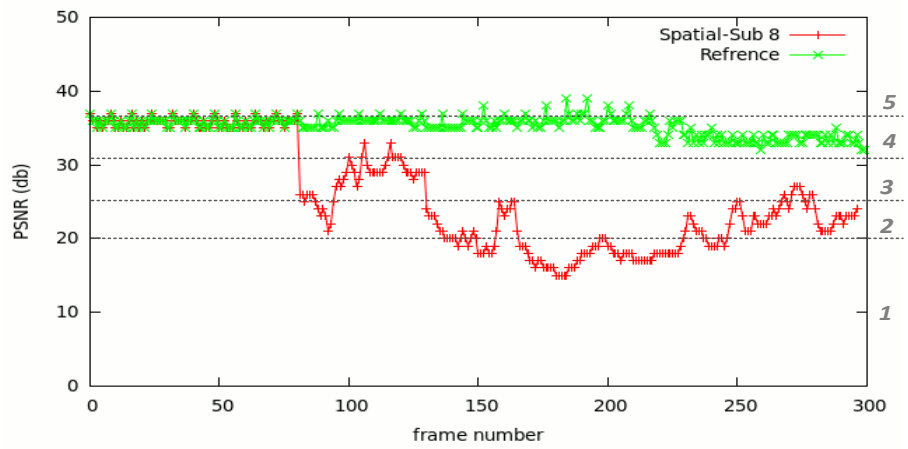


Figure 5.15: PSNR, Spatial SVC, Subscriber =8, AVG = 26.29 dB

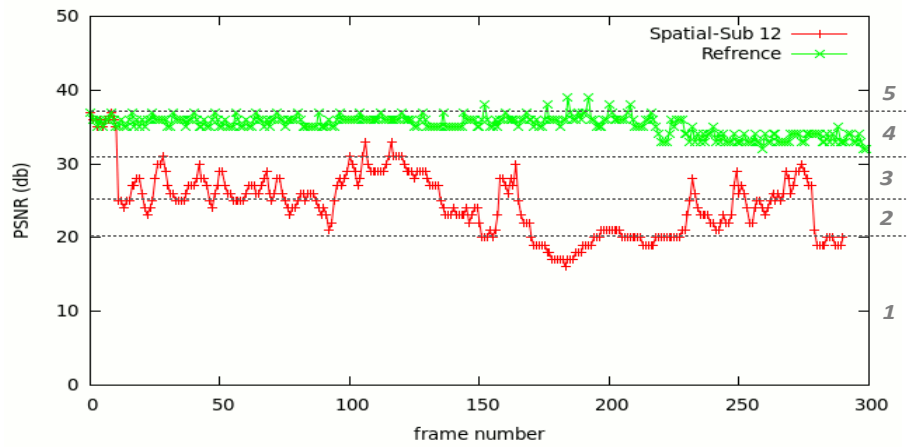


Figure 5.16: PSNR, Spatial SVC, Subscriber =12, AVG = 25.22 dB

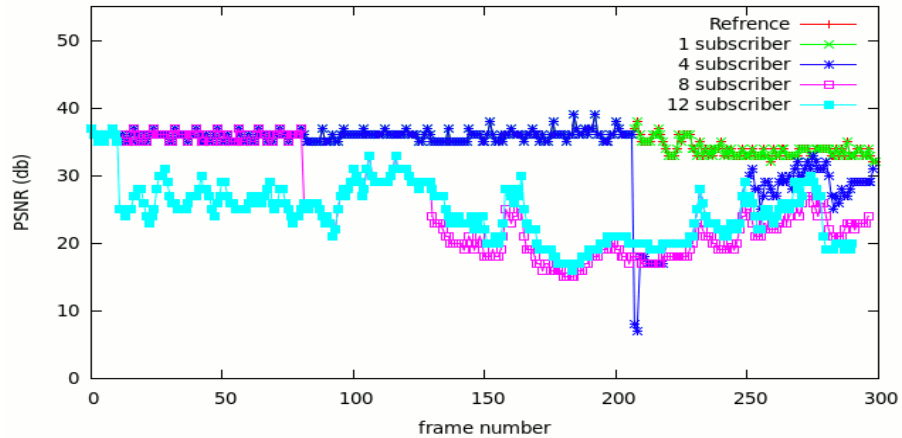


Figure 5.17: Y- PSNR Results, Spatial type

Figures 5.18 to 5.22 report PSNR and MOS level for Quality (SNR) SVC. We see from figure 5.18 that PSNR value approximately around reference at low load (subscriber=1) with the highest average PSNR equals 38.68 db (compare to other type). In figure 5.19 at medium load (subscriber=4), Frames PSNR remains around reference one for the first 170 frames, negative sharp peak occurred to the PSNR but with short duration, the average PSNR equal to 33.88 dB. For 8 and 12 subscribers in figures 5.20 and 5.21 respectively; we see long degradation in the PSNR especially in the case of 12 but it's more smooth compare to temporal and spatial type. The average PSNR is 29.97 dB and 27.07 dB. For MOS level, the perfect MOS values occurred at low load, half of frames ar at excellent and good level.

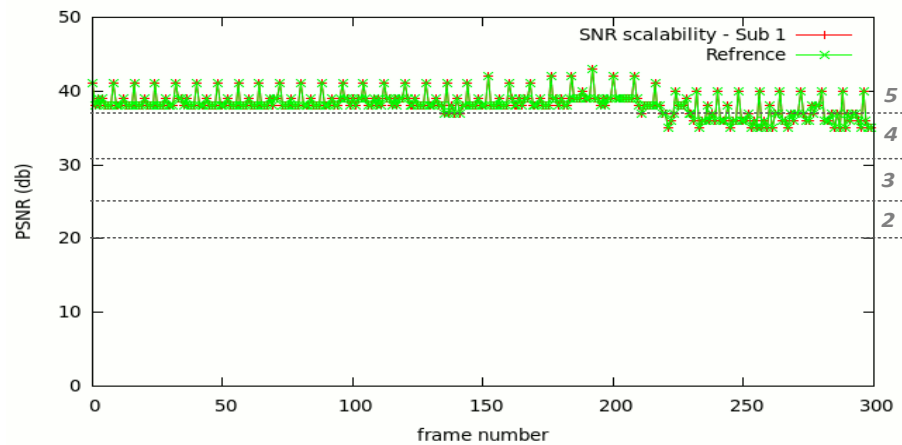


Figure 5.18: PSNR, SNR SVC, Subscriber =1, AVG = 38.68 dB

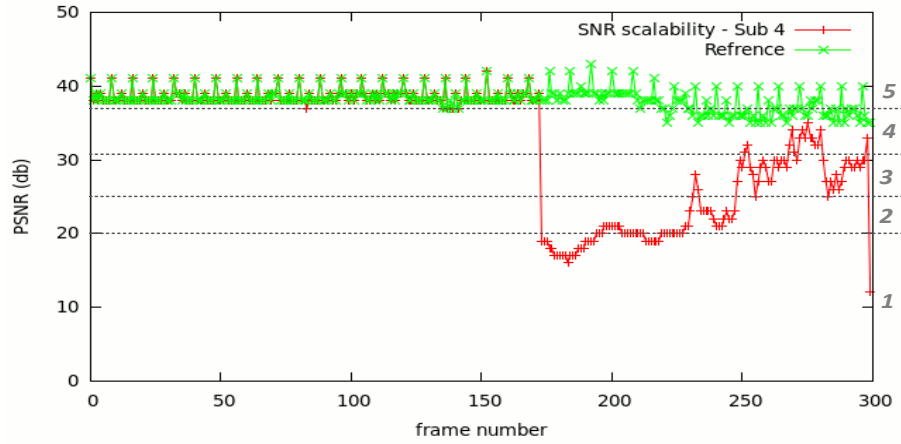


Figure 5.19: PSNR, SNR SVC, Subscriber =4, AVG = 32.88 dB

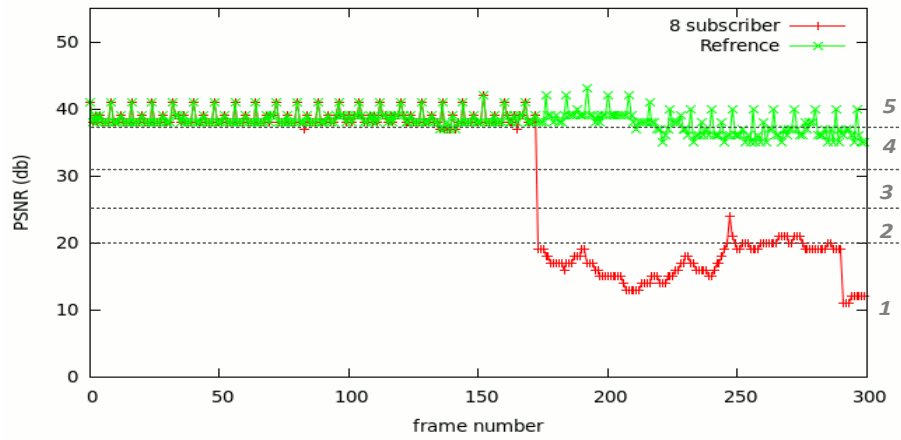


Figure 5.20: PSNR, SNR SVC, Subscriber =8, AVG = 29.97 dB

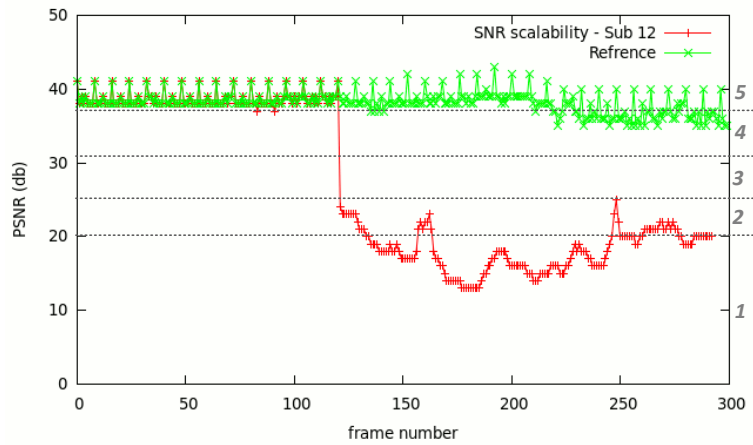


Figure 5.21: PSNR, SNR SVC, Subscriber =12, AVG = 27.07 dB

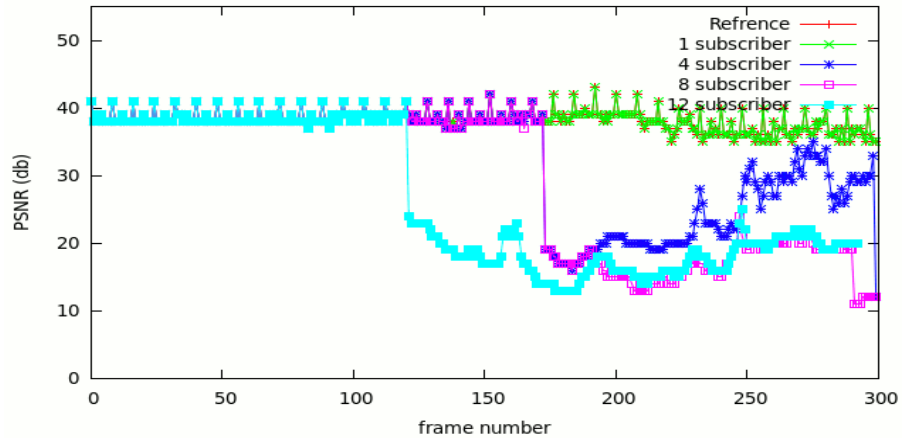


Figure 5.22: Y- PSNR Results, SNR type

Figures 5.23 to 5.27 report PSNR and MOS level for combined SVC. We see from figure 5.23 that PSNR value approximately around reference at low load (subscriber=1) as in other SVC types with average PSNR equals 36.22 dB. In figure 5.24 at medium load (subscriber=4), Frames PSNR remain around reference one for the first 225 frame, negative sharp peak occurred to the PSNR but with short duration, the average PSNR equal to 33.64 dB. For 8 and 12 subscriber in figures 5.25 and 5.26 respectively: we see long degradation in the PSNR especially in the case of 12 but it's more smooth compare to temporal and spatial type, also it's better than SNR type . The average PSNR is 30.1 dB and 27.36 dB for the 8 and 12 subscriber respectively. For MOS level, the perfect MOS values occurred at low load. Half of frames are at good MOS and half and poor and fair level

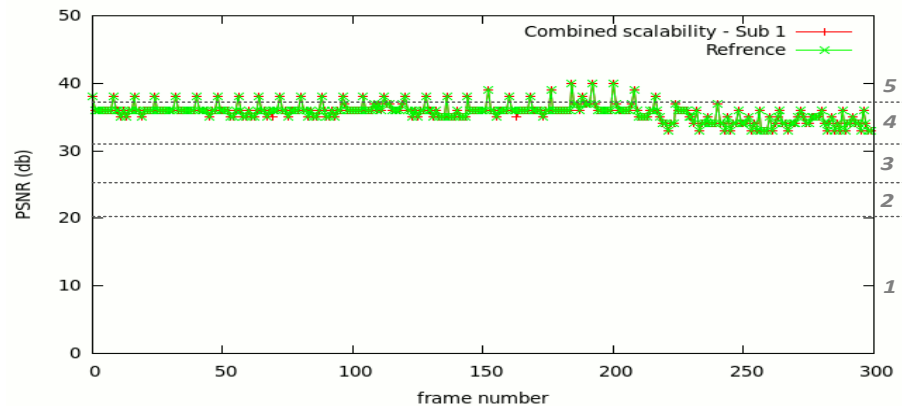


Figure 5.23: PSNR, Combined SVC, Sub=1, AVG = 36.22 dB

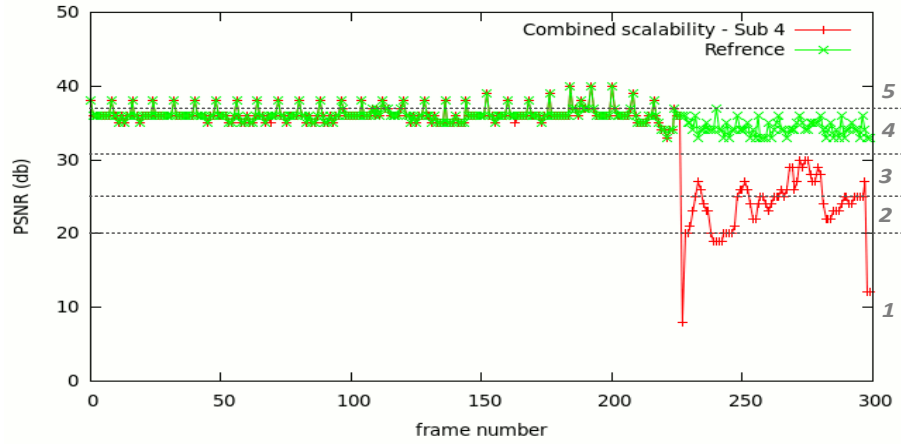


Figure 5.24: PSNR, Combined SVC, Sub=4, AVG = 33.64 dB

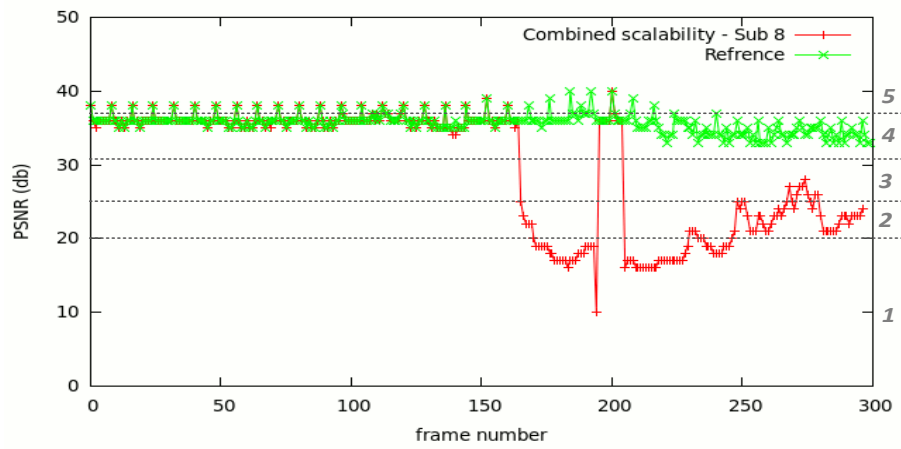


Figure 5.25: PSNR, Combined SVC, Sub =8, AVG =30.1 dB

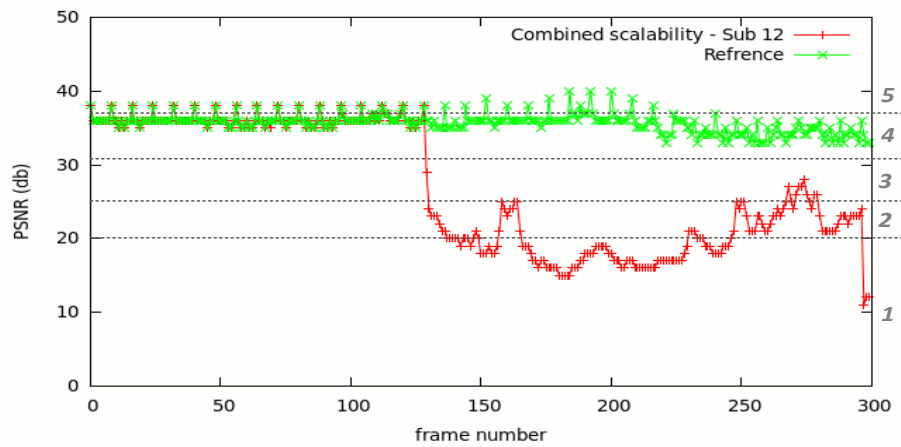


Figure 5.26: PSNR, Combined type, Sub=12, AVG = 27.36 dB

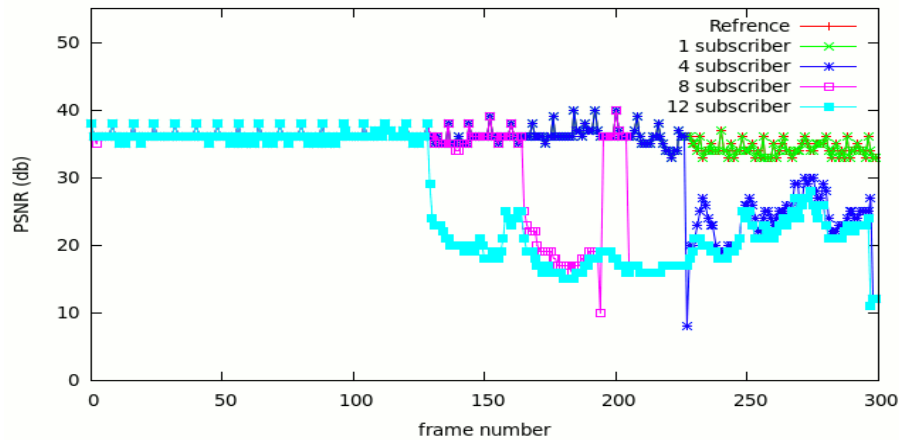


Figure 5.27: Y- PSNR Results, Combined SVC type

We see from PSNR results for all types a smooth degradation when subscriber increases from 1 to 4 and it's become sharp in case of 8 and 12 Subscribers. SNR scalability type reports the best PSNR values compare to other type with average PSNR equals 38.68 dB in the case of 1 subscriber, smooth degrading to video quality is achieved compare to other type. For MOS level, SNR also gives best MOS value especially at low load, combined scalability give good MOS value, only few frames have bad value. The justification for this is due to better coding efficiency achieved with SNR type where frame is fragmented to multiple packets while with temporal type whole frames are encapsulated to single packet, when dropped occurred the frame always lost in temporal which only in the worst case frame dropped in SNR. Furthermore, Packet loss with SNR is smaller compare to other type and this due to smallest video encoded size compare to other, network load will be minimum with SNR and quality scalability.

Figure 5.28 reports video snapshot from the receiving videos after use the four SVC types. First row represent video after using temporal type with different number of receivers, then spatial, SNR and combined respectively. Snapshots reflect the actual video quality seen by users. We see from the figure that SNR and combined types give a clear videos without interrupts even when receivers increased. In the next side, we see an interruption in videos when using temporal type and apply more load. Spatial do a better compare to temporal expect some jerks at 12 subscribers.

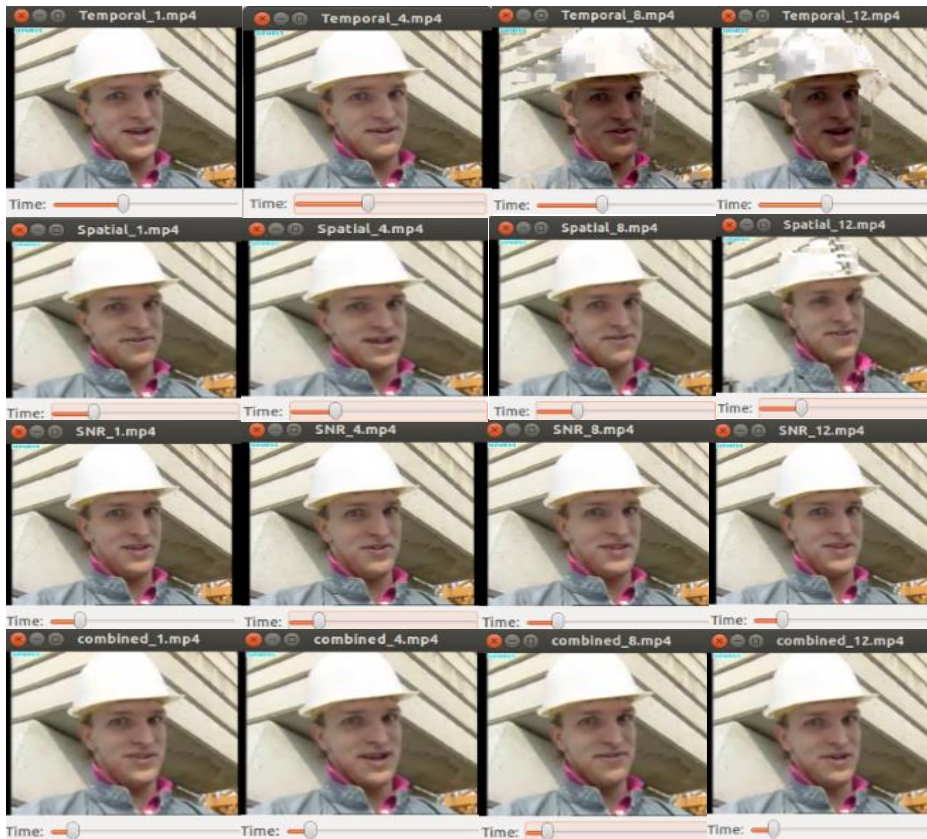


Figure 5.28: Video Snapshots for different SVC types and Subscribers

5.3.2 Frame Delay Results

Figure 5.29-5.32 report frame end-to-end delay using different SVC type for 1 and 12 receiver. Table 5.3 reports average delay using different SVC type for different number of receiver. For frame delay, it should be within play-out buffer deadline to be useful (below 150 ms).

Figure 5.29 reports frame delay for temporal SVC. We see from figure that there are a noticeable delay when subscribers are increased to 12 especially the values above 150 ms. Average delay increases from 46.25 ms in the case of 1 subscriber to 72.36 ms when 12 receivers applied. Spatial type has a better delay results with low load compare to temporal ,but at high load of 12 receivers average delay becomes worse with 82.17 ms, see Figure 5.30.

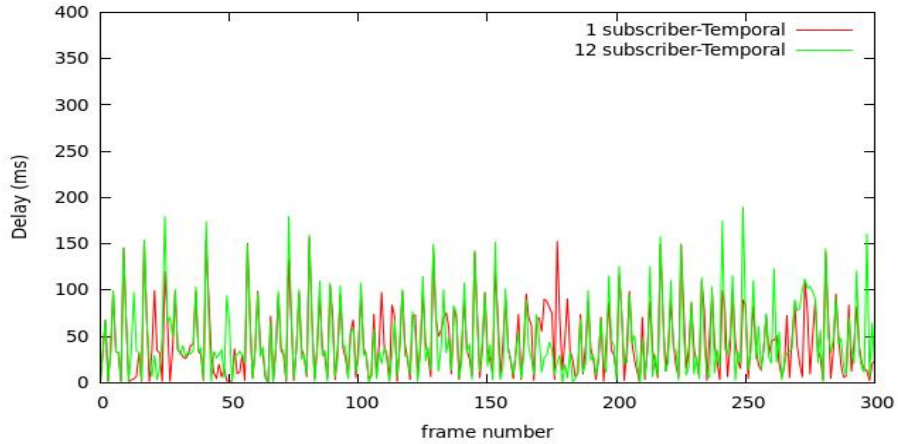


Figure 5.29: Frames Delay, Temporal SVC.

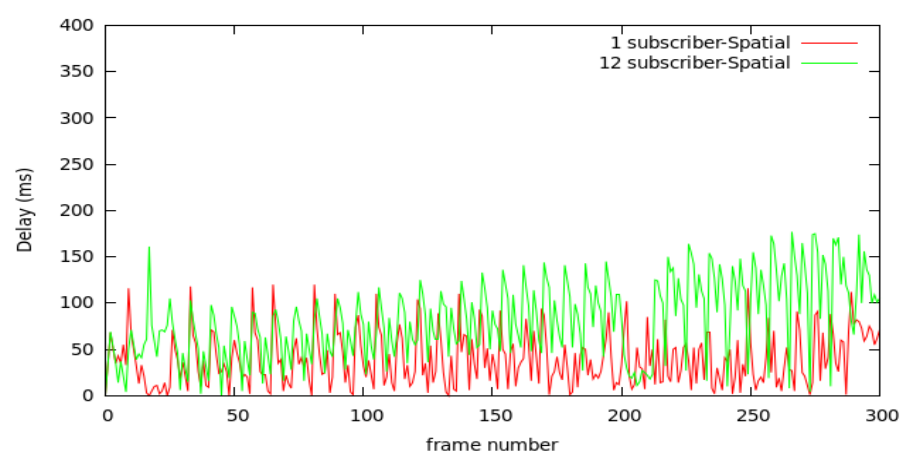


Figure 5.30: Frames Delay, Spatial SVC.

Figure 5.31-5.32 report frame delay for SNR and Combined respectively. We see that combined gives the best delay results at low load with minimum average 35.55 ms. But at high load, SNR reports minimum delay average 57.09 ms when 12 receivers exist. From Delay results we see that SNR reports best delay values specially when subscriber increased which mean that it's more endure to network load, this is because more encoding efficiency which lead to minimum encoded size as we mentioned earlier. The source of the frames delay is mainly due to transmission time for every frame. In Additional, there are another sources of delay; which are delay due to buffering at receiver which that is used to minimize jitter, also frames re-ordering based on encoding order is a source of delay before it was decoded. Processing time for video trace file after encoding at sender side and before decoding at receiver side is a causing and effect on frames delay.

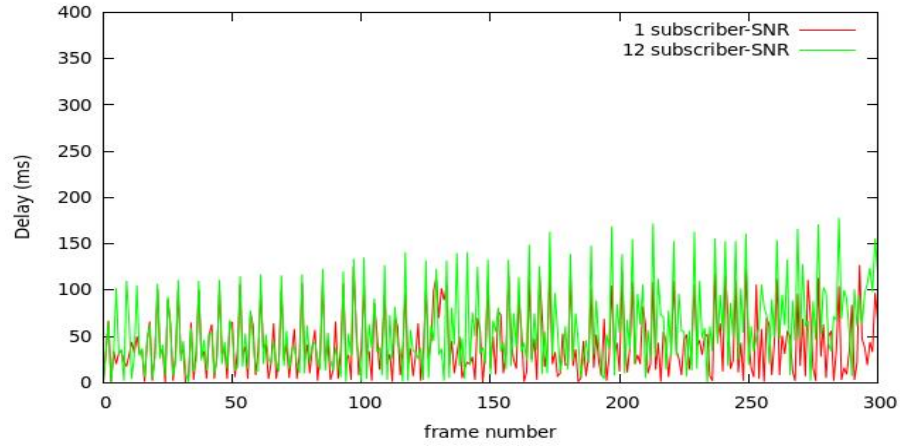


Figure 5.31: Frames Delay, SNR SVC.

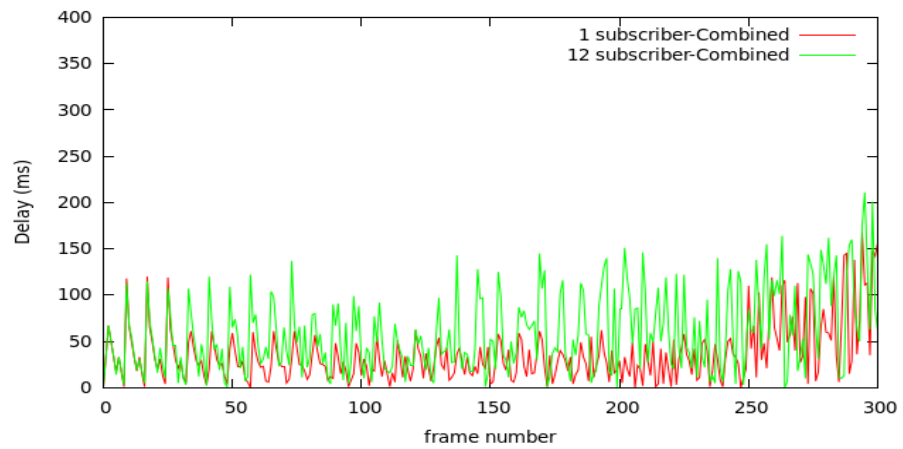


Figure 5.32: Frames Delay, Combined SVC.

Table 5.3: Average frame delay in (ms) for different SVC types

Subscriber	Temporal	Spatial	SNR	Combined
1	46.25	39.12	42.27	35.55
4	46.62	49.39	50.33	43.42
8	53.16	56.79	50.48	51.82
12	72.36	82.17	57.09	59.3

If we compare all delay results, we see that video packets experienced the higher transmission delay with temporal and spatial type, this is because the higher video encoded size compares to the other type, this lead to higher network load. SNR type gives the best end-to-end delay because of minimum encoded video size. Furthermore, video in SNR was encoded to maximum number of NALU which leads to reduction in packets size and this reduce its drooping priority.

5.3.3 Jitter Results

Figure 5.33-36 report frame jitter using different SVC types for 1 and 12 receivers. Table 5.4 reports average jitter using different SVC types for different number of receivers. We see from figures and table 4 that combined type give the minimum jitter at low load, at high load temporal and SNR type give the better jitter results except some positive peak that exceed 20 ms in temporal type.

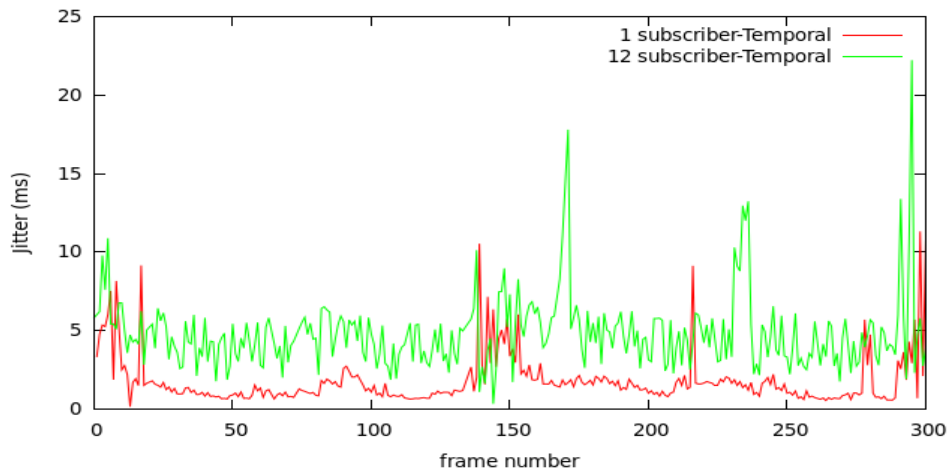


Figure 5.33: Frame Jitter, Temporal SVC.

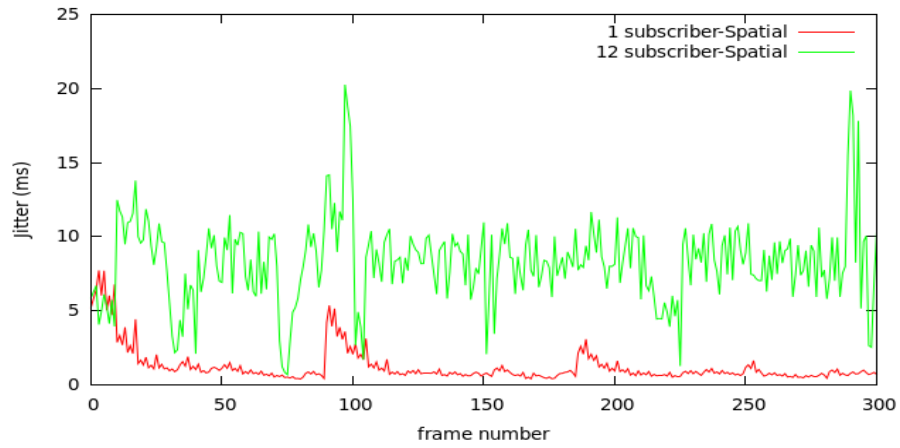


Figure 5.34: Frame Jitter, Spatial SVC.

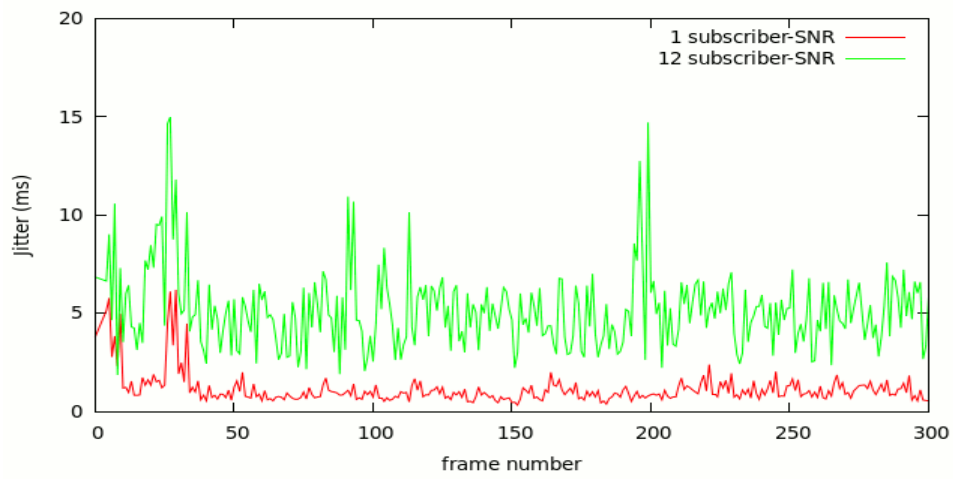


Figure 5.35: Frame Jitter, SNR SVC.

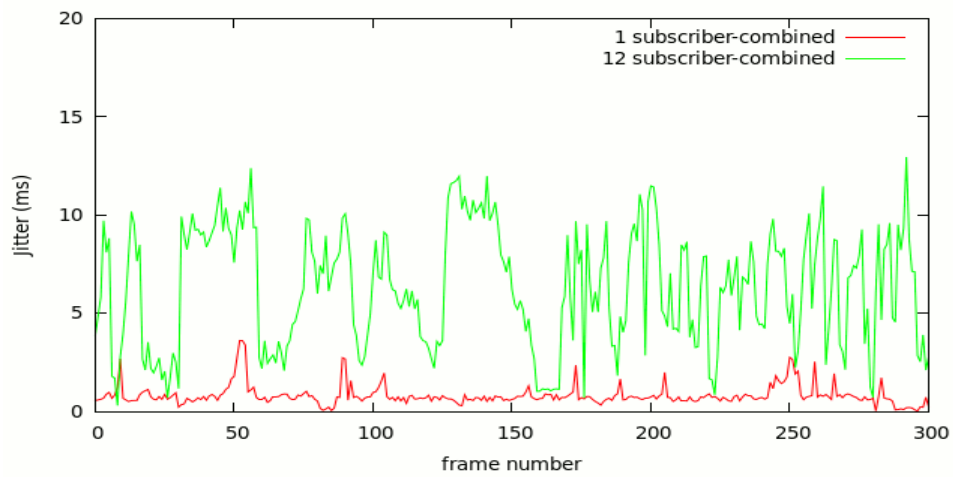


Figure 5.36: Frame Jitter, Combined SVC.

Table 5.4 Average jitter in (ms) for different SVC type

Subscriber	Temporal	Spatial	SNR	Combined
1	1.78	1.23	1.1366	0.82
4	3.6	2.36	2.323	1.52
8	4.65	5.15	3.43	3.67
12	4.88	8.72	5.16	6.32

CHAPTER 6

CONCLUSION AND FUTURE WORK

Real-Time video streaming over Wireless has a lot of difficulties since it requires a high bandwidth channel and rigorous transport delay. Wireless networks have problems of the time-varying losses and varying available bandwidth. Different solutions are proposed for efficient real-time video over Wi-Fi. Scalable video coding (SVC) or layer coding is proposed to achieve graceful degradation in video quality for lossy transmission environments by dropping part of enhancement layer without the need of re-encoding. SVC or layer coding face a problem of unequal layer protection, base layer may be dropped while there is a chance for higher enhancement layer to be dropped first. In Addition, the whole scalable layer may dropped while there is chance to drop packet by packet. Drooping Base layer affect video quality and lead to sharp degradation, graceful degradation can be achieved when enhancement layers are dropped first.

In this work, I studied the behavior of real-time video streaming over Wi-Fi with the associated. I proposed and developed a solution based on new standard video encoding H.264/SVC over Data distribution service middle ware (DDS). My proposed solution is based on unequal packet dropping priority, regardless of dropping whole scalable layer (when switching to low quality are needed) dropping was done packet by packet to achieve more graceful degradation. Video was encoded to four H.264/SVC type: Temporal, Spatial, SNR (Quality) and Combined scalability. SVEF (Scalable Video-streaming Evaluation Framework) test tool was used to assess receive video performance, we make the use of specific metrics like PSNR (Peak signal to noise ratio) or MOS (Mean Opinion Score) which related to quality preserved by end user (Quality of experience), Frames delay and jitter. DDS was used to implement the proposed approach achieve required scalability throw QoS like partition and deadline and communication scalability throw publisher/subscriber model.

Experimental results show a graceful degradation is achieved to Video Quality (PSNR) in all SVC type as load increased. Performance is sensitive to encoding types and network load, The analysis shows that SNR (Quality) performed better performance in terms of PSNR, MOS level, and frame delay, this is due to better coding efficiency, which lead to minimum video encoding size and minimum packet size.

APPENDIX A

Video Encoding and Instruction

APPENDIX A contains a details discussion regards the programs, commands, inputs and outputs data that are associated with encoding and decoding process. Furthermore, this part discuss command instruction and tools that are used in performance evaluation.

The following sections explain the main steps for experimental work which are: Video Encoding, Trace File extraction, Publishing, Subscription and decoding.

A.1 Video Encoding

I mentioned earlier that Video compression (encoding) is an important concept in video streaming by removing or reducing redundant content from video file, so that it will be effectively sent over networks, in this part I explained encoding and decoding process which are performed using JSVM tools [38] with associated input parameters file and output data in each phase.

In encoding part, Scalable video coding standard (SVC) [28] is used with its four type four: Temporal, Spatial, SNR (Quality) and Combined Scalability. In all encoding type: the raw video sequence Foreman.yuv (300 frame) is used see table 5.1 which contain video details.

The JSVM *H264AVCEncoderLibTestStatic* tool is used to perform encoding by configuring all relevant encoding setting and the type of scalability. After Encoding, Video trace will be extracted using The JSVM *H264AVCEncoderLibTestStatic* tool. In the following subsections, I explain in details the encoding process based on different encoding type.

A.1.1 Temporal type.

In Temporal type, two parameter file are used: main.cfg and layer0.cfg. The output will be an encoded video with (.264) format The following Unix command line is used to perform encoding:


```

JSVM 9.19.14 Encoder

profile & level info:
=====
DQ= 0:  Main @ Level 2.1

```

AU	0:	I	T0	L0	Q0	QP 26	Y 39.1541	U 42.0347	V 45.1828	73536 bit
AU	8:	P	T0	L0	Q0	QP 26	Y 38.7813	U 42.0895	V 45.0952	39784 bit
AU	4:	B	T1	L0	Q0	QP 29	Y 37.6244	U 41.8723	V 44.7600	13472 bit
AU	2:	B	T2	L0	Q0	QP 31	Y 37.4366	U 41.8395	V 44.6682	7392 bit
AU	1:	B	T3	L0	Q0	QP 32	Y 37.4572	U 41.9035	V 44.9427	3048 bit
AU	3:	B	T3	L0	Q0	QP 32	Y 37.0191	U 41.7916	V 44.5182	3608 bit
AU	6:	B	T2	L0	Q0	QP 31	Y 37.2729	U 41.8385	V 44.8710	7184 bit
AU	5:	B	T3	L0	Q0	QP 32	Y 37.0346	U 41.7872	V 44.7472	3832 bit
AU	7:	B	T3	L0	Q0	QP 32	Y 37.1397	U 41.8597	V 44.8633	3640 bit
AU	16:	P	T0	L0	Q0	QP 26	Y 38.4945	U 41.8488	V 45.1749	46536 bit
AU	12:	B	T1	L0	Q0	QP 29	Y 37.4345	U 41.7527	V 44.5879	14672 bit
AU	10:	B	T2	L0	Q0	QP 31	Y 37.0726	U 41.6730	V 44.6142	7632 bit
AU	9:	B	T3	L0	Q0	QP 32	Y 37.0313	U 41.8043	V 44.7028	3696 bit
AU	11:	B	T3	L0	Q0	QP 32	Y 36.6776	U 41.5409	V 44.4636	4008 bit
AU	14:	B	T2	L0	Q0	QP 31	Y 36.9673	U 41.3851	V 44.3662	8344 bit
AU	13:	B	T3	L0	Q0	QP 32	Y 36.4679	U 41.4806	V 44.2933	5304 bit
AU	15:	B	T3	L0	Q0	QP 32	Y 36.9851	U 41.5121	V 44.8542	3896 bit
AU	24:	P	T0	L0	Q0	QP 26	Y 38.5363	U 41.9500	V 45.0240	40272 bit
AU	20:	B	T1	L0	Q0	QP 29	Y 37.2133	U 41.7125	V 45.0120	10256 bit

Figure A.3: Decoding output using temporal scalability

After encoding is done, Video trace File will be extracted from the encoded .264 video based on this command:

```

> '/home/pc2/jsvm/bin/BitStreamExtractorStatic' -pt originaltrace.txt
foreman_cif.264

```

Figure A.4 reports the supported temporal layers after extraction is done, we see from figure that there are four temporal layer from TID = 0 to TID = 3, with four frame rates. We see also that all layers have the same resolution. The output of the extraction is the original video trace file which contains a details information for each NALU like: size, Scalable number (TID, QID and DID) and Packet type. In temporal type we see that Spatial DID and Quality QID are 0. See figure A.5.

```
JSVM 9.19.14 BitStream Extractor
Contained Layers:
=====
```

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	352x288	3.7500	182.50	182.50	(0,0,0)
1	352x288	7.5000	230.30	230.30	(0,1,0)
2	352x288	15.0000	276.90	276.90	(0,2,0)
3	352x288	30.0000	327.10	327.10	(0,3,0)

Figure A.4: Supported layer using Temporal Scalability.

<u>Start-Pos.</u>	<u>Length</u>	<u>LId</u>	<u>TId</u>	<u>QId</u>	<u>Packet-Type</u>	<u>Discardable</u>	<u>Truncatable</u>
=====	=====	===	===	===	=====	=====	=====
0x00000000	118	0	0	0	StreamHeader	No	No
0x00000076	14	0	0	0	ParameterSet	No	No
0x00000084	8	0	0	0	ParameterSet	No	No
0x0000008c	18	0	0	0	SliceData	No	No
0x0000009e	9183	0	0	0	SliceData	No	No
0x0000247d	18	0	0	0	SliceData	No	No
0x0000248f	4964	0	0	0	SliceData	No	No
0x000037f3	18	0	1	0	SliceData	Yes	No
0x00003805	1675	0	1	0	SliceData	Yes	No
0x00003e90	18	0	2	0	SliceData	Yes	No
0x00003ea2	915	0	2	0	SliceData	Yes	No
0x00004235	18	0	3	0	SliceData	Yes	No
0x00004247	371	0	3	0	SliceData	Yes	No

Figure A.5: Video Trace file using temporal SVC.

A.1.2 Spatial type:

For spatial encoding, the same command will be used to encoding raw video to .264 format with minor different

```
> '/home/pc2/jsvm/bin/H264AVCEncoderLibTestStatic' -pf main.cfg -lqp 0 30 -lqp
1 32
```

The main different is the encoding parameter in configuration files. The main file in spatial is shown in figure A.6, the major different from temporal type is that spatial supports

two scalable layers each one with certain resolution, thus there are two layer configuration file layer0.cfg and layer1.cfg see figure A.7 and A.8 respectively. We see from figures that layers file for spatial type support two resolution: CIF (352×288) and QCIF (176×144).

```
# JSVM Main Configuration File

OutputFile          foreman.264 # Bitstream file
FrameRate           30.0  # Maximum frame rate [Hz]
FramesToBeEncoded   300   # Number of frames (at input frame rate)
GOPSize             8     # GOP Size (at maximum frame rate)
BaseLayerMode       2     # Base layer mode (0,1: AVC compatible,
                        #                               2: AVC w subseq SEI)
SearchMode          4     # Search mode (0:BlockSearch, 4:FastSearch)
SearchRange         32    # Search range (Full Pel)
NumLayers           2     # Number of layers
LayerCfg            layer0.cfg # Layer configuration file
LayerCfg            layer1.cfg # Layer configuration file
```

Figure A.6: main.cfg configuration file for Spatial SVC

```
# JSVM Layer Configuration File

InputFile           foreman_qcif15.yuv # Input file
SourceWidth         176    # Input frame width
SourceHeight        144    # Input frame height
FrameRateIn         15     # Input frame rate [Hz]
FrameRateOut        15     # Output frame rate [Hz]
```

Figure A.7: layer0.cfg configuration file for Spatial SVC

```
# JSVM Layer Configuration File

InputFile           foreman_cif30.yuv # Input file
SourceWidth         352    # Input frame width
SourceHeight        288    # Input frame height
FrameRateIn         30     # Input frame rate [Hz]
FrameRateOut        30     # Output frame rate [Hz]
InterLayerPred      2     # Inter-layer Pred
                        # (0: no, 1: yes, 2: adap.)
```

Figure A.8: layer1.cfg configuration file for Spatial SVC

After decoding, trace file will be generated the (the same command in temporal type). Figure A.9 reports the supported scalable layer extracted after using spatial scalability. As see from figure, there are two spatial layer with DID=0 and 1. Every layer supports certain resolution. Video trace file is shown in figure A.10.

```

JSVM 9.19.14 BitStream Extractor

Contained Layers:
=====

```

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	176x144	3.7500	49.90	49.90	(0,0,0)
1	176x144	7.5000	63.50	63.50	(0,1,0)
2	176x144	15.0000	77.80	77.80	(0,2,0)
3	352x288	3.7500	166.60	166.60	(1,0,0)
4	352x288	7.5000	208.10	208.10	(1,1,0)
5	352x288	15.0000	251.50	251.50	(1,2,0)
6	352x288	30.0000	291.20	291.20	(1,3,0)

Figure A.9: Supported layer using Spatail Scalability

0x0000ca10	18	0	2	0	SliceData	Yes	No
0x0000ca22	874	0	2	0	SliceData	Yes	No
0x0000cd8c	2756	1	2	0	SliceData	Yes	No
0x0000d850	18	0	3	0	SliceData	Yes	No
0x0000d862	537	0	3	0	SliceData	Yes	No
0x0000da7b	1599	1	3	0	SliceData	Yes	No
0x0000e0ba	1025	1	4	0	SliceData	Yes	No
0x0000e4bb	1079	1	4	0	SliceData	Yes	No
0x0000e8f2	18	0	3	0	SliceData	Yes	No
0x0000e904	591	0	3	0	SliceData	Yes	No
0x0000eb53	1632	1	3	0	SliceData	Yes	No
0x0000f1b3	1056	1	4	0	SliceData	Yes	No

Figure A.10: Video Trace file using Spatail SVC

A.1.3 SNR (Quality Scalability)

The same encoding and trace file extraction that are done before was performed for SNR type. The following is command line used in SNR encoding:

```

>' /home/pc2/jsvm/bin/H264AVCEncoderLibTestStatic' -pf main.cfg -lqp 0 30 -rqp
0 32

```

Figures A.11-A.13 represent main, layer 0 and layer 1 configuration file respectively

```
# JSVM Main Configuration File
OutputFile          foreman.264 # Bit stream file
FrameRate           30.0 # Maximum frame rate [Hz]
FramesToBeEncoded   300 # Number of frames
GOPSize             8 # GOP Size
BaseLayerMode       2 # Base layer mode (0,1: AVC compatible,
                    # 2: AVC w subseq SEI)
CgsSnrRefinement    1 # SNR refinement as 1: MGS; 0: CGS
EncodeKeyPictures   1 # Key pics at T=0 (0:none, 1:MGS, 2:all)
MGSControl          2 # ME/MC for non-key pictures in MGS layers
                    # (0:std, 1:ME with EL, 2:ME+MC with EL)
SearchMode          4 # Search mode (0:BlockSearch, 4:FastSearch)
SearchRange         32 # Search range (Full Pel)
NumLayers           2 # Number of layers
LayerCfg            layer0.cfg # Layer configuration file
LayerCfg            layer1.cfg # Layer configuration file
```

Figure A.11: main.cfg configuration file for SNR SVC

```
# JSVM Layer Configuration File
InputFile           foreman_cif.yuv # Input file
SourceWidth         352 # Input frame width
SourceHeight        288 # Input frame height
FrameRateIn         30 # Input frame rate [Hz]
FrameRateOut        30 # Output frame rate [Hz]
MGSVectorMode       0 # MGS vector usage selection
```

Figure A.12: layer0.cfg configuration file for SNR SVC

```
# JSVM Layer Configuration File
InputFile           foreman_cif.yuv # Input file
SourceWidth         352 # Input frame width
SourceHeight        288 # Input frame height
FrameRateIn         30 # Input frame rate [Hz]
FrameRateOut        30 # Output frame rate [Hz]
InterLayerPred      1 # Inter-layer Prediction (0: no, 1: yes,
MGSVectorMode       1 # MGS vector usage selection
MGSVector0          4 # Specifies 0th position of the vector
MGSVector1          4 # Specifies 1st position of the vector
MGSVector2          8 # Specifies 2nd position of the vector
```

Figure A.13: layer1.cfg configuration file for SNR SVC

Figure A.14 reports the supported scalable layer extracted after using SNR scalability, we see for quality layer from Q= 0 to 3, every layer support certain quality. The extracted Video trace file is shown in figure A.15.

Layer	Resolution	Frame rate	Bit rate	MinBitrate	DTQ
0	352x288	3.7500	153.40	153.40	(0,0,0)
1	352x288	7.5000	195.60	195.60	(0,1,0)
2	352x288	15.0000	236.80	236.80	(0,2,0)
3	352x288	30.0000	283.60	283.60	(0,3,0)
4	352x288	3.7500	273.30		(0,0,1)
5	352x288	3.7500	345.00		(0,0,2)
6	352x288	3.7500	439.10		(0,0,3)
7	352x288	7.5000	343.30		(0,1,1)
8	352x288	7.5000	426.60		(0,1,2)
9	352x288	7.5000	535.00		(0,1,3)
10	352x288	15.0000	406.80		(0,2,1)
11	352x288	15.0000	498.30		(0,2,2)
12	352x288	15.0000	618.90		(0,2,3)
13	352x288	30.0000	477.10		(0,3,1)
14	352x288	30.0000	578.10		(0,3,2)
15	352x288	30.0000	713.50		(0,3,3)

Figure A.14: Supported layer using SNR Scalability

0x00004078	19	0	0	0	SliceData	No	No
0x0000408b	4197	0	0	0	SliceData	No	No
0x000050f0	3807	0	0	1	SliceData	Yes	No
0x00005fcf	2035	0	0	2	SliceData	Yes	No
0x000067c2	2654	0	0	3	SliceData	Yes	No
0x00007220	18	0	1	0	SliceData	Yes	No
0x00007232	1605	0	1	0	SliceData	Yes	No
0x00007877	980	0	1	1	SliceData	Yes	No
0x00007c4b	338	0	1	2	SliceData	Yes	No
0x00007d9d	302	0	1	3	SliceData	Yes	No
0x00007ecb	18	0	2	0	SliceData	Yes	No
0x00007edd	785	0	2	0	SliceData	Yes	No
0x000081ee	417	0	2	1	SliceData	Yes	No

Figure A.15: Video Trace file using SNR SVC

A.2 Video publishing and receiving

After raw video (YUV) was encoded and video trace file was extracted. We use SVEF tool *f-nstamp* which isn't provided by JSVM. *f-nstamp* add corresponding frames number for every entry (NULUs) in the video trace before it was transmitted.

```
> '/home/pc2/svef-1.5/f-nstamp' originaldecoderoutput.txt originaltrace.txt >
originaltrace-frameno.txt
```

"originaltrace-frameno.txt" represent video trace file after add frame number, this file is source that will be publishing in addition to the encoded foreman.264 video.

The following command used to start publisher program which will read video trace file entries (one by one) and publishing them. 10 refers to DDS domain id.

```
> objs/i86Linux2.6gcc4.1.1/DDS_VideoStream_publisher 10
```

Any subscriber want to receive video data must use the following command, where 10 refers to domain id. I run this command n time (on receiver side) where n represent the required number of receiver (subscriber). Each subscriber receive the published video NALU one by one and stored it in the file “receivedtrace.txt” which represent video trace file after transmission over network

```
> objs/i86Linux2.6gcc4.1.1/DDS_VideoStream_subscriber 10
```

A.3 NALU Filtering and Decoding

After video trace file (receivedtrace.txt) was received by subscriber, NALUs Filter is performed using SVEF tool *nalufilter*:

```
> '/home/pc6/svef-1.5/nalufilter' originaltrace-frames.txt  
receivedtrace.txt 5000 30 > ft.txt
```

nalufilter filtered the received trace by remove NALUs that have excessive delay, discard NALUs that unsatisfied decoding dependency (if NALU **W** depends on NALU **Z** and **Z** was not exist in received trace file, then **W** will discarded). Furthermore, NALU filter will recorded received packet according to sending order, reorder is important since decoded unordered frame will construct inconsistent video. After trace file received and filtered it passed to JSVM decoder. Ft.txt represent filtered trace file.

New copy of the original encoded video (foreman.264) is extracted by applying the filtered trace file which represents video after transmission experienced in network. This performed by JSVM BitStreamExtractorStatic using the following command.

```
> '/home/pc6/jsvm/bin/BitStreamExtractorStatic' foreman.264 foreman-  
filtered.264 -et ft.txt
```

YUV video format is generated (represent reconstructed video after transmission) from raw video using JSNM decoder.

```
> '/home/pc6/jsvm/bin/H264AVCDecoderLibTestStatic' foreman-filtered.264  
forman-filtered.yuv
```

Frame filter is used to conceal generated video by copy the previous frame where a missing frames are found.

```
> '/home/pc6/svef-1.5/framefiller' ft.txt 152064 300 forman-filtered.yuv forman-  
concealed.yuv
```

A.4 PSNR , Delay and jitter Calculation

PSNR was computed using JSVM *PSNRStatic* tool, reference PSNR compare original video with video “foreman_cif.yuv” with decoded video (before transmission to see the effect of compression and decompression on video quality) “forman_cif_new.yuv”

```
> '/home/pc6/jsvm/bin/PSNRStatic' 352 288 foreman_cif.yuv forman_cif_new.yuv  
>psnr_ref.txt
```

PSNR was computed by comparing original video with reconstructed video after transmission and decoding

```
'/home/pc6/jsvm/bin/PSNRStatic' 352 288 foreman_cif.yuv forman-concealed.yuv  
>psnr.txt
```

Delay was computed using python script exist with SVEF tool called “computedelay.py”. After trace file was received every frame has a receiving time stamp. Delay was computed by subtracting actual time stamp from expected time stamp using the following command

```
> python computedelay.py receivedtrace.txt 30
```

Jitter was computed using python script exist with SVEF tool called “computejitter.py”. jitter was computed for every frame packets by comparing packet timestamp before sending and after receiving see the following command

```
> Python computejitter.py sent_t.txt receivedtrace_t.txt
```

APPENDIX B

DDS Implementation And QoS

B.1 XML QoS Profile for DDS_VideoStream

The following XML lines represent QoS configuration used in my experiment for video streaming. Data Reader QoS are activated at publisher side and Data writer QoS are activated at subscriber side.

```
<?xml version="1.0"?>
```

```
<!--
```

Description

XML QoS Profile for DDS_VideoStream

The QoS configuration of the DDS entities in the generated example is loaded from this file.

This file is used only when it is in the current working directory or when the environment variable

NDDS_QOS_PROFILES is defined and points to this file.

For more information about XML QoS Profiles see Chapter 15 in the

RTI Connext user manual.

```
-->
```

```
<dds xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:noNamespaceSchemaLocation="/home/pc6/RTI/ndds.5.0.0/scripts/./resource  
/rtiddsgen/./qos_profiles_5.0.0/schema/rti_dds_qos_profiles.xsd" version="5.0.0">
```

```
<!-- QoS Library containing the QoS profile used in the generated example.
```


A QoS library is a named set of QoS profiles. -->

```
<qos_library name="DDS_VideoStream_Library">
```

```
<!-- QoS profile used to configure reliable communication between the DataWriter  
and DataReader created in the example code. A QoS profile groups a set of related  
QoS. -->
```

```
<qos_profile name="DDS_VideoStream_Profile" is_default_qos="true">
```

```
<!-- QoS used to configure the data writer created in the example code -->
```

```
<publisher_qos>
```

```
<!-- Example of using partition for spatial and quality scalability -->
```

```
<partition>
```

```
<name>
```

```
<element>P0.0</element>
```

```
</name>
```

```
<name>
```

```
<element>P0.1</element>
```

```
</name>
```

```
<name>
```

```
<element>P0.2</element>
```

```
</name>
```

```
<name>
```

```
<element>P0.3</element>
```

```
</name>

<name>
    <element>P1.0</element>
</name>

<name>
    <element>P1.1</element>
</name>

<name>
    <element>P1.2</element>
</name>

<name>
    <element>P1.3</element>
</name>

</partition>

</publisher_qos>

<!-- Here we set two partitions the Subscriber -->

<subscriber_qos>

    <partition>

        <name>

            <element>P0.0</element>

        </name>

    </partition>

</subscriber_qos>
```

```
<name>
    <element>P0.1</element>
</name>
<name>
    <element>P0.2</element>
</name>
<name>
    <element>P0.3</element>
</name>
<name>
    <element>P1.0</element>
</name>
<name>
    <element>P1.1</element>
</name>
<name>
    <element>P1.2</element>
</name>
<name>
    <element>P1.3</element>
</name>
```

```
        </subscriber_qos>

<datawriter_qos>

  <reliability>

    <kind>BEST_EFFORT_RELIABILITY_QOS</kind>

  </reliability>

  <history>

    <kind>KEEP_LAST_HISTORY_QOS</kind>

  </history>

  <durability>

    <kind>TRANSIENT_LOCAL_QOS</kind>

  </durability>

  <protocol>

    <rtps_reliable_writer>

      <min_send_window_size>50</min_send_window_size>

      <max_send_window_size>50</max_send_window_size>

    </rtps_reliable_writer>

  </protocol>

</datawriter_qos>

<!-- QoS used to configure the data reader created in the example code -- >

<datareader_qos>

  <reliability>
```

```
<kind>BEST_EFFORT_RELIABILITY_QOS</kind>
</reliability>
<history>
  <kind>KEEP_LAST_HISTORY_QOS</kind>
  <depth>2</depth>
</history>
<durability>
  <kind>TRANSIENT_LOCAL_QOS</kind>
</durability>

<deadline>
  <period>
    <nanosec>150 000 000</nanosec> <!-- 150 ms -->
  </period>
</deadline>

<time_based_filter>
  <minimum_separation>
    <nanosec>30 000 000</nanosec> <!-- 30 ms -->
  </minimum_separation>
</time_based_filter>

</datareader_qos>
```

```
<participant_qos>

  <participant_name>

    <name>Video Streaming Over Wireless, Project</name>

  </participant_name>

</discovery>

  <initial_peers>

    <element>239.255.0.1</element>

    <element>builtin.shmem://</element>

    <element>builtin.udpv4://127.0.0.1</element>

    <element>builtin.udpv4://192.168.1.10</element>

    <element>builtin.udpv4://192.168.1.20</element>

    <element>builtin.udpv4://192.168.1.30</element>

    <element>builtin.udpv4://192.168.1.40</element>

  </initial_peers>

  <multicast_receive_addresses>

    <element>293.255.0.1</element>

  </multicast_receive_addresses>

</discovery>

</participant_qos>

</qos_profile>

</qos_library>
```

</dds>

B.2 Sample OF DDS Implementation with SVEF

As we mentioned, switching between different subs streams depends on network condition which are done based on available partitions using different DDS Data Writer. DW-8 represent the highest quality sub-stream and all partition are copied to. DW-1 represent lowest quality sub-stream and contains only *P0.0*.

See the following sample code for using spatial type where DID refers to spatial layer,

```
// the higher resolution partition P1.3 in each GOP, is copied only to writer 8
```

```
if (nalutosend-> lid == 3 && nalutosend-> Did == 1){  
  
retcode = Video_writer8->write(*instance, instance_handle);  
  
    if (retcode != DDS_RETCODE_OK) {  
  
        printf("write error %d\n", retcode); } }
```

```
// partition P1.2 in each GOP, is copied to the writer 7 and 8
```

```
else if (nalutosend-> lid == 2 && nalutosend-> Did== 1){  
  
retcode = Video_writer8->write(*instance, instance_handle);  
  
retcode = Video_writer7->write(*instance, instance_handle);  
  
    if (retcode != DDS_RETCODE_OK){  
  
        printf("write error %d\n", retcode); } }
```

```
// partition P1.1 in each GOP, has been copied to the writer 6 , 7 and 8
```

```

else if (nalutosend-> lid == 1 && nalutosend-> Did== 1) {

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK){

        printf("write error %d\n", retcode);}}

// partition P1.0 in each GOP, is copied to the writer 5 ,6 , 7 and 8

if (nalutosend-> lid == 0 && nalutosend-> Did== 1){

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

retcode = Video_writer5->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK){

        printf("write error %d\n", retcode);}}

// partition P0.3 in each GOP, is copied to the writer 4, 5 ,6 , 7 and 8

if (nalutosend-> lid == 3 && nalutosend-> Did== 0){

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

retcode = Video_writer5->write(*instance, instance_handle);

```



```
retcode = Video_writer4->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK){

        printf("write error %d\n", retcode);}}
```

// partition P0.2 in each GOP, is copied to the writer 3 , 4 , 5 ,6 , 7 and 8

```
if (nalutosend-> lid == 2 && nalutosend-> Did== 0){

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

retcode = Video_writer5->write(*instance, instance_handle);

retcode = Video_writer4->write(*instance, instance_handle);

retcode = Video_writer3->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK) {

        printf("write error %d\n", retcode); }}
```

// partition P0.1 in each GOP, is copied to the writer 2 , 3 , 4 , 5 ,6 , 7 and 8

```
if (nalutosend-> lid == 1 && nalutosend-> Did== 0) {

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

retcode = Video_writer5->write(*instance, instance_handle);

retcode = Video_writer4->write(*instance, instance_handle);
```

```
retcode = Video_writer3->write(*instance, instance_handle);

retcode = Video_writer2->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK) {

        printf("write error %d\n", retcode); }}
```

// partition P0.0 in each GOP, is copied to the writer 1, 3, 4, 5,6, 7 and 8

```
if (nalutosend-> lid == 0 && nalutosend-> Did== 0) {

retcode = Video_writer8->write(*instance, instance_handle);

retcode = Video_writer7->write(*instance, instance_handle);

retcode = Video_writer6->write(*instance, instance_handle);

retcode = Video_writer5->write(*instance, instance_handle);

retcode = Video_writer4->write(*instance, instance_handle);

retcode = Video_writer3->write(*instance, instance_handle);

retcode = Video_writer2->write(*instance, instance_handle);

retcode = Video_writer1->write(*instance, instance_handle);

    if (retcode != DDS_RETCODE_OK)

        printf("write error %d\n", retcode); }}
```

REFERENCES

- [1] Deshpande, Sachin Govind, and Petrus JL Van Beek. "Wireless video transmission system." U.S. Patent 8,018,850, issued September 13, 2011.
- [2] Shankar, N. Sai, and Mihaela van der Schaar. "Performance analysis of video transmission over IEEE 802.11 a/e WLANs." *Vehicular Technology, IEEE Transactions on* 56, no. 4 (2007): 2346-2362.
- [3] Al-Madani, Basem, Mohammed Al-Saeedi, and Anas A. Al-Roubaiey. "Scalable Wireless Video Streaming over Real-Time Publish Subscribe Protocol (RTPS)." In *Distributed Simulation and Real Time Applications (DS-RT), 2013 IEEE/ACM 17th International Symposium on*, pp. 221-230. IEEE, 2013.
- [4] Cranley, Nicola, and Mark Davis. "Study of the Behavior of Video Streaming over IEEE 802.11 b WLAN Networks." In *Wireless and Mobile Computing, Networking and Communications, 2006.(WiMob'2006). IEEE International Conference on*, pp. 349-355. IEEE, 2006.
- [5] Lei, Zhijun, and Nicolas D. Georganas. "Adaptive video transcoding and streaming over wireless channels." *Journal of Systems and Software* 75, no. 3 (2005): 253-270.
- [6] "The Scalable Video Coding Amendment of the H.264/AVC Standard (2008)", web Site: <http://blog.csdn.net/worldpharos/article/details/3369933>.
- [7] Luo, Haiyan, Song Ci, Dalei Wu, and Hui Tang. "End-to-end optimized TCP-friendly rate control for real-time video streaming over wireless multi-hop networks." *Journal of Visual Communication and Image Representation* 21, no. 2 (2010): 98-106.
- [8] Yang, Fan, Qian Zhang, Wenwu Zhu, and Ya-Qin Zhang. "End-to-end TCP-friendly streaming protocol and bit allocation for scalable video over wireless Internet." *Selected Areas in Communications, IEEE Journal on* 22, no. 4 (2004): 777-790.
- [9] Korhonen, Jari, Andrew Perkis, and Ulrich Reiter. "Congestion control in wireless links based on selective delivery of erroneous packets." *Signal Processing: Image Communication* 26, no. 2 (2011): 105-115.

- [10] Chen, Minghua, and Avidesh Zakhor. "Rate control for streaming video over wireless." In INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, pp. 1181-1190. IEEE, 2004.
- [11] Zainaldin, Ahmed, Ioannis Lambadaris, and Biswajit Nandy. "Adaptive rate control low bit-rate video transmission over wireless Zigbee networks." In Communications, 2008. ICC'08. IEEE International Conference on, pp. 52-58. IEEE, 2008.
- [12] Lee, Changhyun, Kwanwoong Song, Younghun Joo, and Yongserk Kim. "Adaptive rate control for real-time video streaming over the mobile WiMAX." In Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on, pp. 1454-1457. IEEE, 2008.
- [13] Loiacono, Michael, Jeffrey Johnson, Justinian Rosca, and Wade Trappe. "Cross-layer link adaptation for wireless video." In Communications (ICC), 2010 IEEE International Conference on, pp. 1-6. IEEE, 2010.
- [14] Li, Zhiling, and Jinhe Zhou. "Adaptive Cross-layer QoS for priority-based video streaming over wireless channel." In Future Information Networks, 2009. ICFIN 2009. First International Conference on, pp. 108-112. IEEE, 2009.
- [15] Setton, Eric, Taesang Yoo, Xiaoqing Zhu, Andrea Goldsmith, and Bernd Girod. "Cross-layer design of ad hoc networks for real-time video streaming." *Wireless Communications*, IEEE 12, no. 4 (2005): 59-65
- [16] Luo, Haiyan, Song Ci, Dalei Wu, Jianjun Wu, and Hui Tang. "Quality-driven cross-layer optimized video delivery over LTE." *Communications Magazine*, IEEE 48, no. 2 (2010): 102-109.
- [17] Wang, Chia-Hui, Ray-I. Chang, Jan-Ming Ho, and Shun-Chin Hsu. "Rate-sensitive ARQ for real-time video streaming." In Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE, vol. 6, pp. 3361-3365. IEEE, 2003.
- [18] Tsai, Ming-Fong, Ce-Kuen Shieh, Chih-Heng Ke, and Der-Jiunn Deng. "Sub-packet forward error correction mechanism for video streaming over wireless networks." *Multimedia Tools and Applications* 47, no. 1 (2010): 49-69.

- [19] Nafaa, Abdelhamid, Tarik Taleb, and Liam Murphy. "Forward error correction strategies for media streaming over wireless networks." *Communications Magazine*, IEEE 46, no. 1 (2008): 72-79.
- [20] Nguyen, Dong, Tuan Tran, Thinh Nguyen, and Bella Bose. "Wireless broadcast using network coding." *Vehicular Technology, IEEE Transactions on* 58, no. 2 (2009): 914-925.
- [21] kram Hakiri, Pascal Berthou, Aniruddha Gokhale, Douglas C. Schmidt, Thierry Gayraud, "Supporting end-to-end quality of service properties in OMG data distribution service publish/subscribe middleware over wide area networks, *Journal of Systems and Software*, Volume 86, Issue: October 2013, Pages 2574-2593.
- [22] Tusch, Roland. "Towards an adaptive distributed multimedia streaming server architecture based on service-oriented components." In *Modular Programming Languages*, pp. 78-87. Springer Berlin Heidelberg, 2003.
- [23] Hoffert, Joe, Douglas C. Schmidt, and Aniruddha Gokhale. "Evaluating transport protocols for real-time event stream processing middleware and applications." In *On the Move to Meaningful Internet Systems: OTM 2009*, pp. 614-633. Springer Berlin Heidelberg, 2009.
- [24] García-Valls, Marisol, Pablo Basanta-Val, and Iria Estévez-Ayres. "Adaptive real-time video transmission over DDS." In *Industrial Informatics (INDIN), 2010 8th IEEE International Conference on*, pp. 130-135. IEEE, 2010.
- [25] Detti, Andrea, Pierpaolo Loreti, Nicola Blefari-Melazzi, and Francesco Fedi. "Streaming H. 264 scalable video over data distribution service in a wireless environment." In *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*, pp. 1-3. IEEE, 2010.
- [26] Al-madani, Basem, A. Al-Roubaiey, and Taher Al-shehari. "Wireless video streaming over Data Distribution Service middleware." In *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*, pp. 263-266. IEEE, 2012.
- [27] Al-Madani, Basem, Anas Al-Roubaiey, and Zubair A. Baig. "Real-Time QoS-Aware Video Streaming: A Comparative and Experimental Study." *Advances in Multimedia 2014* (2014).

- [28] Schwarz, Heiko, Detlev Marpe, and Thomas Wiegand. "Overview of the scalable video coding extension of the H. 264/AVC standard." *Circuits and Systems for Video Technology*, IEEE Transactions on 17, no. 9 (2007): 1103-1120.
- [29] Kofler, Ingo, Martin Prangl, Robert Kuschnig, and Hermann Hellwagner. "An H.264/SVC-based adaptation proxy on a WiFi router." In *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 63-68. ACM, 2008.
- [30] Hillestad, Odd Inge, Andrew Perkis, Vasken Genc, Seán Murphy, and John Murphy. "Adaptive H. 264/MPEG-4 SVC video over IEEE 802.16 broadband wireless networks." In *Packet Video 2007*, pp. 26-35. IEEE, 2007.
- [31] Migliorini, Daniele, Enzo Mingozzi, and Carlo Vallati. "Performance evaluation of H. 264/SVC video streaming over mobile WiMAX." *Computer Networks*, Elsevier 55, no. 15 (2011): 3578-3591.
- [32] Li, Xiang, Peter Amon, Andreas Hutter, and André Kaup. "Performance analysis of inter-layer prediction in scalable video coding extension of H. 264/AVC." *Broadcasting*, IEEE Transactions on 57, no. 1 (2011): 66-74.
- [33] Hsiao, Yi-Mao, Jeng-Farn Lee, Jai-Shiarng Chen, and Yuan-Sun Chu. "H. 264 video transmissions over wireless networks: Challenges and solutions." *Computer Communications* 34, no. 14 (2011): 1661-1672.
- [34] Zhang, Honghai, Yanyan Zheng, Mohammad Ali Khojastepour, and Sampath Rangarajan. "Cross-layer optimization for streaming scalable video over fading wireless networks." *Selected Areas in Communications*, IEEE Journal on 28, no. 3 (2010): 344-353.
- [35] Schierl, Thomas, Cornelius Hellge, Shpend Mirta, K. Gruneberg, and Thomas Wiegand. "Using H. 264/AVC-based scalable video coding (SVC) for real time streaming in wireless IP networks." In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pp. 3455-3458. IEEE, 2007.
- [36] Apostolopoulos, John G., Wai-tian Tan, and Susie J. Wee. "Video streaming: Concepts, algorithms, and systems." HP Laboratories, report HPL-2002-260 (2002).

- [37] Schierl, Thomas, Thomas Stockhammer, and Thomas Wiegand. "Mobile video transmission using scalable video coding." *Circuits and Systems for Video Technology*, IEEE Transactions on 17, no. 9 (2007): 1204-1217.
- [38] Joint Video Team (JVT), "JSVM (Joint Scalable Video Model) Software Manual". V9.18 (last update June-2009) [Online]. Available: <https://evalsvc.googlecode.com/files/SoftwareManual.doc>.
- [39] OMG. "Data Distribution Service for Real-time systems". Object Management Group, 1.2 formal/07-01-01 edition, January 2007.
- [40] Oh, Sangyoon, Jai-Hoon Kim, and Geoffrey Fox. "Real-time performance analysis for publish/subscribe systems." *Future Generation Computer Systems* 26, no. 3 (2010): 318-323.
- [41] Network Working Group, RFC: 3550, "RTP: A Transport Protocol for Real-Time Applications", July 2003.
- [42] T. Wiegand, G. J. Sullivan, J. Reichel, H. Schwarz, and M. Wien, Joint Draft 11 of SVC Amendment, Joint Video Team, Jul.2007.
- [43] A. Detti, G. Bianchi, W. Kellerer, et al. "SVEF: an Open-Source Experimental Evaluation Framework" in *Proc. of IEEE MediaWIN 2009*, Sousse, Tunisia.
- [44] Schwarz, Heiko, Detlev Marpe, Thomas Schierl, and Thomas Wiegand. "Combined scalability support for the scalable extension of H. 264/AVC." In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pp. 4-pp. IEEE, 2005.
- [45] Pardo-Castellote, Gerardo. "Omg data-distribution service: Architectural overview." In *Distributed Computing Systems Workshops, 2003. Proceedings. 23rd International Conference on*, pp. 200-206. IEEE, 2003.
- [46] Lopez-Vega, Jose M., Javier Sanchez-Monedero, Javier Povedano-Molina, and Juan M. Lopez-Soler. "QoS policies for audio/video distribution over DDS middleware." In *Workshop on Distributed Object Computing for Real-time and Embedded Systems*. 2008.
- [47] Quality of Service Design Overview, Book: "Enterprise QoS Solution Reference Network Design Guide" by Cisco (2005). Available in:

http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/WAN_and_MAN/QoS_SRND/QoS-SRND-Book.pdf

- [48] Ke, Chih-Heng, Ce-Kuen Shieh, Wen-Shyang Hwang, and Artur Ziviani. "An Evaluation Framework for More Realistic Simulations of MPEG Video Transmission." *Journal of Information Science & Engineering* 24, no. 2 (2008).
- [49] Le, Tien Anh, Hang Nguyen, and Hongguang Zhang. "EvalSVC—an evaluation platform for scalable video coding transmission." In *Consumer Electronics (ISCE), 2010 IEEE 14th International Symposium on*, pp. 1-6. IEEE, 2010.
- [50] Andrea Detti, Claudio Pisa. SVEF: Scalable Video-streaming Evaluation Framework. Reference software: <http://svef.netgroup.uniroma2.it/>.
- [51] Migliorini, Daniele, Enzo Mingozzi, and Carlo Vallati. "QoE-oriented performance evaluation of video streaming over WiMAX." In *Wired/wireless Internet communications*, pp. 240-251. Springer Berlin Heidelberg, 2010.
- [52] Piamrat, Kandaraj, Cesar Viho, J. Bonnin, and Adlen Ksentini. "Quality of experience measurements for video streaming over wireless networks." In *Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on*, pp. 1184-1189. IEEE, 2009.
- [53] Definition of Quality of Experience (QoE), Reference: TD 109rev2 (PLEN/12), International Telecommunication Union – ITU. Jan, 2007.

Vitae

Name: Mohammad Fawzi Ahmad Al-Hammouri.

Nationality: Jordan.

Date of Birth : 11/12/1986

Email : g201203060@kfupm.edu.sa.

Address: King Fahd University of Petroleum and Minerals Dhahran, 31261, Saudi Arabia.

Education:

2003 to 2004	General Secondary School Certificate- GPA "92.2%", ISS, Jordan – Irbid.
2004 to 2009	Bachelor in Computer Engineering , GPA "89.9, Excellent, 2nd Rank out of 70, Deans Honor List". Yarmouk University, Hijjawi faculty for Engineering Technology. Irbid-Jordan
2012- 2014	MSc degree In computer engineering , king Fahd University for petroleum and mineral, GPA"3.65 out of 4", Dhahran-Saudi Arabia

Work Experience

Jun-2008 till Jan-2009	Software Developer Trainee ((ASP.NET, JS, MS-SQL), ESKADENIA Software , Amman - Jordan
2009 till 2010	Lab Instructor, Yarmouk University , Irbid-Jordan.

Feb-2010 till Aug-2012 Oracle Programmer, **Computer Center/Yarmouk University**, Irbid-Jordan.

Publication

- Al-Madani, Basem, Anas Al-Roubaiey, and Mohammad F. Al-Hammouri. **"Performance Enhancement of Limited-Bandwidth Industrial Control Systems."** *Advanced Materials Research* 739 (2013): 608-615
- Al-Madani, Basem, Mohammad F. Al-Hammouri and Mohammed Alsaedi. **"Scalable Unequal Packet Priority For Real-Time Wireless Video Streaming Over DDS Based Middleware"**. (To be submitted soon)