Major Project

On

Pre-Transform Scanning for Video Compression

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science & Engineering

By

Trushal Thaker

(06MCE017)

Under Guidance of

Dr. Hari Kalva (Florida Atlantic University)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERCITY OF SCIENCE & TRCHNOLOGY

Ahmedabad 382481

May 2008

This is to certify that Dissertation entitled

Pre-Transform Scanning for Video Compression

Submitted by

Trushal Thaker

has been accepted towards fulfillment of the requirement

for the degree of

Master of Technology in Computer Science & Engineering

Dr. S. N. Pradhan
P.G.Coordinator

Prof. D. J. Patel
Head of the Department

Prof. A. B. Patel
Director, Institute of Technology

CERTIFICATE

This is to certify that the work presented here by **Mr. Trushal Thaker** entitled "*Pre-Transform Scanning for Video Compression***"** has been carried out at **Florida Atlantic University, Florida, USA,** during the period **September 2007 – April 2008** is the bonafide record of the research carried out by him under my guidance and supervision and is up to the standard in respect of the content and presentation for being referred to the examiner. I further certify that the work done by him is his original work and has not been submitted for award of any other diploma or degree.

Dr. S. N. Pradhan
Professor, Department of Computer Engineering,
Institute of Technology, Ahmedabad.

Dr. Hari Kalva
Associate Professor, Department of Computer Science and Engineering,
Florida Atlantic University, Florida, USA.

Date:

ACKNOWLEDGMENT

It gives me immense pleasure in expressing my regards and profound gratitude to **Dr**. **Hari Kalva**, Associate Professor, Department of Computer Science & Engineering, Florida Atlantic University, Florida, USA, for his valuable guidance and continual encouragement throughout my Major project. I am heartily thankful to him for his precious time, suggestions and sorting out the difficulties of my topic that helped me a lot during this study.

My heartfelt gratitude to **Dr. Borko Furht**, Chairman, Department of Computer Science & Engineering, Florida Atlantic University, Florida, USA, for his immense support in providing me the required resources and making me feel at home in a foreign land.

The invaluable support and guidance provided by **Dr. S.N. Pradhan,** P.G. Coordinator, Computer Science and Engineering Department, Nirma University, Ahmedabad, needs a special mention and my heartfelt gratitude to him. Had it not been his support, I would have not completed the project in the desired fashion.

I would like to extend my special regards to **Prof. D.J Patel**, Head, Computer Science & Engineering Department, Nirma University, Ahmedabad for his encouragement and motivation throughout the Major Project. I am also thankful to **Prof. A. B. Patel**, Director, Institute of Technology, Nirma University, Ahmedabad for his kind support in all respect during my study.

I am thankful to all faculty members of Computer Science & Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

Trushal Thaker

Roll No. 06MCE017

ABSTRACT

Most video compression algorithms today use block based coding. An image or video frame data is coded one block at a time. A block of data (or residual data block in inter frames) is de-correlated using transform coding to exploit spatial redundancies in the data blocks. Transforms such as 2D DCT are used to de-correlate the video/image data to improve compression performance. The transforms used have a strong energy compaction property; after a transform is applied, most of the signal information is concentrated in a few low-frequency coefficients and results in fewer non-zero coefficients. Fewer non-zero coefficients lead to a lower bitrate and higher compression. In this thesis we have explored novel ways of improving the correlation of the data in a block in order to reduce the number of non-zero coefficients. The number of non-zero coefficients in the transformed data decrease with the increase in spatial correlation in the input data. Similarities among neighboring blocks are exploited by permuting block data using the sorting permutation of the neighbors to improve the correlation. We show that this permutation based approach reduces the number of non-zero coefficients significantly and improves compression. Performance of this method is demonstrated in an implementation in Xvid MPEG-4 video encoder.

From the results we were able to come to certain conclusions regarding the Peak Signal to Noise Ratio (PSNR) and the average number of non zero coefficients (NZC) after transformation. We were able to decrease the average NZC per frame but with loss in PSNR. Consistency was maintained as the approach was extended and implemented for various approaches and also implementation in the Xvid MPEG-4 encoder. Here in the Xvid Mpeg-4 encoder implementation we also concluded that we need a different Variable length Coder (VLC) that supports this approach as the final bitstream resulting out of the VLC was increasing in size.

From the study of the various results from all the different approaches we came to a conclusion that we need to have a new VLC for this approach as concluded from the Xvid implementation results. The proposed approach can also be extended and explored as a post transform scanning approach to see if this approach can be made to loose less PSNR.

TABLE OF CONTENTS

ACKNOWLEDGEMENT IV

ABSTRACT V

LIST OF FIGURES VIII

LIST OF TABLES X

LIST OF ABREVIATIONS XI

1 INTRODUCTION 1

1.1 MOTIVATION 1

1.2 PROBLEM STATEMENT 1

1.3 PROJECT DESCRIPTION 2

1.4 OVERVIEW OF THE THESIS 3

2 BACKGROUND & RELATED STUDY 4

2.1 DIGITAL VIDEO CONCEPTS 4

2.1.1 The need for compression 4

2.1.2 Perceptual Redundancies 5

2.1.3 The human visual system 5

2.1.4 Sensitivity to Temporal Frequencies 6

2.1.5 Sensitivity to Spatial Frequencies 6

2.1.6 EXPLOITING TEMPORAL REDUNDANCIES 7

2.1.7 BLOCK BASED MOTION ESTIMATION 7

2.1.8 EXPLOITING SPATIAL REDUNDANCIES 8

2.1.9 Exploiting statistical redundancies 9

2.2 MPEG 10

2.2.1 History 10

2.2.2 MPEG-4 10

2.2.3 BASIC MPEG-4 CONCEPTS 11

2.2.4 MPEG-4 VISUAL 13

2.2.5 SIMPLE PROFILE 14

2.2.6 ADVANCED SIMPLE PROFILE 18
2.3 Basic Encoder/Decoder 20
2.4 Hybrid Video Codec Model 21
3. PROJECT DESCRIPTION 23
3.1 SORTING PERMUTATION OF A BLOCK 23
3.2 DYNAMIC PRE-TRANSFORM BLOCK SCANS 24
3.3 DISCRETE COSINE TRANSFORM: 26
3.3.1 INTRODUCTION 26
3.3.2 ONE DIMENSIONAL DCT 27
3.3.3 2-DIMENSIONAL DCT 27
3.3.4 EFFECT ON DCT USING PRE TRANSFORM SCAN 28
4. EXPERIMENTAL SETUP AND RESULTS 33
4.1 EXPERIMENTAL SETUP 33
4.2 RESULTS 35
4.2.1 16X16 NEIGHBOURHOOD SCAN 35
4.2.2 8X8 NEIGHBOURHOOD SCAN 39
4.2.3 16X16 PREV-NEIGHBOURHOOD SCAN 42
4.2.4 8X8 PREV-NEIGHBOURHOOD SCAN 47
4.2.5 IMPLEMENTATION IN XVID ENCODER 52
5. CONCLUSION & FUTURE WORK 66
5.1 CONCLUSIONS 66
5.2 FUTURE WORK 67
REFERENCES 68
APPENDIX A PLOTS 70
APPENDIX B XVID MPEG-4 ENCODER 110
APPENDIX C XVID STATISTICS 112

LIST OF FIGURES

2.1 Structure of the human eye 6

2.2 Successive frames in a video. 8

2.3 Motion estimation 9

2.4 I-VOP encoding and decoding stages. 15

2.5 P-VOP encoding and decoding stages. 16

2.6 Video packet structure. 17

2.7 Components of a basic Encoder/Decoder 20

2.8 Hybrid Video Codec model 21

3.1 Sorting permutation 23

3.2 Generating sorting permutations.....24

3.3 (a) Blocks in a video frame (b) previously coded blocks used for determining the scan patterns for the current block X 25

3.4 Evaluation of a neighboring block permutation for encoding 26

3.5 8x8 Block of Data to Fed for DCT 28

3.6 DCT blocks (a) Without Permutation (b) With Permutation.....29

3.7 Quantized blocks (a) Without Permutation (b) With Permutation.....30

3.8 Quantization block for sorted data.....31

3.9 IDCT blocks (a) Without Permutation (b) With Permutation.....32

4.1 Implementation of dynamic pre-transform scanning 33

4.2 Sequences used for experiments.....34

4.3 Range VS NZC plots for Crew and Ice Sequences 36

4.4 QP VS NZC plots for Crew and Ice Sequences 37

4.5 RD curves for Crew and Ice Sequences 38

4.6 Gain plots for Crew and Ice Sequences 39

4.7 QP VS NZC plots for 8x8 ME for Crew and Ice Sequences 40

4.8 RD curves for 8x8 ME for Crew and Ice sequences.....42

4.9 QP Vs NZC plots for Prev-Neighbourhood Scan for Crew and Ice sequences 43

4.10 RD curves for Prev-Neighbourhood Scan for Crew and Ice sequences 44

4.11 RD curves for Comparing Prev-Neighbourhood and Neighbourhood Approaches.....45

4.12 QP Vs NZC curves for 8x8 Prev-Neighbourhood Approach 48

4.13 RD curves for 8x8 Prev-Neighbourhood Scan 49

4.14 RD curves for comparing 8x8 Prev-Neighbourhood and 8x8 Neighbourhood approach 50

4.15 QP Vs NZC plots for Crew and Ice for Xvid implementation 64

4.16 RD curves for Crew and Ice for Xvid implementation.....65

LIST OF TABLES

2.1 MPEG-4 Visual Profiles for Coding Natural Video 12

4.1 PSNR values for 16x16 approaches for Crew (4CIF) 46

4.2 PSNR values for 16x16 approaches for Ice (4CIF) 47

4.3 PSNR values for 8x8 approaches for Crew (4CIF) 51

4.4 PSNR values for 8x8 approaches for Ice (4CIF)..... 51

4.5 Histogram data for levels in Xvid implementation for City Sequence 53

4.6 Histograms for run lengths for City Sequence..... 53

4.7 Last Run level statistics for City sequence (Normal approach)..... 55

4.8 Last Run level statistics for City sequence (Neighborhood approach).... 59

C1 Last, Run, Level statistics (Normal scan) 112

C2 Last, Run, Level statistics (Neighbourhood scan) 116

LIST of ABBRVIATIONS

PSNR	Peak Signal to Noise Ratio
DCT	Discrete Cosine Transform
QP	Quantization Parameter
SSD	Sum of Squared Differences
SAD	Sum of Absolute Differences
RGB	Red Green Blue
FPS	Frames per Second
MPEG	Motion Pictures Experts Group
VHDL	Verilog Hardware Definition Language
CODEC	Coder Decoder
ISO	International Organization for Standardization
AVC	Advanced Video Coding
VOP	Vide Object Plane
ITU-T	International Telecommunications Union
TV	Television
GMV	Global Motion Vectors
ME	Motion Estimation
VLC	Variable Length Coder
GPL	General Public License
SP	Simple Profile
ASP	Advanced Simple Profile
HEC	Header Extension Code

1 INTRODUCTION

1.1 MOTIVATION

Presently the whole world has started sharing information and the major platform used to share this information is the "Internet". But as time passes the sharing is not only restricted to textual data but has extended to Multimedia data. Photographs, movie clips, etc. are just some of the examples of the type of information that is being shared in addition to text information. Digital multimedia data is used in applications such as pay-per-view TV, confidential video conferencing, military, medical, industrial multimedia systems etc. The size of such multimedia files is more and eventually more bandwidth is required to transmit them. But the surprising factor is that as the size of the shared files is increasing, bandwidth available is not keeping pace with it. So to meet this requirement compression of such files takes place.

One area of Multimedia compression which we have concentrated on is video compression. One basic characteristic of Video (multimedia) files is that it has a lot of redundant data and video compression is a process in which this redundant data is removed in such a way that the quality of the file is retained and the size is reduced considerably. More and more research is done on compression standards, and are also developed. With the advent of these new standards, compression achieved is better with quality of the compressed video being maintained but here the tradeoff is complexity. Our focus here is not to develop a new compression standard but to develop an approach that reduces the bitrate without the cost of additional complexity. This is the key motivation for the development of our approach. We decided to test our approach on the XVID (MPEG-4 Part 2) encoder. This approach can be used to reduce the bitrate without adding additional complexity.

1.2 PROBLEM STATEMENT

The objective of this project is to achieve higher compression efficiency without adding additional complexity in video coding. This approach is based on the hypothesis that motion compensated residual blocks in videos that are close to each other have similar distribution and this distribution when exploited properly will improve the compression. The theoretical and practical challenges of introducing pre transform scanned data in the hybrid encoding/decoding loop must be identified and analyzed [8]. This includes whether the new bit stream

got after quantization is supported by the variable length encoder and more compression is achieved in comparison to the standard compression technique.

The key contribution of this project is the addition of a pre-transform coefficient scanning to the video coding loop that reduces the number of non-zero coefficients after the transform is applied. The proposed approach would lead to reduction in bitrates in video coding standards. Pre-transform coefficient scanning based on neighboring block permutations is developed. This coefficient scanning increases the spatial correlation of the data block before applying DCT. When DCT is applied to the data with better spatial correlation, the number of non-zero coefficients is reduced, leading to improved compression [21]. Applying a block permutation or pre-transform scan is similar to the zig-zag like scanning of coefficients commonly used in image and video coding which is applied after the transform to increase the number of consecutive zeros (referred to as run) in a block. In the proposed approach the permutation or scanning is applied before the transform and the permutation or scan pattern is determined dynamically based on the previously decoded blocks.

The project explores many different types of pre transform scans and all of them have to be tested on the MPEG 4 Part 2 standard. As many different types of camera movements are being introduced in all the recent videos, this approach should be able to support majority of them. This approach has the potential to improve compression performance of video encoders with relatively low complexity.

1.3 PROJECT DESCRIPTION

This approach is based on the hypothesis that motion compensated residual blocks in videos that are close to each other have similar distribution and this distribution when exploited properly will improve the compression. The proposed approach scans the motion compensated residual before the transform (DCT) is applied. The scan patterns are determined dynamically based on the sorting permutations of the previously coded blocks. The proposed method is evaluated by counting the number of non-zero coefficients after DCT and quantization. This coefficient scanning increases the spatial correlation of the data block before applying DCT. When DCT is applied to the data with better spatial correlation, the number of non-zero coefficients is reduced, leading to improved compression. Applying a block permutation or pre-transform scan is similar to the zig-zag like scanning of coefficients commonly used in image and video

coding which is applied after the transform to increase the number of consecutive zeros (referred to as run) in a block. In the proposed approach the permutation or scanning is applied before the transform and the permutation or scan pattern is determined dynamically based on the previously coded blocks.

This approach would lead to an almost proportional decrease in bitrate when applied to standard video compression standards. This approach has the potential to improve compression performance of video encoders with relatively low complexity.

1.4 OVERVIEW OF THE THESIS

The thesis is organized as follows: Chapter 2 consists of all the background and related study pertaining to my work. Chapter 3 explains about the project description and various concepts related to it. Chapter 4 talks about the experimental setup and all the results got from the various experiments carried out. Chapter 5 consists of conclusions and future work.

2. BACKGROUND & RELATED STUDY

2.1 DIGITAL VIDEO CONCEPTS

Digital video is essentially a sequence of pictures displayed over time. Each picture of a digital video sequence is a 2D projection of the 3D world. Digital video thus is captured as a series of digital pictures or sampled in space and time from an analog video signal. A frame of digital video or a picture can be seen as a 2D array of pixels. Each pixel value represents the color and intensity values of a specific spatial location at a specific time. The Red-Green-Blue (RGB) color space is typically used to capture and display digital pictured. Each pixel is thus represented by one R, G and B components. The 2D array of pixels that constitutes a picture is actually three 2D arrays with one array for each of the RGB components. A resolution of 8 bits per component is usually sufficient for typical consumer applications.

2.1.1 THE NEED FOR COMPRESSION

A digital video sequence is huge amount of data and it requires a lot of resources. This can be explained by the following example: Consider a digital video sequence at a standard definition TV picture resolution of 720x480 and a frame rate of 30 frames per second(FPS). If a picture is represented using the RGB color space with 8 bits per component or 3 bytes per pixel, size of each frame is 720*480*3 bytes. The disk space required to store one second of video would be 720*420*3*30=31.1 MB. A one hour video would thus require 112 GB. To deliver video over wired and/or wireless networks, bandwidth required is 31.1*8=249 Mbps. In addition to these high storage and bandwidth requirements, using uncompressed video will add significant cost to the hardware and systems that process digital video. Digital video compression is thus necessary even with exponentially increasing bandwidth and storage capacities.

Fortunately, digital video has a lot of redundancies that can be removed with the help of compression. Compression basically removes all the possible redundancies in the video and reduces the amount of data to be transmitted. Video compression is of two types: Lossy and Lossless. Lossless video compression reproduces the identical video after decompression. Here more focus is given to the Lossy compression which yields perceptually equivalent, but not identical video compared to the uncompressed source. As the name suggest there is some information being lost so the decompression cannot lead to an

identical image that was found at the source before compression. Video compression is achieved by exploiting four types of redundancies: 1) perceptual, 2) temporal, 3) spatial, 4) statistical redundancies.

2.1.2 PERCEPTUAL REDUNDANCIES

Perceptual redundancies refer to the details of a picture that a human eye cannot perceive. Anything that a human eye cannot perceive can be discarded without affecting the quality of a picture. The human visual system gives us an understanding of how perceptual redundancies can be explained.

2.1.3 THE HUMAN VISUAL SYSTEM

The human visual system responds when the incoming light is focused on the retina. The photoreceptors in the eye are sensitive to the visible spectrum and generate a simulation that results in perception. The retina has two types of photo receptors 1) rods and 2) cones. Human eye has about 100 million rods and about 7 million cones. The rods and cones respond differently to the incident light. Rods are sensitive to the variations in intensity (lightness and darkness) and cones are sensitive to color. Rods function well under the low illumination and cones function well under the lit conditions. There are three types of cones Red, Green, Blue each sensitive to the different bands of the visible spectrum. About 64% of the cones are Red, 32% are Green and 4% are Blue. The cones, however, are not uniformly distributed on the retina. The fovea, the central area if the retina, has more green cones than reds and blues resulting in different sensitivity to different bands of the visual spectrum. This makes the human eye more sensitive to the mid spectrum; i.e., the incoming blues and reds have to be brighter than greens and yellows to give a perception of equal brightness. Because of the large number of rods, the human eye is more sensitive to variations in intensity than variation in color.

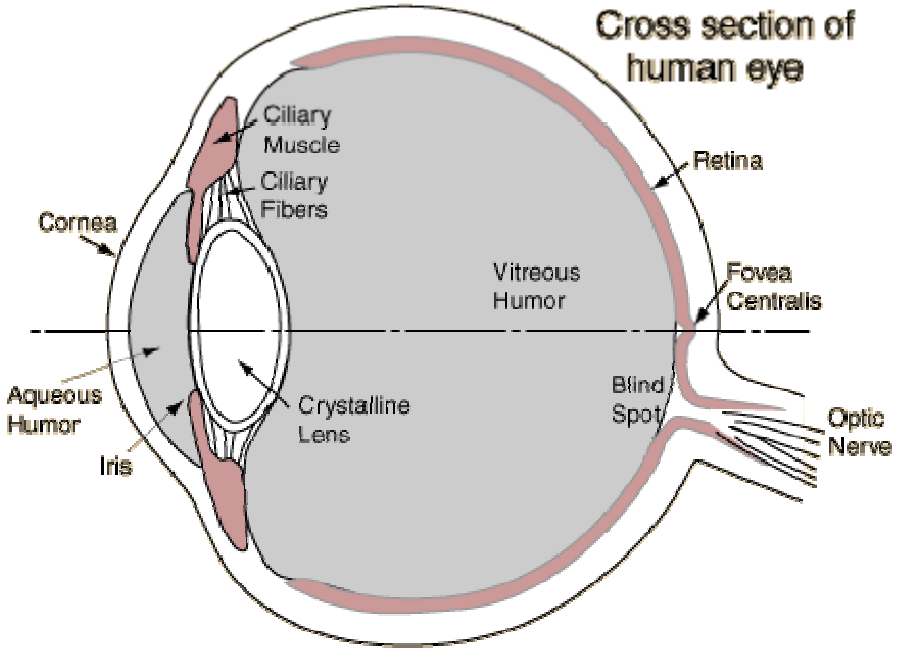


Figure 2.1 Structure of the human eye.

The RGB color space does not closely match the human visual perception. The YCbCr color space (also known as YUV), where Y gives the average brightness of the picture and Cb and Cr give the chrominance components, matches the visual perception better.

2.1.4 SENSITIVITY TO TEMPORAL FREQUENCIES

The eye retains the sensation of a displayed picture for a brief period time after the picture has been removed. This property is called persistence of vision. Human visual persistence is about 1/16 of a second under normal lighting conditions and decreases as brightness increases. Persistence property can be exploited to display video sequence as a set of pictures displayed at a constant rate greater than the persistence of vision. For example, movies are shown at 24 frames per second and require very low brightness inside the movie theatre.

2.1.5 SENSITIVITY TO SPATIAL FREQUENCIES

Spatial frequencies refer to the changes in levels in a picture. The sensitivity of the eye drops as spatial frequencies increase, i.e. as the spatial frequencies increase, the ability of the eye to discriminate between the changing levels decreases. The eye can resolve color and detail only to a certain extent. Any detail that cannot be resolved is averaged. This property of the eye is called spatial integration.

2.1.6 EXPLOITING TEMPORAL REDUNDANCIES

Since a video is essentially a sequence of pictures sampled at a discrete frame rate, there is a lot of similarity between two successive frames in a video sequence. Figure 2 shows two successive pictures in a video. The extent of similarity between two successive frames depends on the frame interval and the relative motion of the objects in the scene. Exploiting the temporal redundancies accounts for majority of the compression gains in video encoding.

Since the two frames are very similar, taking the difference of the two frames results in very less data to be encoded. This technique in which the difference between the two frames is taken is known as predictive coding. This technique fails when the object motion in a video sequence increases resulting in a loss of correlation between collocated pixels in two successive frames. In video, object motion is a common thing and in order to maintain the correlation between pixels motion compensation is used. Here for video compression purpose, a simple translational motion is assumed.

2.1.7 BLOCK BASED MOTION ESTIMATION

The amount of changes between two consecutive frames of the same video sequence is very small. Assuming a translational motion, the NxN regions can be better predicted from a previous frame by displacing the NxN region in the previous image by an amount representing the object motion. The process of finding a predicted block that minimize the difference between the original and the predicted block is called motion estimation and the relative displacement is called a motion vector. Video frames are typically coded one block at a time to take the advantage of the motion compensation applied to small NxN blocks. There is a sort of trade off between the size of the block and the accuracy.

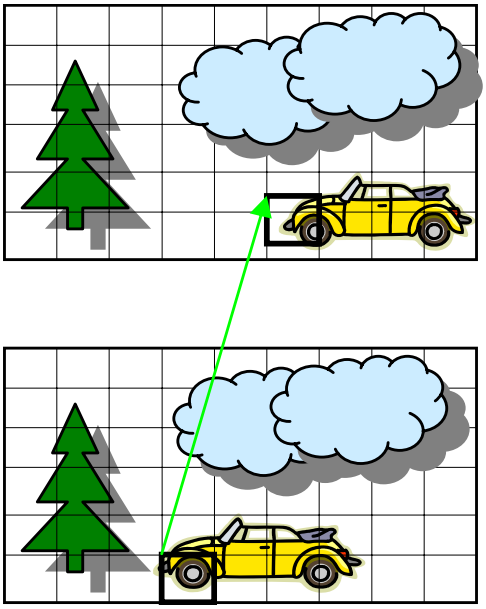


Figure 2.2 Successive frames in a video.

As the size of the block decreases the accuracy of finding a better prediction improves but as the size of the block decreases the total number of blocks in the image increases and thus the motion vector information increases. The best prediction for a given block can be found if the motion of the block relative to a reference picture can be determined. Since translational motion is assumed, the estimated motion is given in terms of the relative displacement of a block in the X and Y planes. The process of forming a prediction thus requires estimating the relative motion of a given NxN block. A simple approach to estimating the motion is to consider all possible displacements in a reference picture and determine which of these displacements gives the best prediction. The best prediction will be very similar to the original block and is usually determined using a metric such as the minimum sum of absolute differences (SAD) of pixels or the minimum sum of squared differences (SSD) of pixels. The SAD has lower computational complexity compared to SSD. The number of possible of a given block is a function of the maximum displacement allowed for motion estimation. If D_{max} is the maximum allowed displacement in the X and Y directions, the number of candidate displacements are $(1+2D_{max})^2$ .An NxN block requires N^2 computations for each candidate motion vector, resulting in an extremely high complexity of motion estimation.

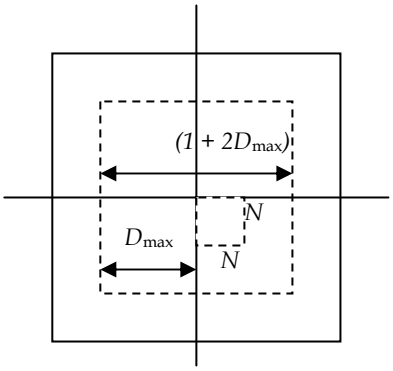


Figure 2.3 Motion estimation.

2.1.8 EXPLOITING SPATIAL REDUNDANCIES

In natural images, there exists a significant correlation between neighboring pixels. Small areas within a picture are usually similar. Redundancies exist even after motion compensation. Exploiting these redundancies will reduce the amount of information to be coded. Transform techniques are used to reduce the spatial redundancies substantially. Discrete Cosine Transform (DCT) is one type of such transforms that is used to exploit the spatial redundancies. It basically transforms an NxN picture block into a NxN block of coefficients into another domain called the frequency domain. The transform coefficients in the frequency domain can be roughly classified into low, medium and high spatial frequencies. Since the human visual system is not sensitive to high spatial frequencies, they can be discarded without affecting the perceptual quality of the image. But the video quality decreases. One way to meet this drawback is to quantize the coefficients. The quantization process reduces the number of levels while still retaining the video quality. As the quantization step size increases, the compression increases and the video quality decreases.

2.1.9 EXPLOITING STATISTICAL REDUNDANCIES

The transform coefficients, motion vectors, and other data have to be encoded using binary codes in the last stage of video compression. The simplest way to code these values is by using fixed length codes. However, these values do not have a uniform distribution and using fixed length codes is wasteful. Average code length can be reduced by assigning shorter code words to values with higher probability. Variable length coding is used to exploit these statistical redundancies and increase compression efficiency.

2.2 MPEG

2.2.1 HISTORY

The "Moving Picture Experts Group" (MPEG) was established in 1988, within the International Organization for Standardization (ISO) Steering Group (SG) 29, which was responsible for the encoding of moving pictures and audio. The MPEG commenced the development of the MPEG-1 standard in 1988, which it released in 1993 and embarked on the standardization of the MPEG-2 scheme in 1990. The MPEG-1 standard was mainly targeted at CD-ROM applications dedicated to recording video at bitrates of up to 1.5 Mbit/s. In contrast, the MPEG-2 standard was designed for substantially higher quality, namely for audiovisual applications such as today's home entertainment systems and digital broadcasting systems requiring video bitrates between 2 and 30 Mbit/s. The MPEG-4 standardization process was initiated in 1994 with the mandate of standardizing algorithms for audiovisual coding in multimedia applications, while allowing for interactivity, and supporting high compression as well as universal accessibility and portability of both the audio and video content [13].

The MPEG-4 Visual standard was developed by the ISO/IEC 14496-2,1 and its Version 1 was released in 1998; additional tools and profiles were added in two amendments of the standard, culminating in Version 2 during late 2001. The operating bitrates targeted by the MPEG-4 video standard are between 5 and 64 kbit/s in the context of mobile or Public Switched Telephone Network (PSTN)-based video applications, spanning up to 4 Mbit/s for digital TV broadcast applications, and even to rates in excess of 100 Mbit/s in High Definition TV (HDTV) studio applications. The MPEG-4 video coding standard is capable of supporting all functionalities already provided by MPEG-1 and MPEG-2.

2.2.2 MPEG-4

MPEG-4 is currently divided into 16 parts which are either published or under development. Parts 1-8 and 10 are published International Standards or Technical Reports whereas the remaining Parts are not yet published in final form. The 16 parts are:

- Part 1, Systems: Scene description, multiplexing of audio, video and related information, synchronization, buffer management, intellectual property management.
- Part 2, Visual: Coding of 'natural' and 'synthetic' visual objects.

- Part 3, Audio: Coding of natural and synthetic audio object.
- Part 4, Conformance Testing: Conformance conditions, test procedures, test bitstreams.
- Part 5, Reference Software: Publicly –available software that implements most tools in the Standard.
- Part 6, Delivery Multimedia Integration Framework: A session protocol for multimedia streaming.
- Part 7, Optimized Visual Reference Software: Optimized software implementation of selected video coding tools.
- Part 8, Carriage of MPEG-4 over IP: Specifies the mechanism for carrying MPEG-4 coded over the IP networks.
- Part 9, Reference Hardware Description: VHDL descriptions of MPEG-4 coding tools.
- Part 10, Advanced Video Coding: Efficient coding of natural video.
- Part 11, Scene Description and Application Engine.
- Part 12, ISO Based Media File Format.
- Part 13, Intellectual Property Management and Protection Extensions.
- Part 14, MPEG-4 file format.
- Part 15, AVC file format.
- Part 16, Animation framework extension.

2.2.3 BASIC MPEG 4 CONCEPTS

MPEG-4 Visual provides its various coding functions through a combination of tools, objects and profiles. A tool is a subset of coding functions to support a specific feature. An object is a video element that is coded using one or more tools. A profile is a set of object types that codec is capable of handling. All manufacturers when make the codecs, do not follow any particular standard. Hence there is no interoperability between them. And to include all the coding functionalities within a single codec is also difficult. Profiles are an important mechanism for encouraging interoperability between CODECs from different manufacturers. Each manufacturer implements the CODEC based on a specific

profile that consists of adequate tools for the target application. To date, some profiles have had more of an impact on the marketplace than others. The Simple and Advanced Simple profiles are particularly popular with the manufacturers. Table 1 lists the MPEG-4 visual profiles invoked for coding video scenes. These profiles range from the so-called simple profile derived for the encoding of rectangular video frames through profiles designed for arbitrary-shaped and scalable object coding to profiles contrived for the encoding of studio-quality video.

One of the major contributions of MPEG-4 Visual is the treatment of video sequences as a collection of one or more video objects. MPEG-4 Visual defines a video object as a flexible entity that a user is allowed to access and manipulate. A video object (VO) is a part of the video scene that may occupy an arbitrarily shaped region and may exist for an arbitrary length of time. An instance of a VO at a particular point in time is a video object plane (VOP). The introduction of the VO concept makes video coding more flexible. A video scene is a combination of a background object and a number of separate foreground objects. In comparison to the rectangular frame structures of the earlier standards, this approach is potentially much more flexible fixed. With respect to their importance in the final scene, all the separate objects may be coded with different visual qualities and temporal resolutions.

Table 2.1 MPEG-4 Visual Profiles for Coding Natural Video[1]

MPEG-4 Visual profile	Main features
☐ Simple	Low-complexity coding of rectangular video frames
☐ Advanced Simple	Coding rectangular frames with improved efficiency and support for interlaced video
☐ Advanced Real-Time Simple	Coding rectangular frames for real-time streaming
☐ Core	Basic coding of arbitrary-shaped video objects
☐ Main	Feature-rich coding of video objects
☐ Advanced Coding Efficiency	Highly efficient coding of video objects
☐ N-Bit	Coding of video objects with sample resolutions other than 8 bits
☐ Simple Scalable	Scalable coding of rectangular video frames
☐ Fine Granular Scalability	Advanced scalable coding of rectangular frames
☐ Core Scalable	Scalable coding of video objects
☐ Scalable Texture	Scalable still texture with improved efficiency and object-based features
☐ Advanced Scalable Texture	Scalable still texture with improved efficiency and object-based features
☐ Advanced Core	Combines the features of Simple, Core and Advanced Scalable Texture profiles
☐ Simple Studio	Object-based coding of high-quality video sequences
☐ Core Studio	Object-based coding of high-quality video with improved compression efficiency

2.2.4 MPEG 4 VISUAL

MPEG-4 Visual (Part 2 of ISO/IEC 14496, 'Coding of Audio-Visual Objects') is a large document that covers a very wide range of functionalities, all related to the coding and representation of visual representation. The standard deals with the following types of data:

- Moving video (rectangular frames);
- Video objects (arbitrary-shaped regions of moving video);
- 2D and 3D mesh objects (representing deformable objects);
- Animated human faces and bodies;
- Static texture (still images);

The standard specifies a set of coding 'tools' that are designed to represent these data types in compressed (coded) form. With a diverse set of tools and supported data types, the MPEG-4 Visual standard can support many different applications, including the following:

- 'legacy' video applications such as digital TV broadcasting, video-conferencing etc.
- 'object based' video applications in which a video scene may be composed of a number of distinct video objects, each independently coded.
- Rendered computer graphics using 2D and 3D deformable mesh geometry and/or animated human faces and bodies.
- 'hybrid' video applications combining real-world video, still images and computer generated graphics.
- Streaming video over the internet and mobile channels.
- High quality video editing and distribution for the studio production environment.

Despite the wide range of coding tools specified by the standard, at the heart of MPEG- 4 Visual is a rather simple video coding mechanism, a block based video CODEC that uses motion compensation followed by DCT, quantization and entropy coding.

2.2.4.1 FEATURES

Some of the key features that distinguish MPEG-4 Visual from visual coding standards include[1]:

- Efficient compression of progressive and interlaced video sequences. The core compression tools are based on the ITU-T H.263 standard and can out-perform MPEG-1 and MPEG-2 video compression.
- Coding the video objects (foreground and background objects) independently.
- Support of effective transmission over practical networks.
- Coding of still 'texture'.
- Coding of animated visual objects such as 2D and 3D polygonal meshes, animated faces and animated human bodies.
- Coding for specialist applications such as 'studio' quality video.

2.2.5 SIMPLE PROFILE

A CODEC that is compatible with the Simple Profile should be capable of encoding and decoding Simple Video Objects using the following tools:

- I-VOP(Intra-coded rectangular VOP, progressive video format);
- P-VOP(Inter-coded rectangular VOP; progressive video format);
- Short header (mode for compatibility with H.263 CODEC).
- Compression efficiency tools.
- Transmission efficiency tools.

2.2.5.1 BASIC CODING TOOLS

I-VOP:

A rectangular I-VOP is a frame of video encoded in Intra mode. The encoding and decoding stages are shown in figure 2.4.

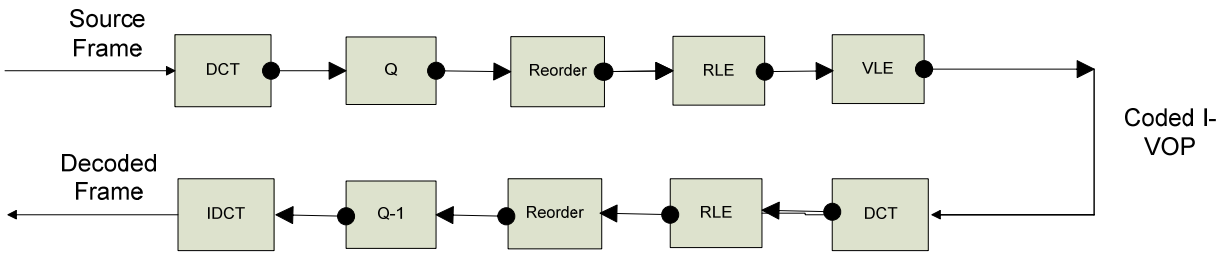


Figure 2.4 I-VOP encoding and decoding stages. [1]

Blocks of luma and chroma samples are transformed using an 8x8 Forward DCT during encoding and an 8x8 inverse DCT during decoding. The MPEG-4 Visual standard specifies the method of rescaling quantized transform in a decoder. Rescaling is controlled by a quantiser scale parameter QP, which can take values from 1 to 31. Quantized DCT coefficients are reordered in a zigzag scan prior to encoding. The array of reordered coefficients corresponding to each block is encoded to represent the zero coefficients efficiently. Each non zero coefficient is encoded as a triplet of (last, run, level), where last indicates whether this is the last non zero coefficient in the block, run corresponds to the number of zeroes proceeding the non zero coefficient and level indicates the coefficient sign and magnitude. Header information and the triplets are represented by variable-length codes (VLCs). These codes are similar to Huffman codes and are defined in the standard, based on pre-calculated coefficient probabilities. A coded I-VOP consists of a VOP header, optional video packet headers and codec macroblocks. Each macroblock is coded with a header followed by coded coefficients for each 8x8 block.

P-VOP:

A P-VOP is coded with Inter prediction from a previously encoded I- or P-VOP. The encoding and decoding stages are shown in figure 2.5. The basic motion compensation scheme is block based compensation of 16x16 pixel macroblocks. The offset between the current macroblock and the compensation region in the reference picture may have half pixel resolution. Predicted samples at sub pixel positions are calculated using bilinear interpolation between samples at integer-pixel positions. The method of motion estimation is left to the designer's discretion. The matching region is subtracted from the current macroblock to produce a residual macroblock (Motion Compensated Prediction, MCP).After motion compensation, the residual data is transformed with the DCT, quantized, reordered, run level coded and entropy coded. The quantized residual is rescaled

and inverse transformed in the encoder in order to reconstruct a local copy of the decoded MB. A coded P-VOP consists of VOPO header, optional video packet headers and coded macroblocks each containing a header and coded residual coefficients for every 8x8 block.

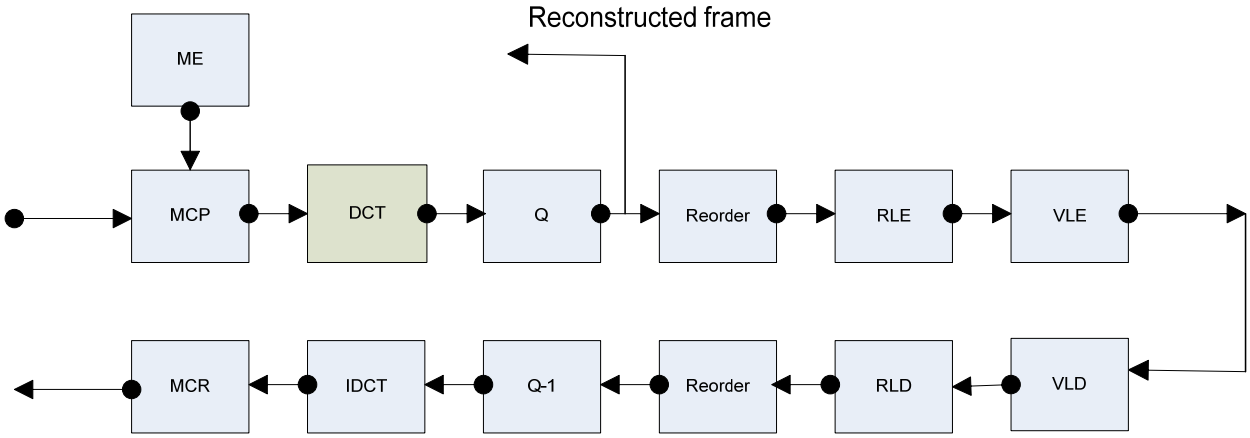


Figure 2.5 P-VOP encoding and decoding stages. [1]

Macroblocks within a P-VOP may be coded in Inter or Intra mode. Inter mode will normally give the best coding efficiency but Intra mode may be useful in regions where there is not a good match in a previous VOP, such as a newly-uncovered region.

Short header:

The short header tool provides compatibility between MPEG-4 Visual and the ITU-T H.263 video coding standard. An I- or P- VOP encoded in a 'short header' mode has identical syntax to an I-picture or a P-picture coded in the baseline mode of H.263. This means that an MPEG-4 I-VOP or P-VOP should be decodable using an H.263 decoder and vice versa. In short header the macroblocks within a VOP are organized in Groups Of Blocks (GOB), each consisting of one or more complete rows of macroblocks. Each GOB may start with a resynchronization marker.

1.2.5.2 CODING EFFICIENCY TOOLS:

Four Motion Vectors per Macroblock:

Motion compensation tends to be more effective with smaller block sizes. The default block size for motion compensation is 16x16 samples (luma) and 8x8 samples (chroma), resulting in one motion vector per macroblock. This tool gives the encoder the option to choose a smaller motion compensation block

size, 8x8 samples (luma) and 4x4 samples (chroma), giving four motion vectors per macroblock. This mode can be more effective in minimizing the energy in the motion compensated residual, particularly in areas of complex motion or near the boundaries of moving objects. There is an increased overhead in sending four motion vectors instead of one, and so the encoder may choose to send one or four motion vectors on a macroblock by macroblock basis.

Unrestricted Motion Vectors:

In some cases, the best match for a macroblock may be a 16x16 region that extends outside the boundaries of the reference VOP. The Unrestricted Motion Vectors (UMV) tool allows motion vectors to point outside the boundary of the reference VOP. If a sample indicated by the motion vector is outside the reference VOP, the nearest edge sample is used instead. UMV mode can improve motion compensation efficiency, especially when there are objects moving in and out of the picture.

Intra Prediction:

Low frequency transform coefficients of neighbouring intra-coded 8x8 blocks are often correlated. In this mode, the DC coefficient and first row and column of AC coefficients in an Intra-coded 8x8 block are predicted from neighbouring coded blocks.

2.2.5.3 TRANSMISSION EFFICIENCY TOOLS:

Video Packet:

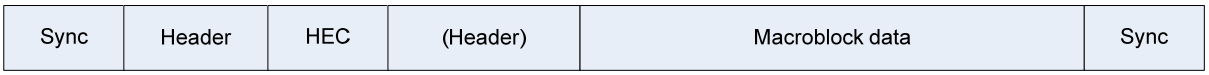


Figure 2.6 video packet structure. [1]

A transmitted VOP consists of one or more video packets. A video packet is analogous to a slice in MPEG-1, MPEG-2 or H.264 and consists of a resynchronization marker; a header field and a series of coded macroblocks in raster scan order (figure 2.6). The resynchronization marker is followed by a count of the next macroblock number, which enables a decoder to position the first macroblock of the packet correctly. After this comes the quantization parameter and a flag, HEC (Header Extension Code).if HEC is set to 1 , it is followed by the duplication of the current VOP header, increasing the amount of

information that has to be transmitted but enabling a decoder to recover the VOP header if the first VOP header is corrupted by an error.

Data Partitioning:

The data partitioning tool enables an encoder to reorganize the coded data within a video packet to reduce the impact of transmission errors. The packet is split into two partitions, the first containing coding mode information for each macroblock together with DC coefficients of each block. The remaining data are placed in the second partition following a resynchronization marker. The information is sent in the first partition is considered to be the most important for adequate decoding of the video packet. If the first partition is recovered, it is usually possible for the decoder to make a reasonable attempt at reconstructing the packet, even if the 2^{nd} partition is damaged or lost due to transmission error(s).

Reversible VLCs:

An optional set of Reversible Variable Length Codes (RVLCs) may be used to encode DCT coefficient data. As the name suggests, these codes can be correctly decoded in both the forward and reverse directions, making it possible for the decoder to minimize the picture area affected by the error. A decoder first decodes each video packet in the forward direction and, if an error is detected, the video packet is decoded in the reverse direction from the next resynchronization marker. Using this approach, the damage caused by an error may be limited to just one macroblock, making it easy to conceal the errored region.

2.2.6 ADVANCED SIMPLE PROFILE

The Advanced simple profile was incorporated into a later version of the standard with added tools to support improved compression efficiency and interlaced video coding. An Advanced Simple Profile CODEC must be capable of decoding Simple objects as well as Advanced Simple objects which may use the following tools in addition to the Simple profile tools:

- B- VOP (Bi-directionally predicted Inter-coded VOP).
- Quarter-pixel motion compensation.
- Global motion compensation.

- Alternate quantizer.
- Interlace.

B-VOP:

The B-VOP uses bidirectional prediction to improve motion compensation efficiency. Each block or macroblock may be predicted using (a) forward prediction from the previous I- or P- VOP, (b) backward prediction from the next I- or P- VOP or (c) an average of the forward and backward predictions. This mode generally gives better coding efficiency than the basic forward prediction; however the encoder must store multiple frames prior to coding each B-VOP which increases the memory requirements and the encoding delay.

Quarter-Pixel Motion Vectors:

The Simple Profile supports motion vectors with half pixel accuracy and this tool supports vectors with quarter-pixel accuracy. The reference VOP samples are interpolated to half pixel positions and then again to quarter pixel positions prior to motion estimation and compensation. This increases the complexity of motion estimation, compensation and reconstruction but can provide a gain in coding efficiency compared with half pixel compensation.

Global Motion Compensation:

Macroblocks within the same video object may experience similar motion. For example, camera pan will produce apparent linear movement of the entire scene, camera zoom or rotation will produce a more complex apparent motion and macroblocks within a large object may all move in the same direction. Global Motion Compensation (GMC) enables an encoder to transmit a small number of motion (warping) parameters that describe a default 'global' motion for the entire VOP. GMC can provide improved compression efficiency when a significant number of macroblocks in the VOP share the same motion characteristics. The global motion parameters are encoded in the VOP header and the encoder chooses either the default GMC parameters or an individual motion vector for each macroblock. When the GMC tool is used, the encoder sends upto four global motion vectors (GMV) for each VOP together with the location of each GMV in the VOP. For each pixel position in the VOP, an individual motion vector is calculated by interpolating between the GMVs and the pixel position is motion compensated.

Interlace:

Interlaced video consists of two fields per frame sampled at different times. An interlaced VOP contains alternate lines of samples from two fields. Because the fields are sampled at different times, horizontal movement may reduce correlation between the lines of samples. The encoder may choose to encode the macroblock in Frame DCT mode, in which each block is transformed as usual, or in Field DCT mode, in which the luminance samples from Field 1 are placed in the top eight lines of the macroblock and the samples from the Field 2 in the lower eight lines of the macroblock before calculating the DCT. Field DCT mode gives a better performance when the two fields are decorrelated.

2.3 Basic Encoder / Decoder:

A typical image/video transmission system is outlined in Figure 2.7. The objective of the source encoder is to exploit the redundancies in image data to provide compression. In other words, the source encoder reduces the entropy, which in our case means decrease in the average number of bits required to represent the image.

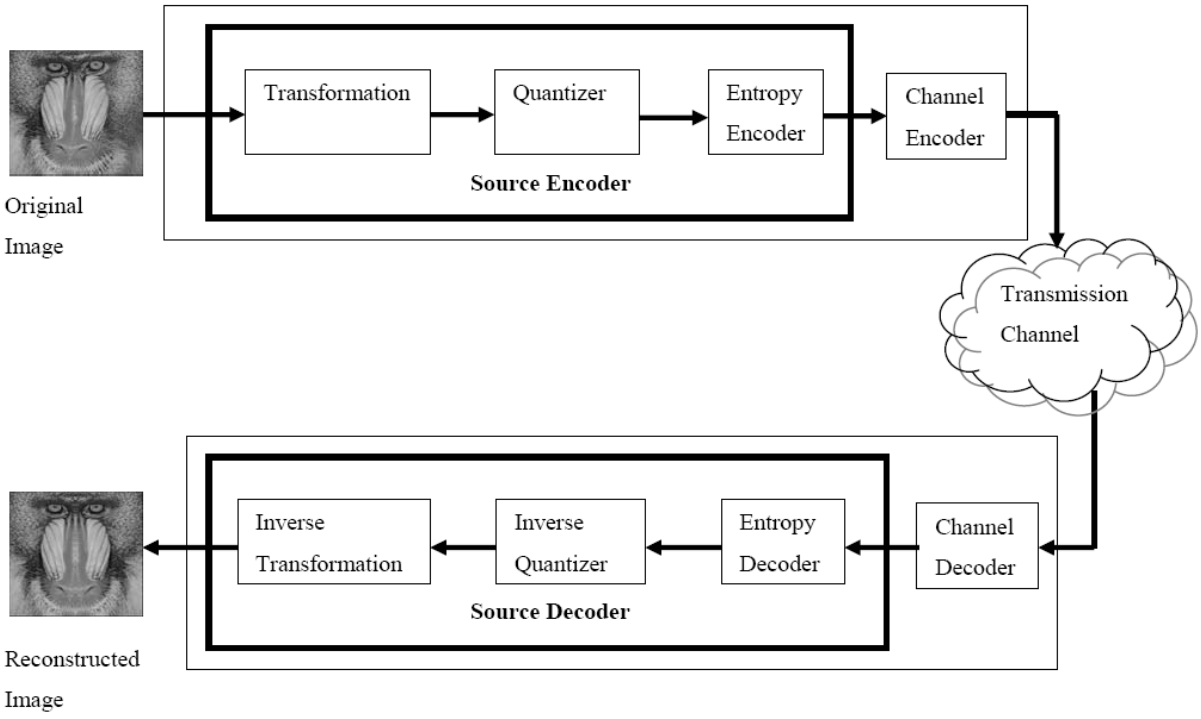


Figure 2.7 Components of a basic Encoder/Decoder

Each sub-block in the source encoder exploits some redundancy in the image data in order to achieve better compression. The transformation sub-block decorrelates the image data thereby reducing (and in some cases eliminating) interpixel redundancy. The quantizer sub-block utilizes the fact that the human

eye is unable to perceive some visual information in an image. Such information is deemed redundant and can be discarded without introducing noticeable visual artifacts. Such redundancy is referred to as psychovisual redundancy. This idea can be extended to low bitrate receivers which, due to their stringent bandwidth requirements, might sacrifice visual quality in order to achieve bandwidth efficiency. This concept is the basis for rate distortion theory, that is, receivers might tolerate some visual distortion in exchange for bandwidth conservation.

Lastly, the entropy encoder employs its knowledge of the transformation and quantization processes to reduce the number of bits required to represent each symbol at the quantizer output.

2.4 Hybrid Video Codec Model

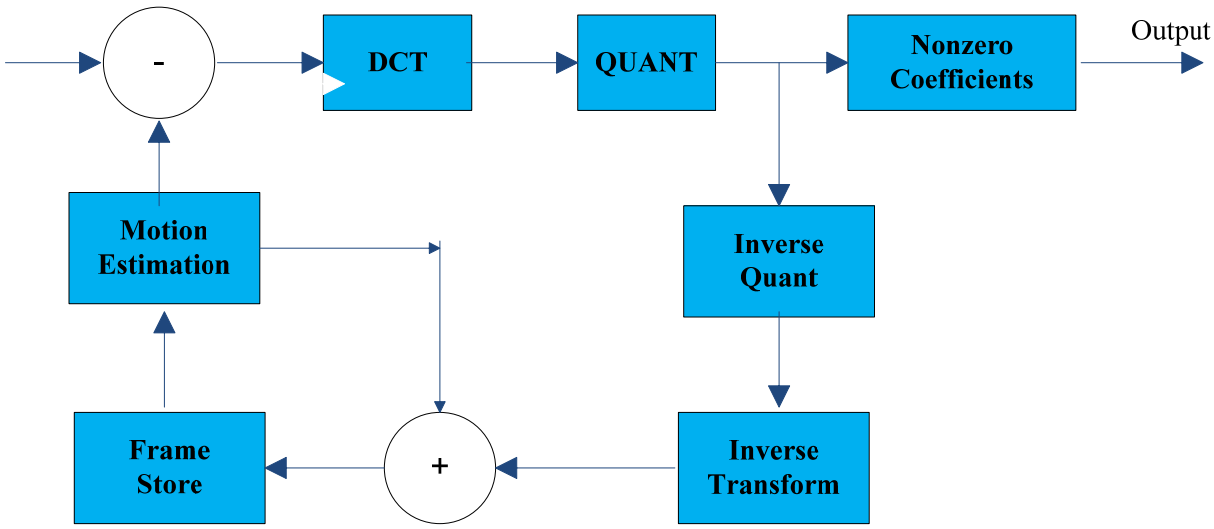


Figure 2.8 Hybrid Video Codec model [1]

The major video codec standards released since the early 1990s have been based on the same generic design (or model) of a video CODEC that incorporates a motion estimation and compensation front end, a transform stage and an entropy encoder. Any codec that is compatible with H.261, H.263, MPEG-1, MPEG-2, MPEG-4 Visual and H.264 has to implement a similar set of basic coding and decoding functions.

Figure 2.8 shows a generic hybrid encoder. In the encoder, the video frame is fed as input and is processed to produce a coded (compressed) bitstream. An

input video stream is presented for encoding in units if macroblocks. The input frame is compared to a reference frame, for example a previously coded frame. A motion estimation function finds a 16x16 region in the reference frame that matches the current macroblock in the current frame. The offset between the current macroblock position and the chosen reference region is a motion vector MV. Based on the chosen motion vector MV, a motion compensated prediction is generated. The motion compensated prediction is subtracted from the current macroblock to produce a residual or difference macroblock. The difference macroblock is transformed using DCT. Typically the residual block is split into 8x8 or 4x4 sub-blocks and each sub block is transformed separately. Each sub block is then quantized. The DCT coefficients of each sub block are then reordered and then run level encoded. Finally the coefficients, motion vector and associated header information for each macro block are entropy encoded to produce the compression bit stream.

3. PROJECT DESCRIPTION

3.1 SORTING PERMUTATION OF A BLOCK

This section gives an overview of the permutations used for determining the dynamic pre-transform scans [8] in the proposed approach. Given an NxN block I, the sorting permutation of the block, P_I, is a permutation that perfectly sorts the elements in the block. An identity permutation, when applied, keeps the block unchanged; i.e., $P_I(I) = I$. When a block is sorted, the resulting data has high spatial correlation and the application of DCT to such blocks results in fewer non-zero coefficients. The computation and application of this sorting permutation is the key to the proposed method. Figure 3.1 shows an example of sorting permutation.

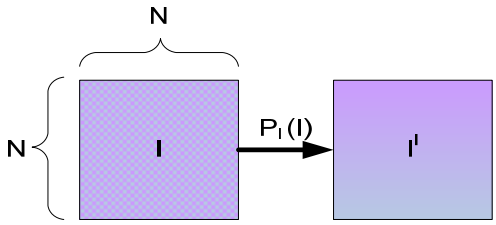


Figure 3.1 Sorting permutation

A sorting permutation is obtained by sorting a block and recording the position of elements in the sorted block. Figure 3.2 shows an example of generating a sorting permutation. Figure 3.2a shows the original block with 6 elements and the corresponding positions. Figure 3.2b shows the block with sorted data and the corresponding position in the original block. The sorting permutation shown in Figure 3.2c is essentially the original index of the data elements in the sorted order. When a sorting permutation is applied to the original data, i.e, element 2 is moved to the first position, element 6 is moved to the second position, element 4 to the third positions and so on, the original data will be perfectly sorted.

If such a sorting permutation is applied to block data in video or image coding, the block data can be highly compressed. To decompress and reconstruct the block at the decoder, the sorting permutation is necessary at the decoder. However, the cost of transmitting permutations is prohibitively expensive; for a 16x16 block, 256 positions representing a permutation must be transmitted in addition to the sorted data. The proposed method hence uses the permutations of previously coded blocks to sort the current block. Since permutations of the

previously coded blocks can be computed from the block data, permutations do not have to be explicitly transmitted to the decoder. If the neighboring blocks are similar or have similar motion, the permutation of the neighboring blocks, when applied to the current block, can result in increased spatial correlation and hence reduce the number of non-zero coefficients after DCT. A similar approach was introduced in [8].

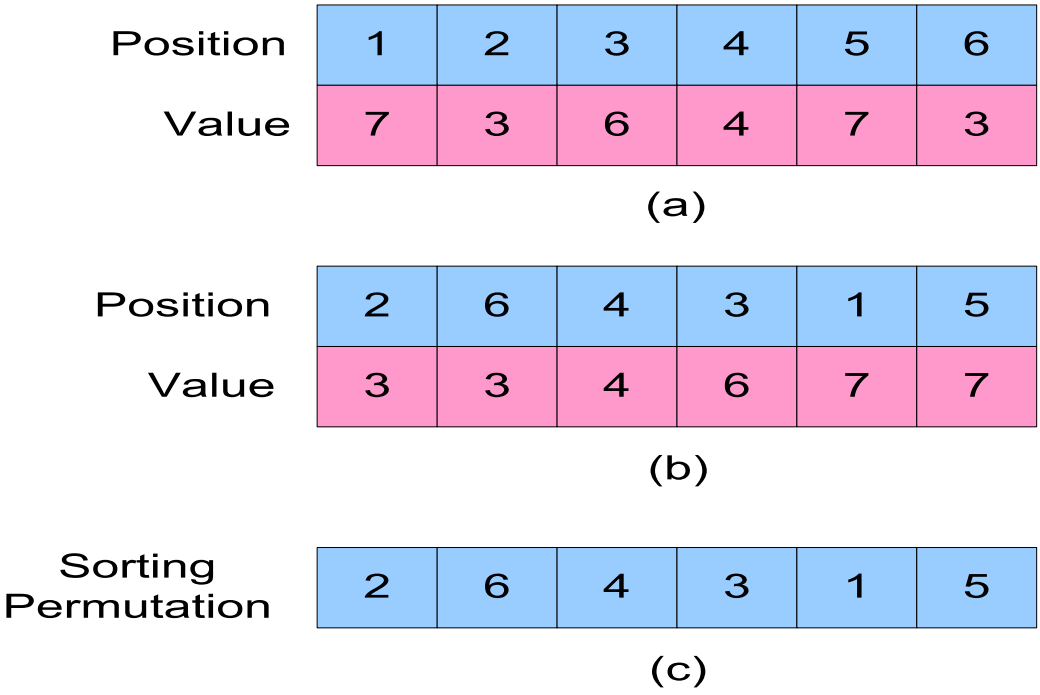


Figure 3.2 Generating sorting permutations

3.2 DYNAMIC PRE-TRANSFORM BLOCK SCANS

Figure 3.3a shows a frame of video with width W and height H. The picture is coded as non-overlapping blocks of size NxN. Blocks are typically coded in raster scan order; left to right and top to bottom. When block X is being encoded, all blocks to the left and top of the current block would have been already coded. The sorting permutation of one of these previously coded blocks can be determined at the decoder without explicitly encoding/transmitting the permutation at the encoder. The key to the proposed method is the use of the sorting permutation of a previously coded block to reorder the elements of the current block before the transform stage. When neighboring blocks have similar motion characteristics or similar data distribution, applying the permutation of a previously coded block can increase the spatial correlation of the current block and hence increase its compressibility.

Figure 3.b shows the current block X and four neighboring blocks, A, B, C, and D, with sorting permutations P_A, P_B, P_C, and P_D respectively. When the current block X is coded, the effect of neighboring permutations is evaluated on X; permutations of the neighboring blocks are applied one at a time and DCT is applied to the resulting scanned block and the permutation that results in the least number of non-zero coefficients after transform is used for encoding the block.

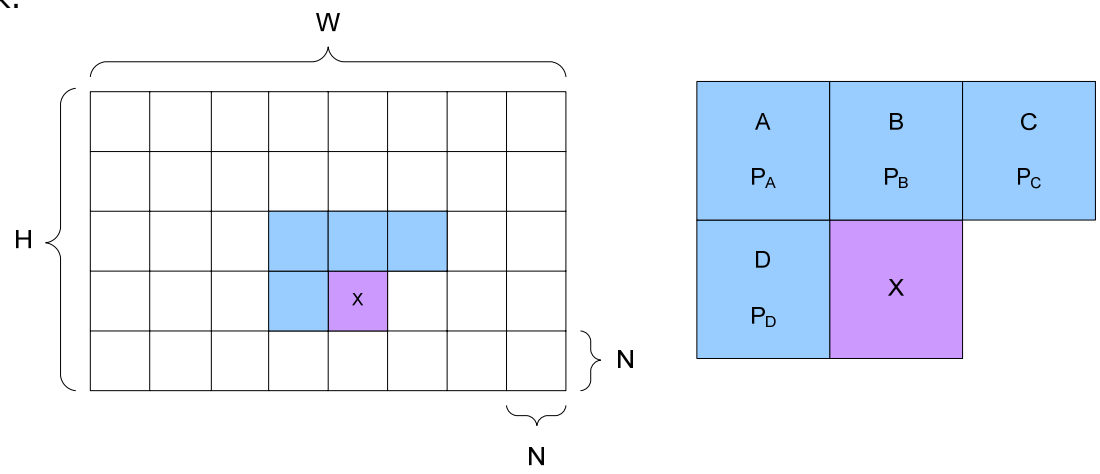


Figure 3.3 (a) Blocks in a video frame (b) previously coded blocks used for determining the scan patterns for the current block X

A block will not use any neighboring permutations if the use of a permutation results in more nonzero transform coefficients than when X is coded normally without using coefficient scanning (i.e. X uses its identity permutation). The index of the block used for permutation or the absence of the use of permutation is encoded as a part of the block.

Figure 3.4 shows the process of the evaluating the impact of applying the permutation of a neighboring block I on the current block X; the permutation of block I, P_I, is applied to block X to get permuted block X'. DCT is applied to the permuted block and the number of nonzero coefficients after the DCT is counted. When a block X is encoded, the permutations of all the neighboring blocks including the current X are evaluated and the permutation that minimizes the number of nonzero coefficients is used for encoding the block. The process essentially evaluates $\underset{I}{argmin}\, f(X,I) \mid I \in \{A,B,C,D,X\}$ where *f(X,I)* gives the number of nonzero coefficients when block X is coded using permutation of block I. In the discussion above, the blocks A, B, C, D, and X represent the motion compensated residual blocks of a video frame.

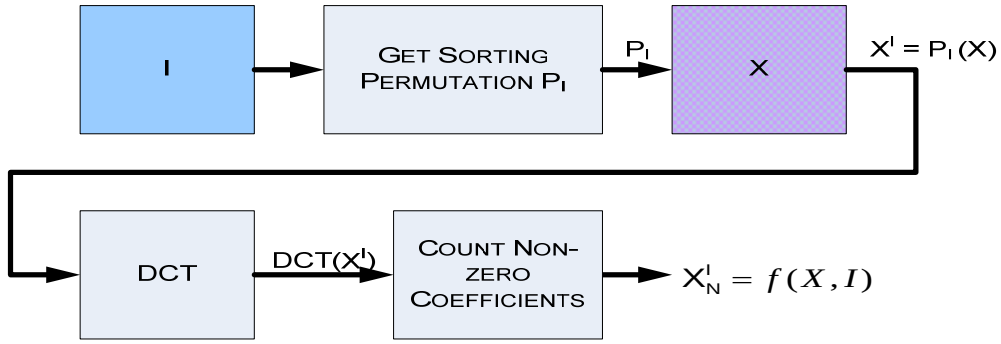


Figure 3.4 Evaluation of a neighboring block permutation for encoding

3.3 DISCRETE COSINE TRANSFORM:

3.3.1 INTRODUCTION

Transform coding constitutes an integral component of contemporary image/video processing applications. Transform coding relies on the premise that pixels in an image exhibit a certain level of correlation with their neighboring pixels. Similarly in a video transmission system, adjacent pixels in consecutive frames show very high correlation. Consequently, these correlations can be exploited to predict the value of a pixel from its respective neighbors. A transformation is, therefore, defined to map this spatial (correlated) data into transformed (uncorrelated) coefficients [21].

Clearly, the transformation should utilize the fact that the information content of an individual pixel is relatively small i.e., to a large extent visual contribution of a pixel can be predicted using its neighbors. The objective of the source encoder is to exploit the redundancies in image data to provide compression. In other words, the source encoder reduces the entropy, which in our case means decrease in the average number of bits required to represent the image. On the contrary, the channel encoder adds redundancy to the output of the source encoder in order to enhance the reliability of the transmission. In the last decade, Discrete Cosine Transform (DCT) has emerged as the de-facto image transformation in most visual systems. DCT has been widely deployed by modern video coding standards, for example, MPEG, JVT etc.

3.3.2 ONE DIMENSIONAL DCT

The most common DCT definition of a 1-D sequence of length N is [21]

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{\pi(2x+1)u}{2N}\right], \tag{1}$$

for $u = 0,1,2,\ldots,N-1$. Similarly, the inverse transformation is defined as [21]

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos\left[\frac{\pi(2x+1)u}{2N}\right], \tag{2}$$

for $x = 0,1,2,\ldots,N-1$. In both equations **(1)** and **(2)** $\alpha(u)$ is defined as

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & for \quad u = 0 \\ \sqrt{\frac{2}{N}} & for \quad u \neq 0. \end{cases} \tag{3}$$

It is clear from **(1)** that for 0 u =0,

$$C(u=0) = \sqrt{\frac{1}{N}} \sum_{x=0}^{N-1} f(x).$$

Thus, the first transform coefficient is the average value of the sample sequence. Here, this value is referred to as the DC Coefficient. All other transform coefficients are called the AC Coefficients.

3.3.3 2-DIMENSIONAL DCT

The 2-D DCT is a direct extension of the 1-D case and is given by [21]

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right], \quad (4)$$

For u, v =0,1,2,…,$N-1$ and $\alpha(u)$ and $\alpha(v)$ are defined in **(3)**. The inverse transform is defined as

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right], \quad (5)$$

3.3.4 EFFECT ON DCT USING PRE TRANSFORM SCAN

In this work DCT plays a very important role. It is the transform that is fed with the permuted data. As mentioned in the previous sections DCT is a transform that de-correlates the data that is fed into it. Suppose we have data as shown below

-21	-9	-5	-10	-5	9	-8	3
-13	0	1	-5	-3	2	-4	3
5	1	-2	4	6	9	-5	-3
7	-5	-5	-2	-3	1	3	3
-2	-3	-2	-9	-6	-6	1	1
-1	-1	-2	-2	5	3	-7	-2
-3	7	0	0	5	-9	-14	6
-1	-5	-11	-12	7	-8	-1	0

Figure 3.5 8x8 Block of Data to Fed for DCT

This data is fed into the DCT block and the data that comes after DCT is shown below. Here two cases are shown, one in which DCT is applied to a normal block and in the other DCT is applied to a permuted block.

-15	-8	0	2	3	-15	5	0
-1	-11	-6	-1	-12	3	9	-9
-11	-8	-6	-4	-2	-12	0	0
-8	-5	-3	-8	-6	-5	5	-3
-11	-11	10	4	-2	7	-1	5
5	3	-2	0	6	-2	3	-5
2	-1	-7	13	1	2	3	-1
-1	4	-1	-3	0	-3	-1	-1

(a)

-14	2	-7	-3	12	3	0	10
1	3	-3	5	-12	-3	-5	2
7	6	-3	-5	-1	3	0	-7
-3	3	1	6	2	3	0	2
-3	-2	2	8	-2	8	5	9
1	6	-4	5	2	-3	-5	-8
6	-7	1	4	-11	-1	-2	0
-8	-4	-1	-4	3	-3	12	3

(b)

Figure 3.6 DCT blocks (a) Without Permutation (b) With Permutation

Then this block is fed for quantization which quantizes the block with an appropriate quantizer that is in synchronization with the final required bitrate. Here we have used a quantization parameter of 8 for illustration purposes.

-1	-1	0	0	0	-1	0	0
0	-1	0	0	-1	0	1	-1
-1	-1	0	0	0	-1	0	0
-1	0	0	-1	0	0	0	0
-1	-1	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0

(b) NZC=16

-1	0	0	0	1	0	0	1
0	0	0	0	-1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	0	0	0	0	1	0	1
0	0	0	0	0	0	0	-1
0	0	0	0	-1	0	0	0

(a) NZC=11

Figure 3.7 Quantized blocks (a) Without Permutation (b) With Permutation

It is clearly seen that the block on the left has more non zero coefficients than the one on the right. In comparison to the perfectly sorted data that is fed for DCT and Quantization the partially sorted data performs substantially well. In our work we have considered the perfectly sorted data as the ideal case and we have tried achieving that sort of compression but then ideal case is always practically not achievable. Hence it was used as a benchmark for the whole research.

-1	-1	0	0	0	0	0	0
-5	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figure 3.8 Quantization block for sorted data

The figures explain how DCT behaves on the normal data and permuted data. Clearly it is seen that the number of zeroes increase in the permuted data case. This is because the blocks are partially sorted. As a result of which correlation improves and hence results in fewer non-zero coefficients. But we are losing some energy from the frame each time it gets permuted and transformed and then subsequently quantized. This is due to the fact that all the energy is trapped in fewer coefficients than before hence small loss of information, results in loss of data with more importance. This is one of the drawbacks of the approach that as the non-zero coefficients decrease, there is also a drop in the PSNR of the resultant video. Figure 3.9 shows the reconstructed blocks after Inverse DCT (IDCT)

-13	-5	-7	-10	-6	3	-3	6
-7	0	1	-1	-1	5	-2	3
2	1	-1	0	4	9	1	-2
0	-1	-1	-1	-2	2	2	2
-2	-2	-1	-1	-3	-5	0	2
0	0	0	0	3	-4	-3	-3
-4	1	1	0	4	-7	-3	0
-2	-2	-9	-8	5	-3	-1	2

(a)

2	-4	1	-3	-2	-2	0	0
-4	1	-1	-2	3	-2	-1	-2
-5	1	0	-1	-1	0	2	-5
6	-9	0	-2	3	-7	-1	2
1	-1	-1	-4	0	-1	-4	3
-1	-1	-4	3	2	-3	-3	-1
0	-5	3	-2	5	-7	4	-5
4	-4	-5	0	1	-3	-7	5

(b)

Figure 3.9 IDCT blocks (a) Without Permutation (b) With Permutation

4. EXPERIMENTAL SETUP AND RESULTS

4.1 EXPERIMENTAL SETUP

The proposed approach is evaluated in a generic video coding loop. Implementation of this process in MPEG-4 video encoder was done. Figure 4.1 shows the block diagram of the system used for evaluation. The system implements a basic hybrid video encoding loop with motion compensation, transform coding, and quantization. Three additional blocks are added to this basic loop to implement pre-transform scan determination that evaluates the permutations of the neighboring blocks and selects the best way to scan and minimize the number of non-zero coefficients after DCT and quantization. The inverse permutation block reconstructs the original residual block and the sorting permutation of the reconstructed residual is computed and saved in the permutation store for use in blocks of the frame yet to be encoded. As shown in figure 4.1, the new components can be added to standard video encoders with minimal changes.

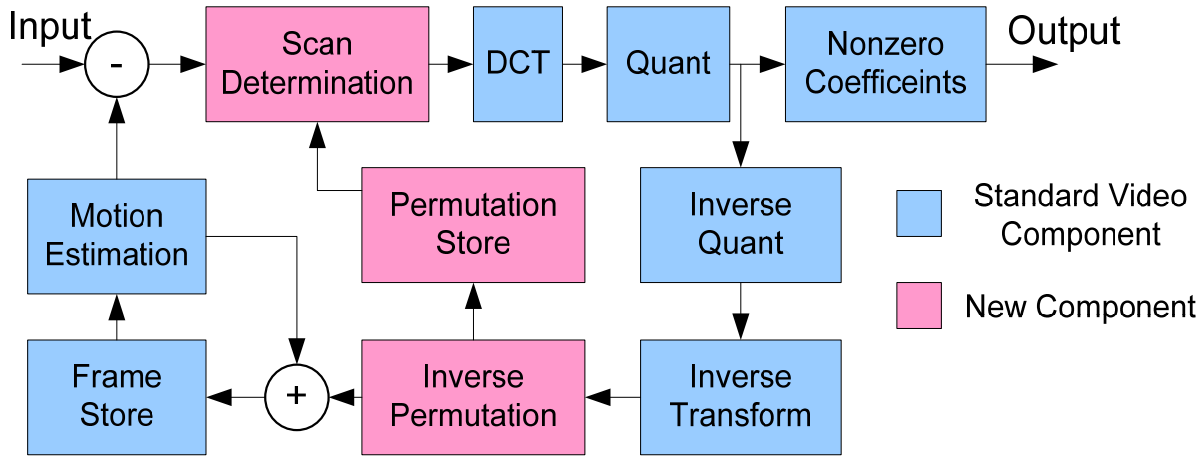


Figure 4.1 Implementation of dynamic pre-transform scanning

The various YUV sequences used for the experiments are shown below. All these sequences have various characteristics. Some of them have high motion while some of them have low motion. Specifically the sequence named Crew and Flower have very high motion and a lot of camera motion. In summary we can say that these two sequences are very complex sequences. The sequence City has a lot of complex camera movement and this makes the sequence very

complicated and makes it difficult for the encoder to code. The sequence ice is sort of a low motion

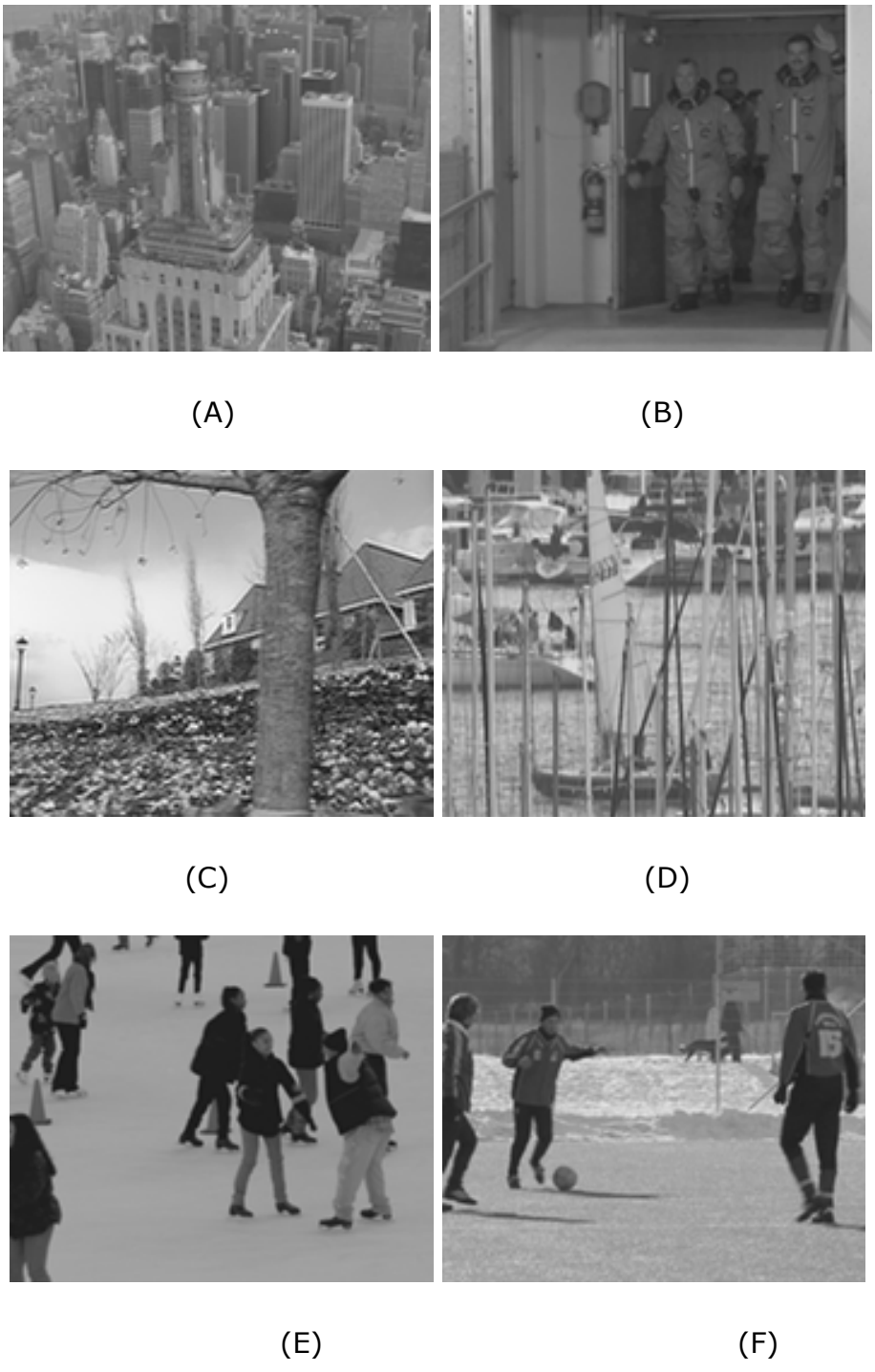


Figure 4.2 Sequences used for experiments

sequence in which there is very less camera movements and only objects keep moving in the sequence. Harbour also is a flat sequence as Ice is and it has similar characteristics.

4.2 RESULTS

A generic encoding loop shown in Figure 4.1 is implemented for this approach. The first frame is encoded without motion estimation and all blocks use transform coding and no pre-transform scanning is applied. For subsequent frames motion compensated residual is computed for 16x16 blocks and the residual coefficients are reordered using a permutation of a previously coded block that leads to minimum number of non-zero coefficients after DCT. For each 16x16 block, a permutation is applied and the four 8x8 sub-blocks blocks are extracted. DCT is applied to each of the four 8x8 blocks followed by quantization, de-quantization, inverse 8x8 DCT, and inverse permutation for the 16x16 block to get the reconstructed residual for the 16x16 block. A sorting permutation is computed for the reconstructed residual of the 16x16 block. This sorting permutation is used to encode future blocks. Here, we have specifically discussed about Crew and Ice. The rest of the results are given in the appendix section.

We started with the hypothesis that motion compensated residual blocks in videos that are close to each other have similar distribution and this distribution when exploited properly will improve the compression. In order to explore this hypothesis we first started implementing the block based motion estimation algorithm called "Full Search". As explained earlier in chapter 2 Full search is an exhaustive search method for motion estimation. Other Motion Estimation algorithms have also being developed but we for experimental purposes thought that this would be a true test for the concept. Later one can implement this approach with other motion estimation algorithms also.

4.2.1 16X16 NEIGHBOURHOOD SCAN

We implemented this approach using 16x16 based motion estimation. We implemented the proposed approach in C++ and measured the output for two metrics. One was average number of non zero coefficients (NZC) and other was Peak Signal to Noise Ratio (PSNR). We had two sets of results for this, one was taken across various search ranges keeping the quantizer constant and other was taken across various quantizers keeping the range constant. The simulations were run for all the sequences as mentioned earlier with three different resolutions, QCIF, CIF and 4CIF with resolutions 176x144, 352x288 and 704x576 respectively. In each case the number of non-zero coefficients after DCT and quantization is measured. We arrived at interesting conclusions from them.

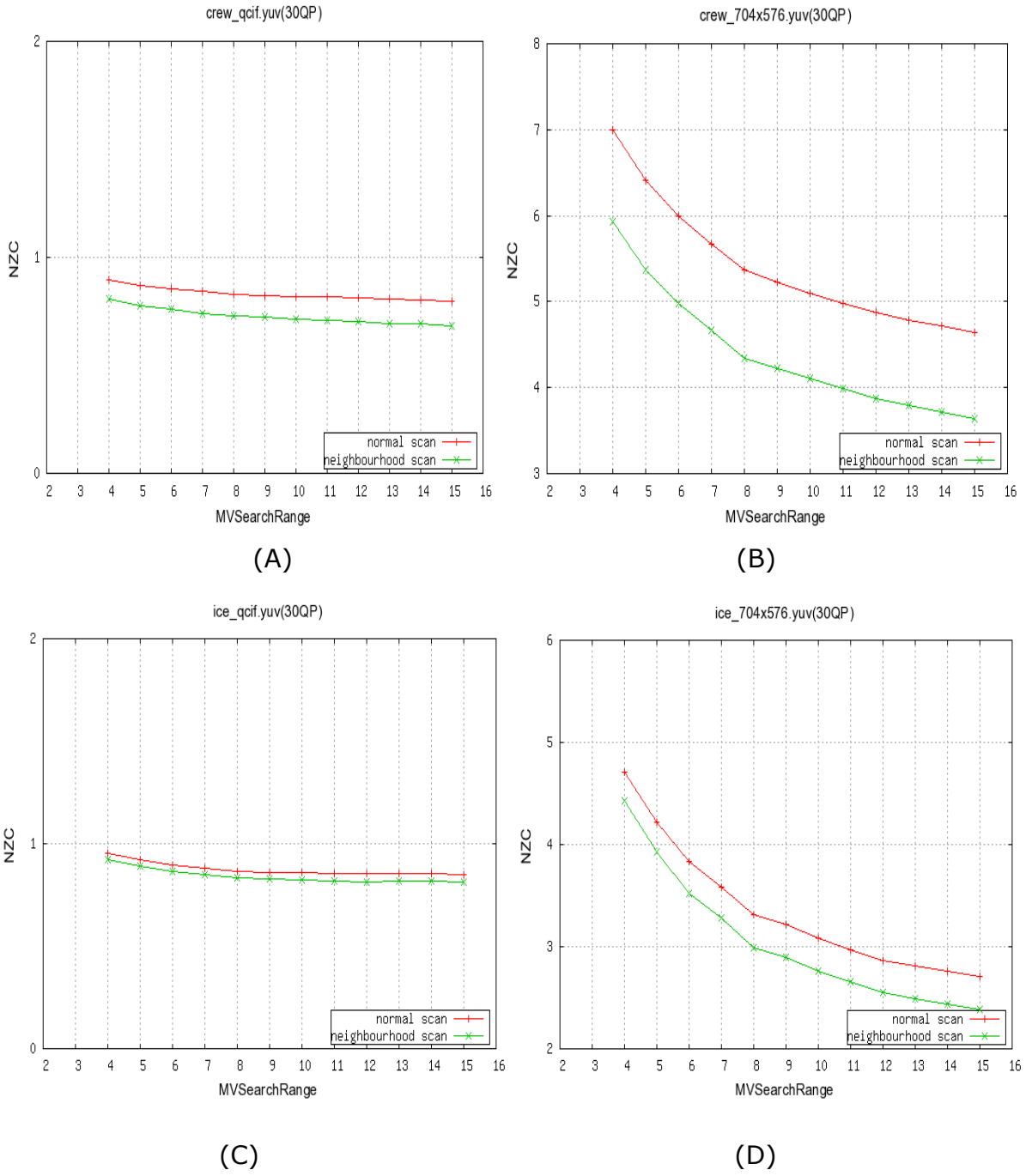


Figure 4.3 Range VS NZC plots for Crew and Ice Sequences

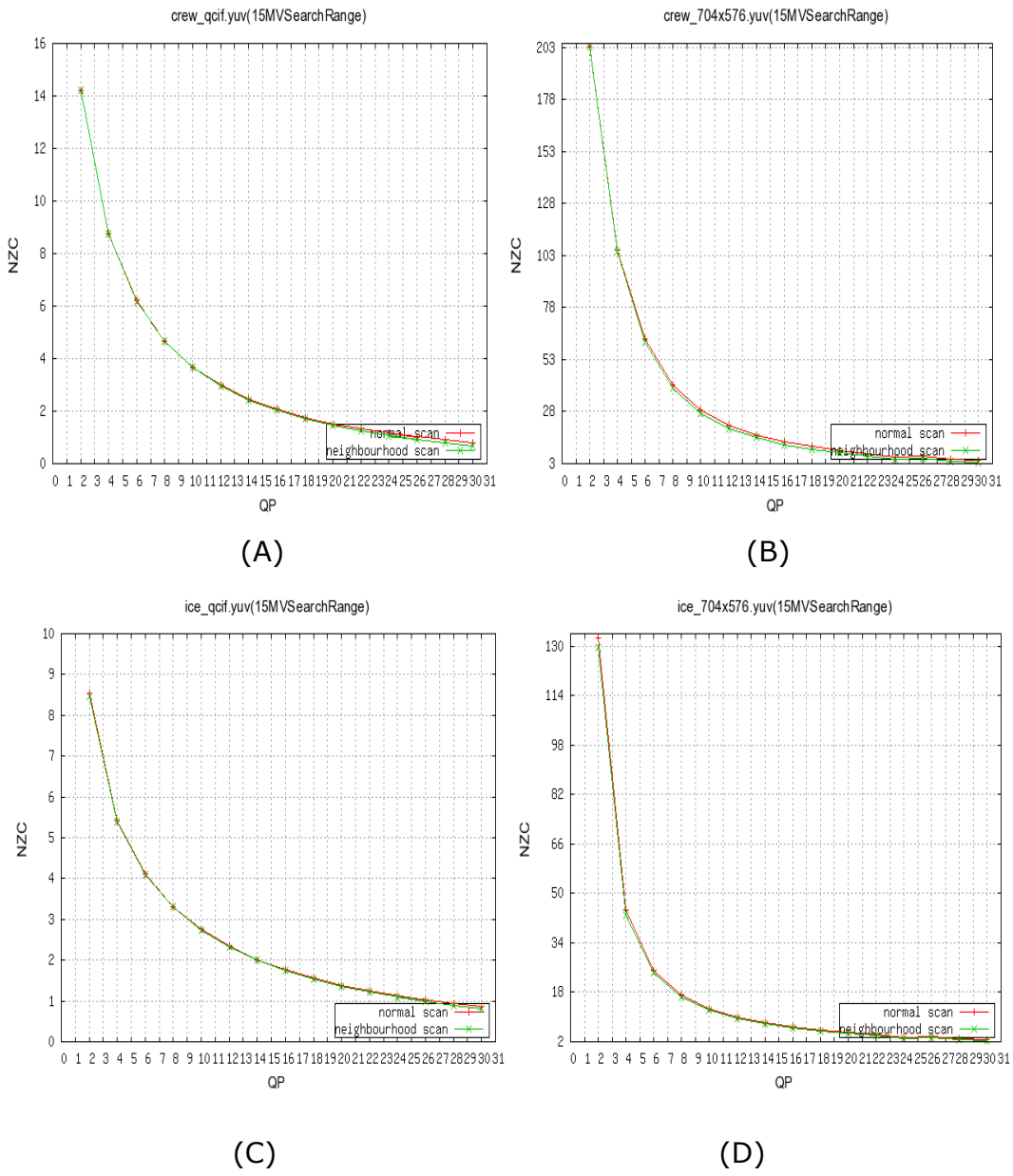


Figure 4.4 QP VS NZC plots for Crew and Ice Sequences

The above results show that as there is significant improvement in the final bitrate. Figure 4.3 shows the Range Vs NZC plots for a constant QP (in this case 30) and Figure 4.4 shows plots for QP Vs NZC for a constant range (in this case 15). As shown in figure 4.3(A) and 4.3(B) for the sequence Crew, it is clearly seen that as the QP increases the average number of nonzero coefficients decreases considerably across ranges and is quite low at a search range of 15. This trend improves as the resolution of the video increases, as here it increases from QCIF to 4CIF. The same is the case for the sequence Ice as shown in figure 4.3(C) and 4.3(D).

The gain is not that much prominent for lower resolution videos as it is for high resolution videos. From figure 4.4(A) and 4.4(B) it is observed that as the QP

increases from 2 to 30 there is a significant drop in the average number of non zero coefficients. This trend is also observed in Figure 4.4(C) and 4.4(D) also. One statistical observation made here was that the gain in the final bitrate (average NZC) improved across resolutions and the proposed method would work really well for higher resolutions.

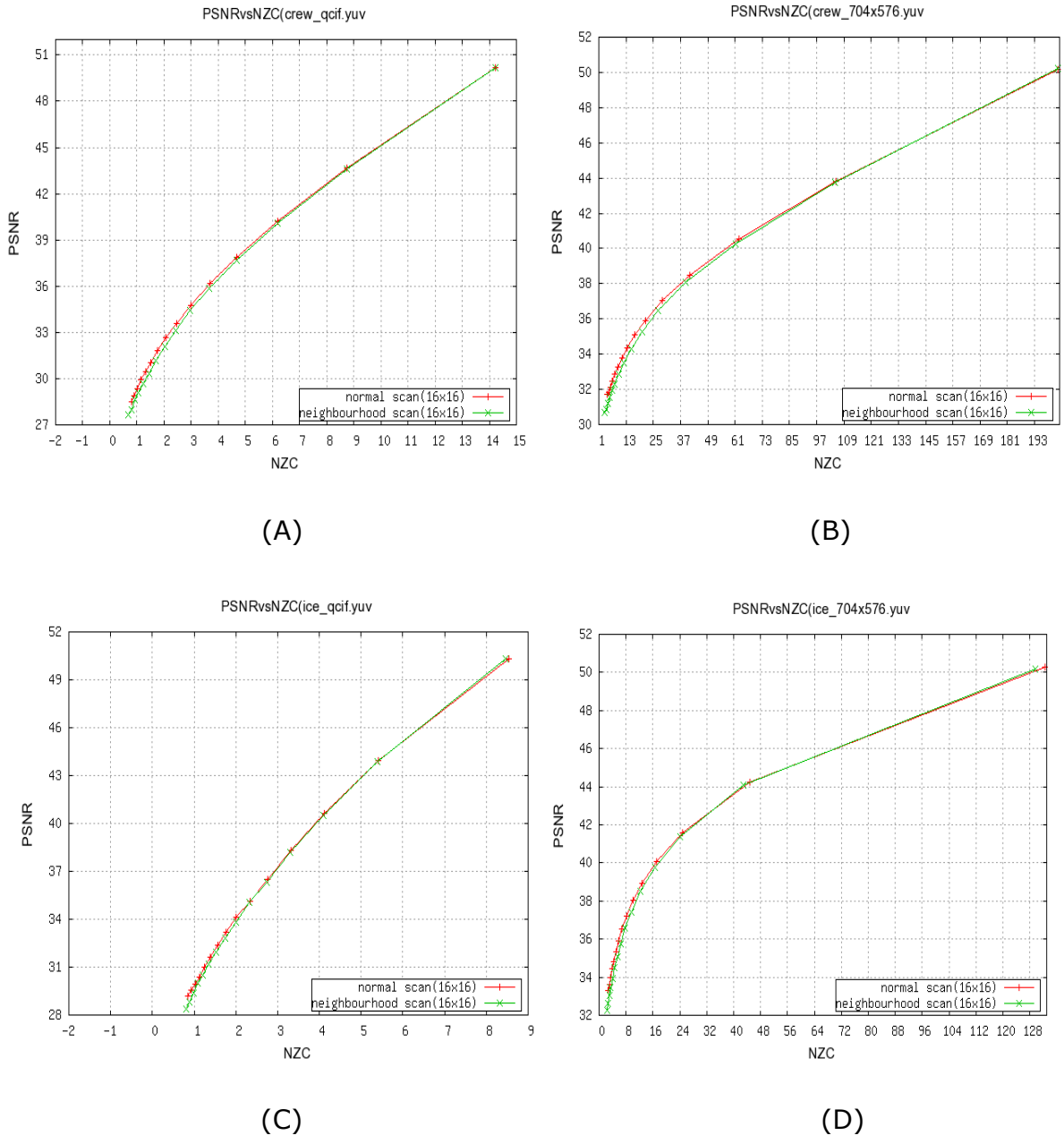
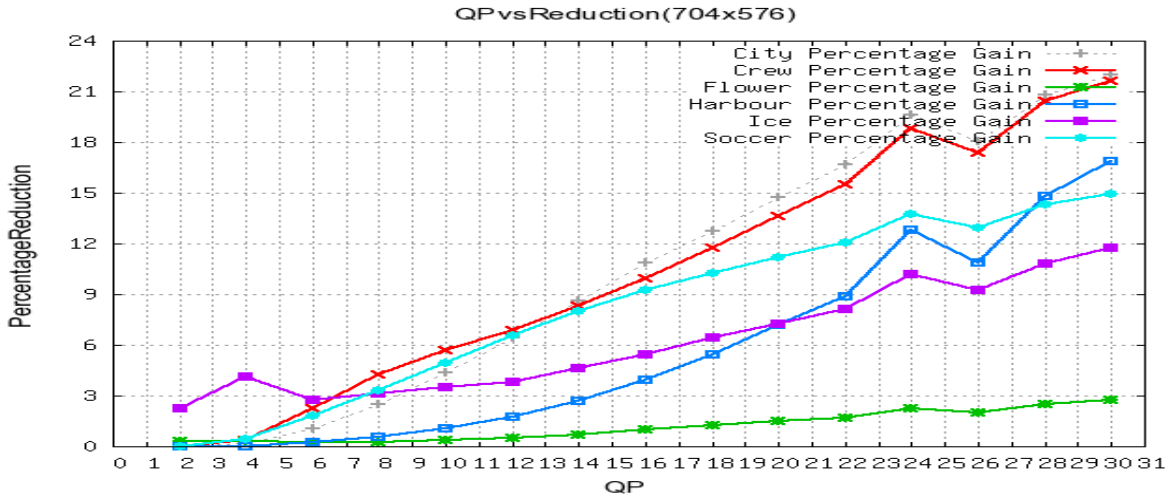


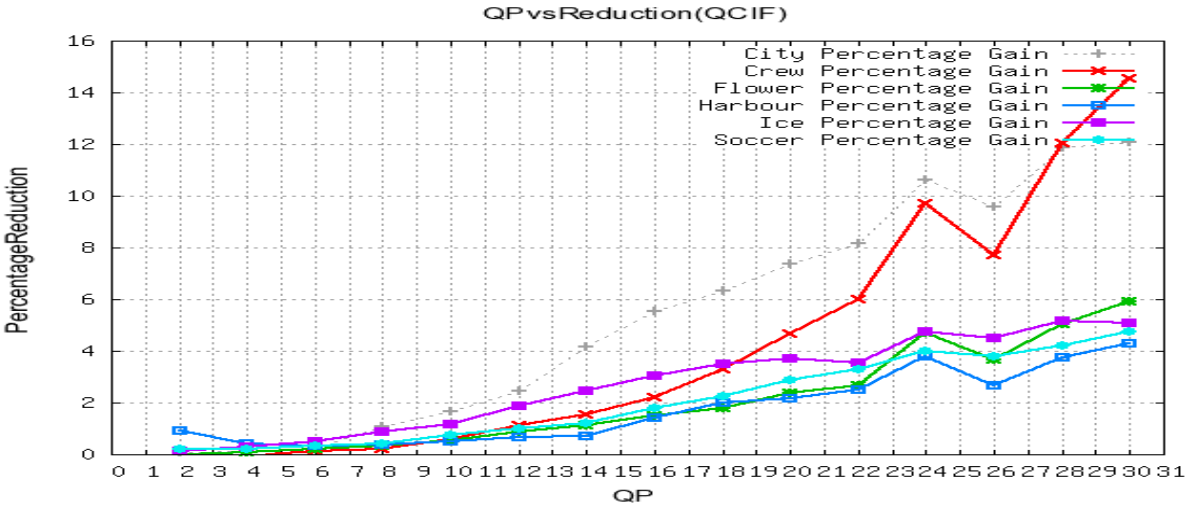
Figure 4.5 RD curves for Crew and Ice Sequences

Figure 4.5 shows the Rate distortion curve for Crew and Ice. It can be observed that the proposed method performs very close to the normal encoding method but along with that there is one more observation here. There is some loss in PSNR in the proposed approach in comparison to the normal approach. This is

due to the energy compaction property of DCT as explained in chapter 3 [21]. Other graphs for all the sequences are given in the appendix and the same observations and inferences can be made there as we did in the case of Crew and Ice.



(A)



(B)

Figure 4.6 Gain plots for Crew and Ice Sequences

4.2.2 8X8 NEIGHBOURHOOD SCAN

After all the analysis from the various graphs we also wanted to study the pattern of the gain across various QPs. The graphs shown in figure 4.6(A) and Figure 4.6(B) show that there is an increase in the gain at higher QPs. In other words at higher QP we have a better gain.

With the inference that we get higher gains at lower bitrates, i.e. at higher QP, we implemented the proposed approach for 8x8 motion estimation as finer the motion estimation more is the accuracy and better is the compression. So in order to explore this approach for more better gains we implemented this approach for 8x8 motion estimation. But this time we ran the simulations for a constant search range of 15 as we were getting the best gain for the search range of 15 and we wanted to get closer to the actual encoding scheme in which the search is automatically set to the best range.

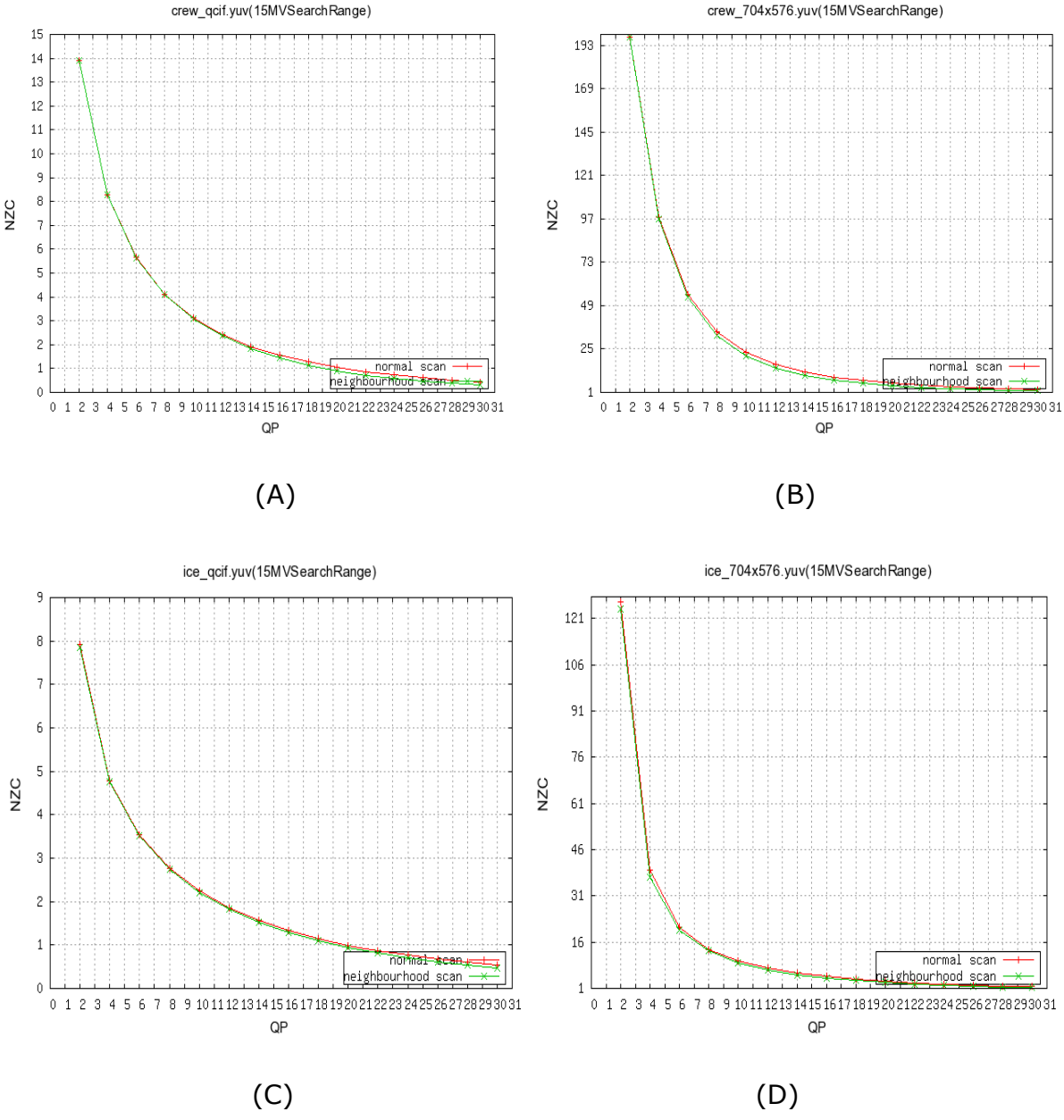
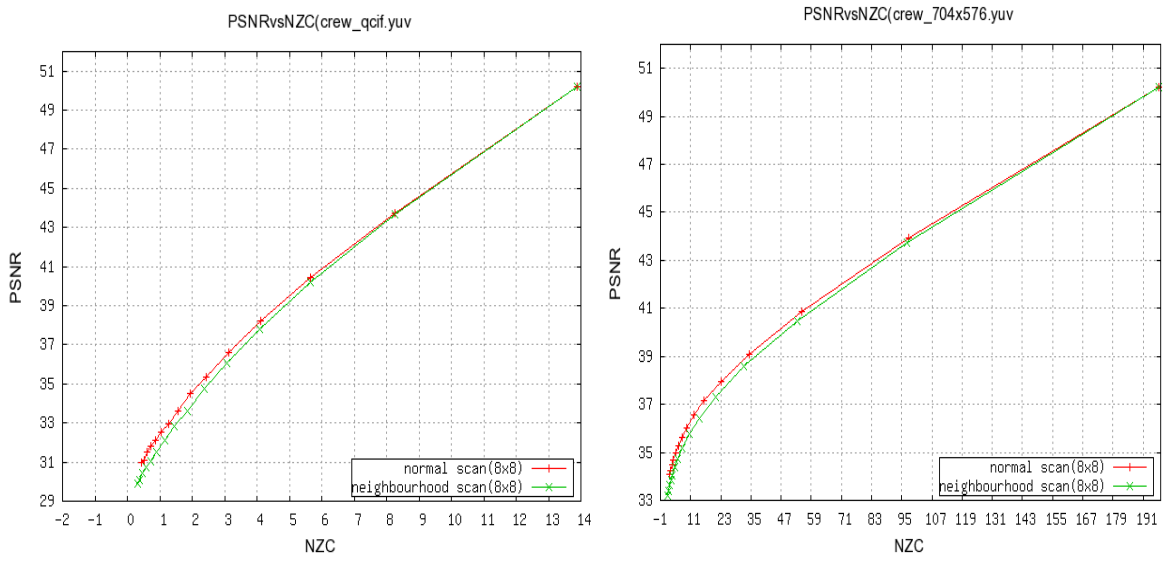
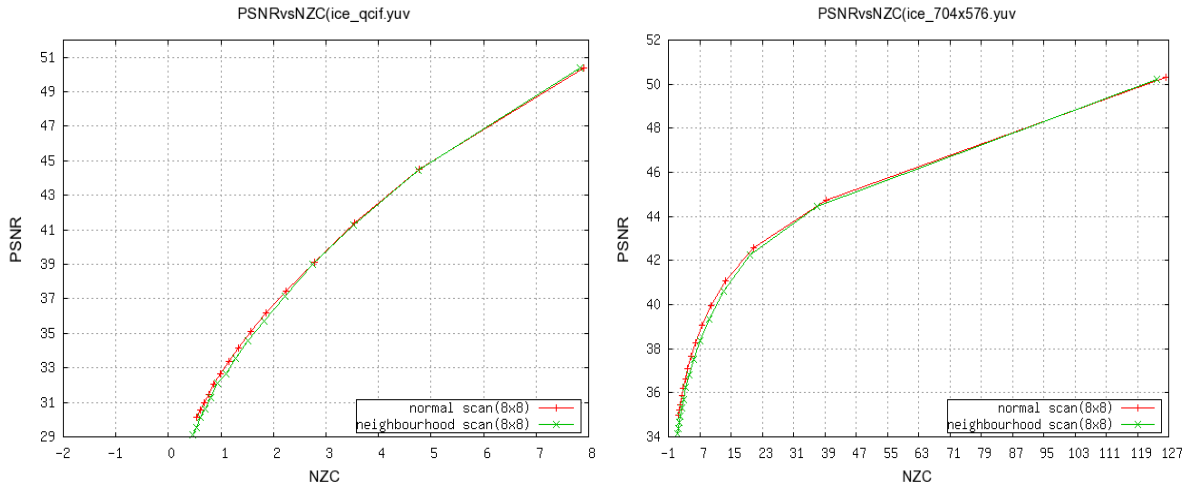


Figure 4.7 QP VS NZC plots for 8x8 ME for Crew and Ice Sequences

As shown in both the sets of plots of figure 4.7, we can observe that the inference that average NZC decreases as the QP increases, still holds for 8x8 motion estimation. This means that the proposed method works well for more accuracy of motion estimation also. This is a very good sign because as the size of motion estimation (ME) reduces, more accuracy for motion vectors is achieved and this reduction in the ME block size is being exploited in many of the latest codecs in the market e.g. H.264 . But this more accuracy comes with more complex encoders. There is a tradeoff between accuracy and complexity where complexity wins as more and more research is being done to write faster encoders with as much complexity as possible. As shown in figure 4.8, the RD curves behaves similarly to the ones we had for 16x16 ME approach but the same drawback is observed here. We lose some PSNR here too and on analysis of various results we came to a conclusion that we were losing on an average 1 db of PSNR across QP but gaining in bitrates i.e. we are able to achieve lesser bitrates for 8x8 ME in comparison to the 16x16 ME approach for Neighbourhood Scan.



(A) (B)



(C) (D)

Figure 4.8 RD curves for 8x8 ME for Crew and Ice sequences

4.2.3 16X16 PREV-NEIGHBOURHOOD SCAN

After we analyzed the proposed approach for 8x8 and 16x16 we tried to exploit the initial hypothesis, that if the residual blocks are closely related they have similar distribution. In order to do so we tried using the permutations of the previously encoded frame, as the area of my research was permutation based pre transform scanning using previously coded blocks in which we decided to use permutations of the previous blocks. These previous block permutations need not have a scope necessarily in the current frame but may also be extended to the previous frames. Theoretically if there is very less motion in the sequence and not much movement of objects then the subsequent frames may have a lot of similarities and hence have similar motion vectors. In order to explore this additional approach in the proposed approach we again implemented this in the neighbourhood method i.e. evaluation of the block using the permutation corresponding to the block number in the previous frame of the sequence. We started saving the permutations in a data structure which were available only to the next subsequent frame. We ran all simulations again for a constant search range of 15 and across a QP ranging from 2 to 30. We decided first we will test this additional scheme with a 16x16 ME.

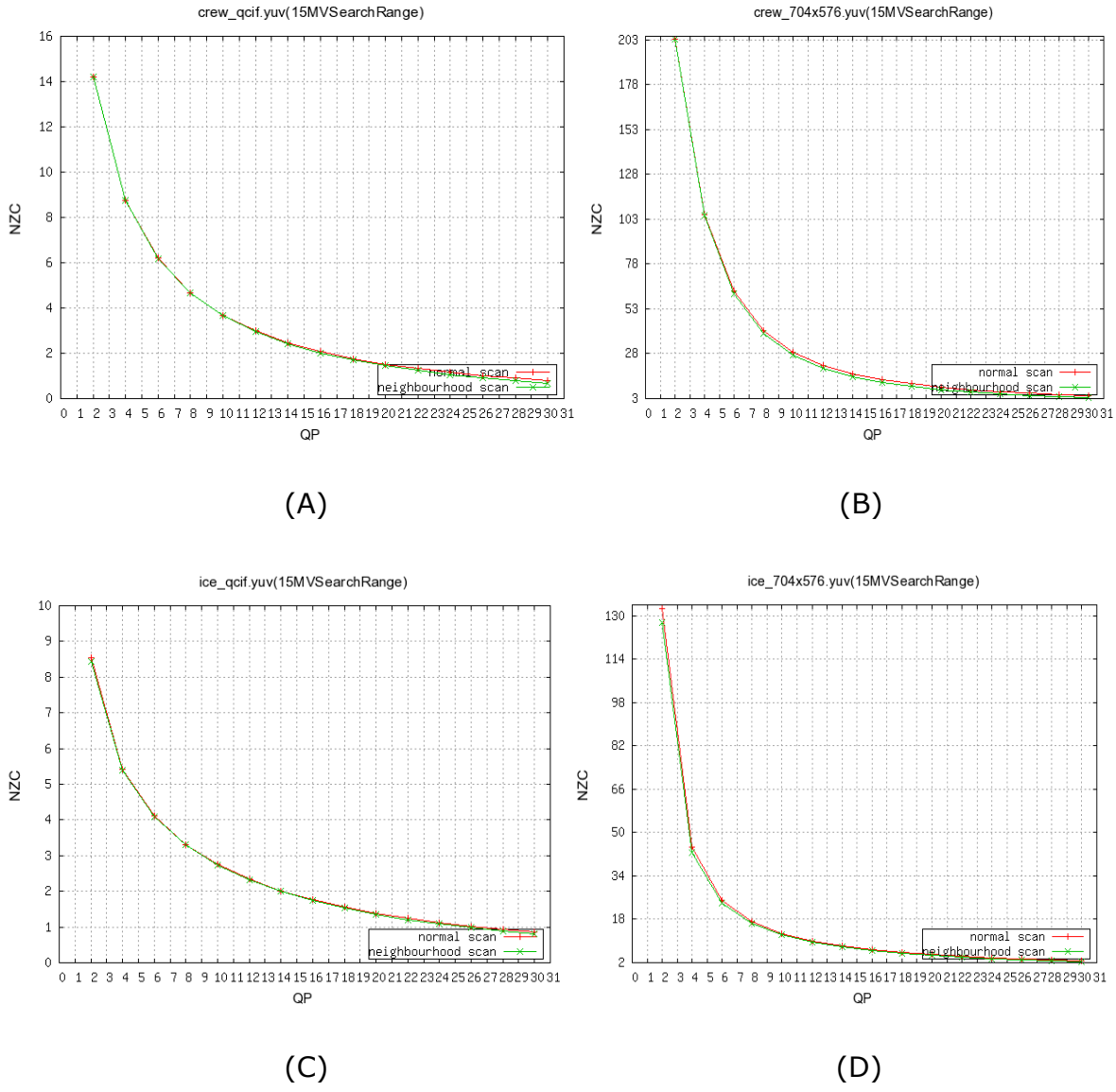


Figure: 4.9 QP Vs NZC plots for Prev-Neighbourhood Scan for Crew and Ice sequences

As shown in the figure 4.9, we still get lesser average NZC compared to the normal encoding procedure, but we have to also check whether still we are loosing PSNR compared to the original sequence and if we are how much are we losing. Are we loosing more or less?

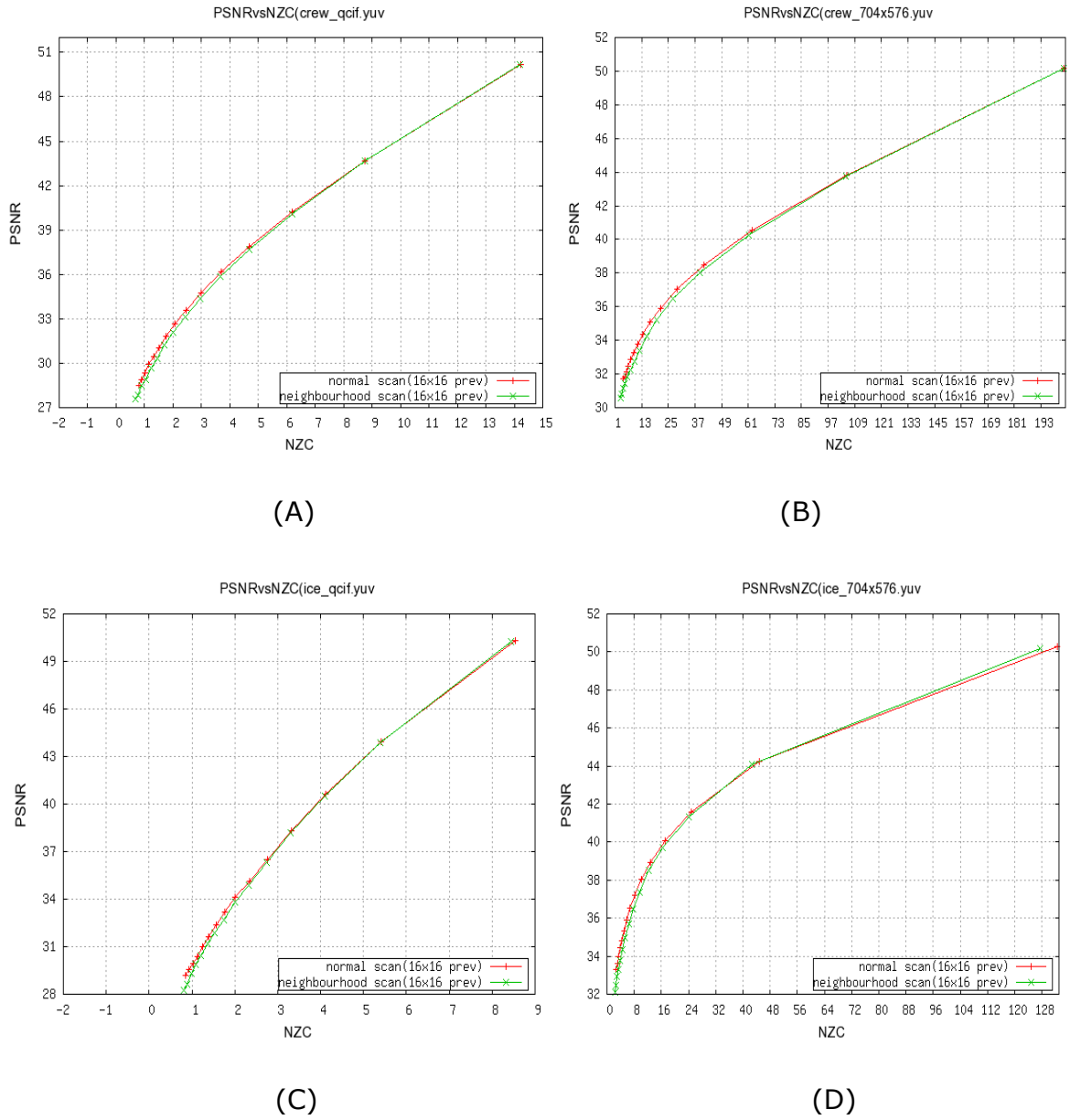


Figure 4.10 RD curves for Prev-Neighbourhood Scan for Crew and Ice sequences

As shown in figure 4.10, we are performing better as the bitrate decreases but still we are loosing PSNR but fortunately after analyzing we came to a conclusion that we were still roughly loosing 1 Db PSNR and not more as we had expected. But the task now in front of us was to compare this Prev- Neighbourhood scheme with the Neighbourhood scheme previously analyzed. For this we compared both approaches on the basis of RD curves because we not only wanted to know what approach is performing better but we also wanted to know which approach loses more PSNR compared to the other.

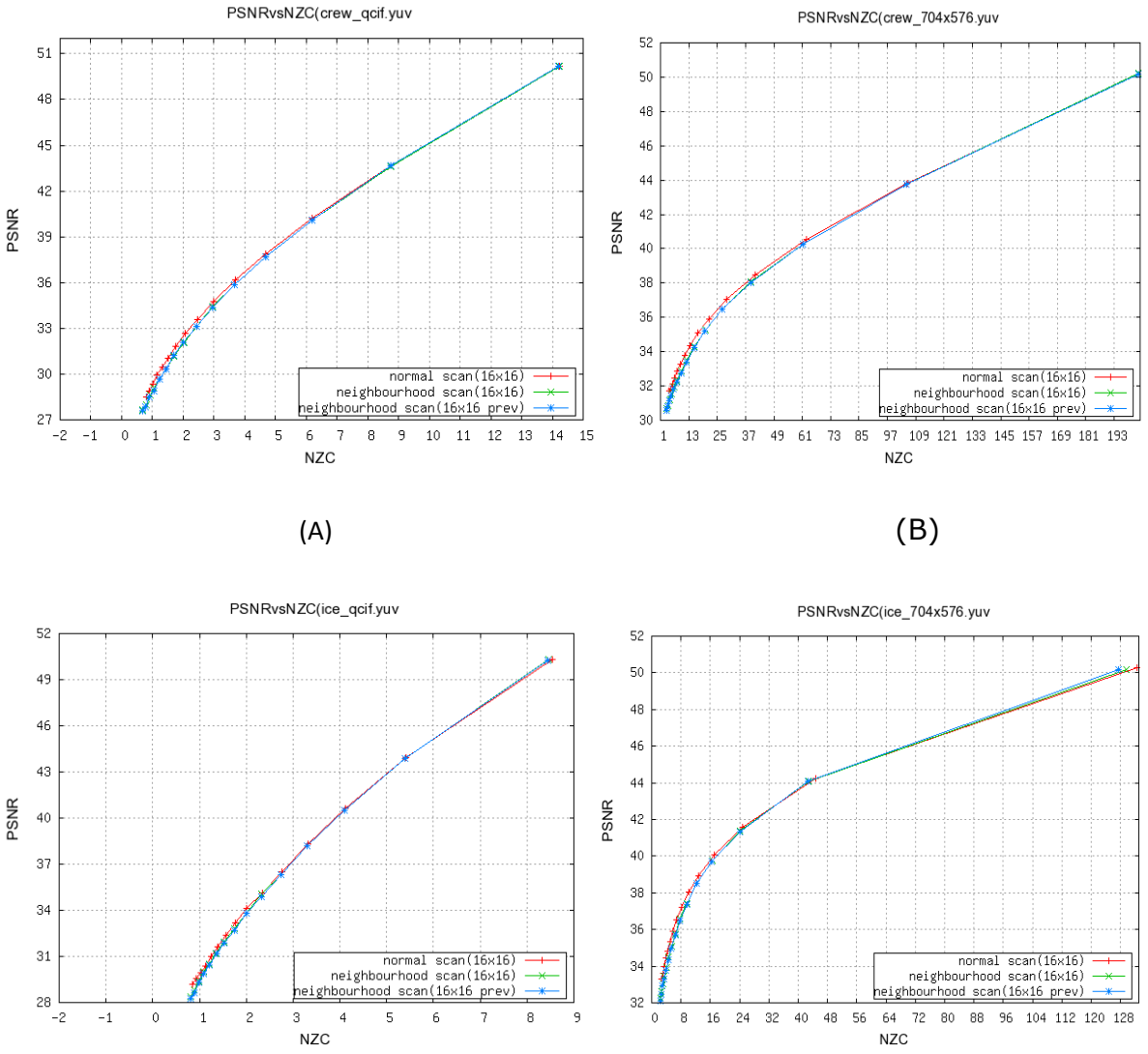


Figure 4.11 RD curves for Comparing Prev-Neighbourhood and Neighbourhood Approaches

It is very difficult to infer from mere observation of the RD plots which approach is performing better than the other. So we did a more intense study on the values and came to certain conclusions. We made some study based on some data collected during the simulations and they were as shown in table 4.1. Table 4.1 shows the simulations for the normal 16x16 approach, Neighbourhood 16x16 approach and Prev-Neighbourhood 16x16 approach for the sequence Crew and table 4.2 shows the results for the sequence Ice. The tables have data for the sequences at a resolution of 704x576 (4CIF). These statistics show that we were able to achieve lesser bitrates with the Prev-Neighbourhood approach in comparison to the Neighbourhood approach but we were losing more PSNR in comparison to the Neighbourhood approach. The loss still remains roughly 1 Db

but when we study the statistics from the simulations we are able to infer that we loose more in the case of the Prev-Neighbourhood approach.

Table 4.1 PSNR values for 16x16 approaches for Crew (4CIF)

16x16 Normal Scan	**16x16 Neighbourhood Scan**	**16x16 Prev-Neighbourhood Scan**
50.1936	50.2083	50.194
43.7999	43.7323	43.7229
40.5398	40.2563	40.2313
38.4849	38.0584	38.0213
37.0593	36.4829	36.4474
35.9204	35.2521	35.2048
35.0834	34.2917	34.2452
34.3641	33.5011	33.4029
33.7826	32.8547	32.7764
33.2443	32.3187	32.2233
33.8399	31.9232	31.8281
33.1289	31.1925	31.4366
32.4594	31.5536	31.1688
31.8413	30.845	30.8005
31.6916	30.6732	30.537

Table 4.2 PSNR values for 16x16 approaches for Ice (4CIF)

16x16 Normal Scan	16x16 Neighbourhood Scan	16x6 Prev-Neighbourhood Scan
50.2599	50.1895	50.1889
44.244	44.0819	44.073
41.5897	41.3616	41.3393
40.079	39.7854	39.7323
38.9143	38.4994	38.4875
38.0176	37.4302	37.3885
37.1907	36.5876	36.4602
36.5129	35.7557	35.6962
35.9021	35.0629	34.9647
35.35	34.478	34.3187
34.8242	33.9055	33.7914
33.9766	33.0223	33.3213
34.4325	33.4797	32.92
33.601	32.6432	32.488
33.2778	32.2822	32.1251

4.2.4 8X8 PREV-NEIGHBOURHOOD SCAN

After we implemented this with 16x16 ME, more curiosity arose as to how this approach will perform with 8x8 as we were able to attain lesser bitrates than our previous Neighbourhood approach and we already had seen that we have lesser bitrates for 8x8 ME compared to 16x16 ME.

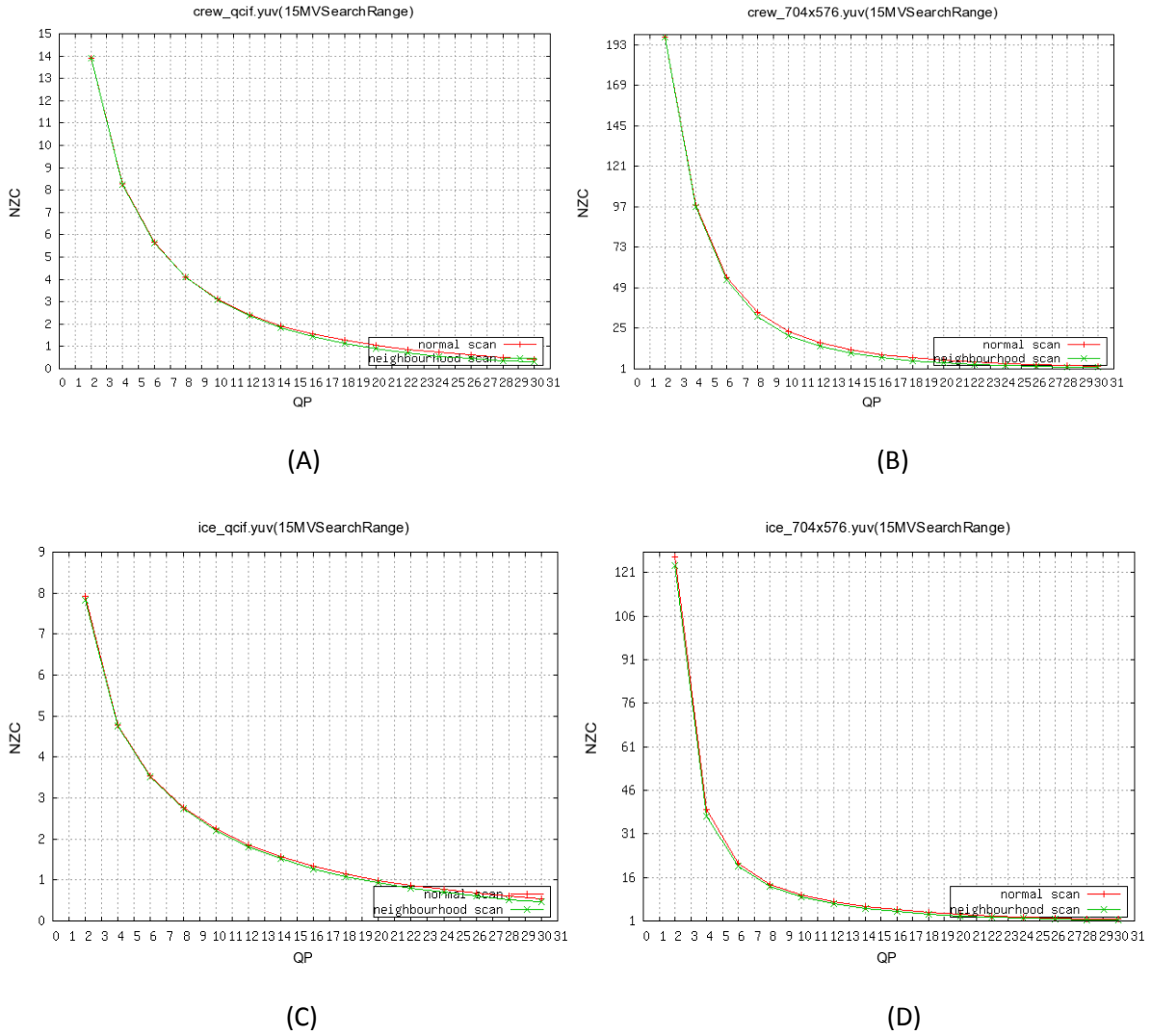


Figure 4.12 QP Vs NZC curves for 8x8 Prev-Neighbourhood Approach

As shown in the figure 4.12, we are able to achieve better bitrates compared to the 16x16 Prev-Neighbourhood approach. This cannot be clearly inferred from the plots but after closely analyzing the statistics got from simulation, we able to infer that we were able to get better bitrates for 8x8 ME approach for Prev-neighbourhood. But again the question arises as to how much PSNR we are losing in this case and for that again the best place to check for it would be the RD Curves.

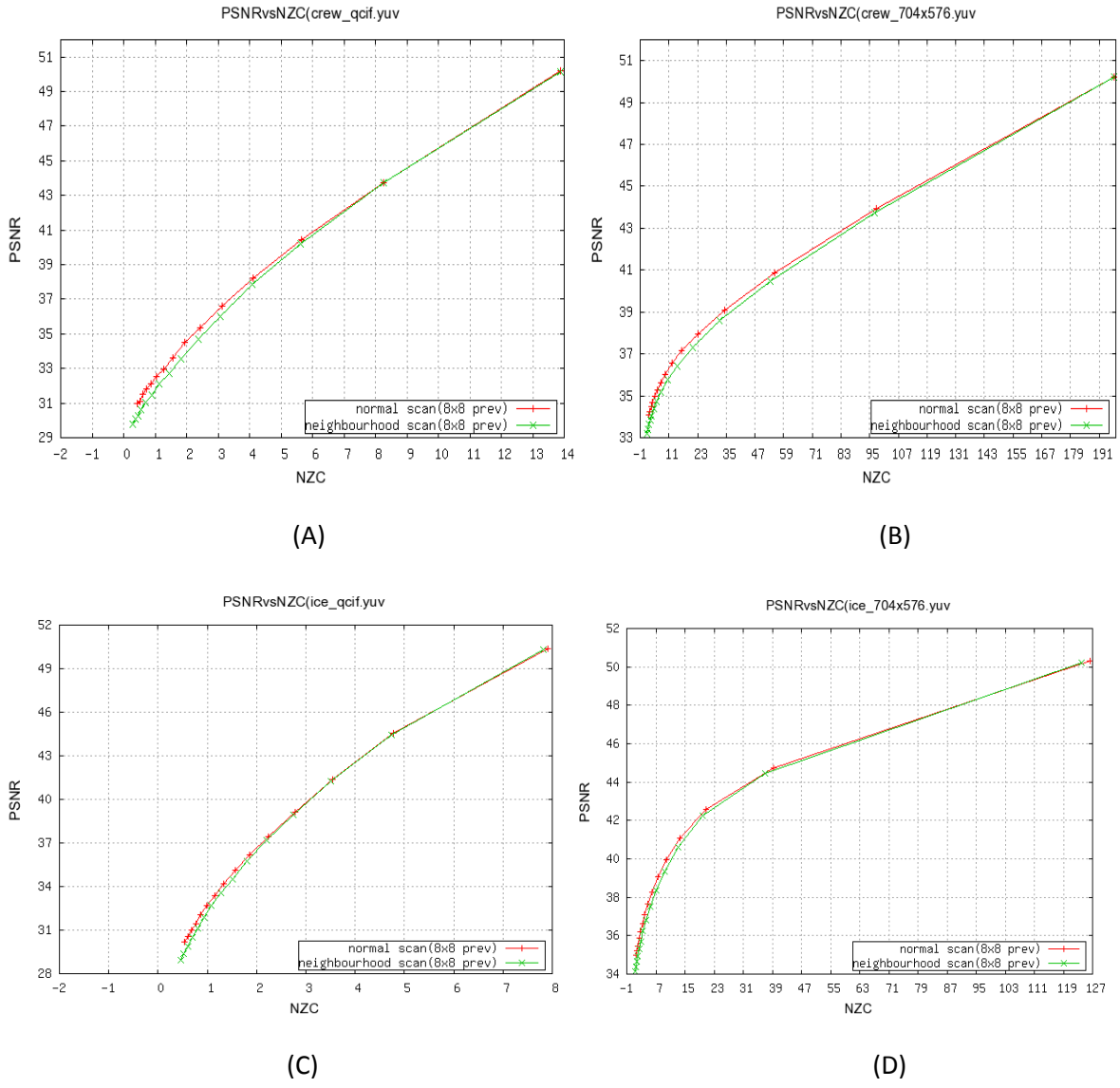


Figure 4.13 RD curves for 8x8 Prev-Neighbourhood Scan

As shown in figure 4.13, we are losing PSNR and close statistical analysis show it is still on an average 1 Db loss. But more interesting here would be to study which approach is losing more PSNR i.e. is 8x8 Neighbourhood or 8x8 Prev Neighbourhood approach.

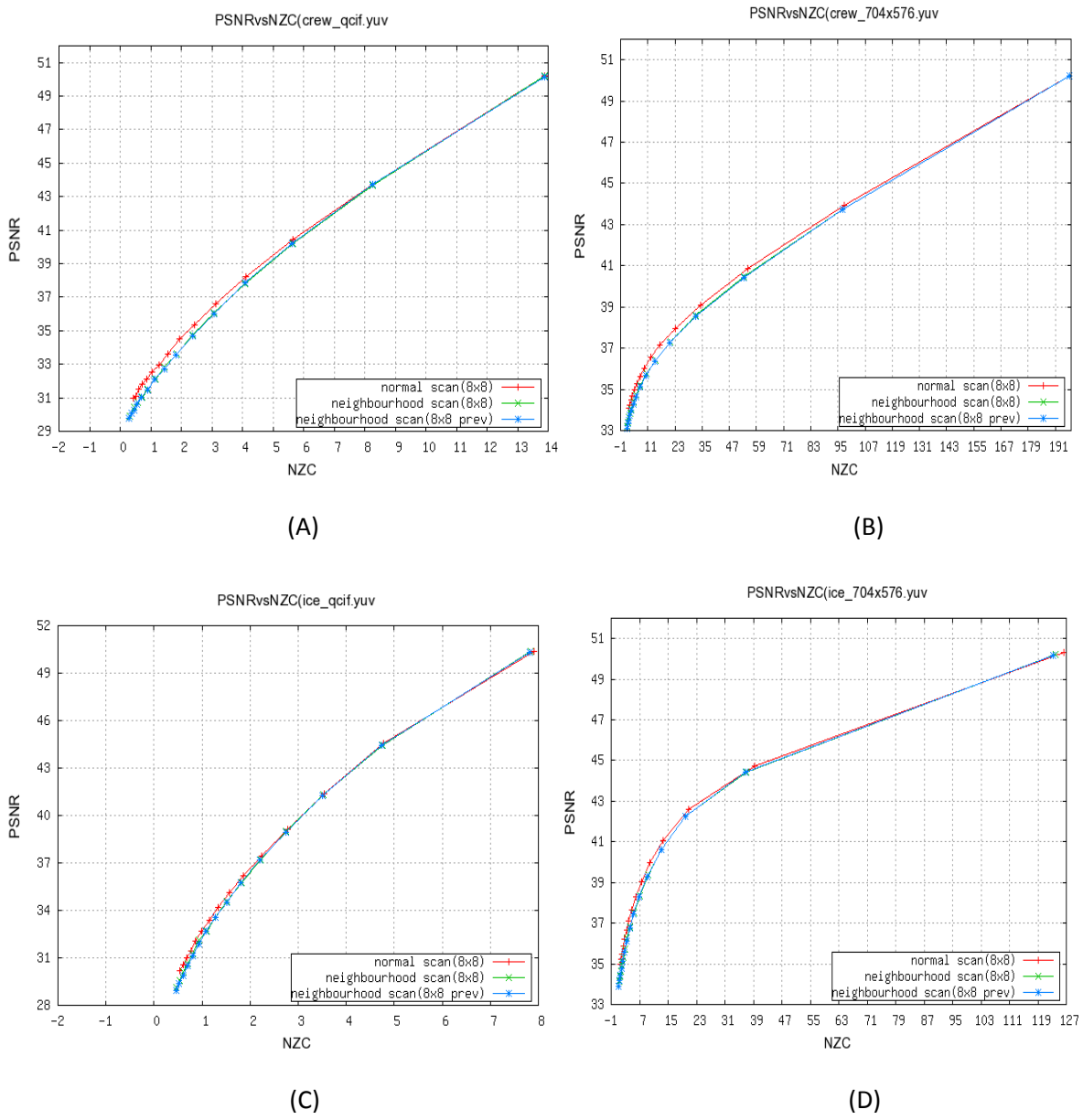


Figure 4.14 RD curves for comparing 8x8 Prev-Neighbourhood and 8x8 Neighbourhood approach

Again it is very difficult to say from the plots by mere observation which is a better approach. We made some study based on some data collected during the simulations and they were as shown in table 4.3. Table 4.3 shows the simulations for the normal 16x16 approach, Neighbourhood 16x16 approach and Prev-Neighbourhood 16x16 approach for the sequence Crew and table 4.4 shows the results for the sequence ice. The tables have data for the sequences at a resolution of 704x576 (4CIF). but stats show that we are able to achieve better bitrates from 8x8 Prev-Neighbourhood approach compared to the Neighbourhood approach but at the cost of PSNR. The average loss is still 1 Db but when

compared looking at stats we find we are losing more in Prev-Neighbourhood approach.

Table 4.3 PSNR values for 8x8 approaches for Crew (4CIF)

8x8 Normal Scan	8x8 Neighbourhood Scan	8x8 Prev-Neighbourhood Scan
50.2196	50.208	50.2148
43.9111	43.7543	43.7515
40.8789	40.4667	40.438
39.1021	38.5666	38.5297
37.9565	37.3199	37.2741
37.1539	36.4193	36.366
36.5505	35.7621	35.6849
36.0274	35.194	351222
36.6117	35.7543	34.654
35.1835	34.3715	34.2943
35.0002	34.0602	33.9817
34.6987	33.8603	33.7444
34.474	33.6309	33.5136
34.2531	33.3955	33.3627
34.0845	33.2212	33.1219

Table 4.4 PSNR values for 8x8 approaches for Ice (4CIF)

8x8 Normal Scan	8x8 Neighbourhood Scan	8x8 Prev-Neighbourhood Scan
50.2981	50.2081	50.1938
44.7408	44.4433	44.4354
42.5218	42.2645	42.2425
41.0701	40.6248	40.5999
39.9614	39.3601	39.3043
39.0452	38.3622	38.2777
38.286	37.5034	37.4291

37.0549	36.8175	36.7433
37.1123	36.2482	36.1378
36.6386	35.7026	35.5876
36.218	35.3056	35.1359
35.8549	34.9278	34.7895
35.4652	34.6502	34.4434
35.2127	34.3872	34.1922
34.9949	34.1385	33.8958

4.2.5 IMPLEMENTATION IN XVID ENCODER

After getting all the high gains in bitrates, we now thought that now we had enough gains to test the proposed approach in some industry encoder. We decided to embed this approach with Xvid MPEG-4 encoder. Details of Xvid MPEG-4 encoder are given in appendix-B. The entire implementation was in C with roughly 1300 lines of code. There were two phases of analysis with Xvid encoder.

First we started studying the results with variable QP. Xvid gives an option where we can use a quantization parameter based on how the quality of the video can be maintained without having artifacts introduced in the video. But as we were experimenting with this we found that the average length of the file (in bytes) was increasing. This file corresponded to the average length contributed by each frame. In Xvid the result were shown frame by frame and hence we were able to understand this. It gave all the stats frame by frame like length of the file, type of frame, time taken to encode it etc. When we observed the average length of file per frame, it increased. We wanted to know where the approach was failing. We experimented with City and Ice in our experiments(QCIF resolution). Here we have explained the results got from the City sequence. The results for the sequence Ice are given in appendix C. We first took the results for the non zero coefficient count for both the approaches. We found that we were getting a reduction there (roughly 7.5 to 8%).Then we started studying all the distributions of all the various levels to see if we were

getting some weird value. But we observed that we had similar distributions. In fact we were able to get the same levels in both the cases.

Table 4.5 Histogram data for levels in Xvid implementation for City Sequence

Level	Normal Scan	Neighbourhood Scan
0	3462847	3478477
1	267837	266230
2	20551	20799
3	5581	5748
4	1949	1969
5	804	819
6	327	351
7	172	173
8	74	79
9	47	39
10	18	22
11	8	7
12	4	2
13	6	4
14	1	0
15	3	1
17	1	0

Then we took the results for the different run length distributions to check whether something is wrong with the encoding schema. Table 4.6 shows the distributions for various run lengths for both the cases, i.e. normal and neighbourhood approach.

Table 4.6 Histograms for run lengths for City Sequence

Run Length	Normal Scan	Neighbourhood Scan
1	41304	35610
2	24813	24399
3	17665	18309
4	15308	15409
5	15061	14068
6	18332	16839
7	14412	11614
8	7202	7807
9	3682	5417
10	2825	4467
11	2454	3744

12	2464	3636
13	2901	3851
14	2519	3026
15	2896	2828
16	2144	2744
17	1380	2296
18	945	1769
19	878	1493
20	855	1340
21	792	1208
22	779	1157
23	1159	1260
24	1394	1477
25	580	936
26	415	844
27	325	664
28	362	545
29	280	498
30	289	487
31	328	561
32	654	691
33	357	546
34	225	384
35	139	329
36	819	256
37	89	237
38	84	262
39	126	258
40	417	451
41	116	282
42	91	191
43	31	144
44	25	116
45	16	98
46	22	99
47	42	153
48	111	224
49	45	150
50	10	104
51	8	65
52	6	52
53	1	34
54	4	34
55	4	46
56	44	143
57	11	88
58	0	82

59	0	40
60	0	25
61	0	13
62	0	16

By observing the table we were able to infer that the run lengths are changing a little than the normal case. The maximum run length is increasing in the neighbourhood approach in comparison to the normal approach which clearly suggests why we are able to get lesser number of non zero coefficients. But then here we got an indication that there definitely something wrong with the run length encoding. Then we started taking the distribution for the various last, run, level combinations statistics for the normal and neighbourhood case.

Table 4.7 Last Run level statistics for City sequence (Normal approach)

LastRunLevel	Count	Probability	Code	Length
0$0$1	89989	0.298934	4	3
0$1$1	42396	0.140835	13	4
0$2$1	21466	0.071308	28	5
0$0$2	13852	0.046015	31	5
0$3$1	11298	0.037531	26	6
1$0$1	11226	0.037292	15	5
0$5$1	11132	0.036979	23	6
0$4$1	10798	0.03587	24	6
0$6$1	5553	0.018446	38	7
0$7$1	5204	0.017287	36	7
0$8$1	4760	0.015812	34	7
0$9$1	4672	0.01552	32	7
1$1$1	4618	0.015341	31	7
0$0$3	3941	0.013092	43	7
0$1$2	3509	0.011657	40	7
1$2$1	3461	0.011497	29	7
1$3$1	3378	0.011221	26	7
1$4$1	3236	0.01075	25	7
1$6$1	2664	0.00885	36	8
1$5$1	2585	0.008587	38	8
1$8$1	2441	0.008109	32	8
1$7$1	2407	0.007996	34	8
0$10$1	2257	0.007498	44	8
0$11$1	1981	0.006581	42	8
0$12$1	1881	0.006248	41	8
1$9$1	1772	0.005886	53	9
1$12$1	1704	0.005661	47	9
1$11$1	1703	0.005657	48	9
1$10$1	1656	0.005501	50	9

0$0$4	1629	0.005411	46	8
1$14$1	1449	0.004813	42	9
1$13$1	1436	0.00477	45	9
1$15$1	1158	0.003847	41	9
0$14$1	1003	0.003332	54	9
0$13$1	861	0.00286	56	9
0$1$3	860	0.002857	60	9
0$2$2	849	0.00282	58	9
1$16$1	804	0.002671	39	9
0$0$5	708	0.002352	63	9
1$20$1	649	0.002156	43	10
1$19$1	577	0.001917	45	10
0$3$2	537	0.001784	70	10
0$15$1	481	0.001598	66	10
1$21$1	476	0.001581	41	10
1$18$1	470	0.001561	46	10
1$17$1	465	0.001545	48	10
0$4$2	454	0.001508	68	10
1$22$1	396	0.001315	39	10
1$24$1	378	0.001256	34	10
0$20$1	372	0.001236	57	10
1$23$1	360	0.001196	37	10
0$16$1	346	0.001149	64	10
0$18$1	331	0.0011	60	10
0$19$1	323	0.001073	59	10
0$17$1	322	0.00107	62	10
1$27$1	318	0.001056	11	11
0$0$6	305	0.001013	75	10
1$25$1	265	0.00088	14	11
0$2$3	240	0.000797	28	11
0$3$3	238	0.000791	26	11
1$26$1	235	0.000781	13	11
0$5$2	233	0.000774	24	11
0$21$1	222	0.000737	54	10
0$7$2	207	0.000688	21	11
0$22$1	195	0.000648	52	10
1$28$1	193	0.000641	8	11
0$6$2	193	0.000641	22	11
0$1$4	187	0.000621	30	11
0$8$2	172	0.000571	18	11
1$0$2	159	0.000528	50	10
0$0$7	157	0.000522	72	10
0$9$2	147	0.000488	16	11
1$30$1	123	0.000409	74	12
1$31$1	104	0.000345	76	12
1$29$1	97	0.000322	73	12
1$33$1	91	0.000302	176	13

1$32$1	88	0.000292	79	12
0$5$3	74	0.000246	166	13
0$0$8	72	0.000239	67	11
0$4$3	70	0.000233	165	13
0$1$5	69	0.000229	67	12
0$27$1	66	0.000219	117	12
1$35$1	63	0.000209	181	13
0$23$1	57	0.000189	68	12
1$0$3	55	0.000183	10	12
0$24$1	54	0.000179	70	12
0$26$1	51	0.000169	175	13
1$34$1	49	0.000163	178	13
0$0$9	45	0.000149	65	11
0$10$2	44	0.000146	171	13
0$6$3	43	0.000143	169	13
0$2$4	43	0.000143	162	13
1$39$1	39	0.00013	189	13
0$3$4	35	0.000116	411	14
1$1$2	34	0.000113	9	12
0$25$1	34	0.000113	173	13
1$36$1	32	0.000106	183	13
1$40$1	27	0.00009	191	13
1$38$1	27	0.00009	186	13
0$11$2	23	0.000076	1578	16
1$37$1	21	0.00007	184	13
1$11$2	19	0.000063	3121	17
0$4$4	18	0.00006	408	14
0$9$3	17	0.000056	801	15
0$2$5	17	0.000056	221	13
0$0$10	17	0.000056	15	12
0$29$1	16	0.000053	476	14
0$28$1	16	0.000053	236	13
0$8$3	15	0.00005	802	15
0$5$4	15	0.00005	407	14
0$1$7	15	0.00005	108	12
1$7$2	14	0.000047	1571	16
1$12$2	14	0.000047	3119	17
0$30$1	14	0.000047	923	15
0$1$6	14	0.000047	161	13
1$8$2	13	0.000043	1569	16
1$41$1	12	0.00004	462	14
1$0$4	11	0.000037	206	13
0$7$3	11	0.000037	804	15
0$31$1	11	0.000037	921	15
1$10$2	10	0.000033	3123	17
0$32$1	10	0.000033	918	15
1$46$1	9	0.00003	3622	17

1$6$2	8	0.000027	1573	16
1$42$1	8	0.000027	1822	16
1$1$3	8	0.000027	799	15
0$0$11	8	0.000027	12	12
1$3$2	7	0.000023	795	15
0$6$4	7	0.000023	807	15
0$10$3	7	0.000023	1580	16
1$9$2	6	0.00002	3125	17
1$14$2	6	0.00002	3114	17
0$33$1	6	0.00002	1830	16
0$0$13	6	0.00002	53	11
1$47$1	5	0.000017	3621	17
1$45$1	5	0.000017	1817	16
1$4$2	5	0.000017	793	15
1$15$2	5	0.000017	3113	17
0$4$5	5	0.000017	6212	18
0$36$1	5	0.000017	1825	16
0$3$5	5	0.000017	6214	18
0$12$2	5	0.000017	1576	16
1$43$1	4	0.000013	1821	16
1$20$2	4	0.000013	6187	18
1$2$2	4	0.000013	796	15
0$38$1	4	0.000013	3627	17
0$34$1	4	0.000013	1828	16
0$0$12	4	0.000013	65	12
1$5$2	3	0.00001	1575	16
1$24$2	3	0.00001	6178	18
0$3$6	3	0.00001	12315	19
0$2$6	3	0.00001	3130	17
0$0$15	3	0.00001	810	15
1$50$1	2	0.000007	7220	18
1$44$1	2	0.000007	1819	16
1$26$2	2	0.000007	12301	19
1$21$2	2	0.000007	6184	18
1$17$2	2	0.000007	6192	18
1$13$2	2	0.000007	3116	17
0$5$6	2	0.000007	49319	21
0$1$9	2	0.000007	3133	17
0$1$8	2	0.000007	808	15
1$8$4	1	0.000003	32645129	30
1$51$1	1	0.000003	7219	18
1$49$1	1	0.000003	3617	17
1$48$1	1	0.000003	3619	17
1$3$4	1	0.000003	32571385	30
1$27$2	1	0.000003	12298	19
1$17$3	1	0.000003	32800763	30
1$16$2	1	0.000003	3110	17

0$8$4	1	0.000003	12306	19
0$47$1	1	0.000003	14392	19
0$41$1	1	0.000003	7223	18
0$40$1	1	0.000003	7225	18
0$39$1	1	0.000003	3625	17
0$21$2	1	0.000003	6198	18
0$16$2	1	0.000003	6208	18
0$14$2	1	0.000003	3126	17
0$12$3	1	0.000003	114854	22
0$10$4	1	0.000003	49323	21
0$1$10	1	0.000003	12319	19
0$0$17	1	0.000003	3134	17

The statistics of the normal approach were compared with those of the neighbourhood approach. They were as shown in table 4.8.

Table 4.8 Last Run level statistics for City sequence (Neighborhood approach)

LastRunLevel	Count	Probability	Code	Length
0$0$1	70527	0.252266	4	3
0$1$1	37050	0.132523	13	4
0$2$1	21066	0.07535	28	5
0$3$1	11272	0.040318	26	6
0$4$1	11140	0.039846	24	6
0$5$1	11062	0.039567	23	6
0$0$2	9562	0.034202	31	5
1$0$1	8123	0.029055	15	5
0$7$1	5830	0.020853	36	7
0$6$1	5693	0.020363	38	7
0$8$1	5602	0.020038	34	7
0$9$1	5448	0.019487	33	7
1$1$1	3895	0.013932	31	7
0$0$3	3450	0.01234	43	7
1$2$1	3428	0.012262	29	7
1$3$1	3131	0.011199	26	7
1$4$1	3036	0.010859	25	7
0$12$1	2983	0.01067	41	8
0$10$1	2831	0.010126	44	8
0$11$1	2824	0.010101	43	8
0$1$2	2423	0.008667	41	7
1$5$1	2276	0.008141	38	8
1$6$1	2198	0.007862	36	8
1$8$1	2012	0.007197	33	8
1$7$1	1940	0.006939	34	8
0$14$1	1675	0.005991	54	9
0$0$4	1622	0.005802	47	8
0$13$1	1543	0.005519	57	9

1$9$1	1426	0.005101	52	9
1$10$1	1390	0.004972	50	9
1$12$1	1384	0.00495	46	9
1$11$1	1323	0.004732	49	9
1$13$1	1251	0.004475	45	9
1$14$1	1181	0.004224	42	9
1$15$1	1144	0.004092	40	9
1$16$1	1040	0.00372	38	9
0$20$1	919	0.003287	56	10
0$18$1	910	0.003255	61	10
0$21$1	903	0.00323	55	10
0$17$1	896	0.003205	62	10
0$22$1	883	0.003158	52	10
0$19$1	859	0.003073	58	10
0$16$1	839	0.003001	64	10
0$15$1	833	0.00298	67	10
0$0$5	792	0.002833	63	9
1$17$1	762	0.002726	49	10
0$1$3	751	0.002686	60	9
1$19$1	732	0.002618	44	10
1$20$1	715	0.002557	42	10
1$18$1	701	0.002507	47	10
1$23$1	628	0.002246	36	10
1$22$1	626	0.002239	38	10
0$2$2	614	0.002196	58	9
1$21$1	610	0.002182	41	10
1$24$1	583	0.002085	34	10
1$25$1	477	0.001706	15	11
1$26$1	430	0.001538	12	11
1$27$1	428	0.001531	11	11
1$28$1	397	0.00142	9	11
0$27$1	379	0.001356	116	12
0$0$6	368	0.001316	75	10
1$31$1	329	0.001177	76	12
1$30$1	324	0.001159	75	12
0$3$2	307	0.001098	71	10
1$32$1	305	0.001091	78	12
0$4$2	305	0.001091	69	10
1$29$1	284	0.001016	72	12
0$24$1	281	0.001005	70	12
1$34$1	279	0.000998	178	13
0$23$1	272	0.000973	68	12
1$35$1	244	0.000873	181	13
1$33$1	232	0.00083	176	13
1$37$1	219	0.000783	185	13
1$40$1	207	0.00074	190	13
0$2$3	207	0.00074	29	11

0$0$7	195	0.000697	72	10
0$1$4	192	0.000687	30	11
0$26$1	191	0.000683	174	13
0$28$1	184	0.000658	236	13
1$38$1	176	0.00063	187	13
1$39$1	172	0.000615	189	13
1$36$1	167	0.000597	183	13
0$3$3	151	0.00054	26	11
0$7$2	143	0.000511	20	11
1$41$1	139	0.000497	462	14
0$25$1	137	0.00049	172	13
0$6$2	124	0.000444	22	11
0$5$2	124	0.000444	25	11
0$8$2	112	0.000401	18	11
0$29$1	102	0.000365	477	14
0$9$2	101	0.000361	16	11
1$42$1	100	0.000358	1823	16
1$0$2	93	0.000333	44	8
1$48$1	92	0.000329	3619	17
1$43$1	91	0.000325	1820	16
1$46$1	87	0.000311	3623	17
0$0$8	80	0.000286	67	11
1$45$1	77	0.000275	1817	16
0$31$1	76	0.000272	920	15
0$30$1	73	0.000261	922	15
0$1$5	72	0.000258	67	12
1$47$1	69	0.000247	3621	17
1$44$1	69	0.000247	1818	16
0$32$1	69	0.000247	918	15
0$33$1	65	0.000232	1831	16
0$34$1	54	0.000193	1829	16
1$50$1	50	0.000179	7221	18
1$49$1	50	0.000179	3616	17
0$4$3	47	0.000168	165	13
1$52$1	46	0.000165	7217	18
1$51$1	43	0.000154	7218	18
0$5$3	43	0.000154	167	13
0$35$1	42	0.00015	1827	16
0$0$9	42	0.00015	65	11
0$2$4	41	0.000147	163	13
0$36$1	38	0.000136	1825	16
0$1$6	38	0.000136	160	13
1$53$1	37	0.000132	7214	18
0$38$1	37	0.000132	3626	17
0$3$4	37	0.000132	411	14
1$54$1	35	0.000125	7213	18
1$55$1	33	0.000118	7211	18

0$39$1	33	0.000118	3625	17
0$37$1	33	0.000118	3628	17
0$6$3	32	0.000114	168	13
1$56$1	29	0.000104	7209	18
0$2$5	28	0.0001	221	13
1$0$3	26	0.000093	11	12
1$58$1	24	0.000086	14385	19
1$1$2	24	0.000086	9	12
1$57$1	23	0.000082	7207	18
0$40$1	23	0.000082	7225	18
0$10$2	23	0.000082	171	13
1$59$1	21	0.000075	14383	19
0$4$4	19	0.000068	409	14
1$60$1	18	0.000064	14381	19
0$42$1	18	0.000064	14403	19
0$1$7	18	0.000064	109	12
0$41$1	17	0.000061	7222	18
0$0$10	17	0.000061	14	12
1$11$2	13	0.000046	3121	17
0$5$4	12	0.000043	407	14
0$0$11	12	0.000043	13	12
0$7$3	11	0.000039	804	15
0$45$1	11	0.000039	14397	19
0$9$3	10	0.000036	801	15
0$8$3	10	0.000036	803	15
0$47$1	10	0.000036	14393	19
0$43$1	10	0.000036	14401	19
0$11$2	10	0.000036	1578	16
1$62$1	9	0.000032	14376	19
1$4$2	9	0.000032	793	15
0$6$4	9	0.000032	807	15
0$46$1	9	0.000032	14395	19
1$61$1	8	0.000029	14378	19
1$63$1	7	0.000025	14375	19
1$7$2	6	0.000021	1571	16
1$1$3	6	0.000021	798	15
1$0$4	6	0.000021	206	13
0$48$1	6	0.000021	14390	19
0$3$5	6	0.000021	6214	18
0$10$3	6	0.000021	1580	16
0$1$8	6	0.000021	809	15
1$9$2	5	0.000018	3125	17
1$8$2	5	0.000018	1569	16
1$13$2	5	0.000018	3117	17
1$12$2	5	0.000018	3118	17
0$44$1	5	0.000018	14398	19
0$0$13	5	0.000018	52	11

0$49$1	4	0.000014	14388	19
0$0$12	4	0.000014	65	12
1$3$2	3	0.000011	795	15
1$23$2	3	0.000011	6180	18
1$20$2	3	0.000011	6186	18
1$10$2	3	0.000011	3122	17
0$51$1	3	0.000011	57415	21
0$12$2	3	0.000011	1576	16
1$6$2	2	0.000007	1572	16
1$5$2	2	0.000007	1575	16
1$24$2	2	0.000007	6178	18
1$2$2	2	0.000007	797	15
1$19$2	2	0.000007	6188	18
1$14$2	2	0.000007	3114	17
0$6$5	2	0.000007	12310	19
0$55$1	2	0.000007	32366595	30
0$50$1	2	0.000007	57413	21
0$2$6	2	0.000007	3130	17
0$18$2	2	0.000007	6205	18
0$1$9	2	0.000007	3132	17
1$22$2	1	0.000004	6182	18
1$21$2	1	0.000004	6184	18
1$18$2	1	0.000004	6190	18
1$17$2	1	0.000004	6192	18
1$16$2	1	0.000004	3110	17
1$15$2	1	0.000004	3112	17
1$10$4	1	0.000004	32677897	30
0$8$4	1	0.000004	12306	19
0$57$1	1	0.000004	32399363	30
0$56$1	1	0.000004	32391167	30
0$54$1	1	0.000004	32350211	30
0$5$5	1	0.000004	12312	19
0$4$5	1	0.000004	6212	18
0$3$6	1	0.000004	12315	19
0$19$2	1	0.000004	6202	18
0$15$2	1	0.000004	6211	18
0$0$16	1	0.000004	1582	16
0$0$15	1	0.000004	810	15

By studying the tables we came to know that there were many combinations of Last-Run-Level that were not there in the normal case but were there in the neighbourhood case. When more study was done on the theory of Variable Length Encoding (VLC) we found that the VLC coding is done on the basis of some static tables. These static tables have been designed based on some distributions of Last-Run-Level combinations. When a particular combination is

not there in these tables they are coded using Escape Sequences. There are 5 modes for these Escape sequences out of which only the first three modes are used in our case. The last two modes are based on reversible VLC which has not been implemented in Xvid MPEG-4 encoder. And more detailed study told us that these escape codes use more bits to code even the smaller values that were new to the VLC coder. So the conclusion was that we were getting the average bitrate more in the neighbourhood case because of this reason. In order to overcome this some changes need to be made to the VLC tables. We tried manipulating the current entries but the result was not getting better. So we concluded that we need to have a new VLC for proposed new approach.

All the above results were got based on variable QP. Then we thought to study how the encoder behaves if we use a constant QP for all the frames. There was an option in the encoder that would allow us to use either fixed QP or variable QP for all the frames. We got results and we thought the best way to analyze them would be using the NZC Vs QP and RD plots like we did earlier. We had taken results for all the sequences used earlier. The results for Crew and Ice are shown here. The rest of the plots are given in appendix A.

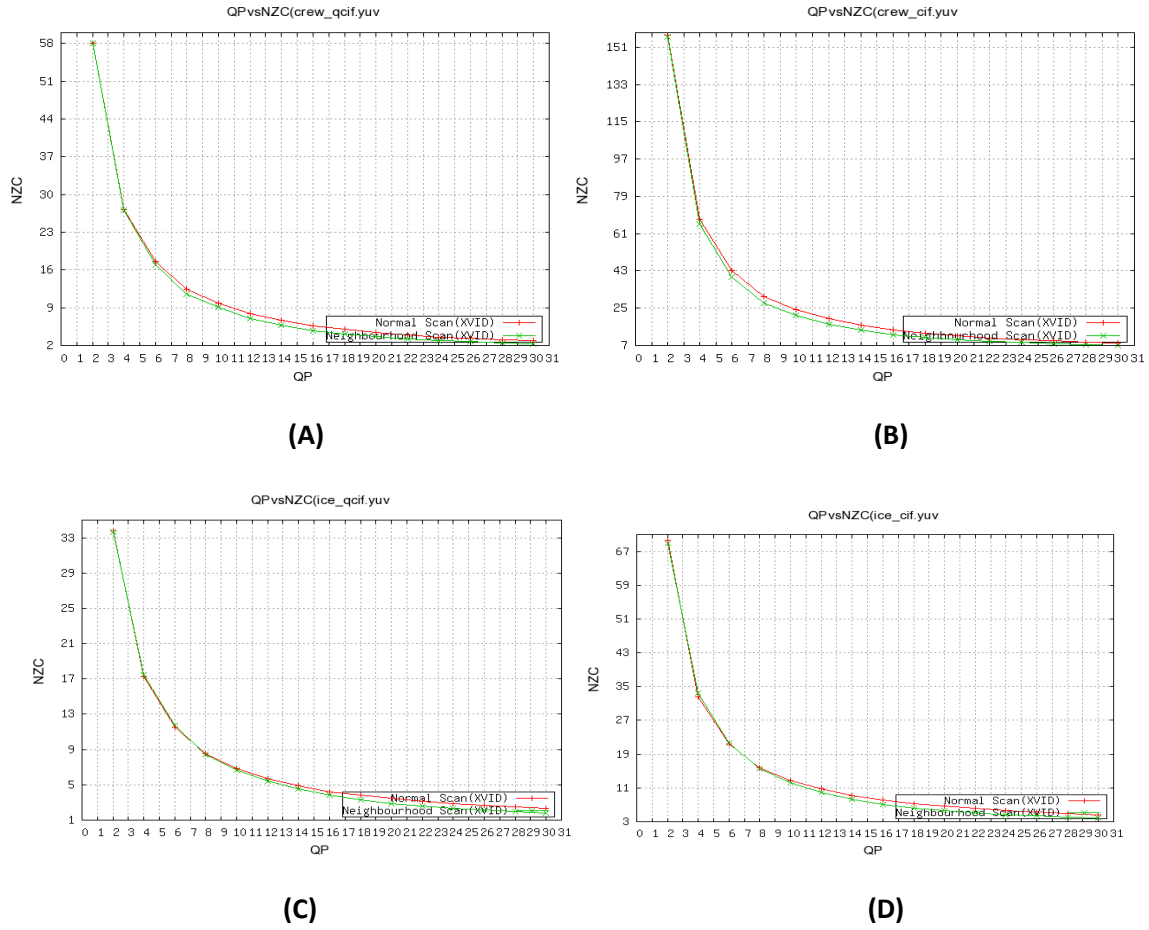


Figure 4.15 QP Vs NZC plots for Crew and Ice for Xvid implementation

The results as shown in the figure 4.15 show that the approach works poorly for lower QP but performance improves as the QP increases. It is clear from these graphs that the initial inference that we got while experimenting with our code that the performance improves as the QP increases. Now we also need to see how much PSNR we are losing when the proposed approach is implemented in the Xvid encoder. This can be clearly studied from the RD curves.

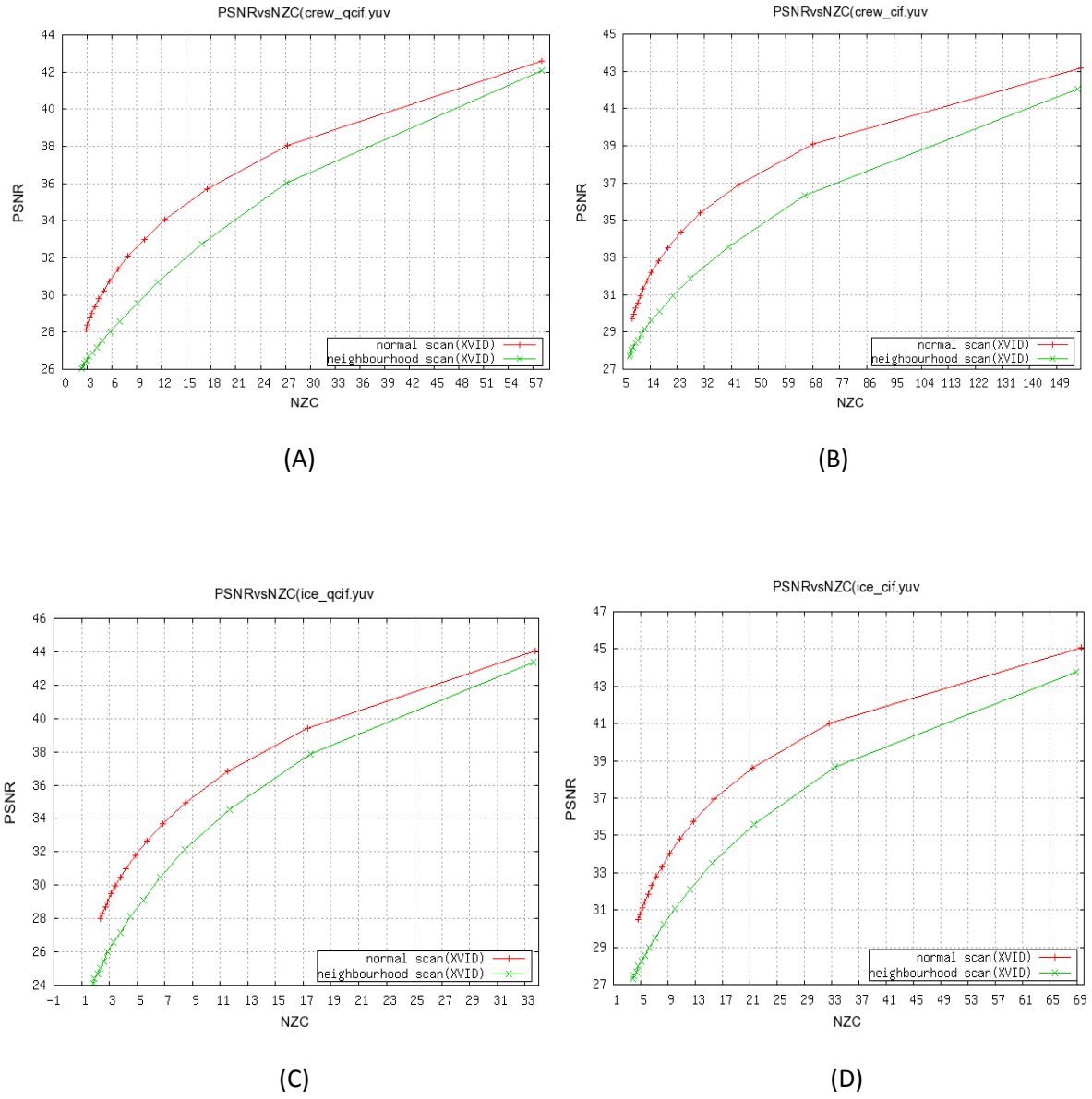


Figure 4.16 RD curves for Crew and Ice for Xvid implementation

From the curves that we got from the experiments, the results showed that we are losing roughly 3 dB of quality here. This is clearly no acceptable as an encoding standard.

5. CONCLUSION & FUTURE WORK

5.1 CONCLUSIONS

Having studied and analyzed all the results got from the experiments we came to certain conclusions. All the conclusions here revolve around the average NZC per frame and the PSNR. Explanations for each approach have been dealt with individually here.

1. **16x16 Neighbourhood approach:**
    - Average NZC per frame Improves by maximum of 25%.
    - As the resolution of the video increases gain in NZC compared to the normal approach increases.
    - Average PSNR loss: 1 dB (roughly).
2. **8x8 Neighbourhood approach:**
    - Average NZC per frame improves in comparison to 16x16 Neighbourhood approach.
    - Compared to normal 8x8 approach, the 8x8 Neighbourhood approach has a maximum gain of 34.2%.
    - Average PSNR loss: 1 dB (roughly).
3. **16x16 Prev-Neighbourhood approach:**
    - Average NZC per frame improves in comparison to the 16x16 Neighbourhood approach.
    - Compared to the normal 16x16 approach, the 16x16 Prev-Neighbourhood has a maximum gain of 24.26%.
    - Average PSNR loss: 1 dB loss (roughly).
4. **8x8 Prev-Neighbourhood approach:**
    - Average NZC per frame improves in comparison to the 8x8 Neighbourhood approach.
    - Compared to the 8x8 Normal approach, the 8x8 Prev-Neighbourhood approach has a maximum gain of 36.28%.

- Average PSNR loss: 1 dB (roughly).

5. Xvid Implementation of Neighbourhood Scan:

- Average NZC per frame improves in comparison to the Normal approach.
- A maximum gain of 25.97% over the normal approach.
- PSNR loss: 3.2 dB (roughly).

5.2 FUTURE WORK

Having studied all the above results and conclusions for all the various approaches explored we observed that due to the rearrangement of the residual blocks, we were able to decrease the non zero coefficients count, but in doing so we were also loosing PSNR (1 dB roughly) for all the cases and in the Xvid implementation we lost 3.2 dB (roughly). As PSNR is one of the most important quality parameters, first we need to think of some approach which reduces this loss in PSNR. This was also seen in the Xvid approach. We also faced a problem in the VLC part of the encoder. This problem was due to the new combinations of Last, Run, Combinations resulting from the neighbourhood approach. In order to fix this problem a new VLC needs to be designed supporting this approach.

Another approach to this approach would be to implement a post transform approach of similar characteristics. If we have a post transform approach using neighbourhood approach, i.e. using the permutations on the quantized blocks and checking which one would result in the least frame size after encoding. By doing so we will not loose PSNR and the quality of the video will be maintained.

REFERENCES

1. I. Richardson, H.264 and MPEG-4 Video Compression, John Wiley & Sons, 2003.

2. J.B. Lee and H. Kalva, "The VC-1 and H.264 Video Compression Standards for Broadband Video Services," Springer, Feb 2008.

3. T. Wiegand, G.J. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Transactions on Circuits and Systems for Video Technology, vol.13, no.7, pp. 560- 576, July 2003.

4. H. Kalva and J.B. Lee, "The VC-1 Video Coding Standard," IEEE Multimedia, Vol. 14, No 4, Oct.-Dec. 2007, pp. 88-91.

5. H. Kalva and B. Furht, "Complexity Estimation for the H.264 Coded Bitstreams," The Computer Journal, Oxford University Press, Vol. 48, No. 5, Sept. 2005, pp. 504-513.

6. J. Jia, E.-K. Jung and H.-K. Kim, "Adaptive Transform Coefficient Scan for H.264 Intra Coding," IEICE Trans Inf & Syst Vol. E90-D, No. 10, 2007, pp. 1709-1711.

7. X. Fan, Y. Lu, and W. Gao, "A novel coefficient scanning scheme for directional spatial prediction-based image compression," Proceedings of the 2003 international Conference on Multimedia and Expo, pp. 557-560.

8. D. Socek, H.Kalva, S. Magliveras, O. Marques, D. Culibrk, and B. Furht, "New Approaches to Encryption and Steganography for Digital Videos", Multimedia Systems Journal, Vol. 13, No.3, Sept. 2007, pp. 191-204.

9. J.B. Lee and H. Kalva, "Video Coding and Standardization," Encyclopedia of Multimedia, Borko Furht, Edts. Springer 2006, ISBN: 0-387-24395-X..

10. Wentao Zheng, Yoshiaki Shishikui, Masahide Naemura, Yasuaki Kanatsugu and Susumu Itoh, "Analysis of Space Dependent Characteristics of Motion Compensated Frame Differences Based on a Statistical Motion Distributed Model".

11. B. Furht, “Multimedia System: An Overview,” IEEE Multimedia, vol. 1, pp. 47-59, 1994.

12. B. Furht, J. Greenberg and R. Westwater, “Motion Estimation Techniques for Video Compression” Kluwer Academic Publishers, Norwell MA, 1996.

13. D.L. Gall, “MPEG: a video compression standard for multimedia applications,” Commun ACM, vol. 34, pp.46-58, 1991.

14. B.G. Haskell, A. Puri and A.N. Netravali, Digital Video: An introduction to MPEG-2, Chapman & Hall, LTD, 1996.

15. M. Liou, “Overview of the P-64 Kbit/s video coding standard,” Commun ACM, vol. 34, pp. 59-63, 1991.

16. A.N. Natravali and B.G. Haskell, “Digital Pictures- Representation, Compression and Standards, “2nd Edition, Plenum 1995.

17. D. Marpe, H. Schwartz and T. Weizand, “Overview of H.264/AVC Coding Standard,” IEEE Trans. On CSVT, vol. 13, No. 7, July 2003, pp. 560-575.

18. J.L. Mitchell, W.B. Pennebaker, C.E. Fogg and D.J. LeGall, “MPEG Video Compression Standard, “Digital Multimedia Standard Series, Chapman & Hall, 1996, pp. 135-169.

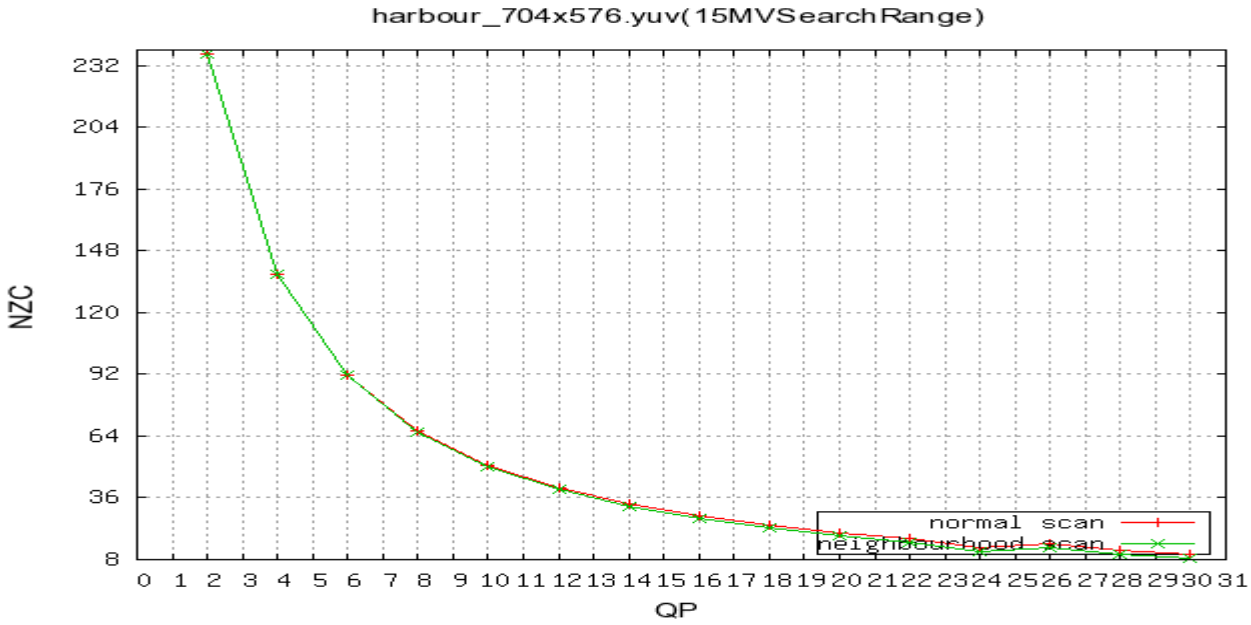
19. F. Pereira and T. Ebrahimi, The MPEG-4 Book, IMSC Press Series, Prentice Hall PTR, 2002.

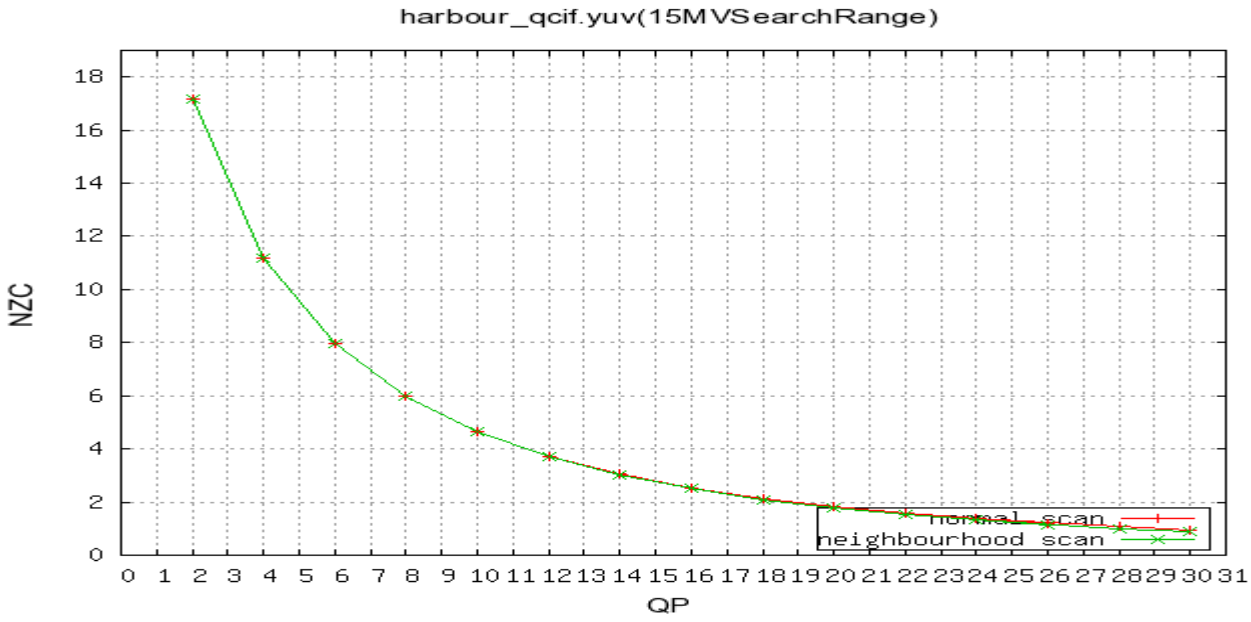
20. W.B. Pennebaker and J.L. Mitchell, JPEG Still Image Data Compression Standard, Kluwer Academic Publishers, 1992.

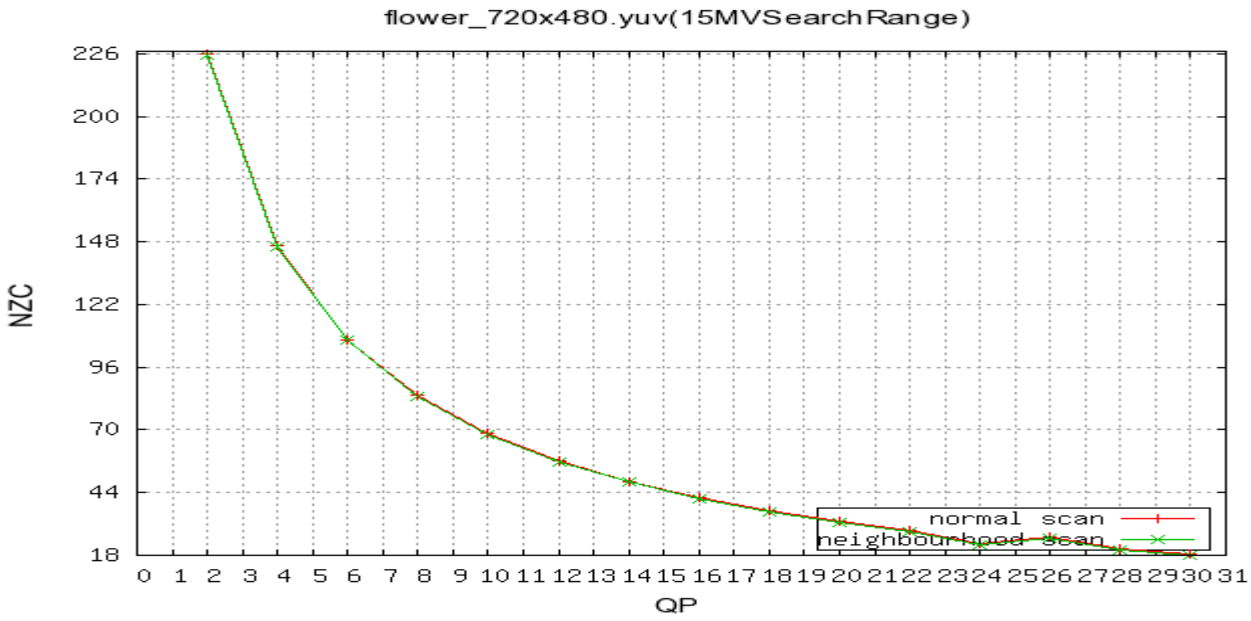
21. K.R. Rao and P. Yip, Discrete Cosine Transform: Algorithms, Advantages and Application, Academic Press, 1990, ISBNO-12-580203-X.

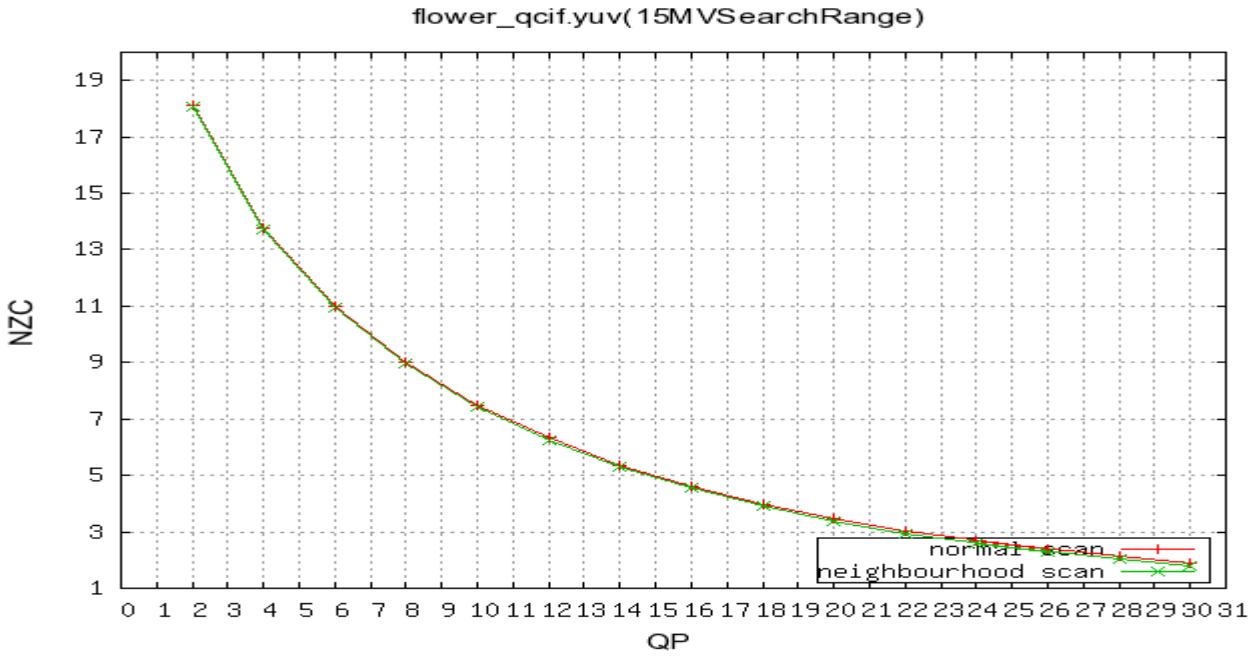
APPENDIX A: - PLOTS

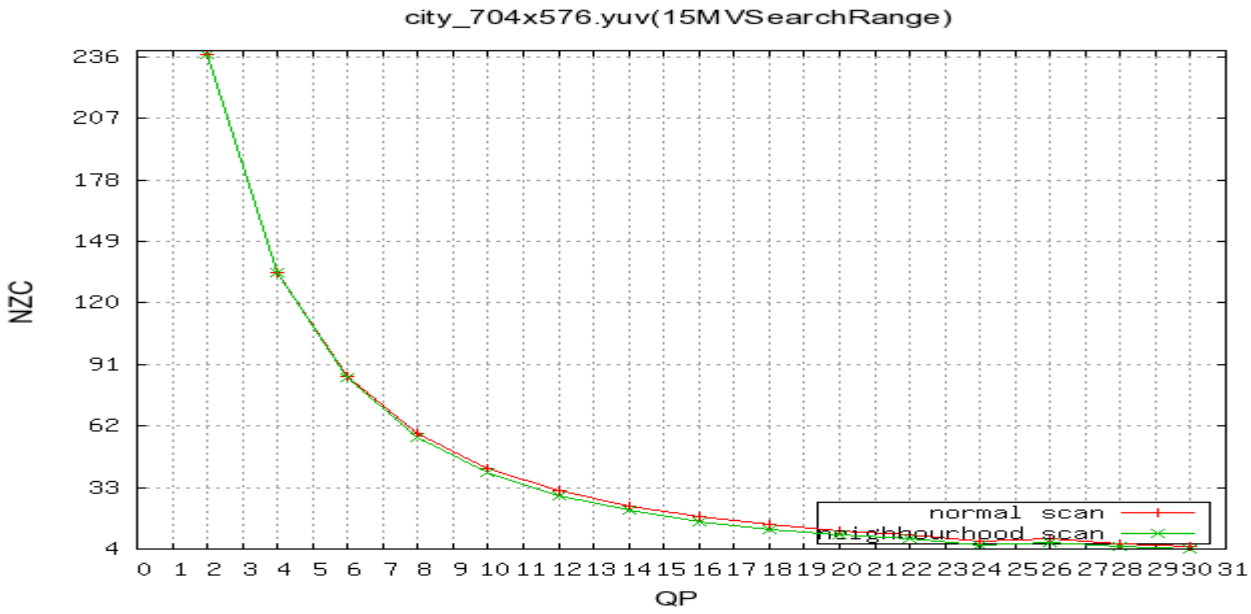
A.1 QP VS NZC (16X16 NEIGHBOURHOOD SCAN)

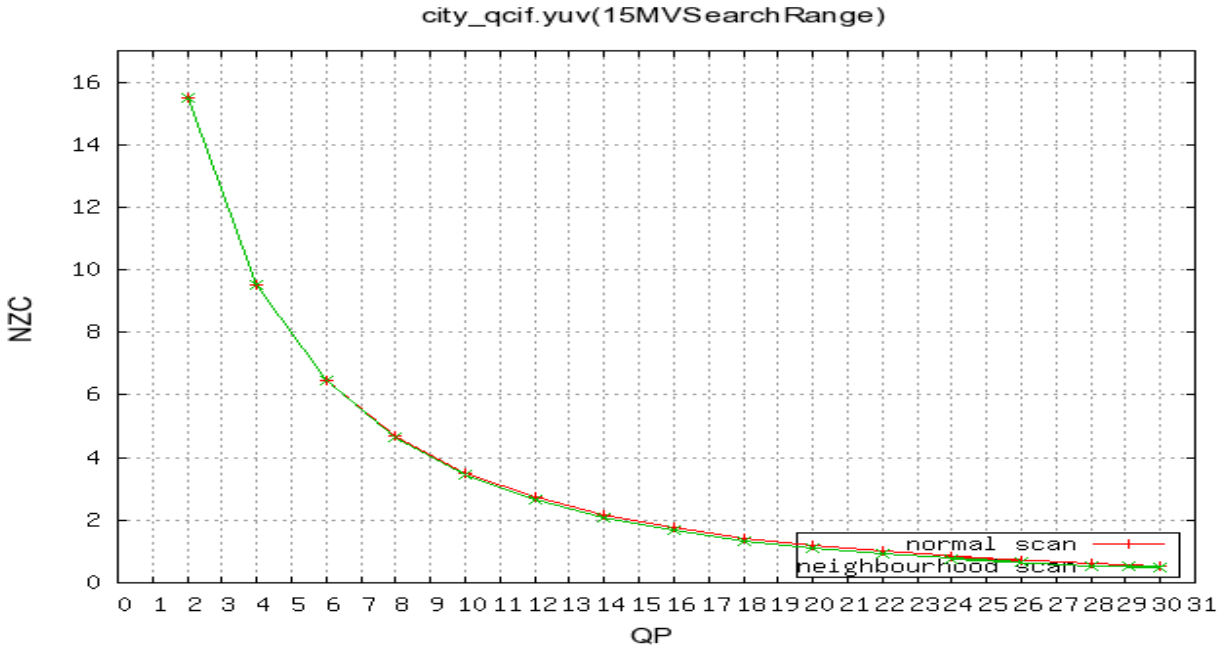


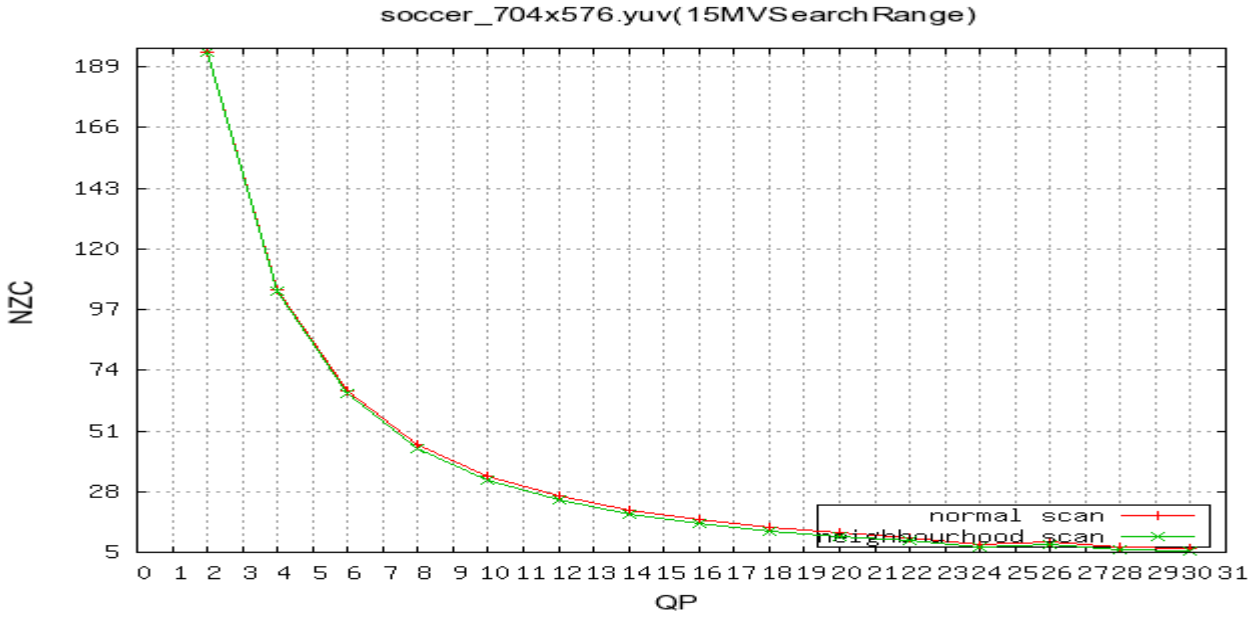


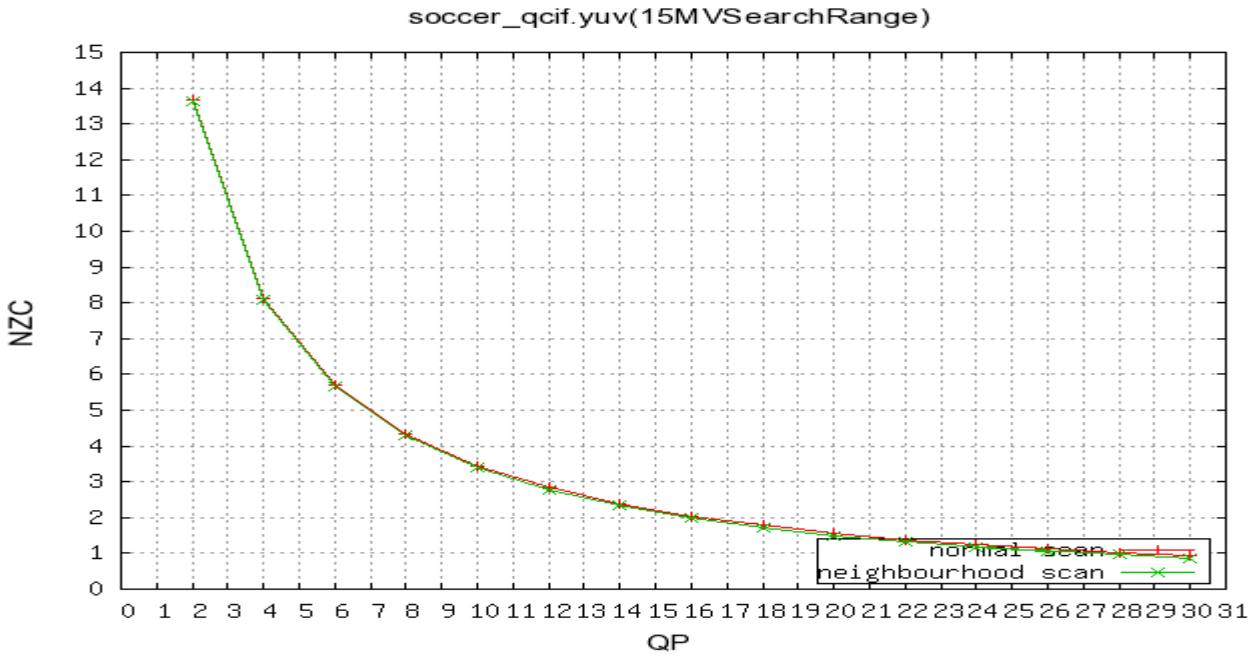

flower_720x480.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan


flower_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

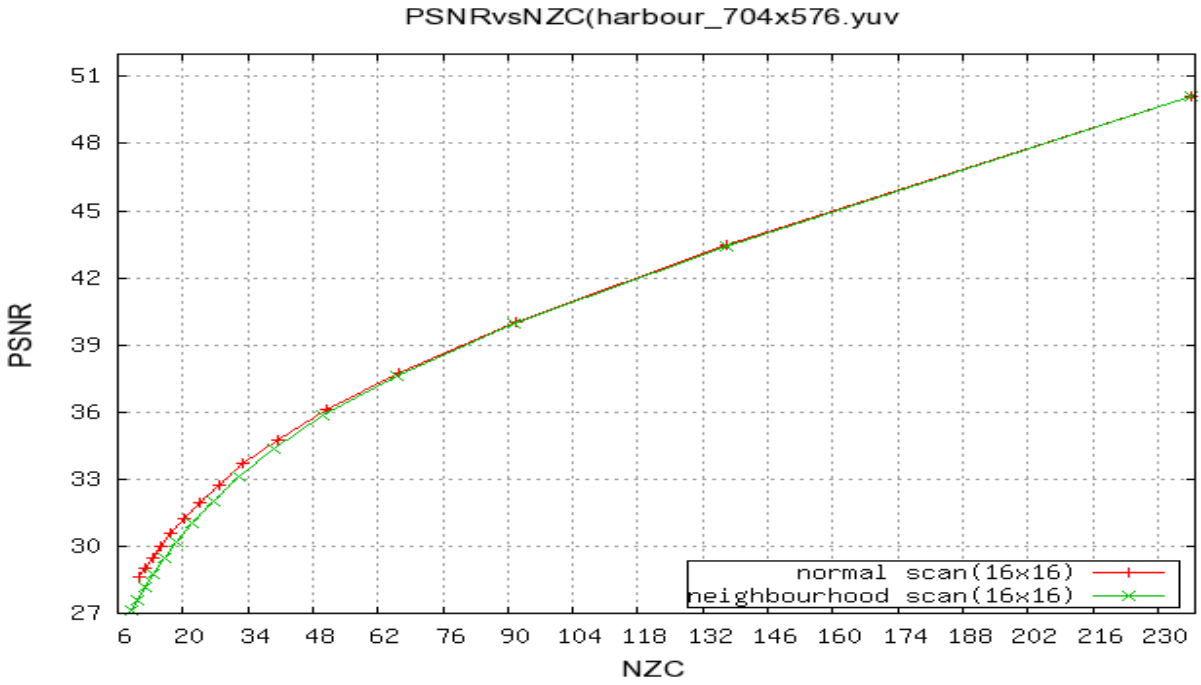

city_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

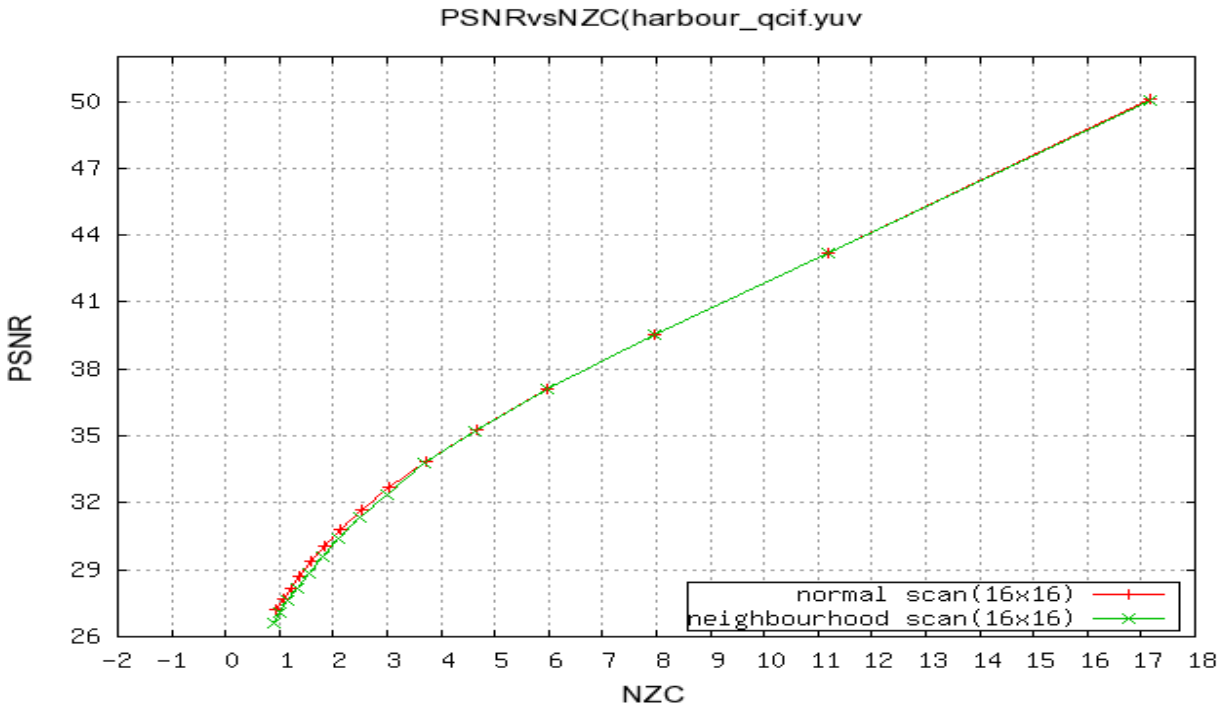

city_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

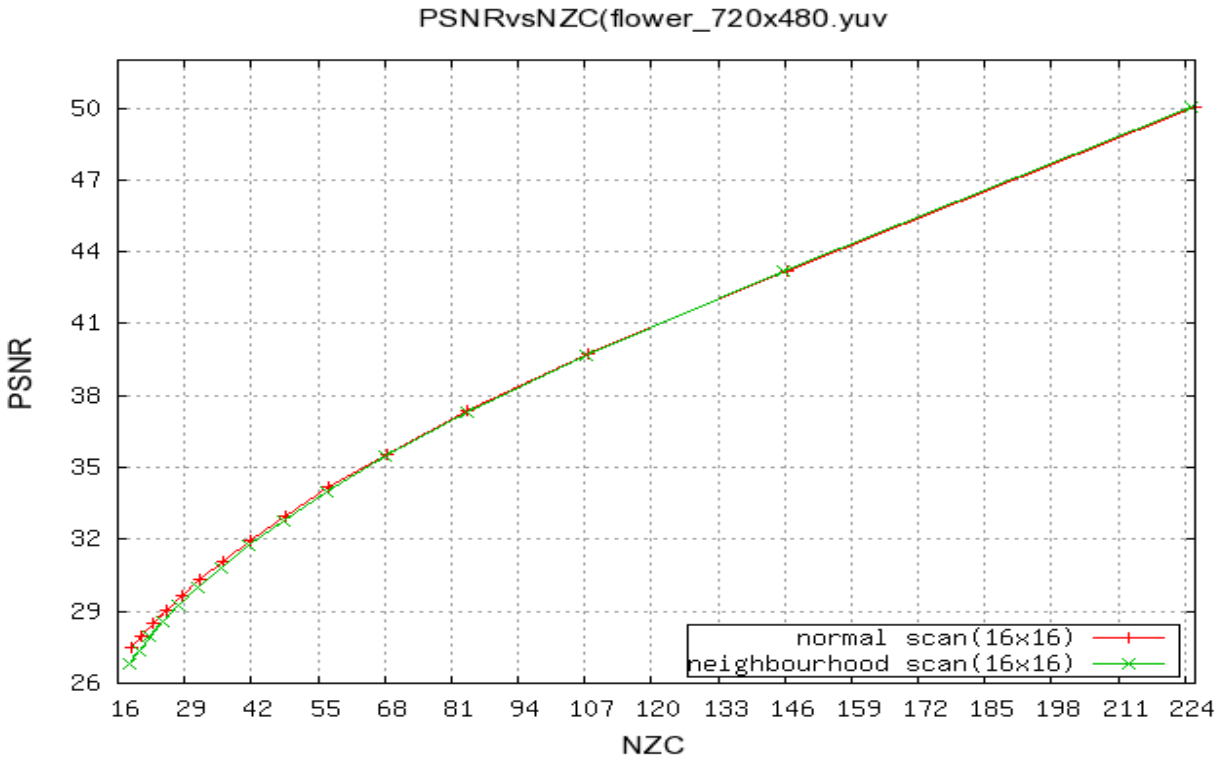

soccer_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

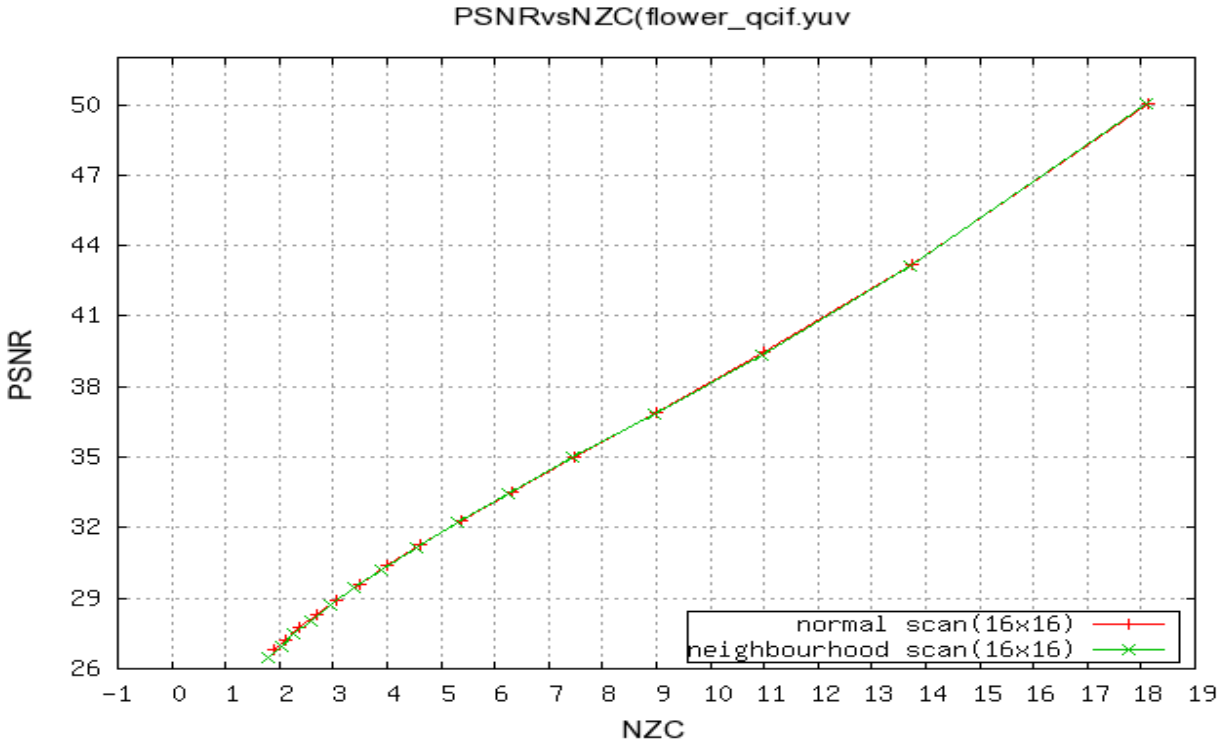

soccer_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

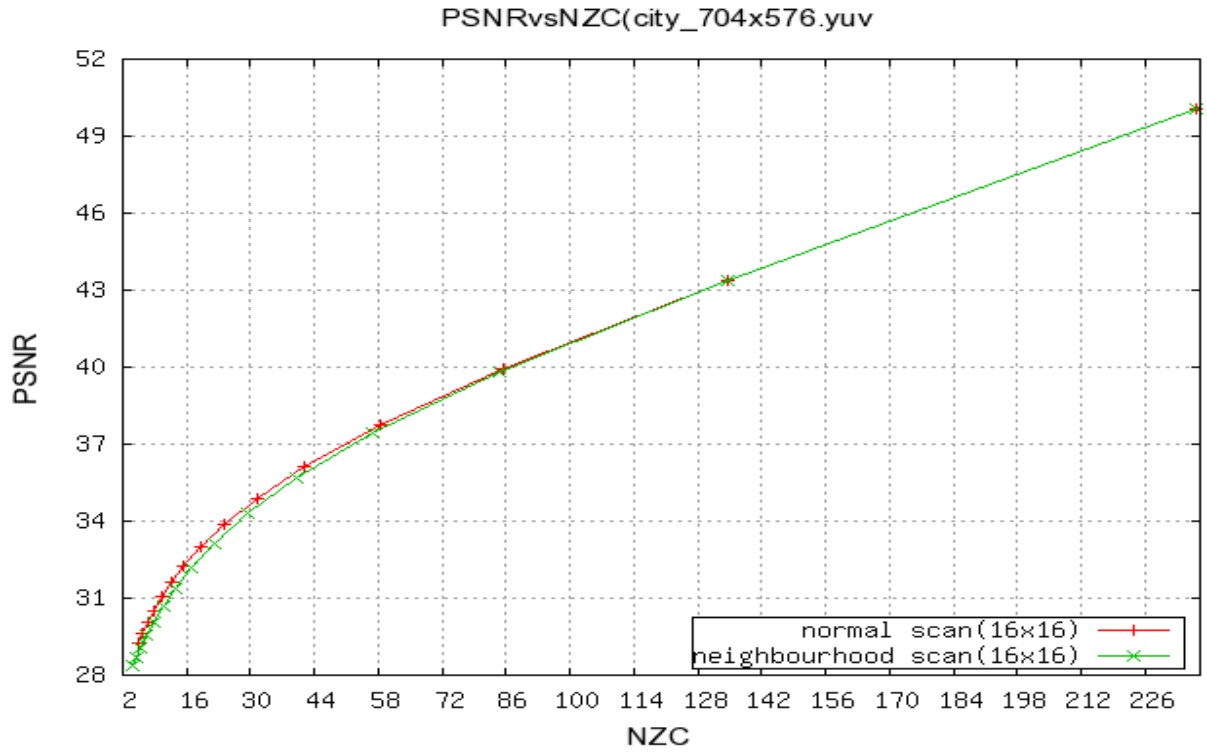
A.2 RD CURVES (16X16 NEIGHBOURHOOD SCAN)

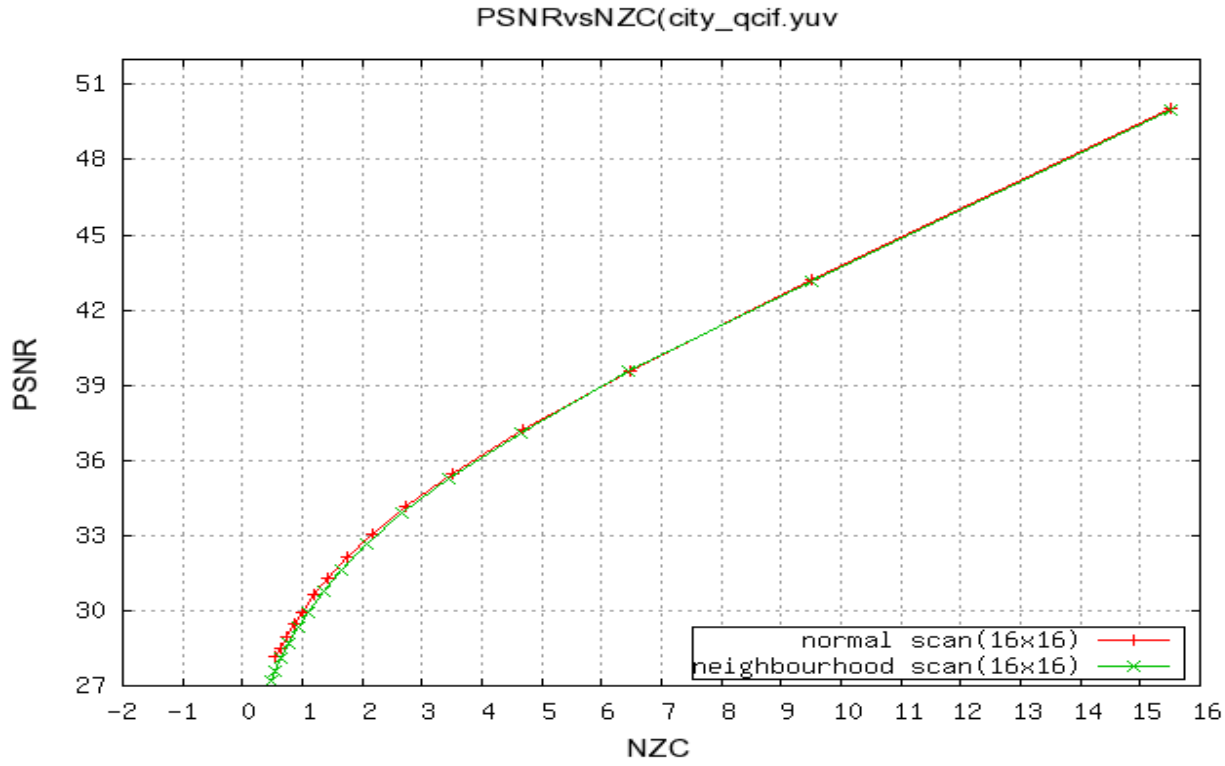


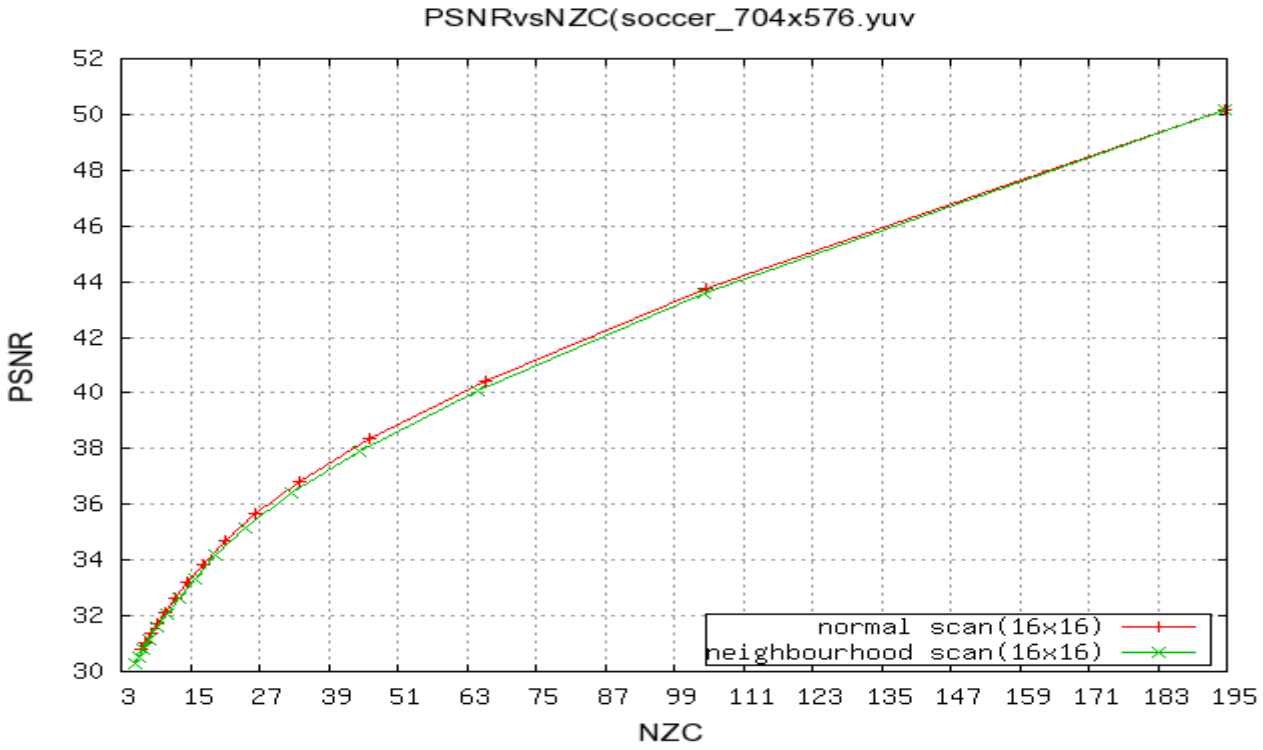


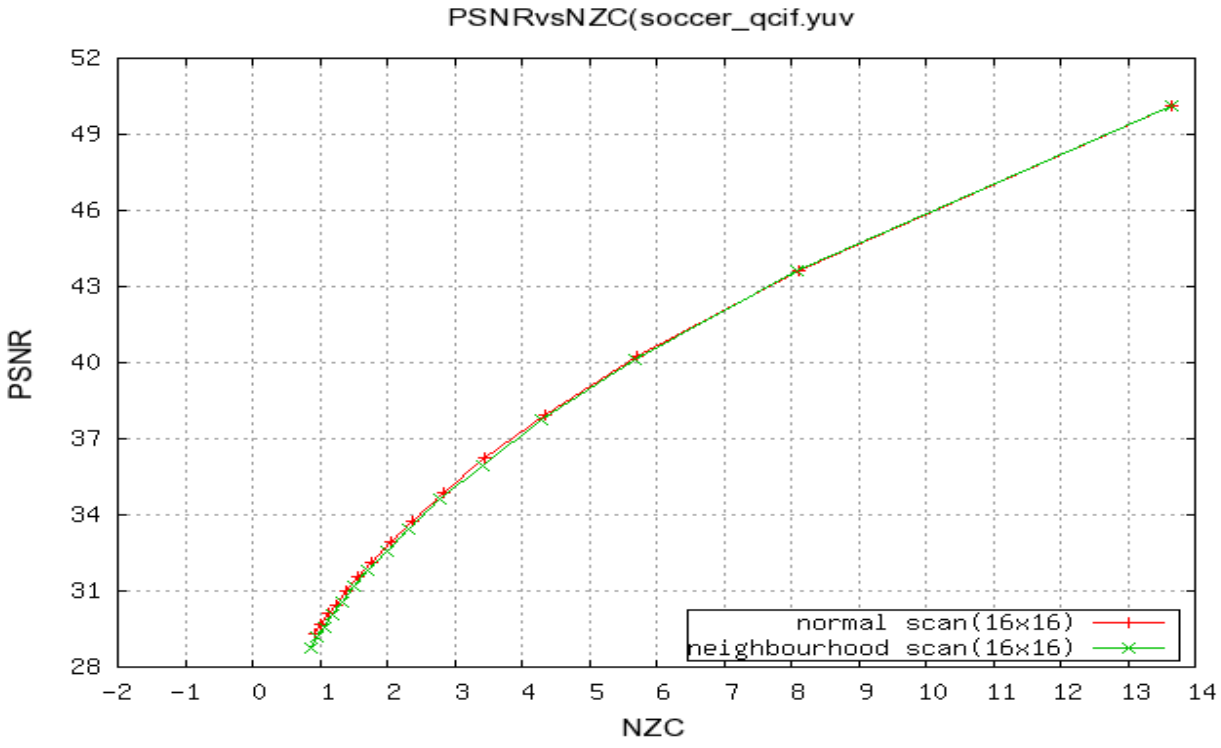

PSNRvsNZC(flower_720x480.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
50
47
44
41
38
35
32
29
26
16
29
42
55
68
81
94
107
120
133
146
159
172
185
198
211
224


PSNRvsNZC(flower_qcif.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
50
47
44
41
38
35
32
29
26
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

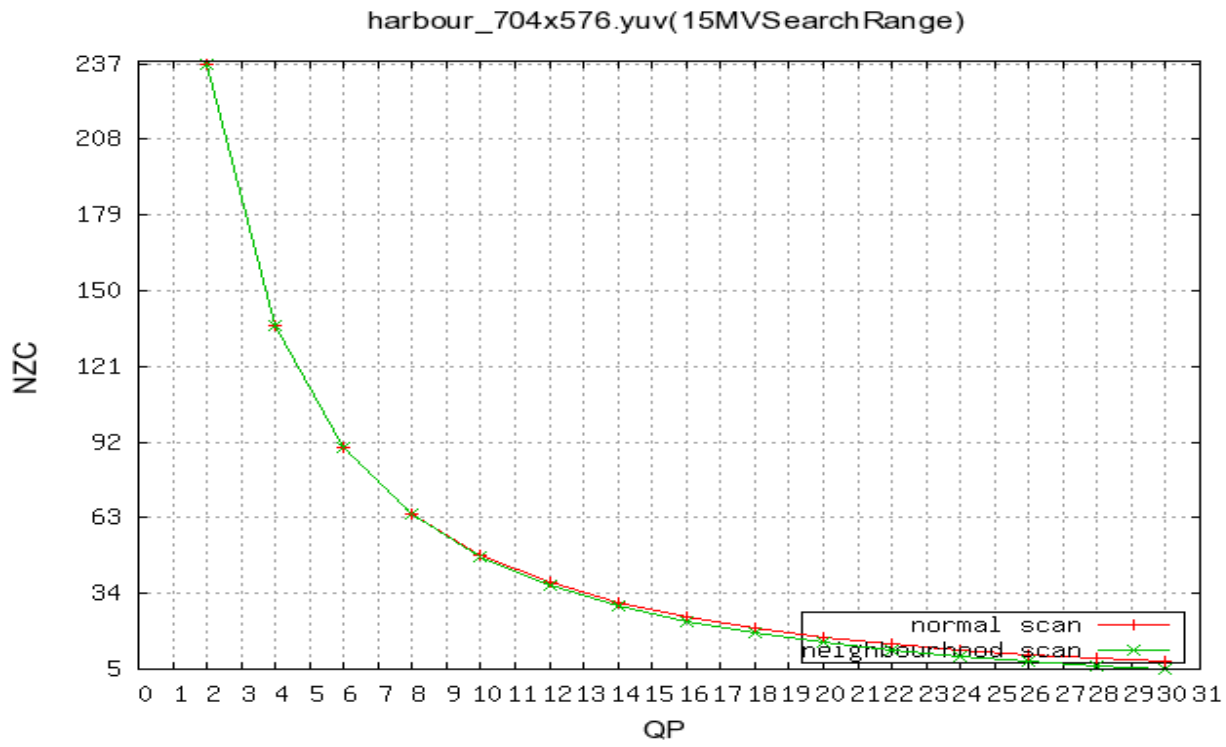

PSNRvsNZC(city_704x576.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
52
49
46
43
40
37
34
31
28
2
16
30
44
58
72
86
100
114
128
142
156
170
184
198
212
226

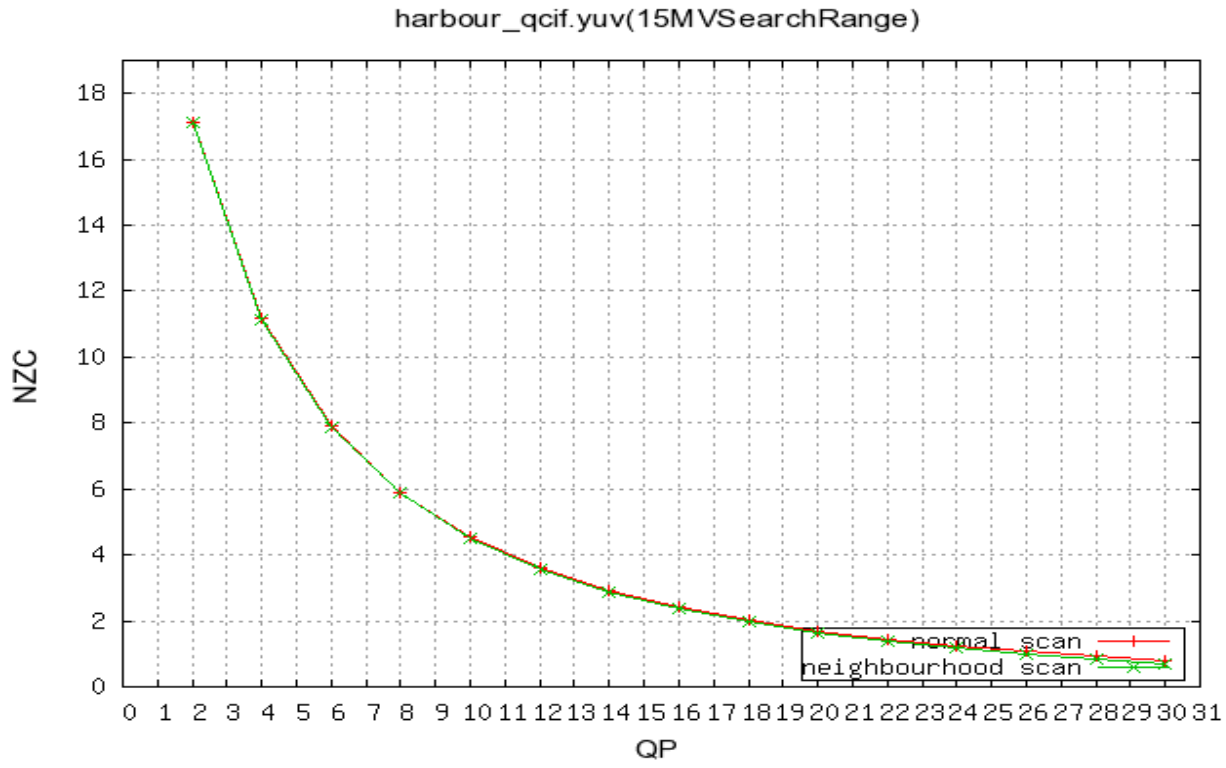

PSNRvsNZC(city_qcif.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
51
48
45
42
39
36
33
30
27
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

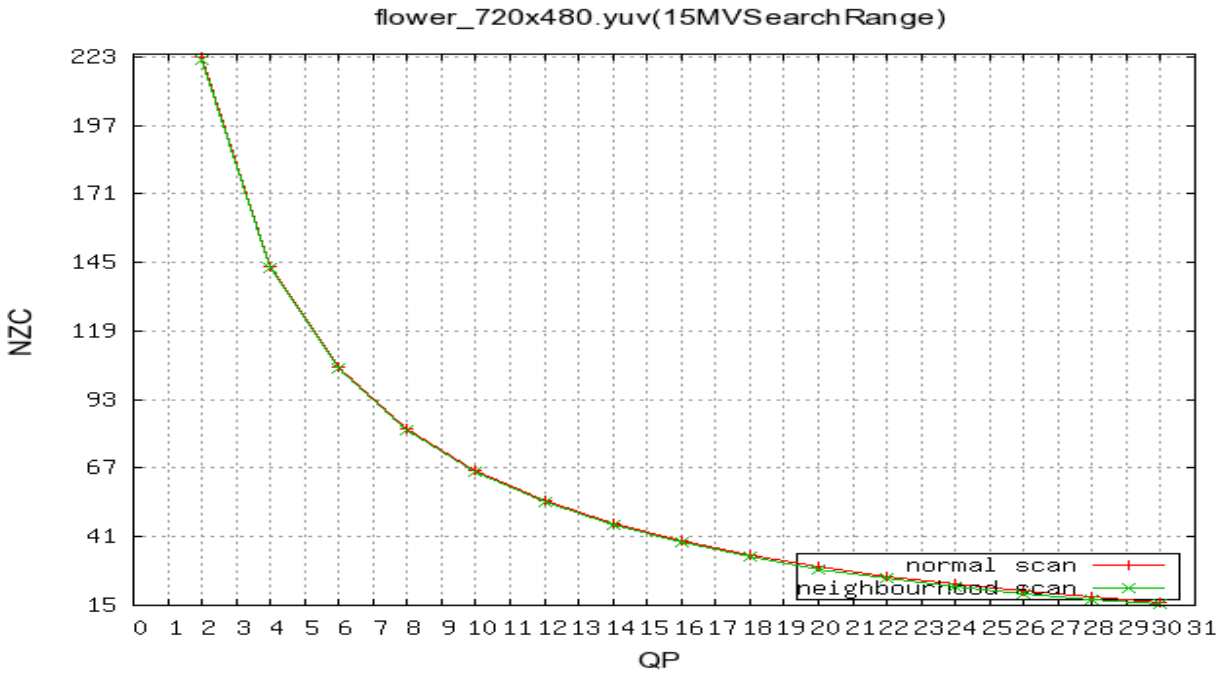

PSNRvsNZC(soccer_704x576.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
52
50
48
46
44
42
40
38
36
34
32
30
3
15
27
39
51
63
75
87
99
111
123
135
147
159
171
183
195

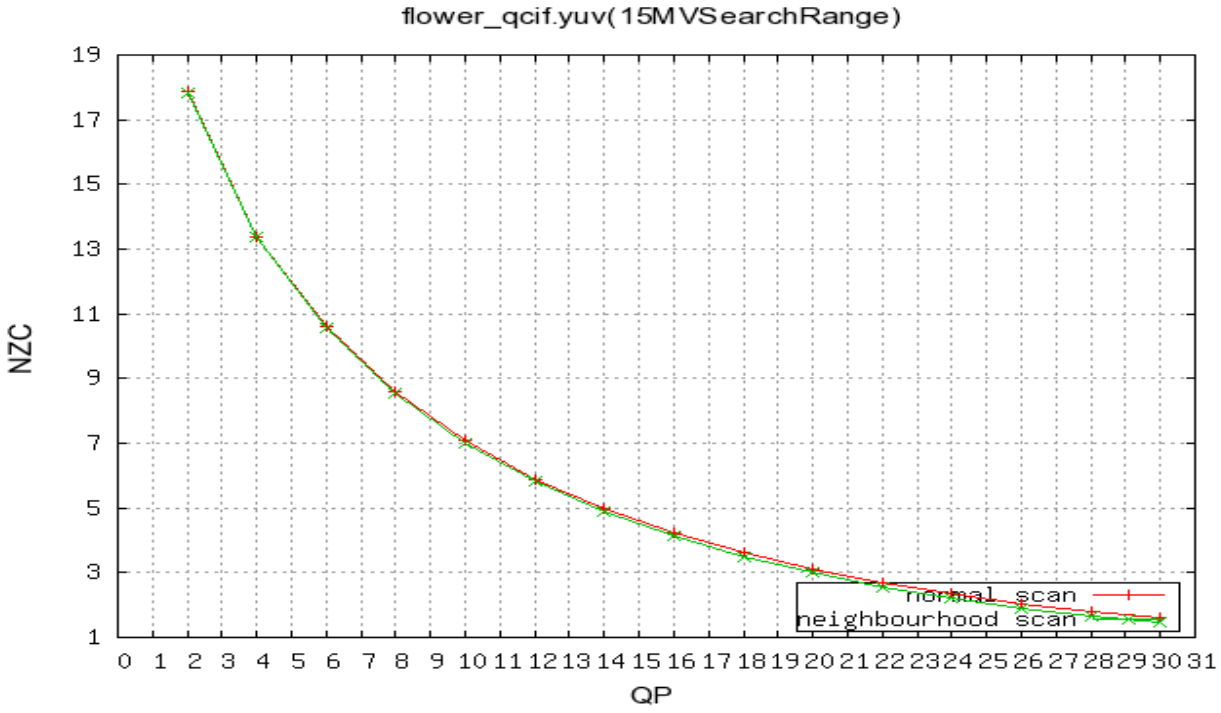

PSNRvsNZC(soccer_qcif.yuv
PSNR
NZC
normal scan(16x16)
neighbourhood scan(16x16)
52
49
46
43
40
37
34
31
28
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

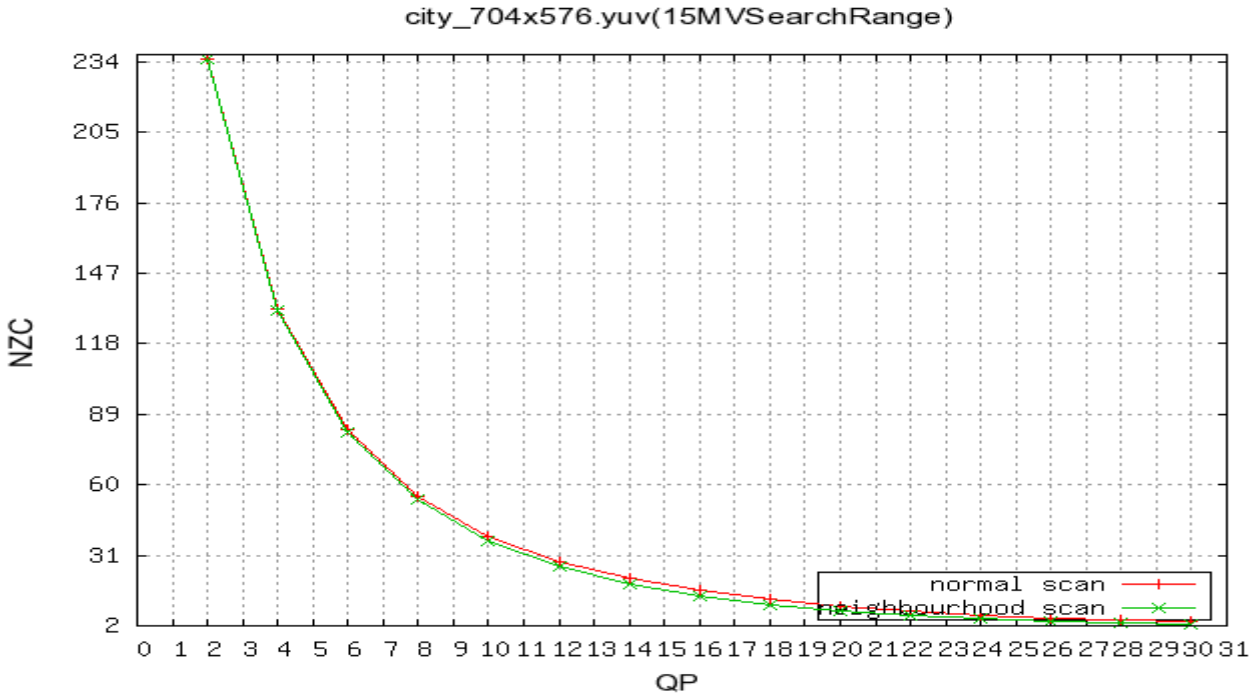
A.3 QP VS NZC (FOR 8X8 NEIGHBOURHOOD SCAN)

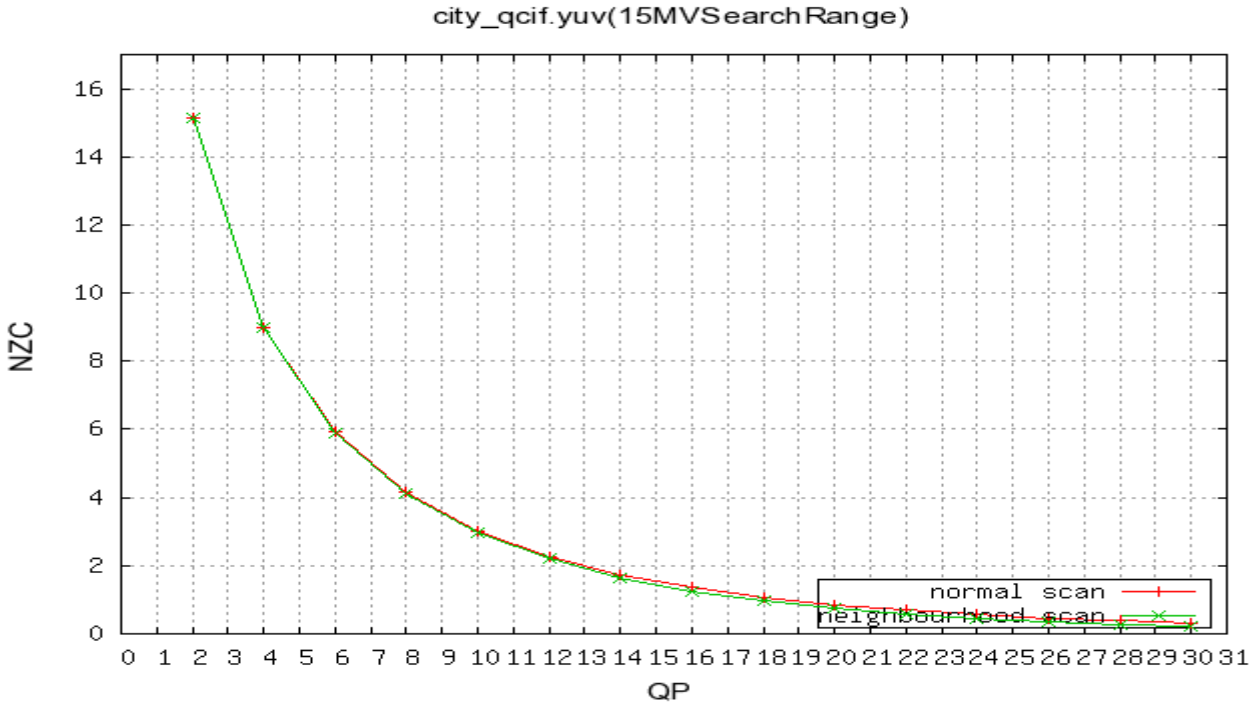


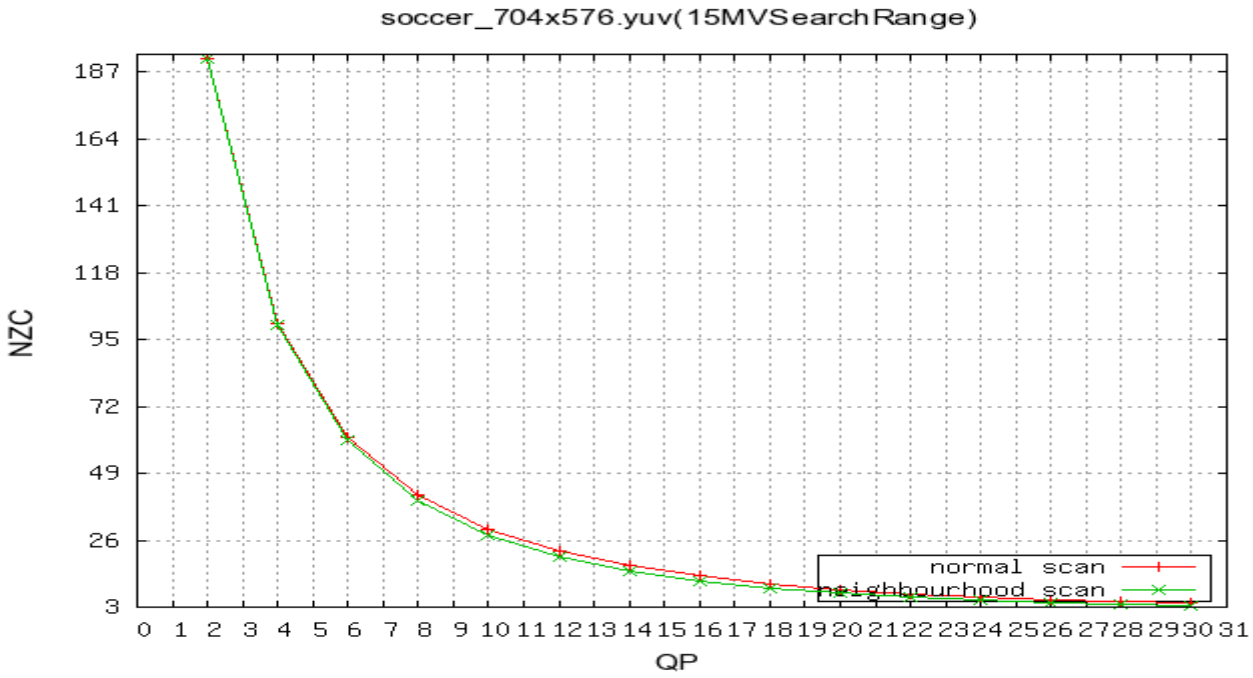


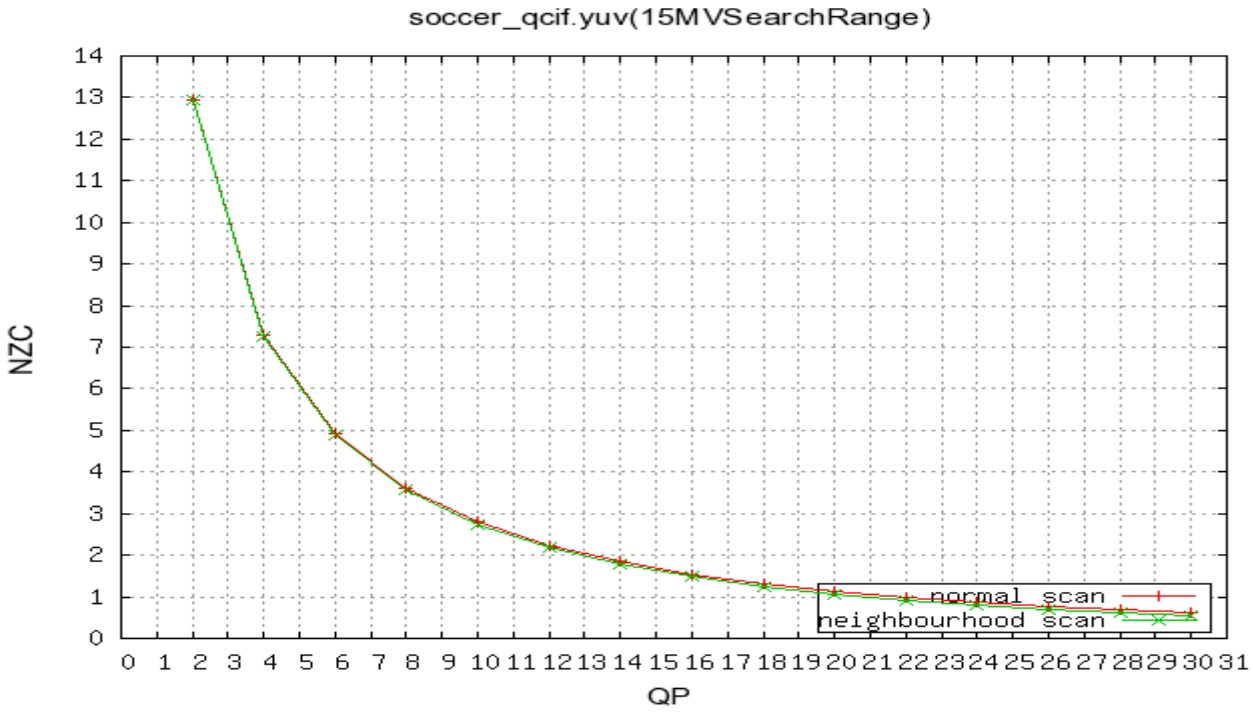

flower_720x480.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan


flower_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

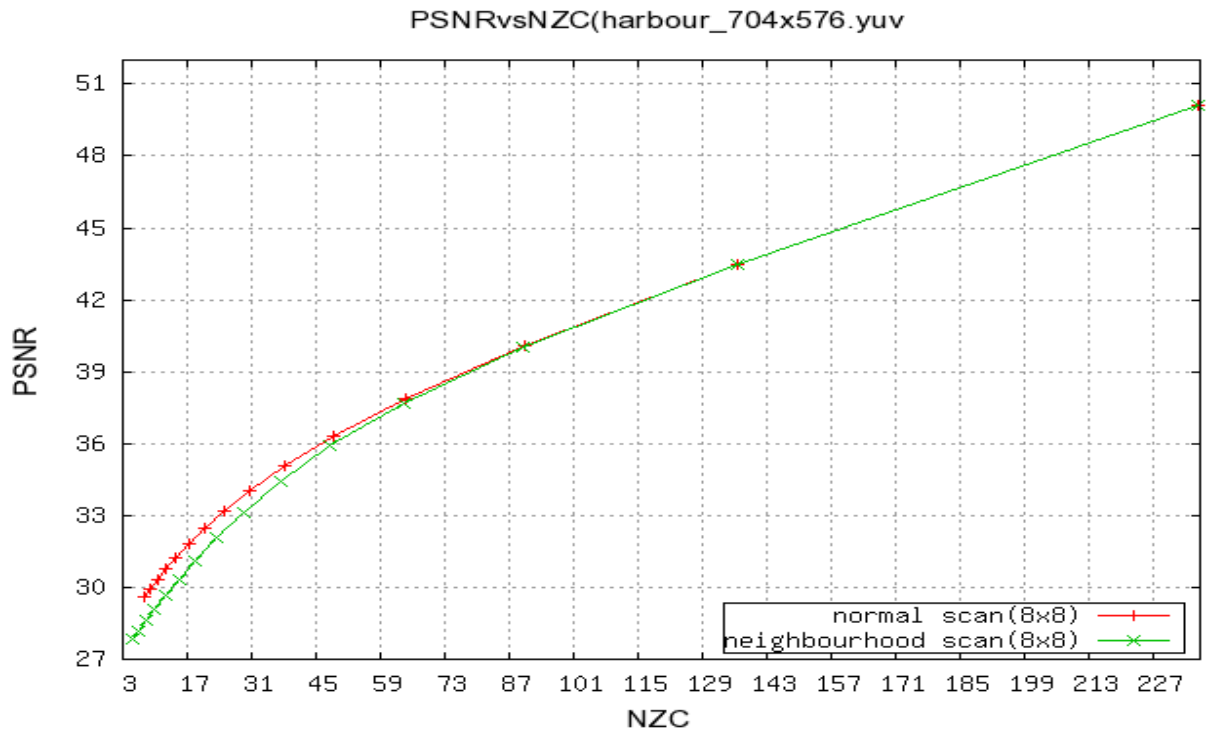

city_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

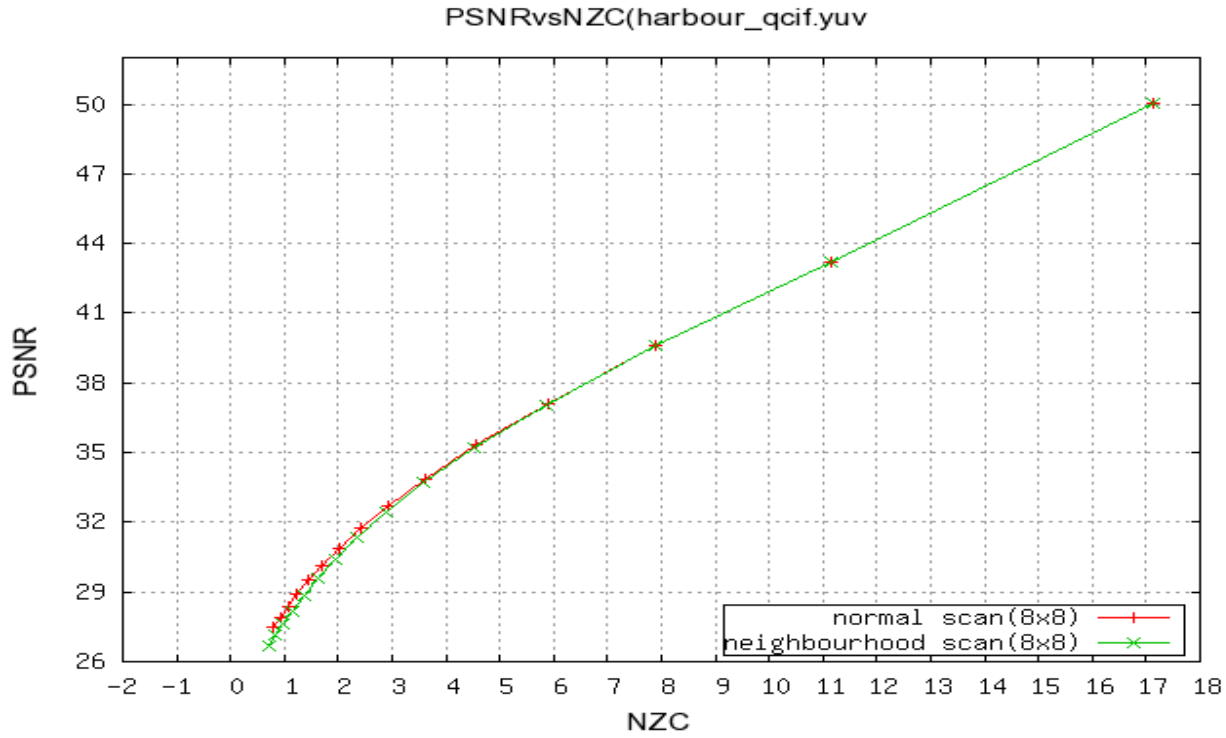

city_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

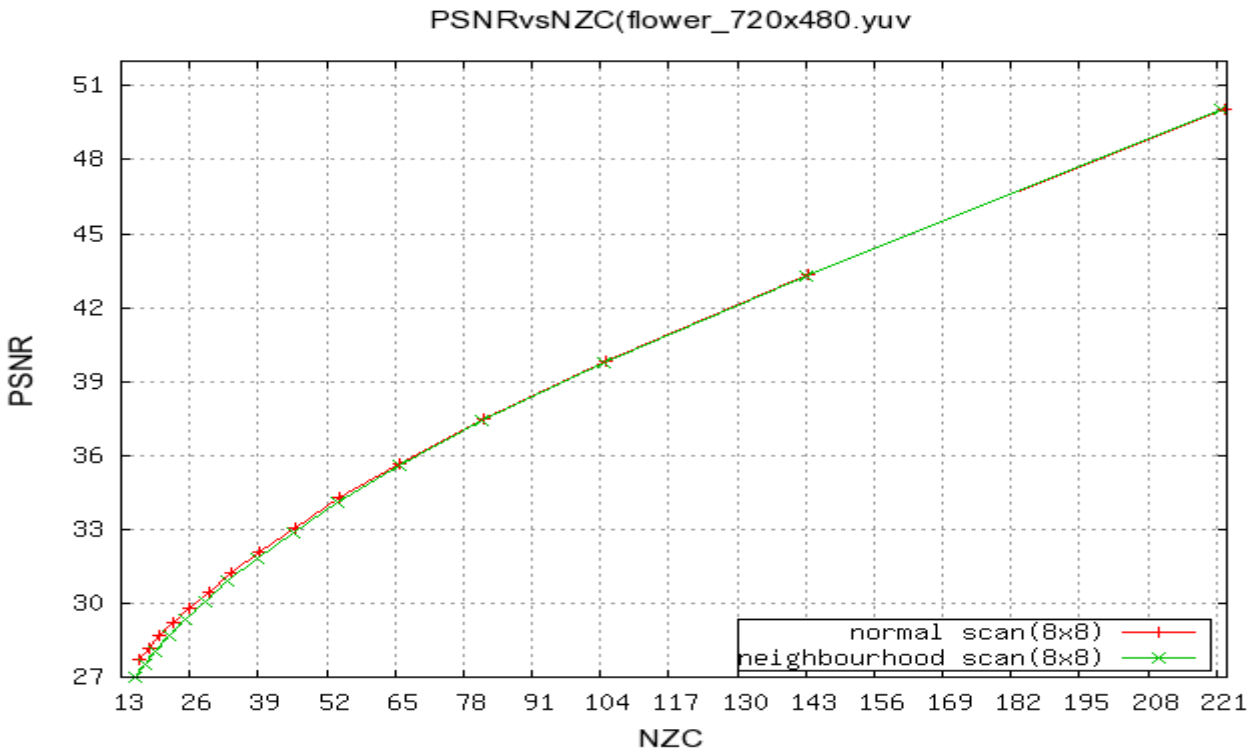

soccer_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

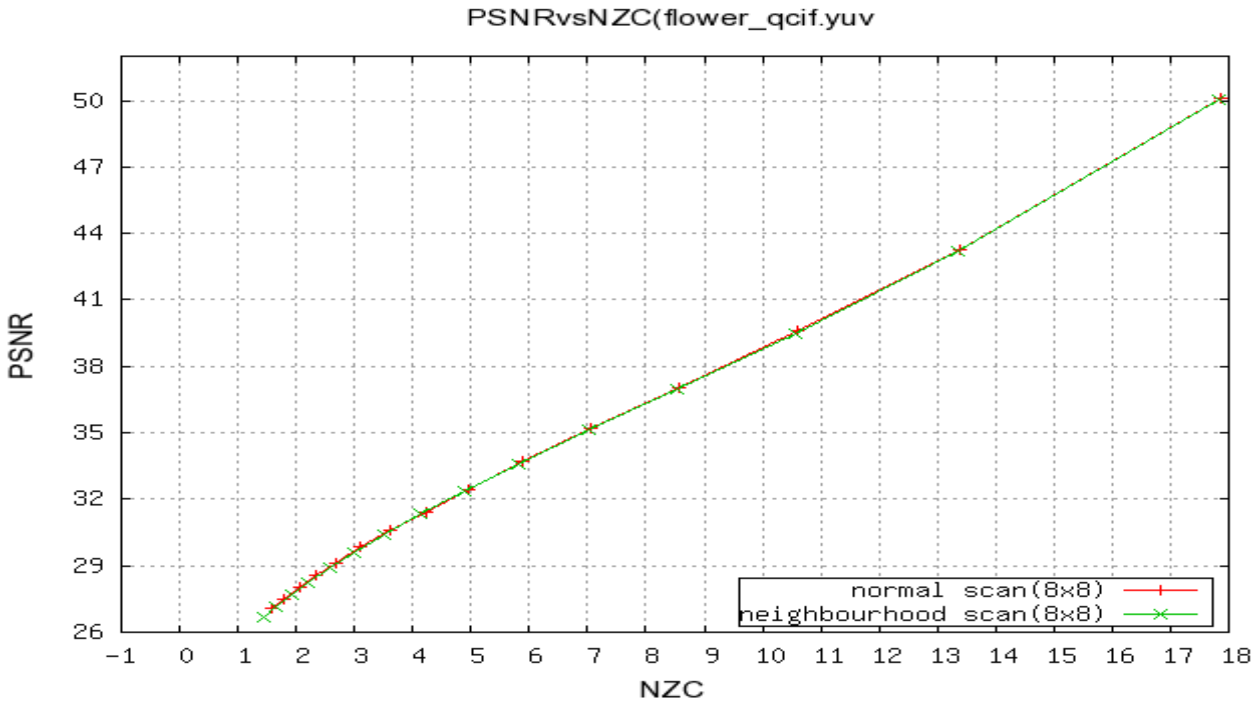

soccer_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

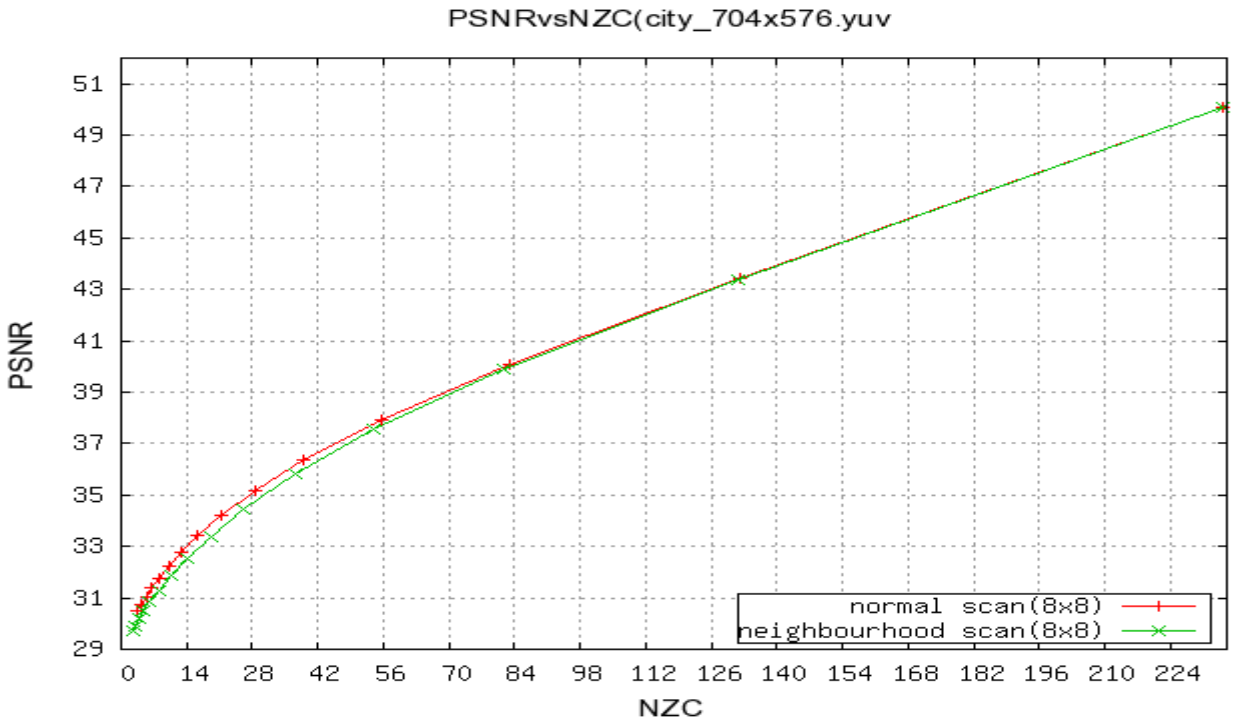
A.4 RD CURVES (8X8 NEIGHBOURHOOD SCAN)

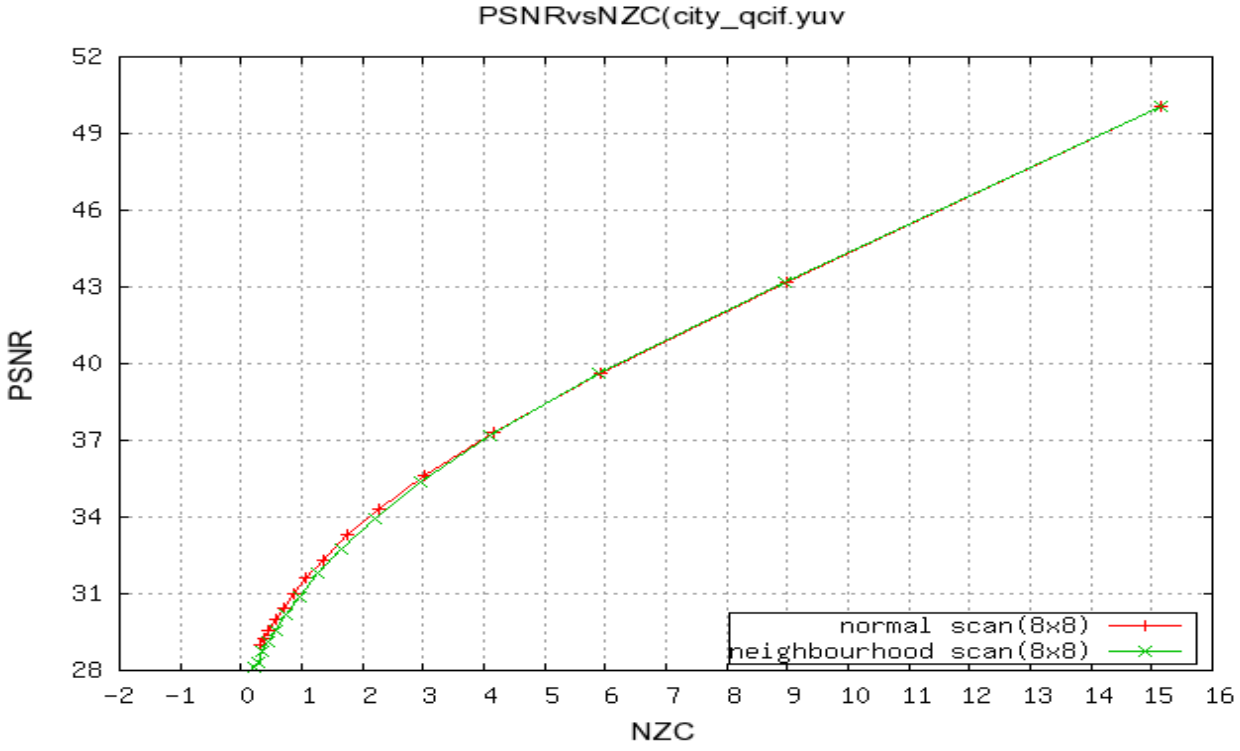


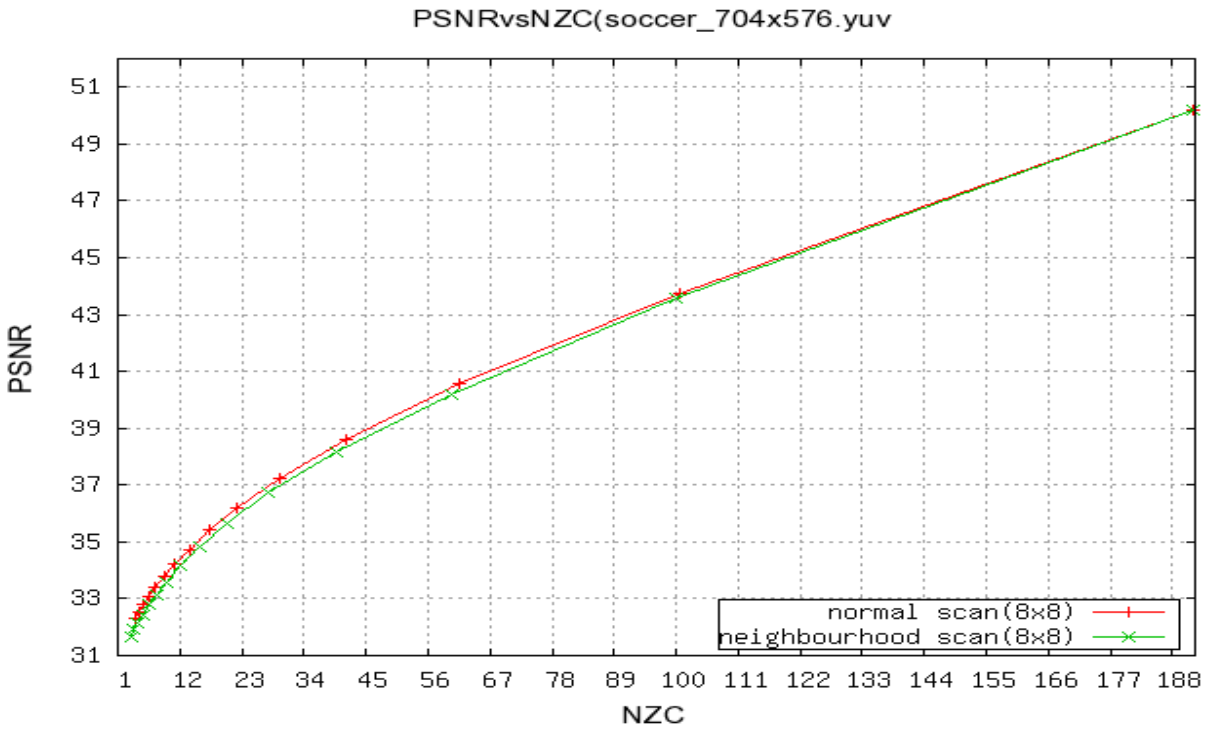


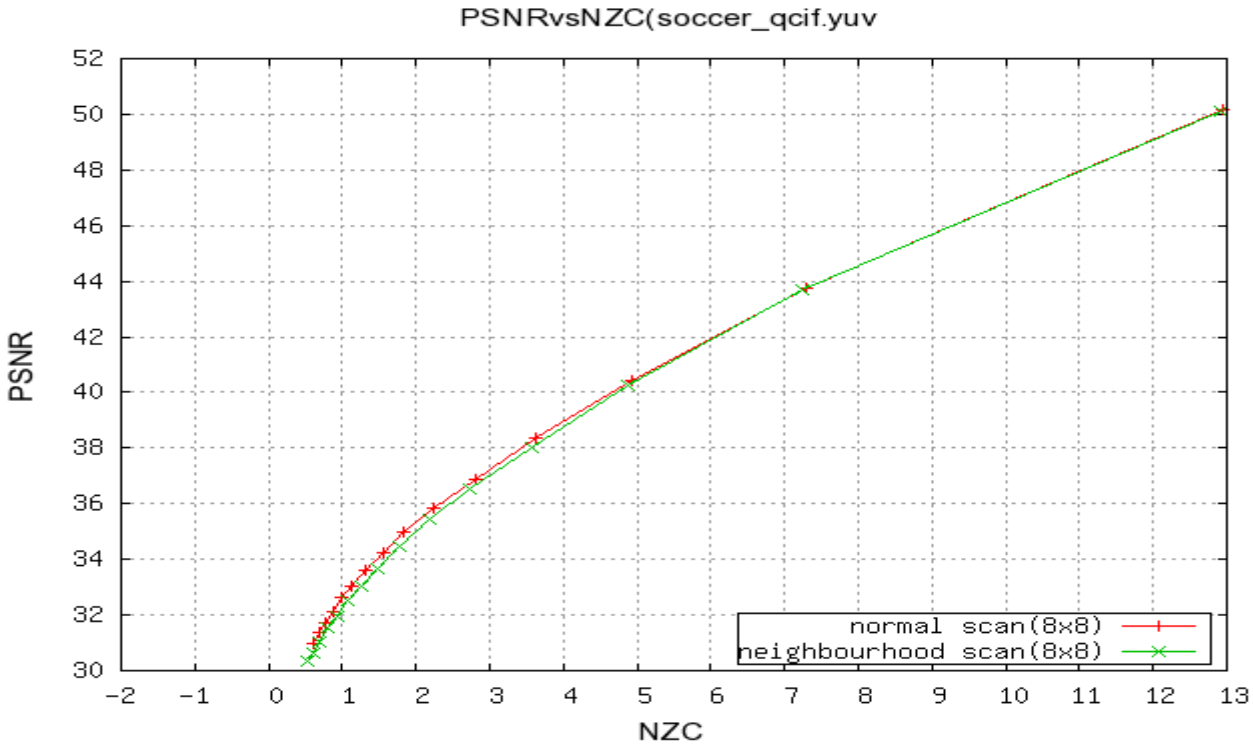

PSNRvsNZC(flower_720x480.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
51
48
45
42
39
36
33
30
27
13
26
39
52
65
78
91
104
117
130
143
156
169
182
195
208
221


PSNRvsNZC(flower_qcif.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
50
47
44
41
38
35
32
29
26
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

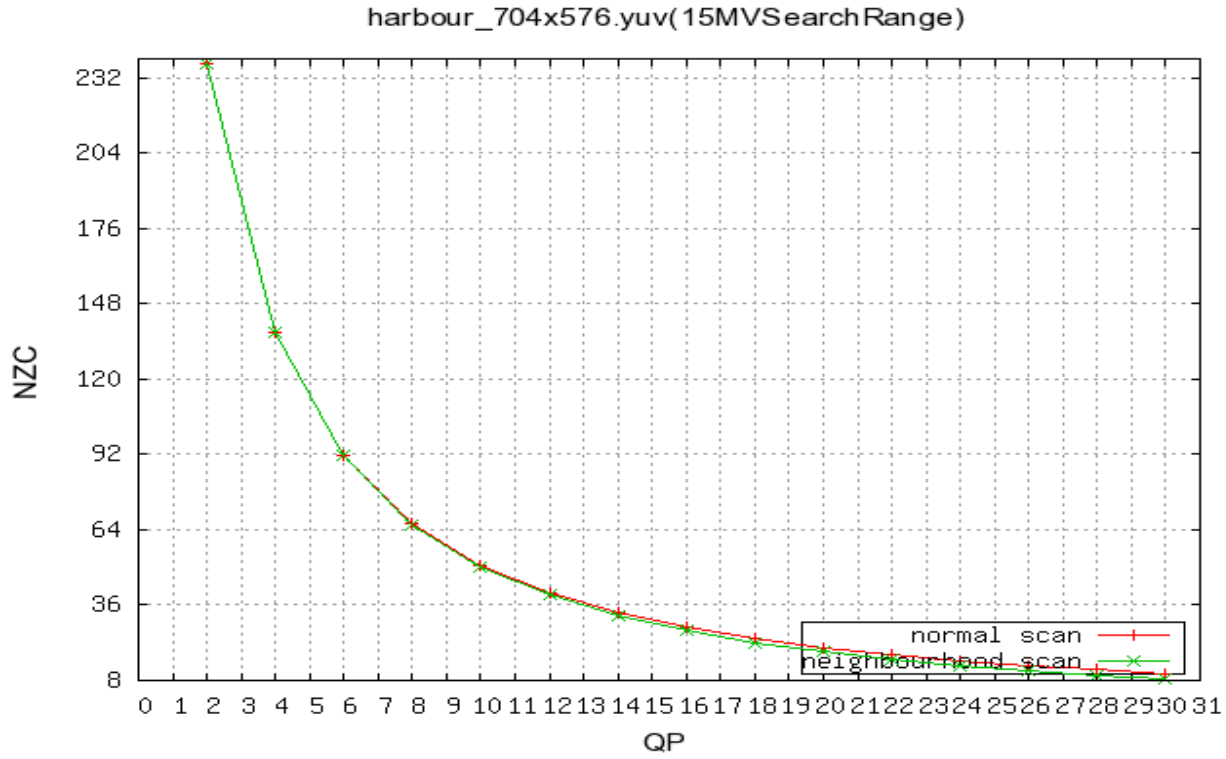

PSNRvsNZC(city_704x576.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
51
49
47
45
43
41
39
37
35
33
31
29
0
14
28
42
56
70
84
98
112
126
140
154
168
182
196
210
224

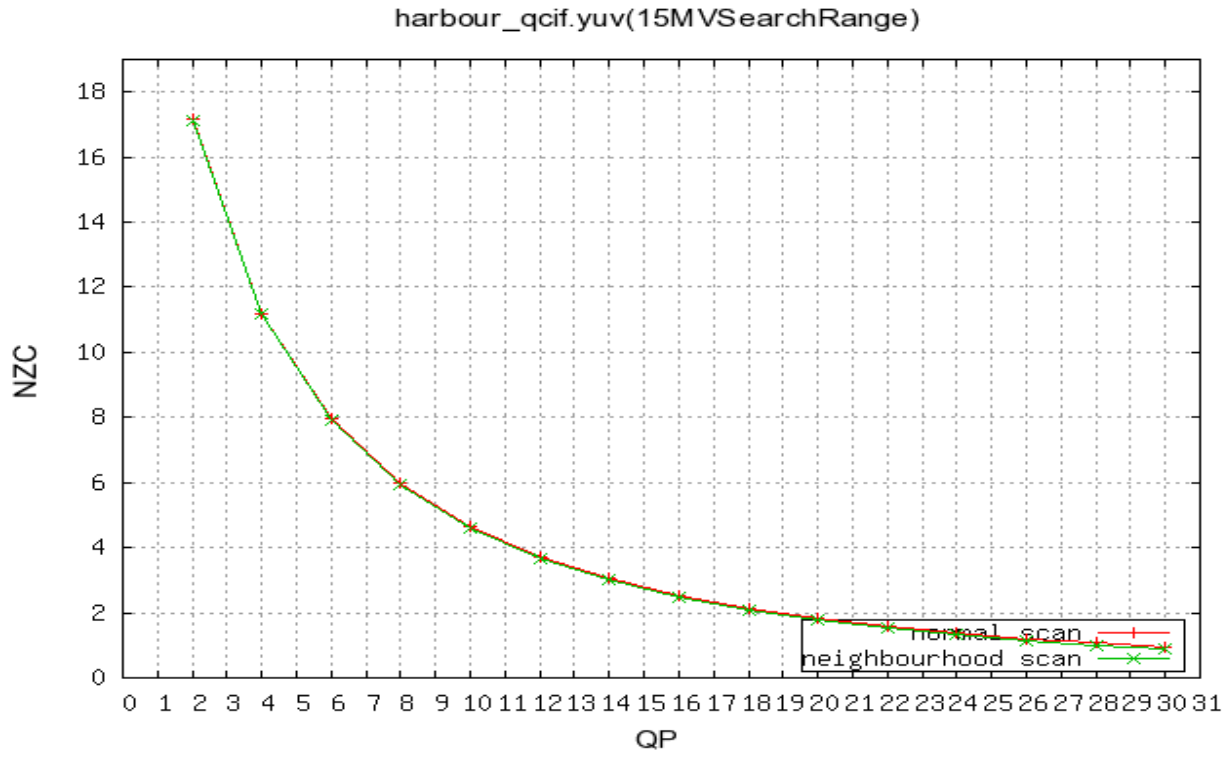

PSNRvsNZC(city_qcif.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
52
49
46
43
40
37
34
31
28
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

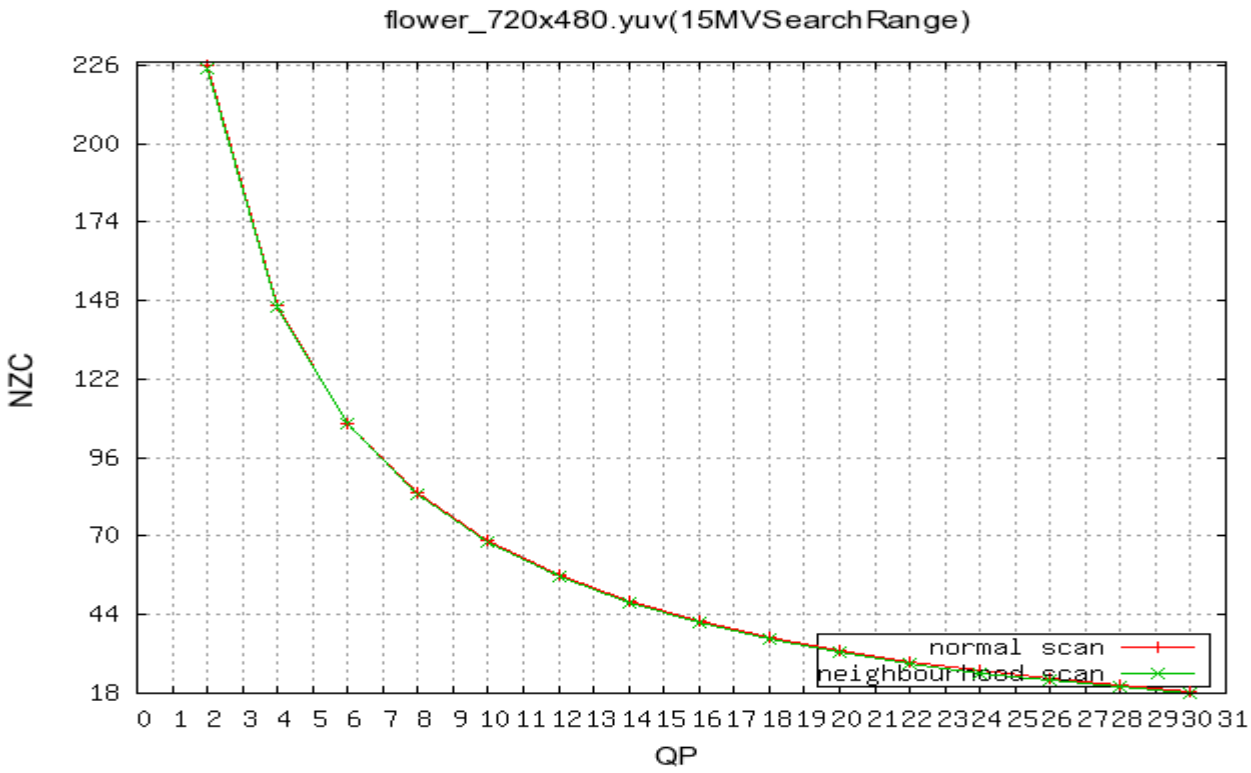

PSNRvsNZC(soccer_704x576.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
51
49
47
45
43
41
39
37
35
33
31
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188

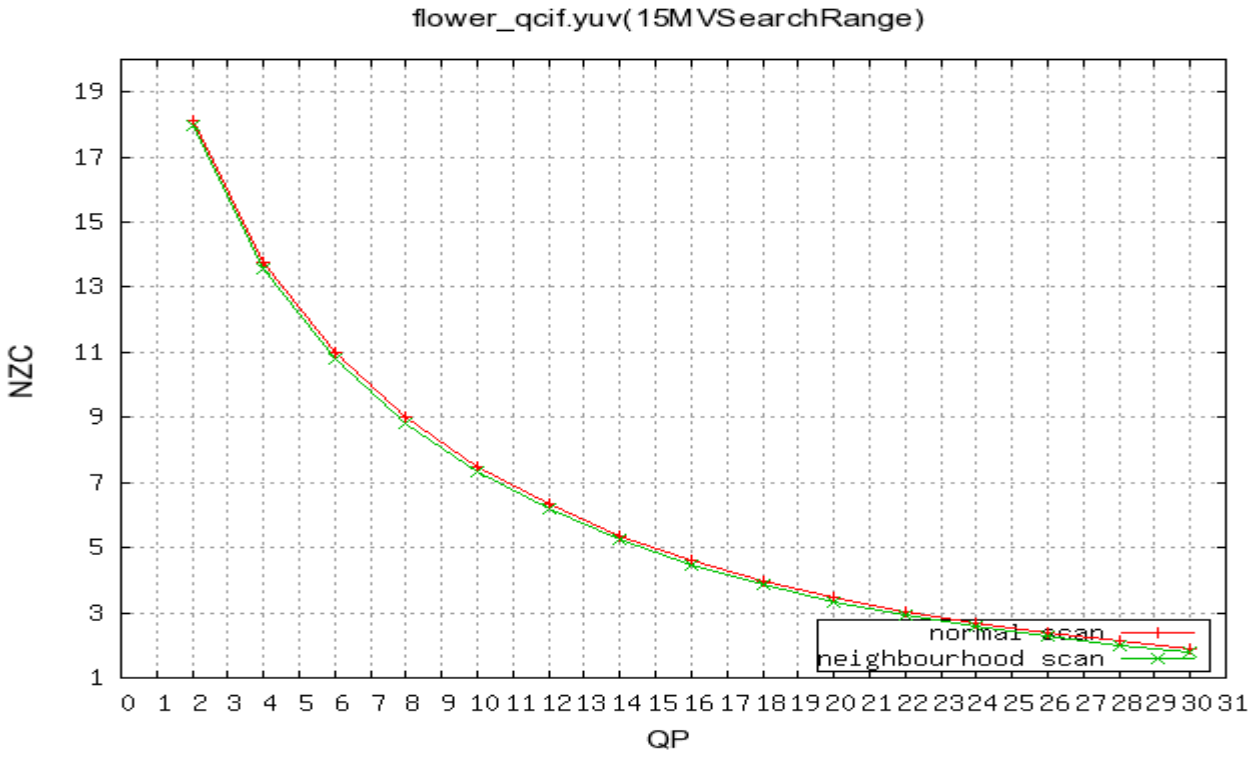

PSNRvsNZC(soccer_qcif.yuv
PSNR
NZC
normal scan(8x8)
neighbourhood scan(8x8)
52
50
48
46
44
42
40
38
36
34
32
30
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13

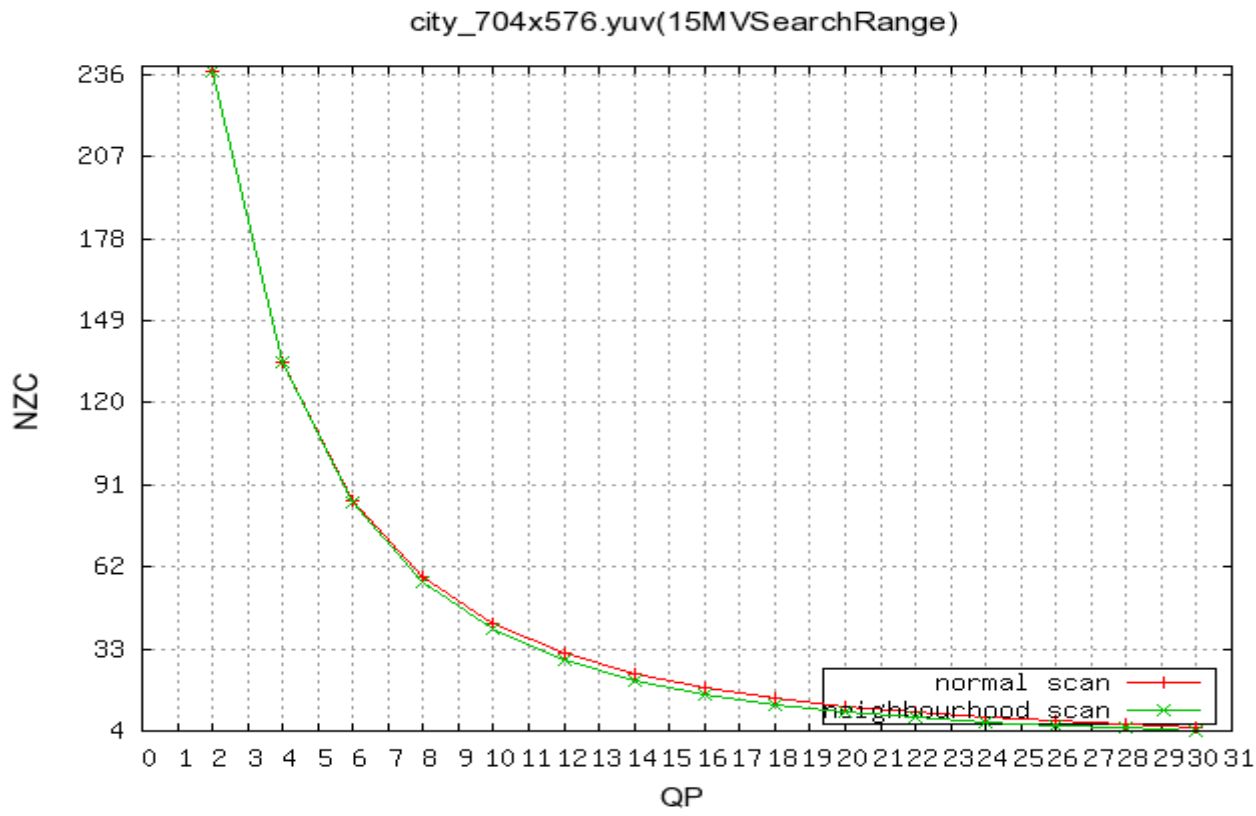
A.5 QP VS NZC (16X 16 PREV-NEIGHBOURHOODS SCAN)

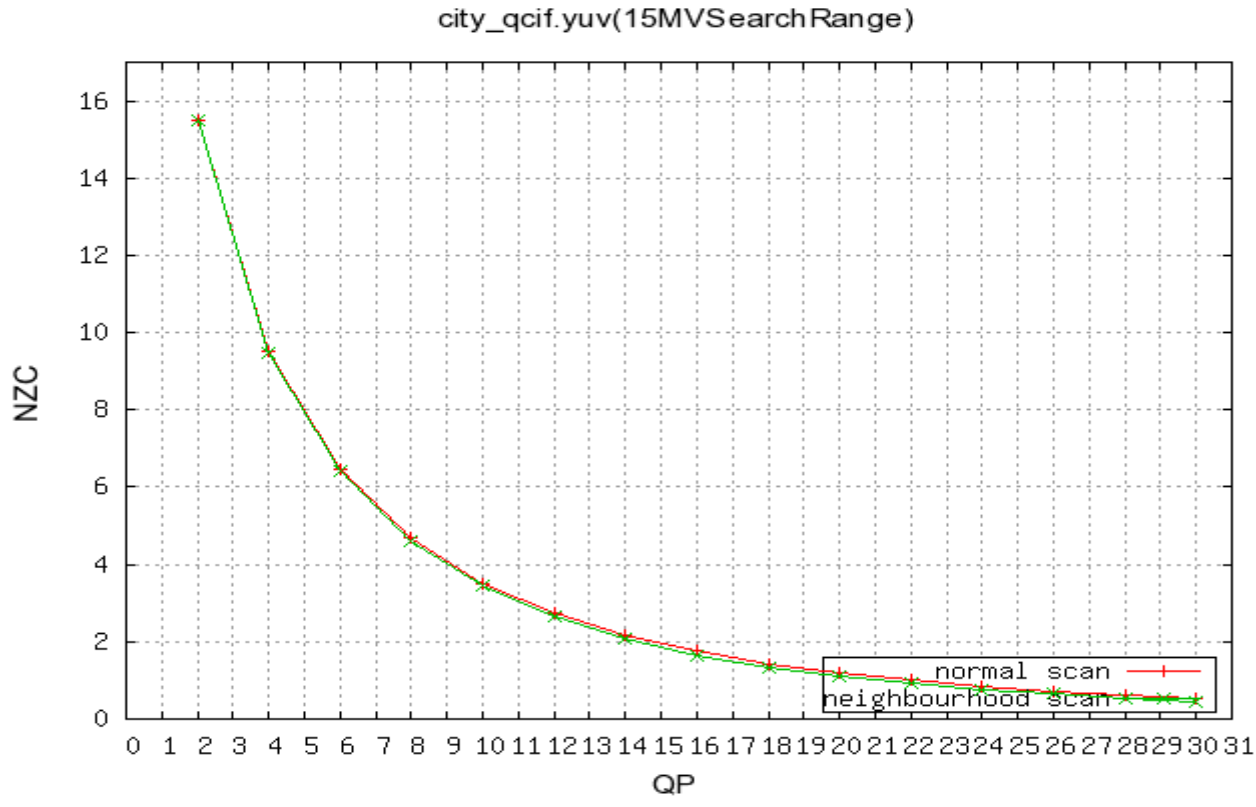


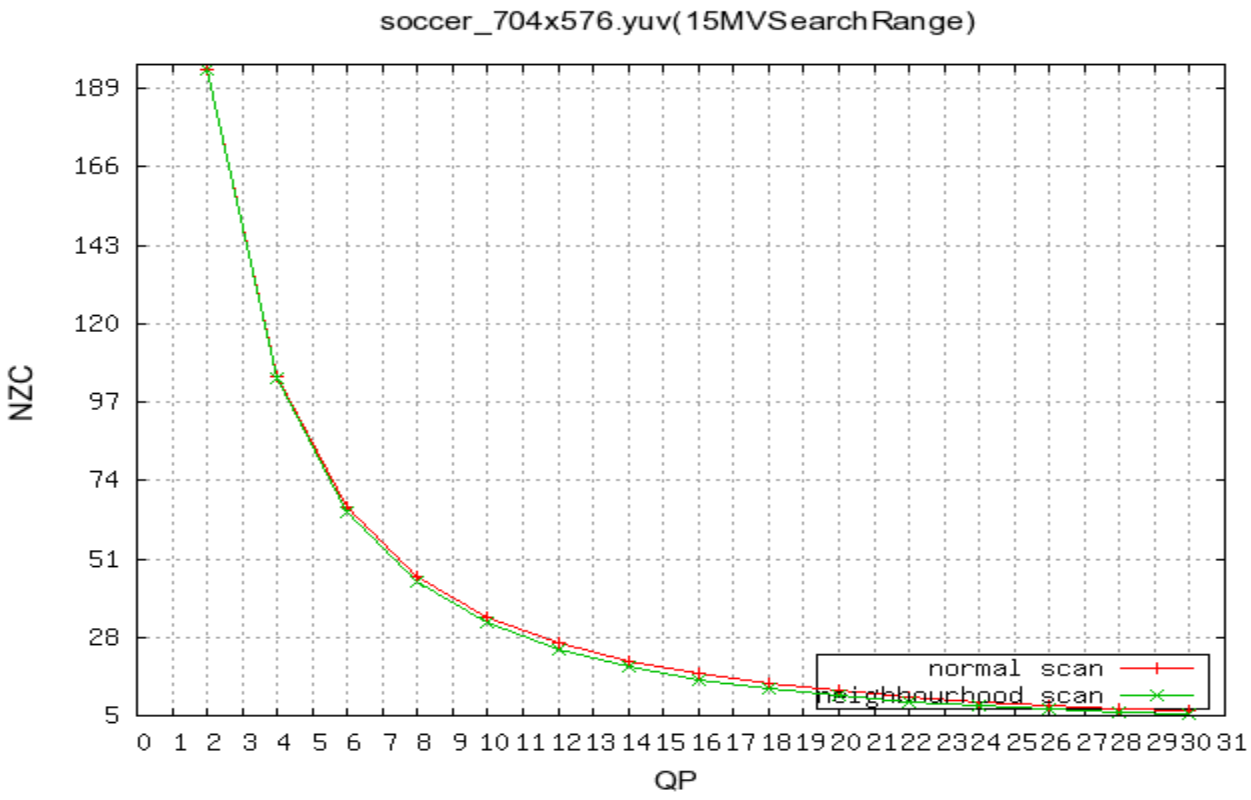


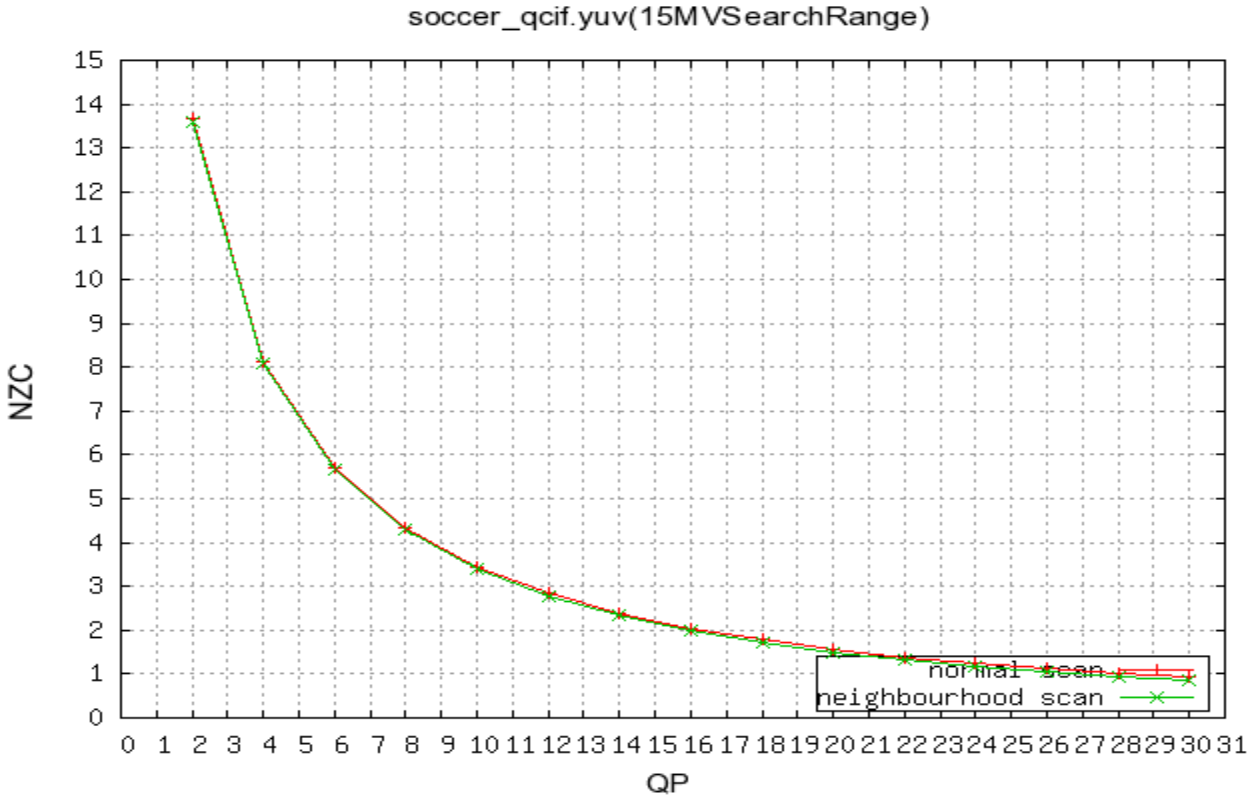

flower_720x480.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan


flower_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

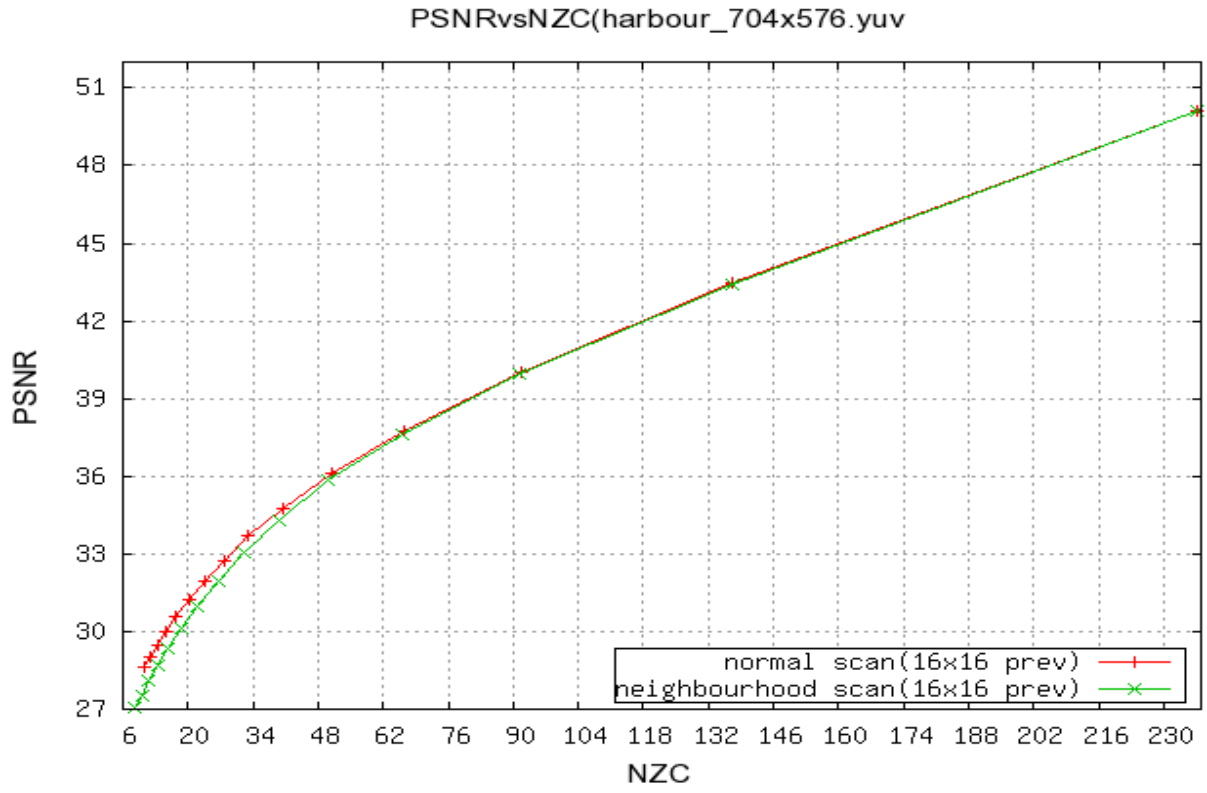

city_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

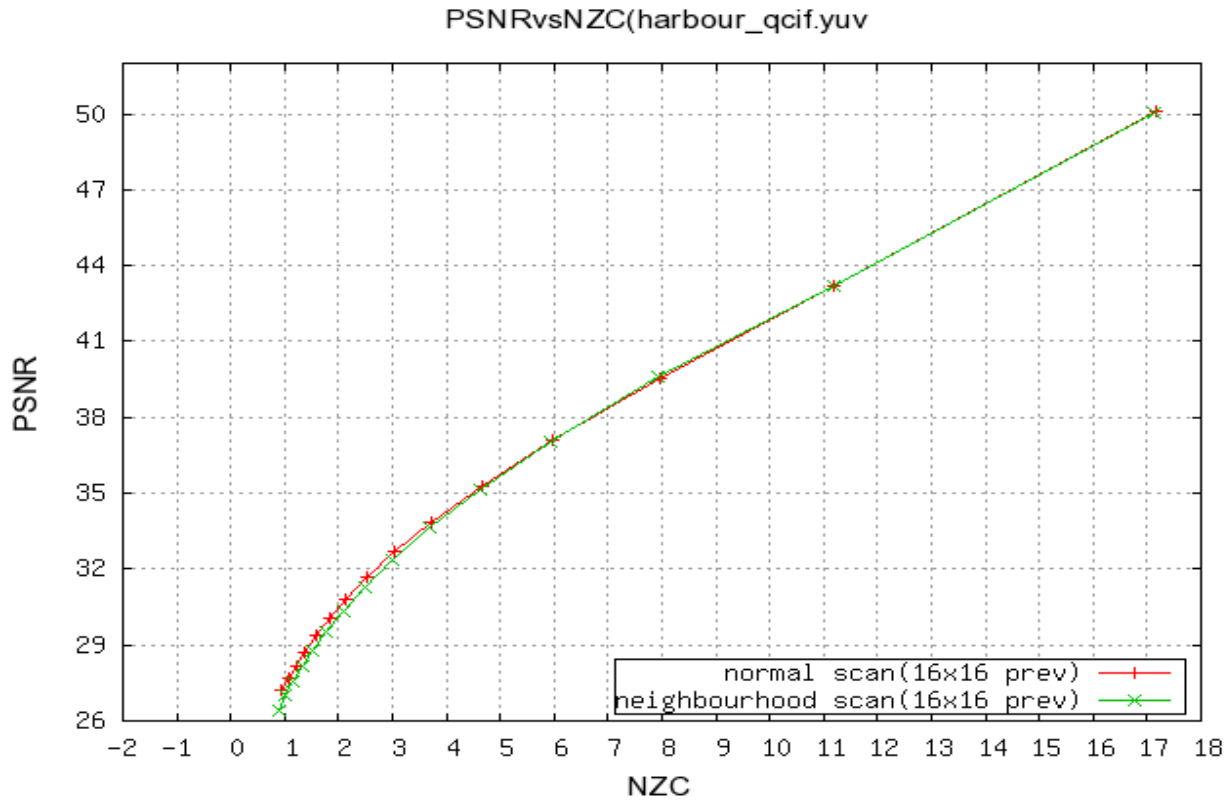

city_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

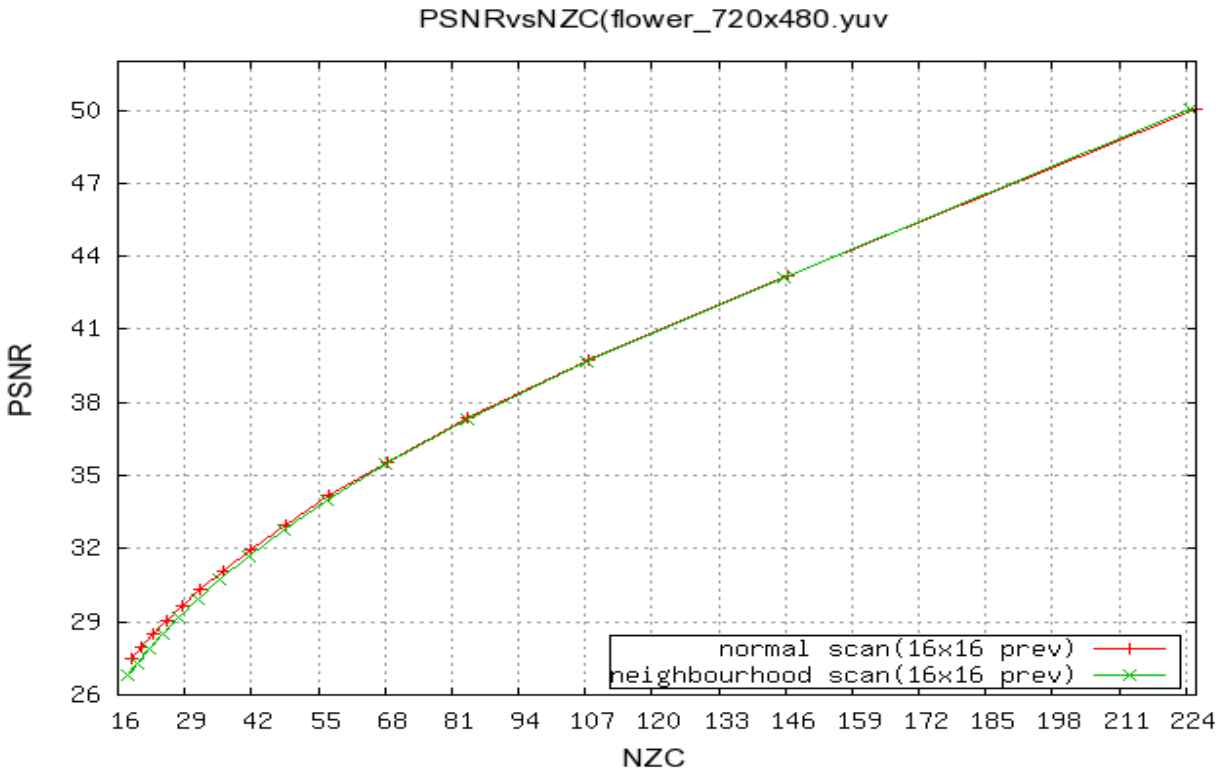

soccer_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

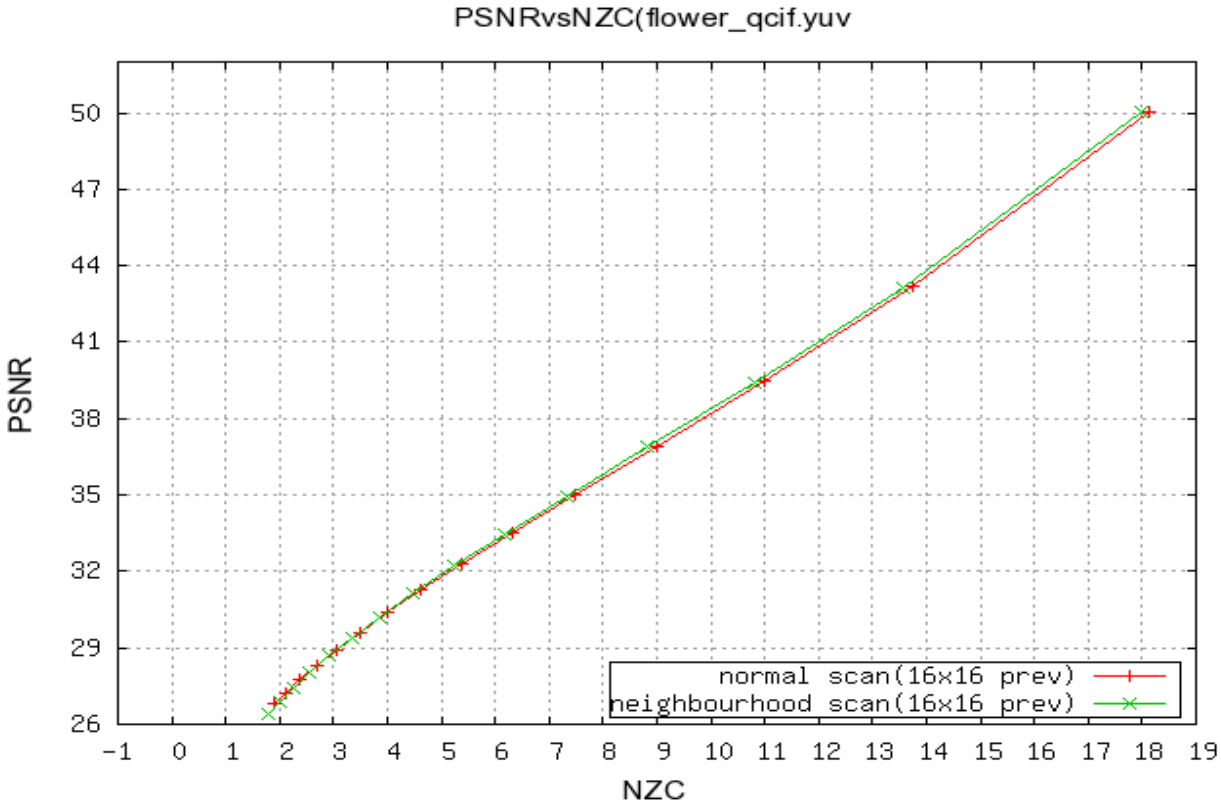

soccer_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

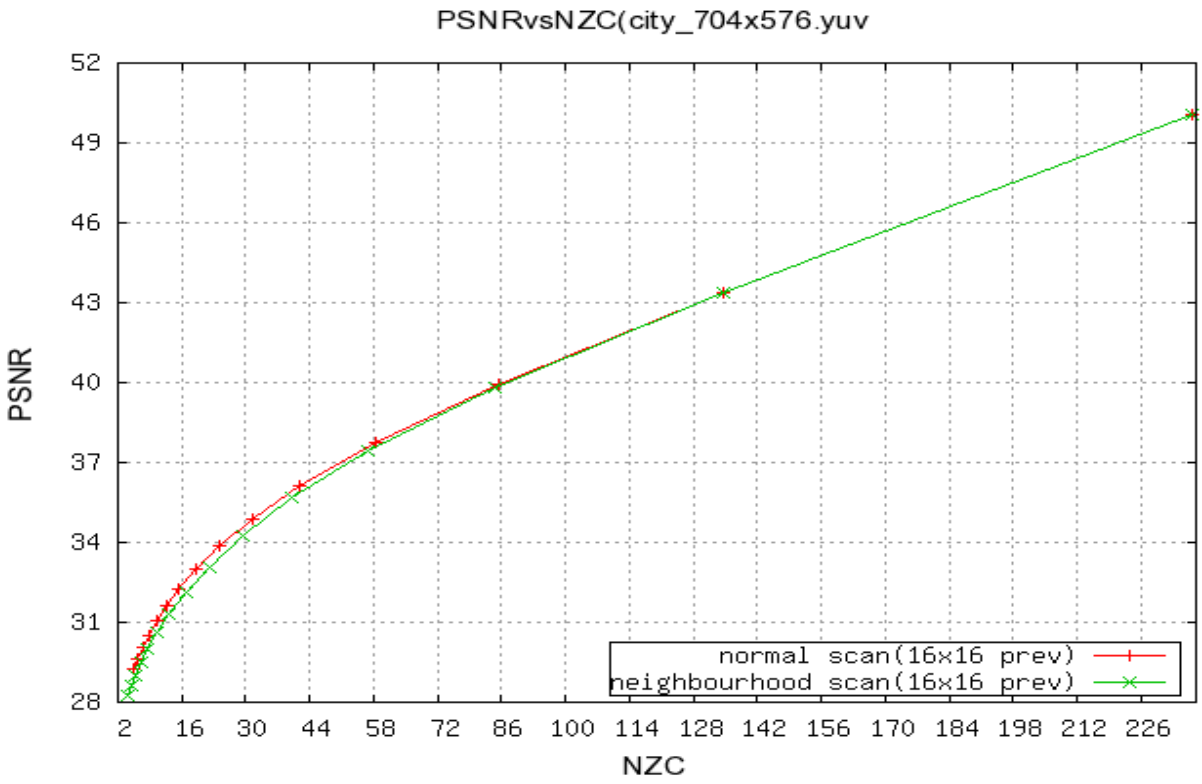
A.6 RD CURVES (16X16 PREV- NEIGHBOURHOOD SCAN)

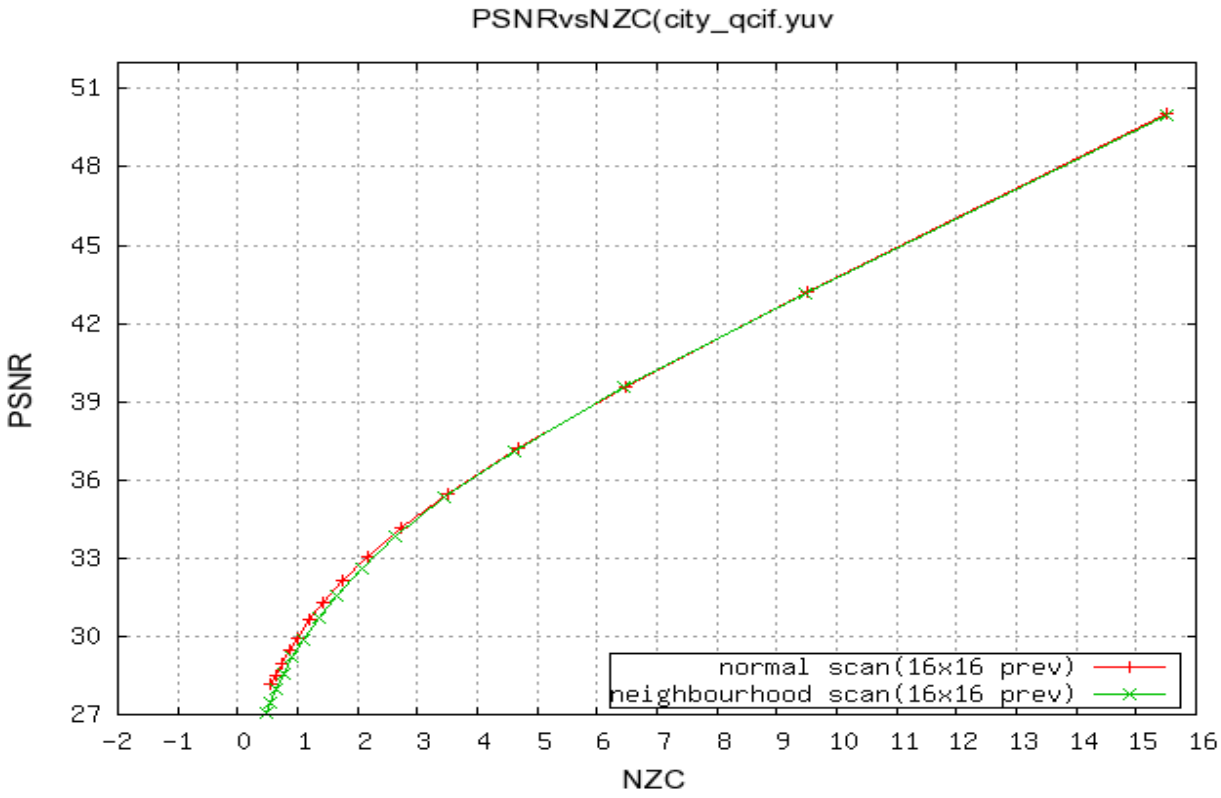


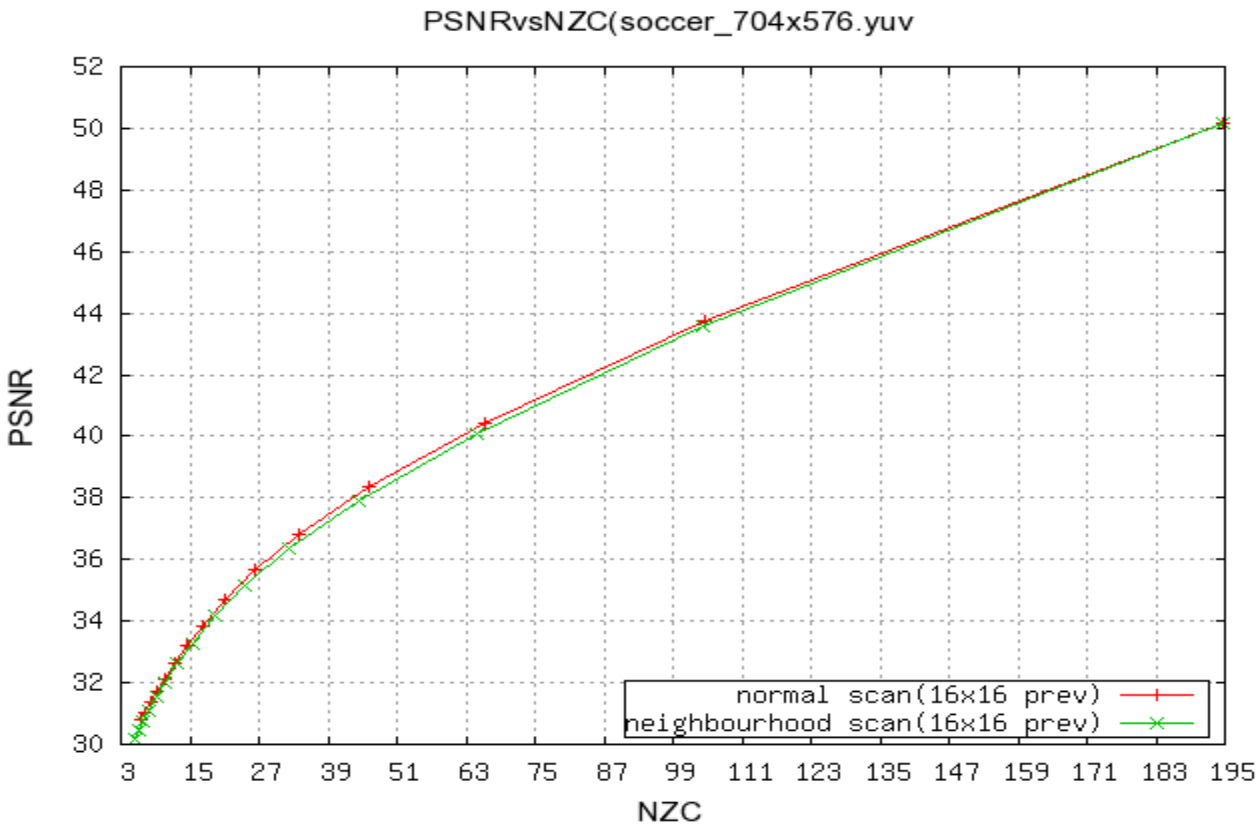


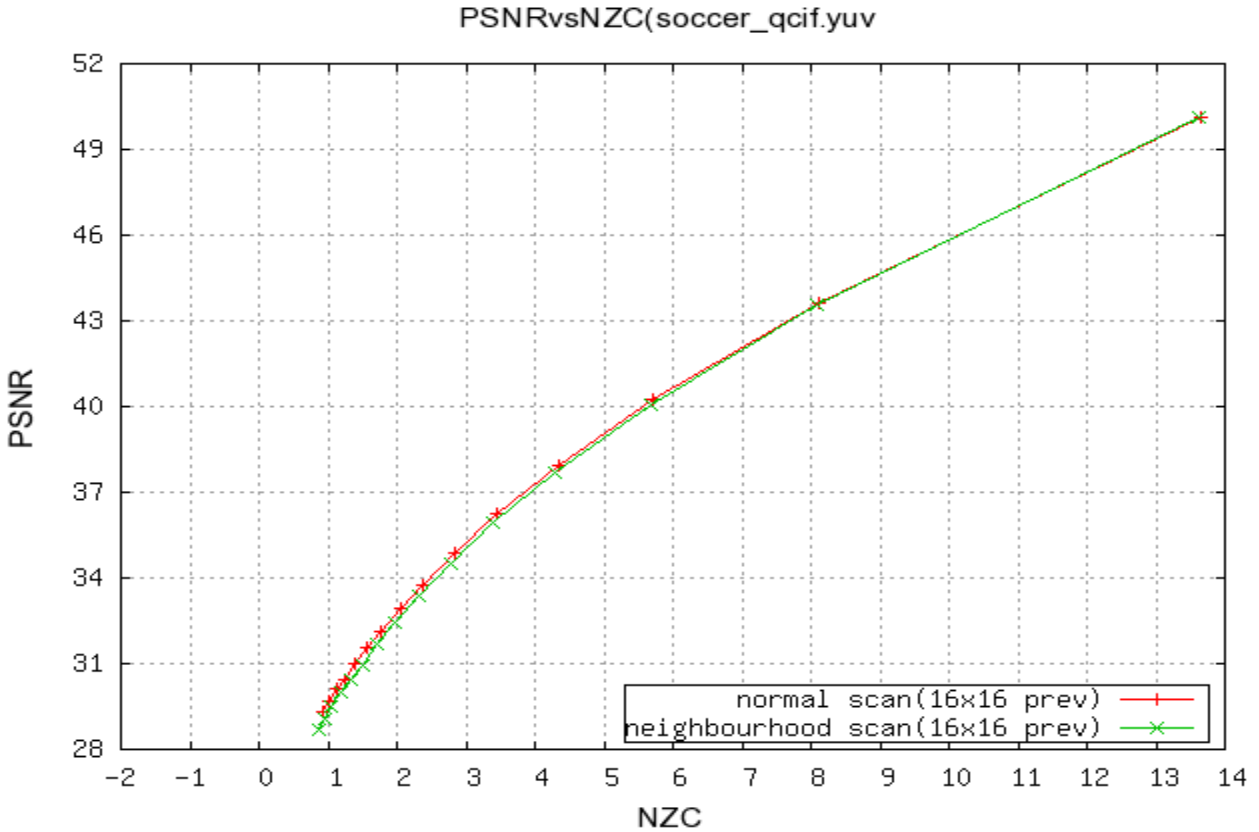

PSNRvsNZC(flower_720x480.yuv
PSNR
NZC
50
47
44
41
38
35
32
29
26
16
29
42
55
68
81
94
107
120
133
146
159
172
185
198
211
224
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)


PSNRvsNZC(flower_qcif.yuv
PSNR
NZC
50
47
44
41
38
35
32
29
26
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)

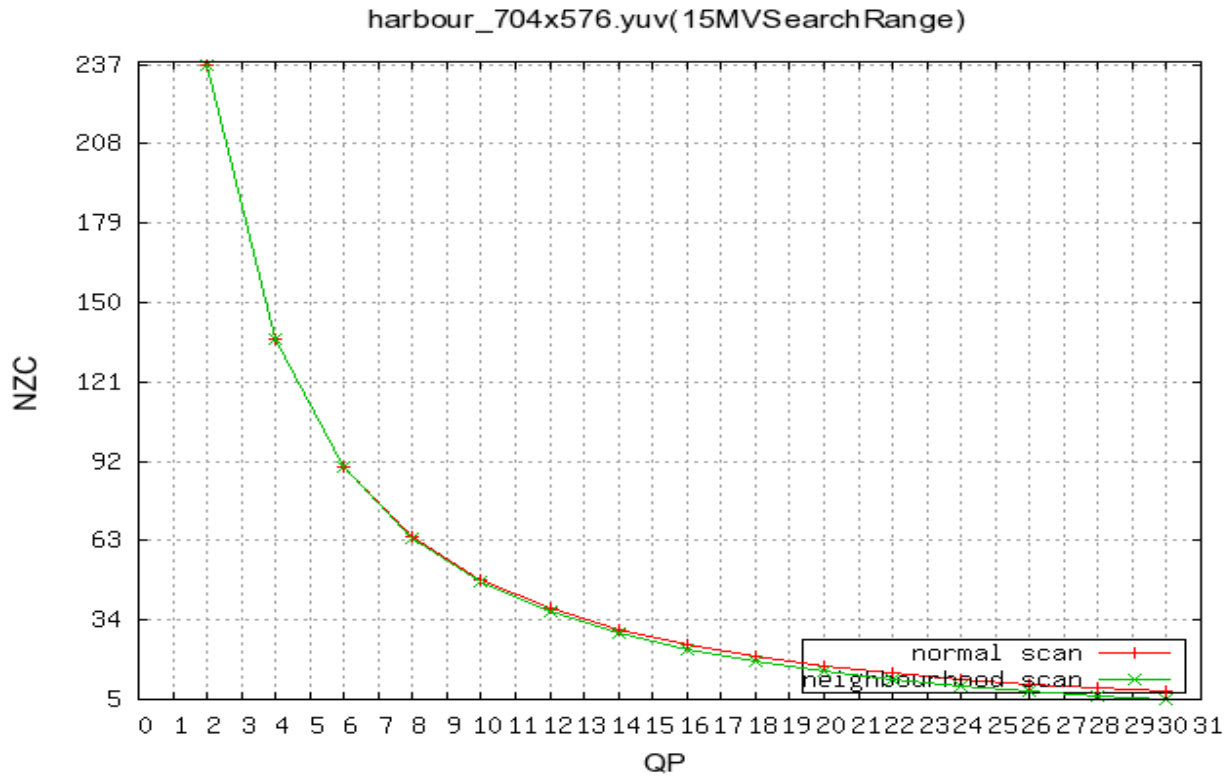

PSNRvsNZC(city_704x576.yuv
PSNR
NZC
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)
52
49
46
43
40
37
34
31
28
2
16
30
44
58
72
86
100
114
128
142
156
170
184
198
212
226

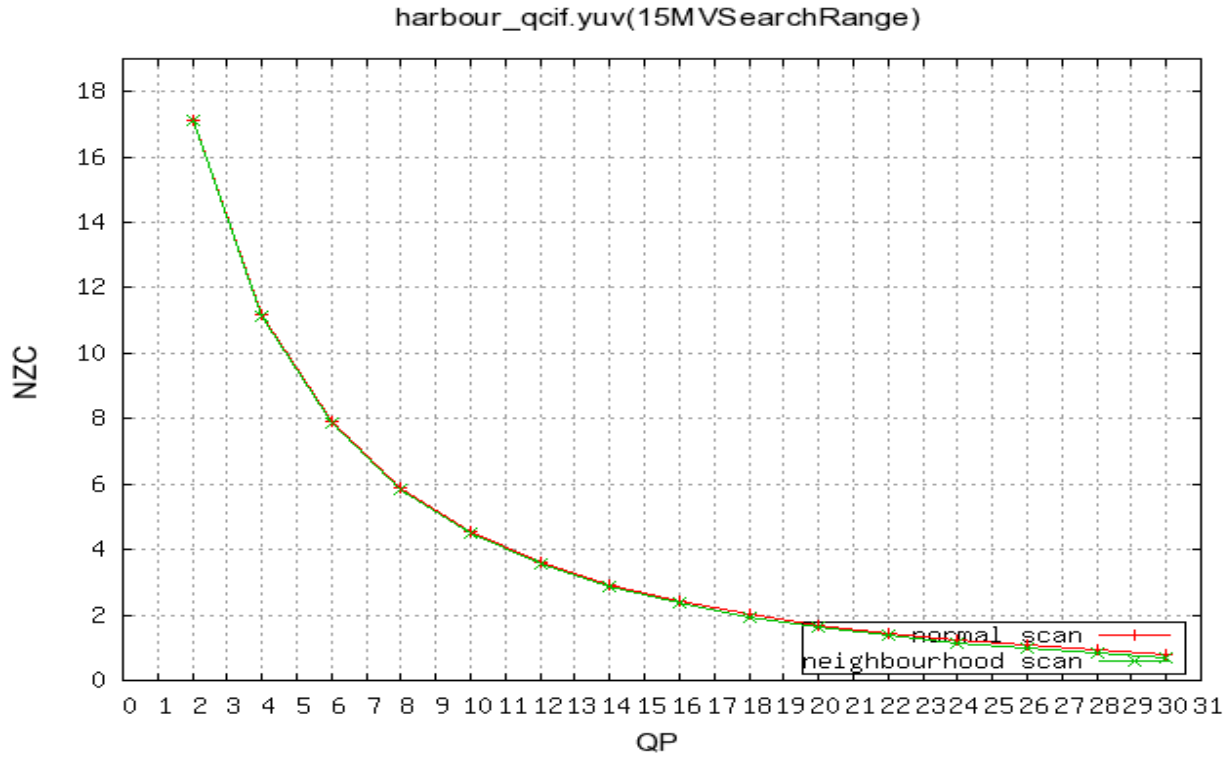

PSNRvsNZC(city_qcif.yuv
PSNR
NZC
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)
51
48
45
42
39
36
33
30
27
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

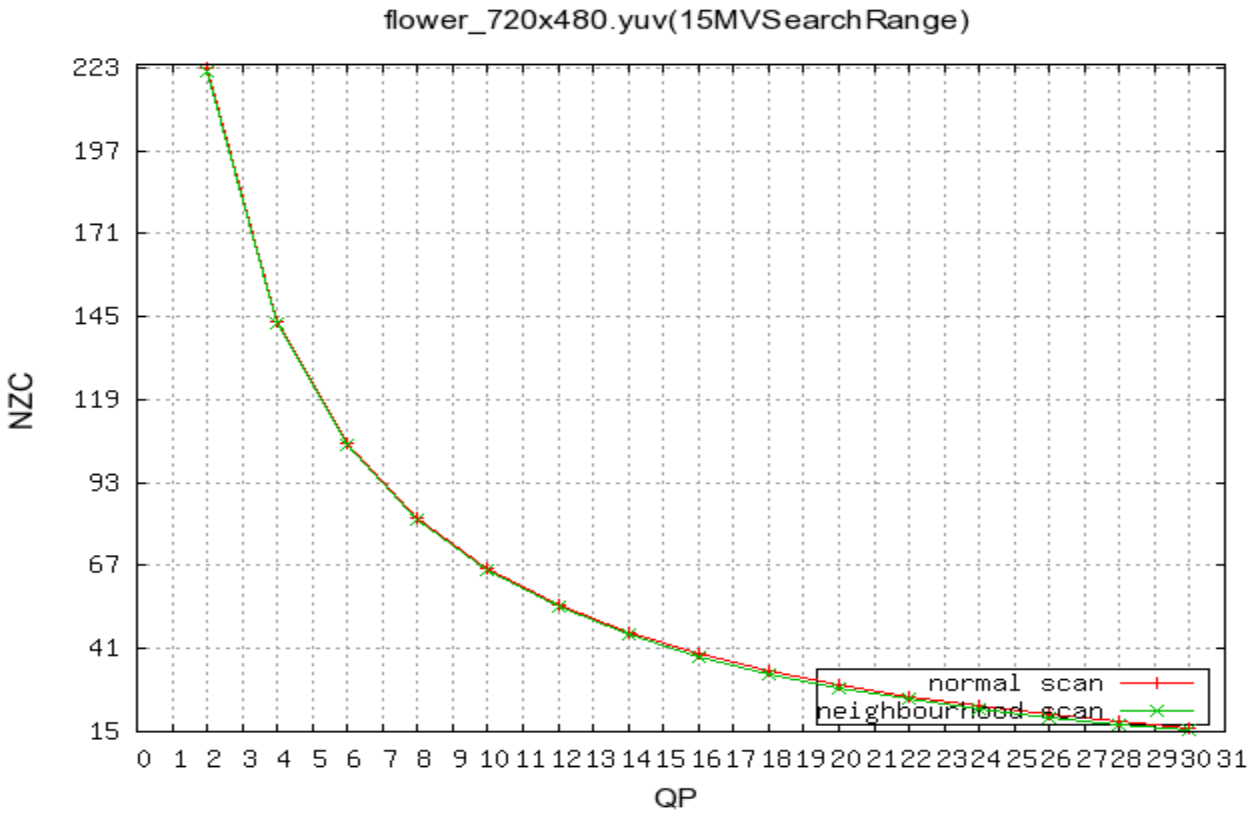

PSNRvsNZC(soccer_704x576.yuv
PSNR
NZC
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)
52
50
48
46
44
42
40
38
36
34
32
30
3
15
27
39
51
63
75
87
99
111
123
135
147
159
171
183
195

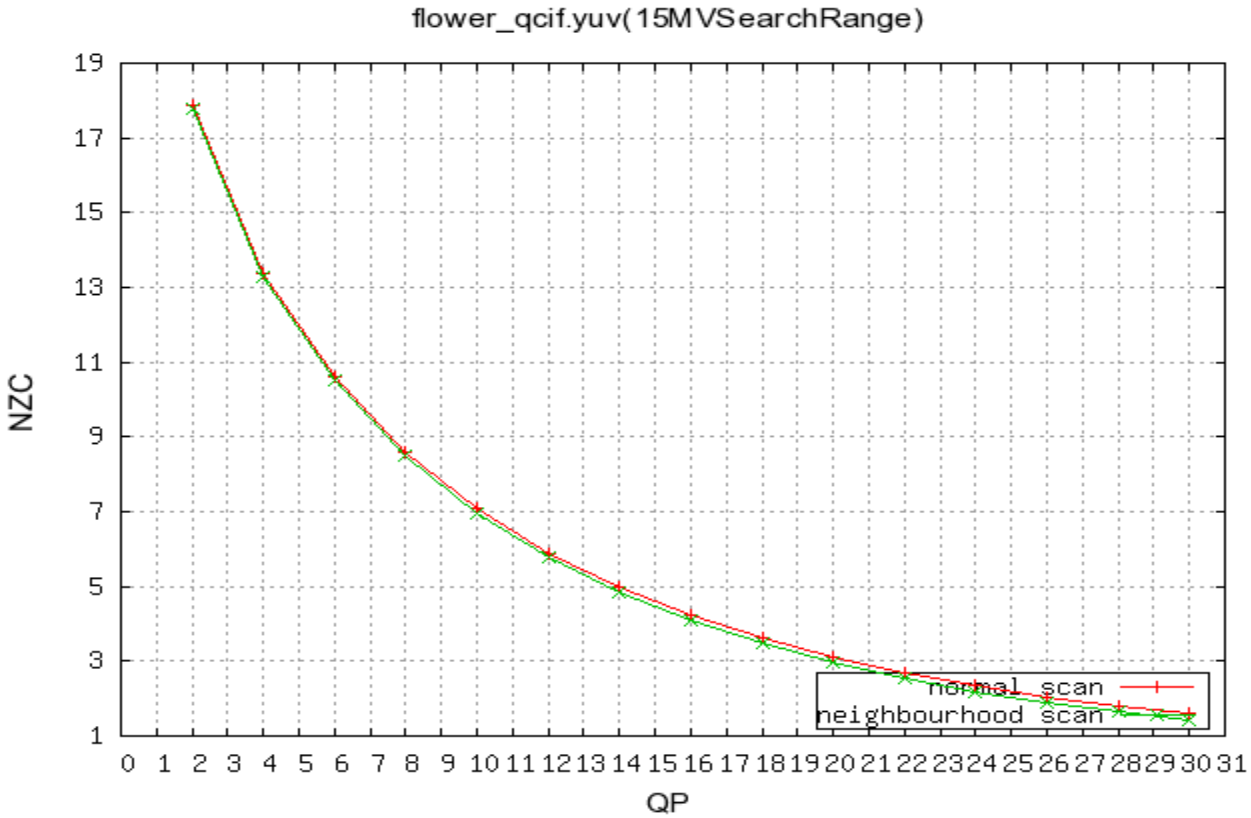

PSNRvsNZC(soccer_qcif.yuv
PSNR
NZC
normal scan(16x16 prev)
neighbourhood scan(16x16 prev)
52
49
46
43
40
37
34
31
28
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

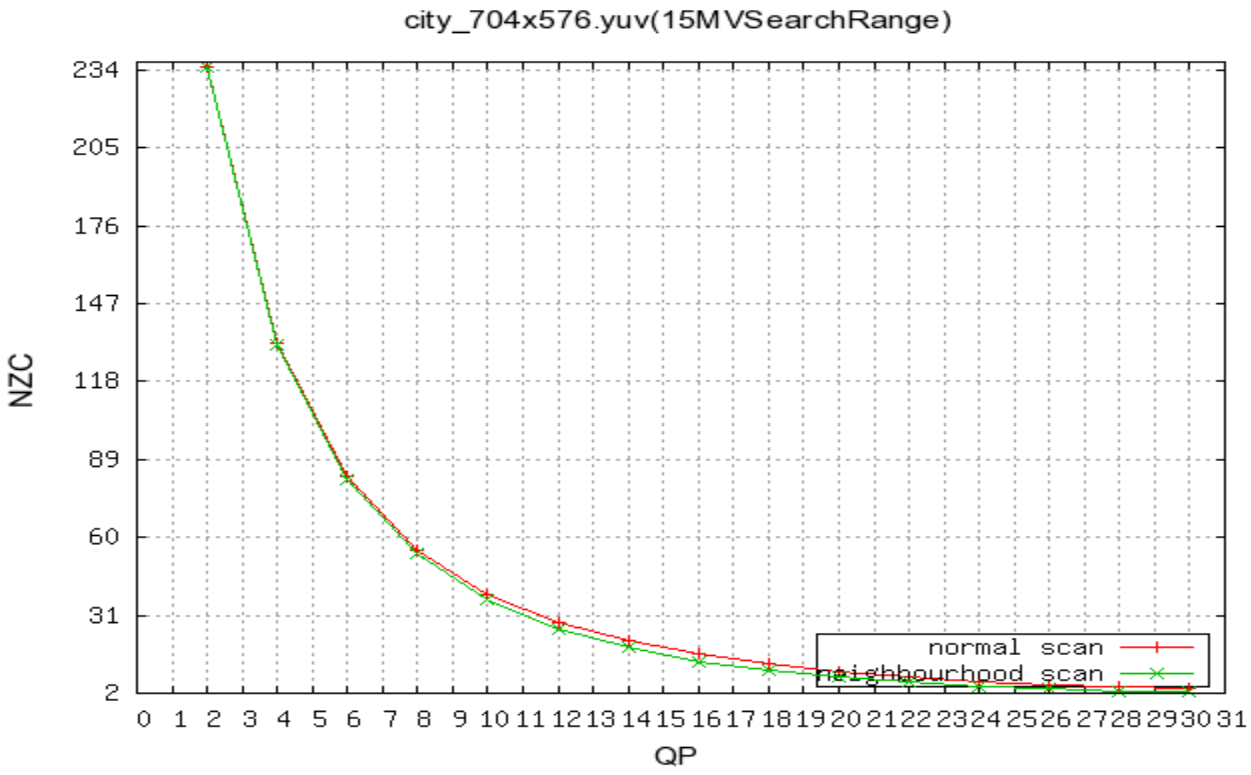
A.7 QP VS NZC (8X8 PREV-NEIGHBOURHOOD SCAN)

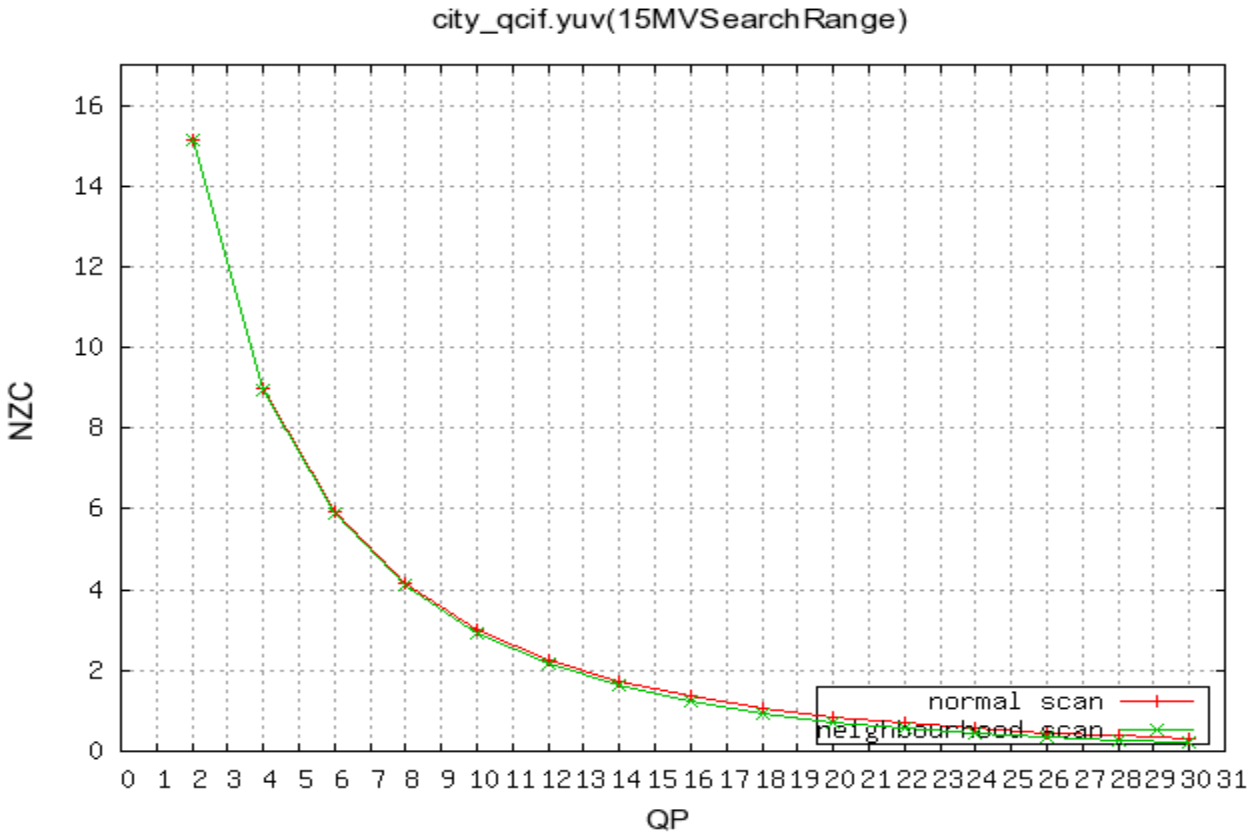


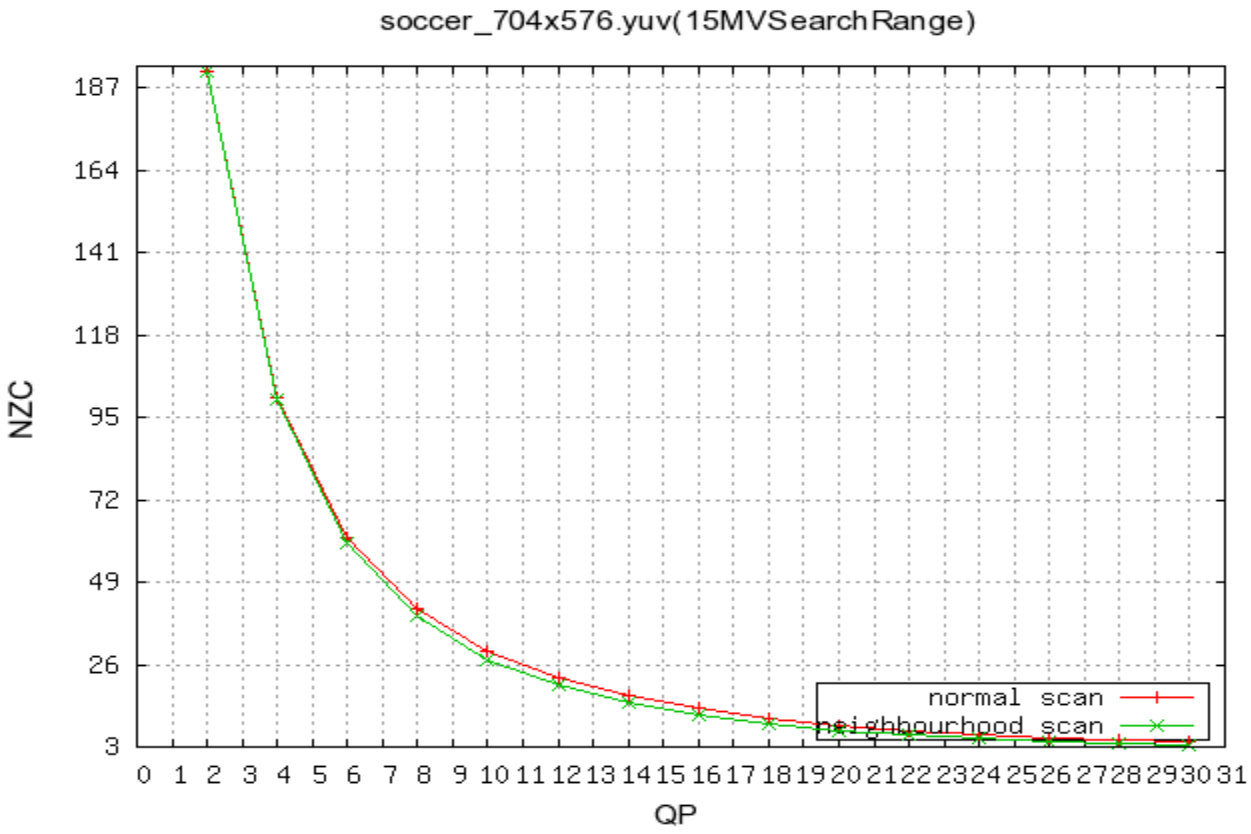


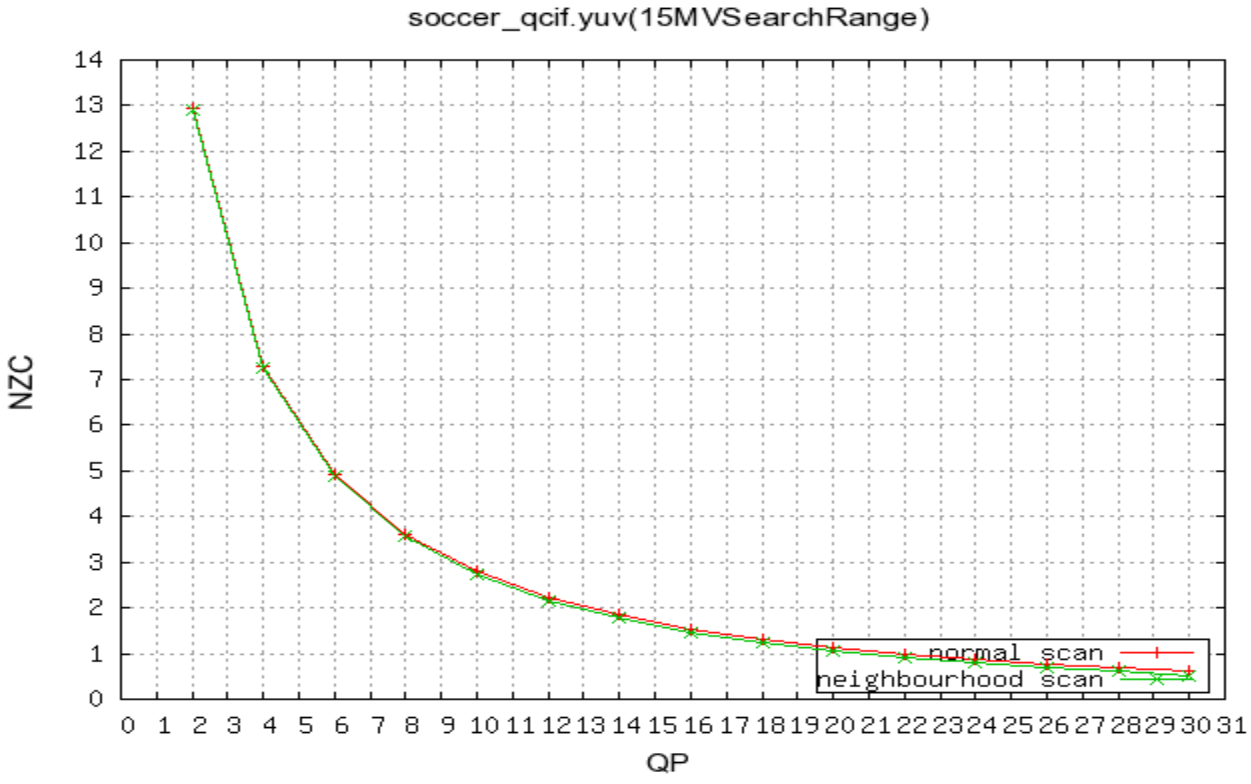

flower_720x480.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan


flower_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

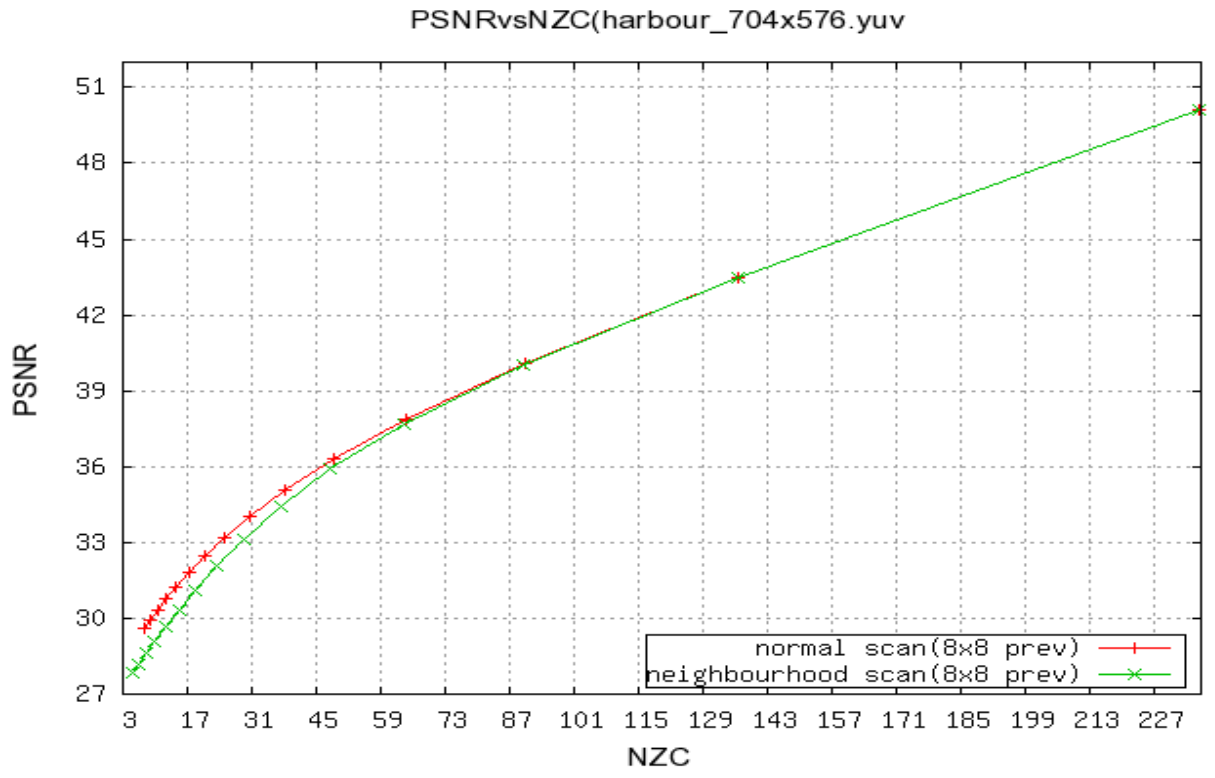

city_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

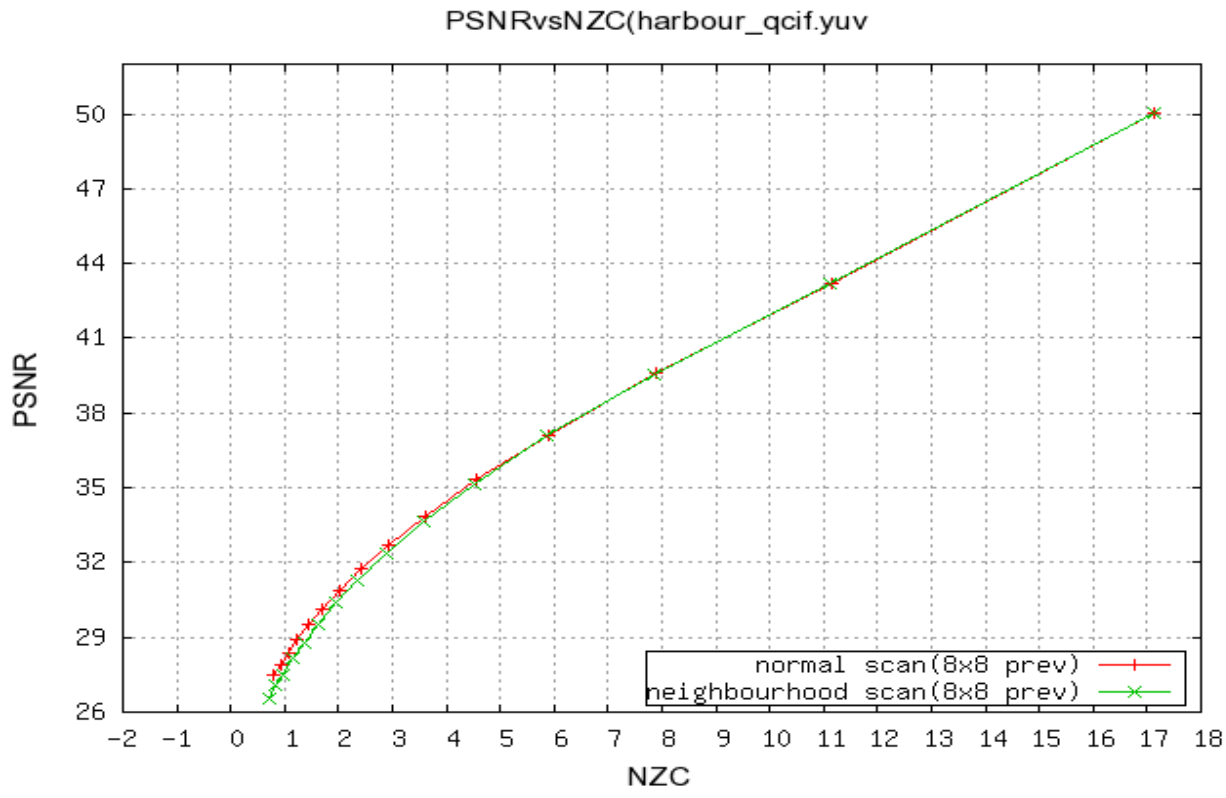

city_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

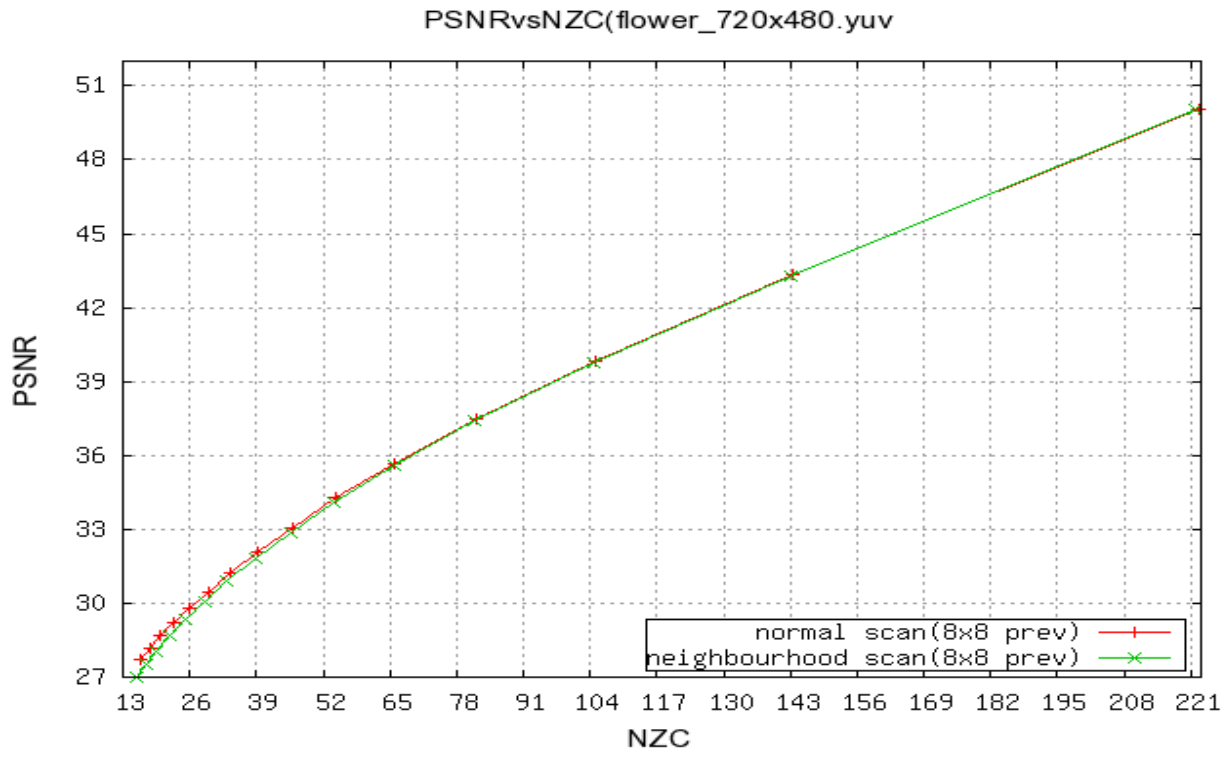

soccer_704x576.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

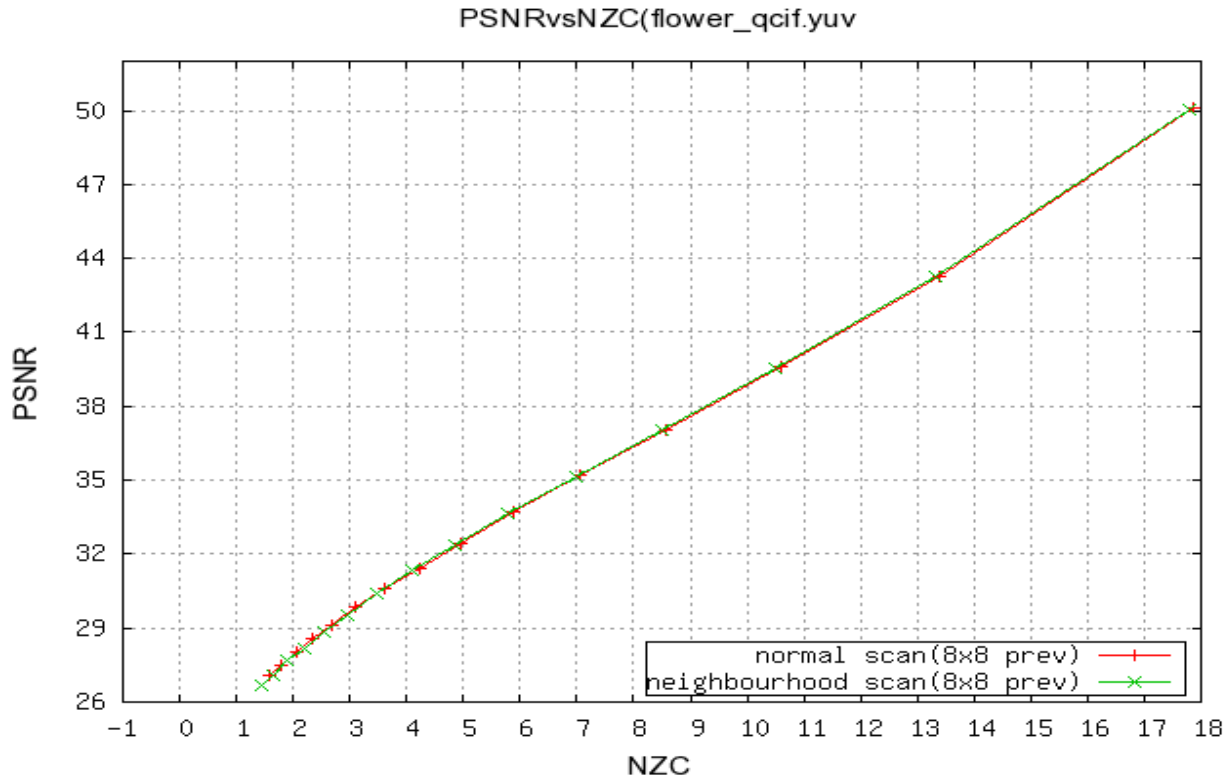

soccer_qcif.yuv(15MVSearchRange)
NZC
QP
normal scan
neighbourhood scan

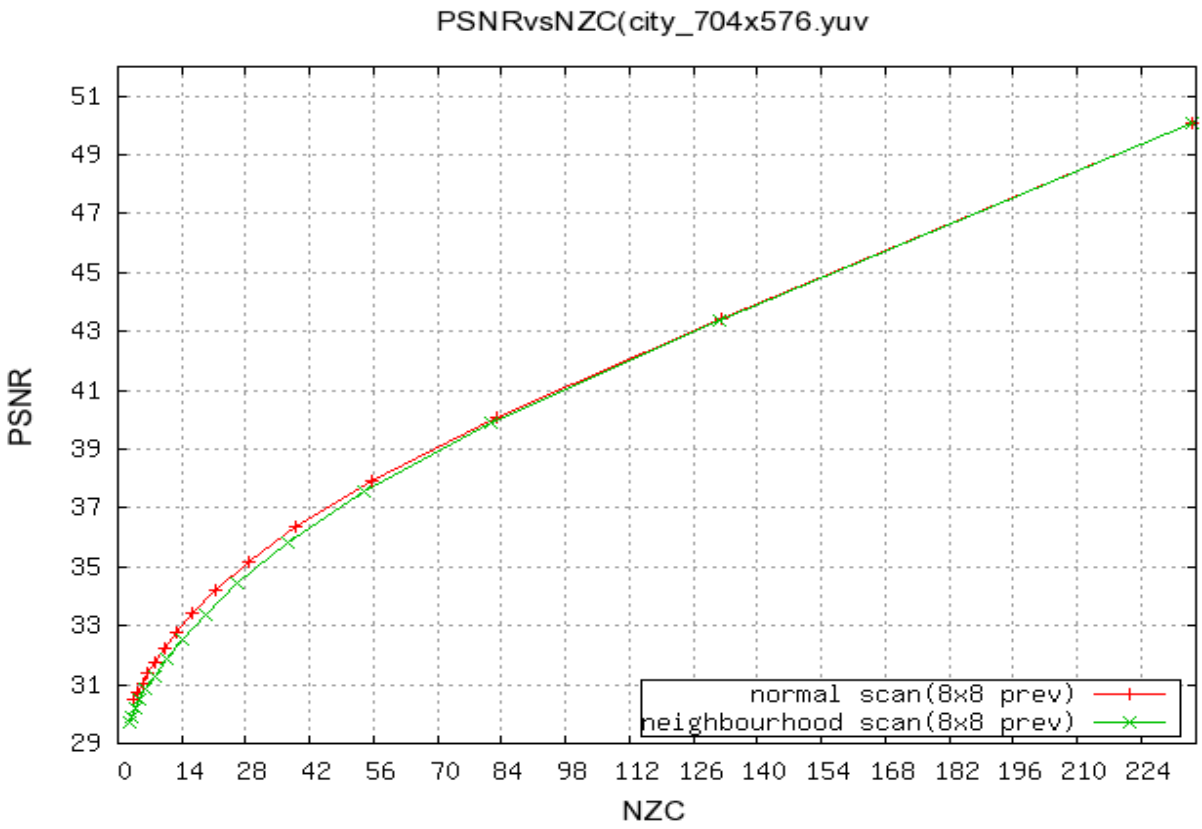
A.8 RD CURVES (8X8 PREV-NEIGHBOURHOOD SCAN)

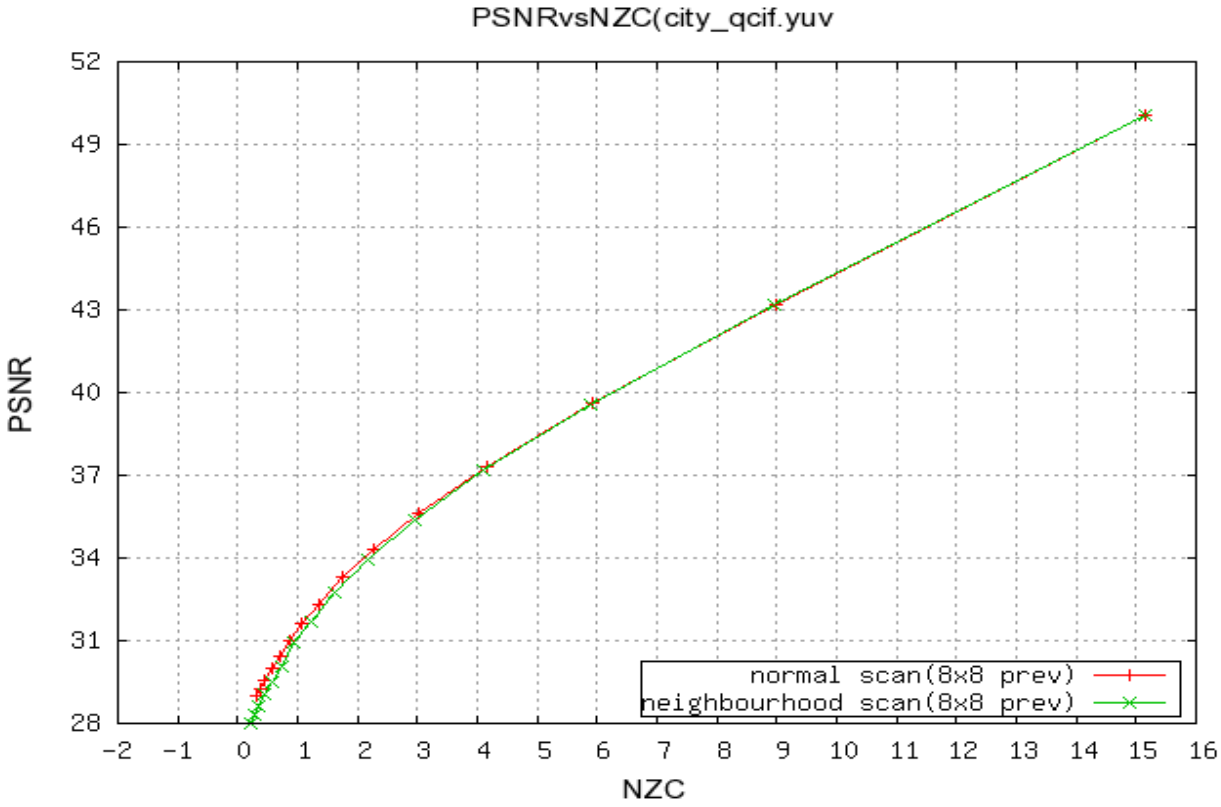


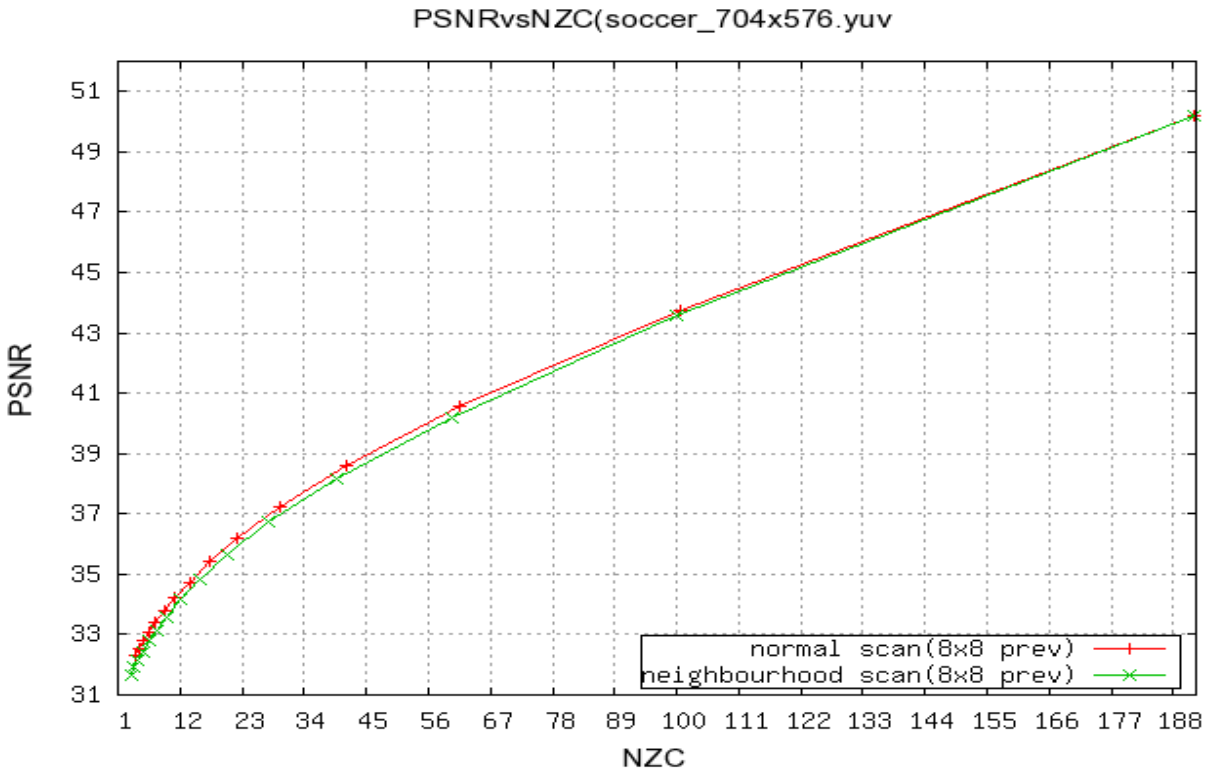


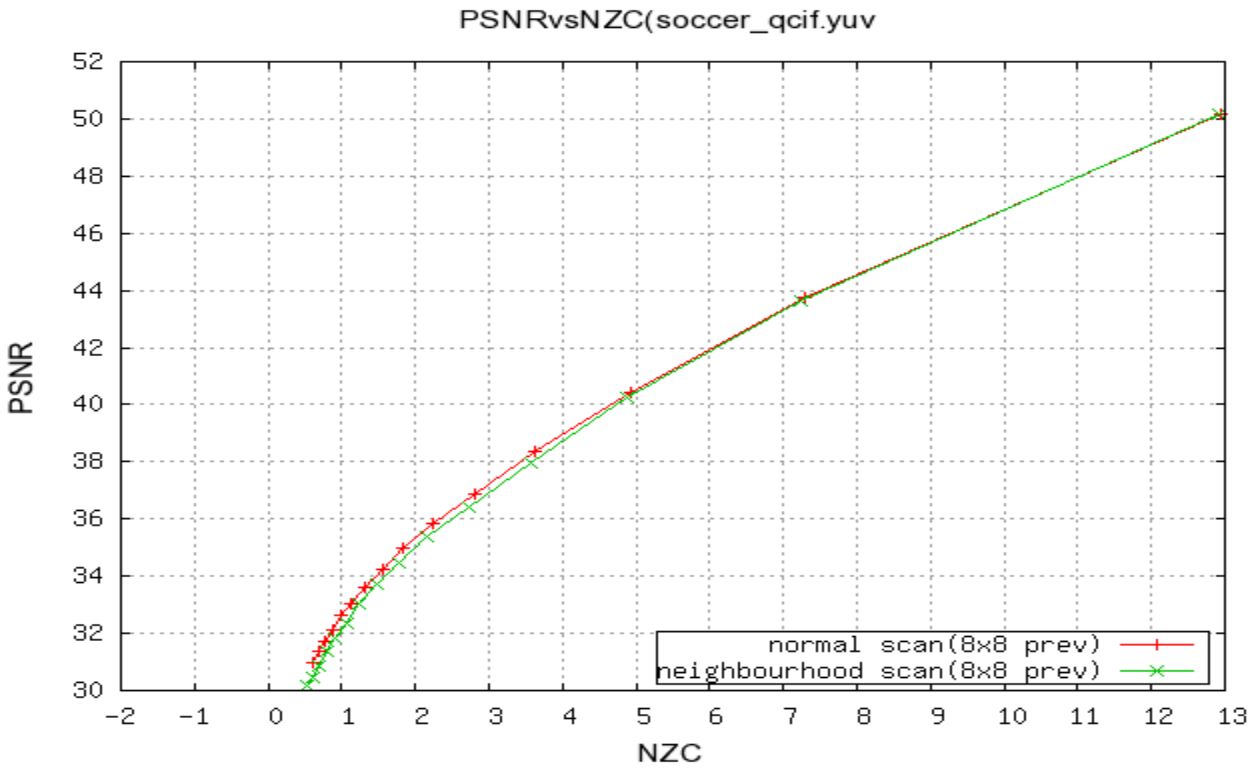

PSNRvsNZC(flower_720x480.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
51
48
45
42
39
36
33
30
27
13
26
39
52
65
78
91
104
117
130
143
156
169
182
195
208
221


PSNRvsNZC(flower_qcif.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
50
47
44
41
38
35
32
29
26
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

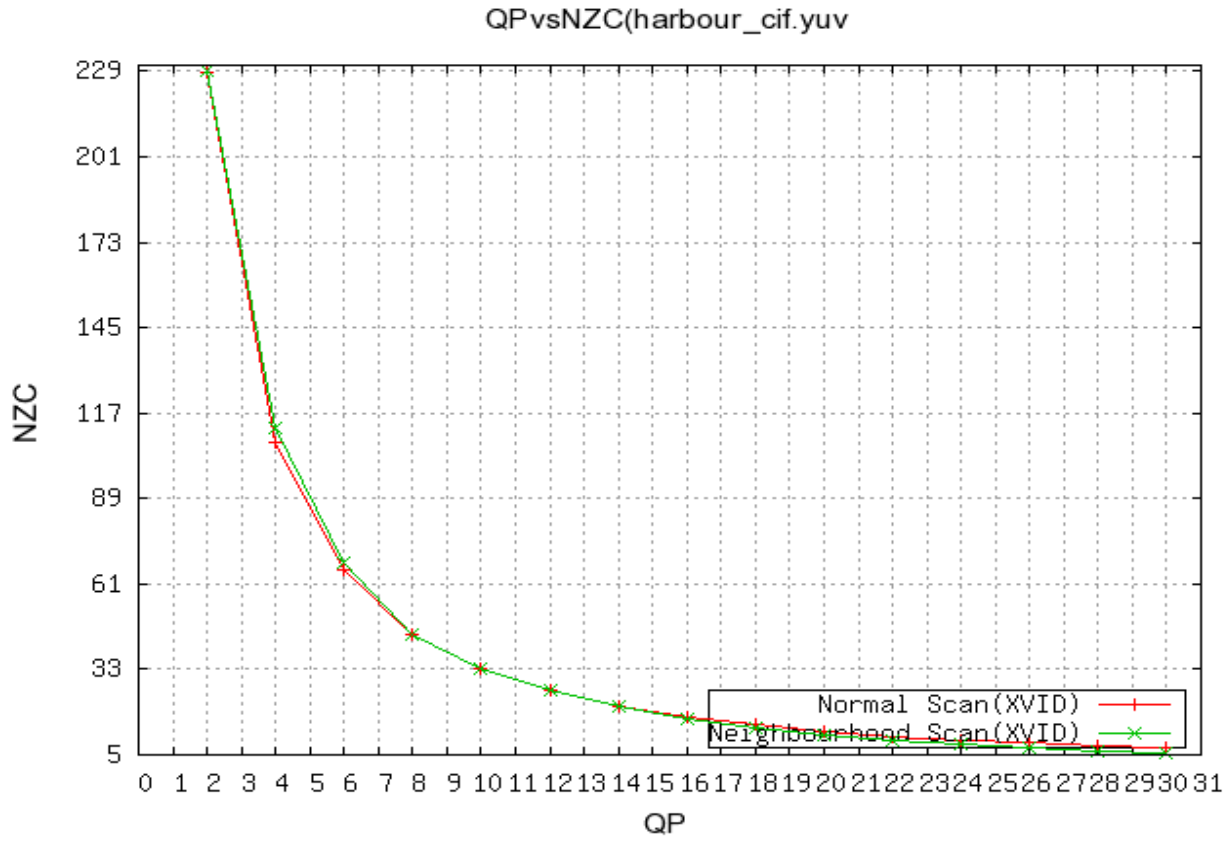

PSNRvsNZC(city_704x576.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
51
49
47
45
43
41
39
37
35
33
31
29
0 14 28 42 56 70 84 98 112 126 140 154 168 182 196 210 224

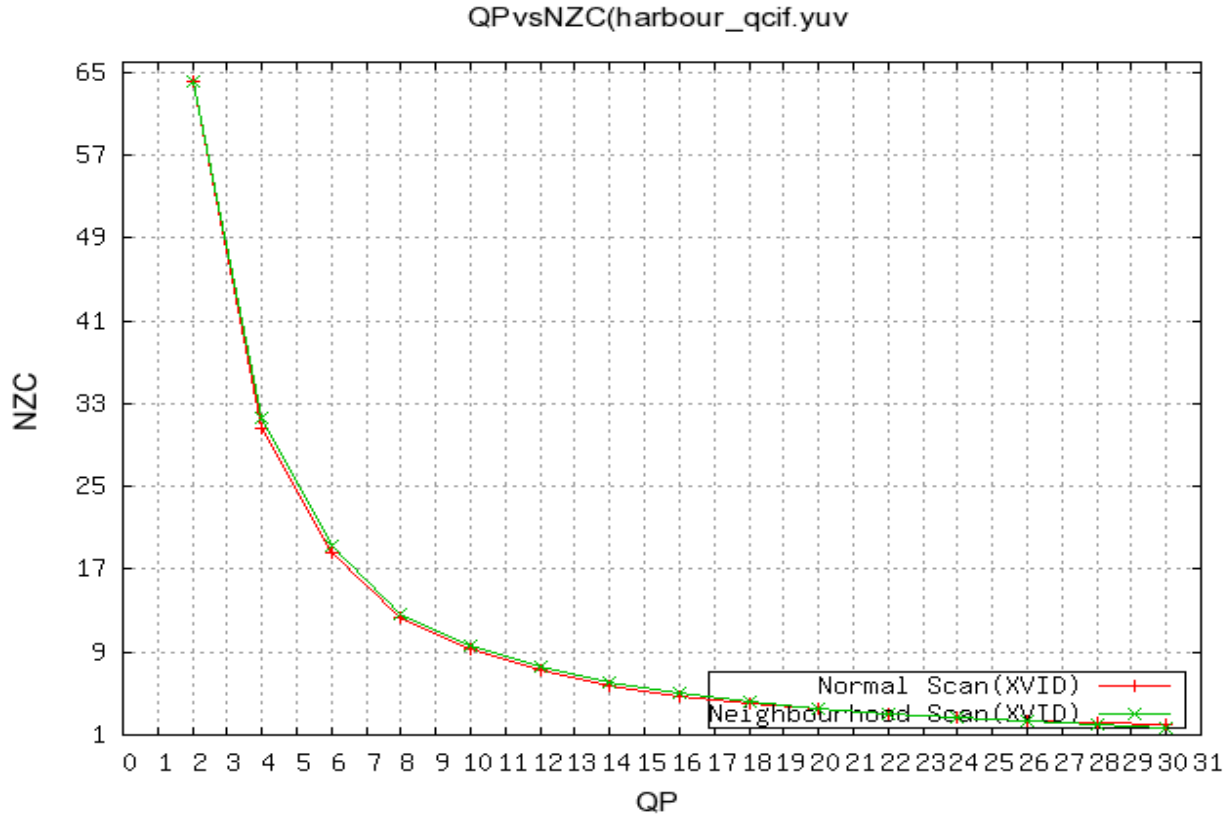

PSNRvsNZC(city_qcif.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
52
49
46
43
40
37
34
31
28
-2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

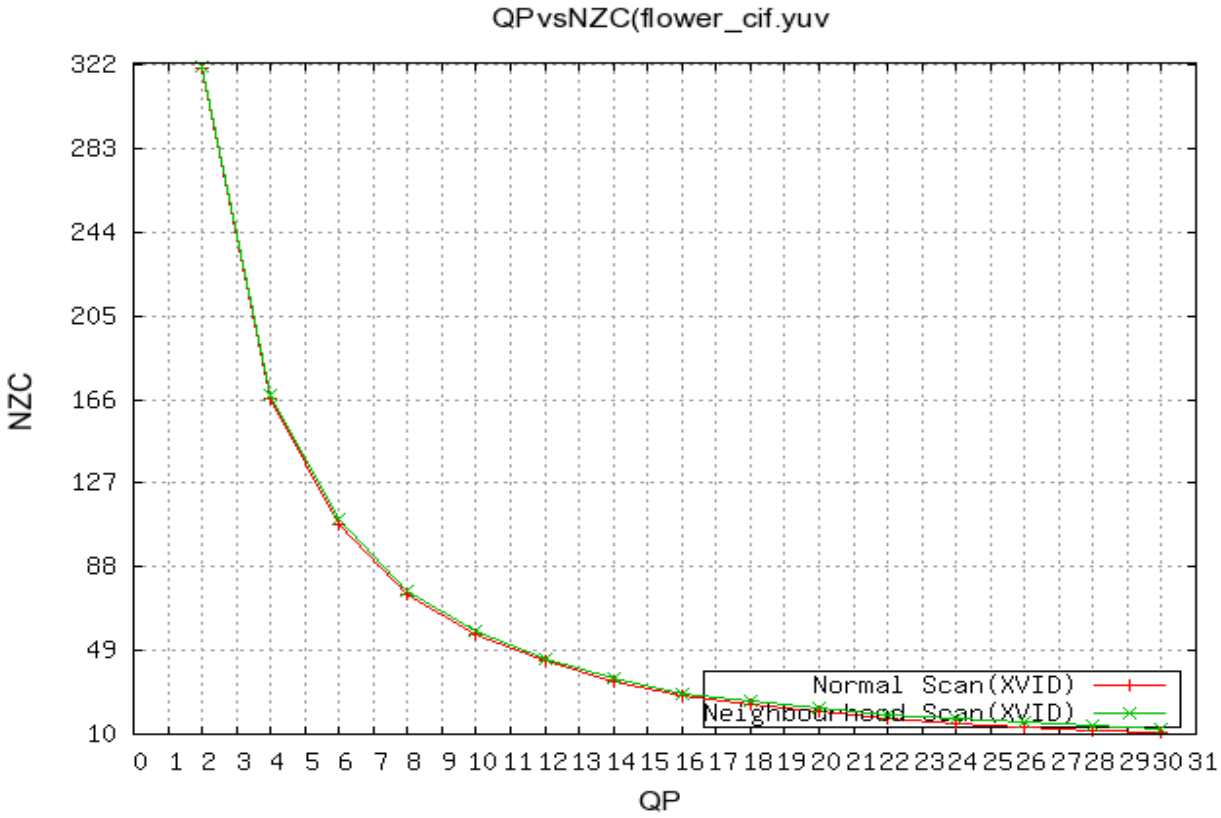

PSNRvsNZC(soccer_704x576.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
51
49
47
45
43
41
39
37
35
33
31
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188

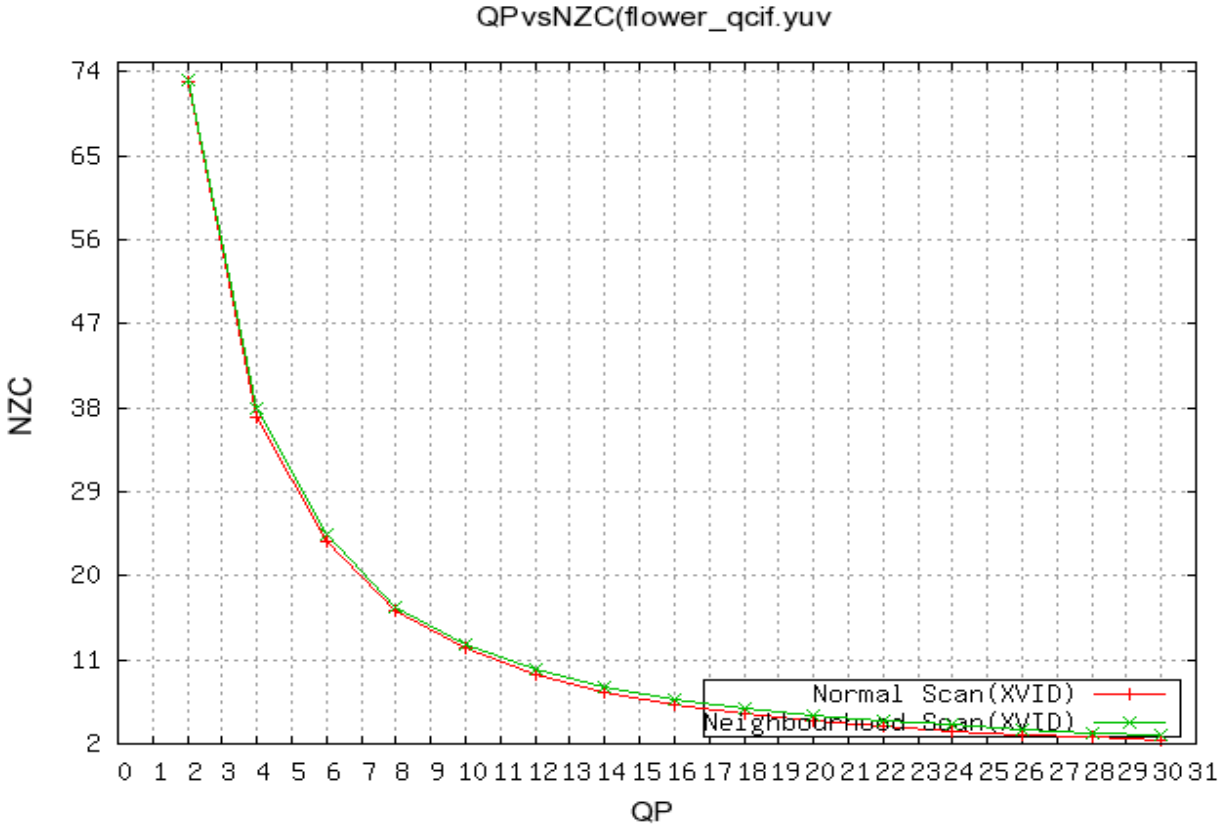

PSNRvsNZC(soccer_qcif.yuv
PSNR
NZC
normal scan(8x8 prev)
neighbourhood scan(8x8 prev)
52
50
48
46
44
42
40
38
36
34
32
30
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13

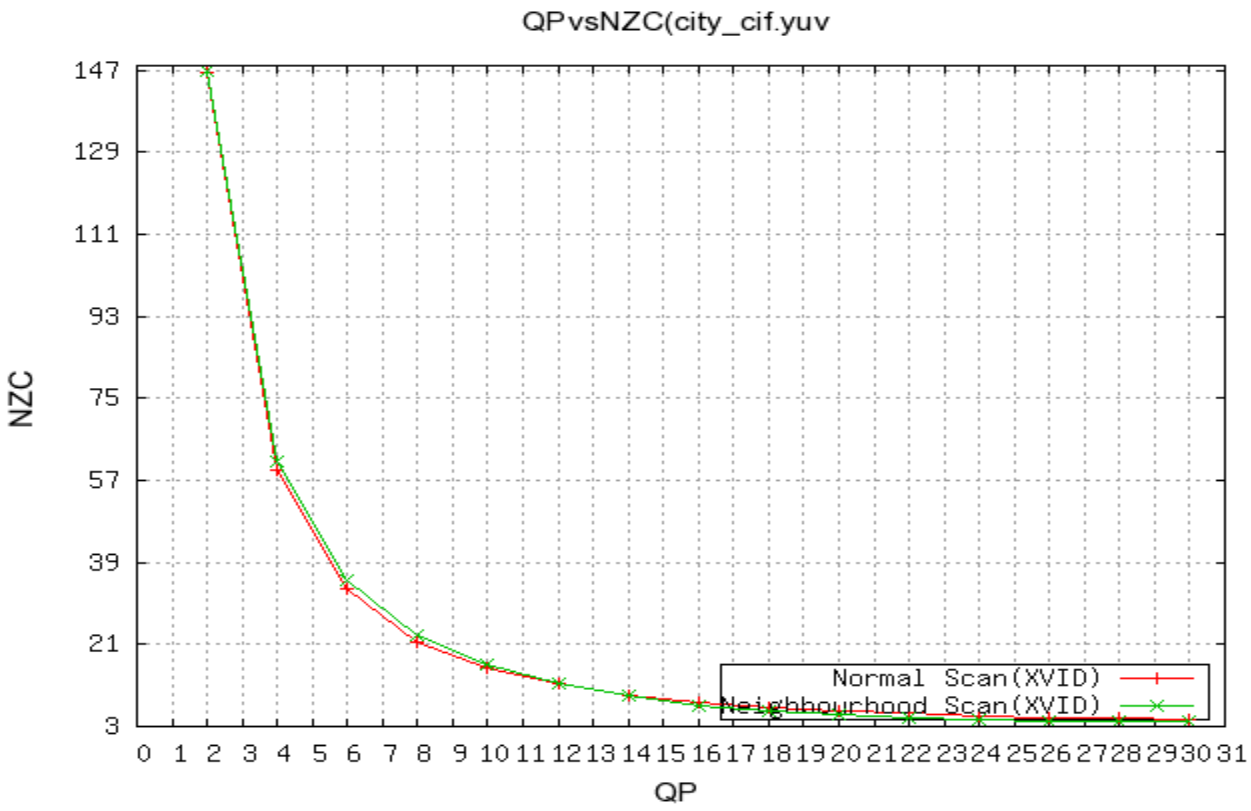
A.9 QP VS NZC (XVID IMPLEMENTATION)

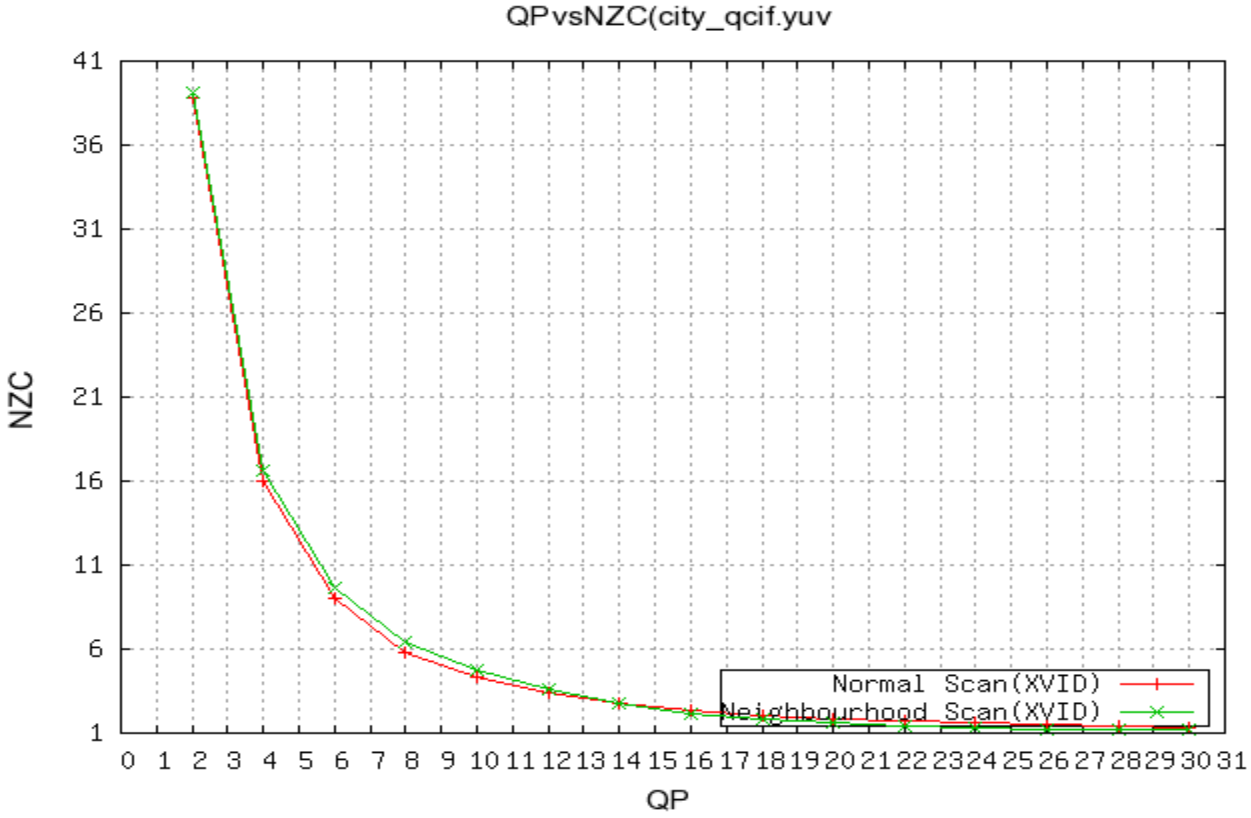


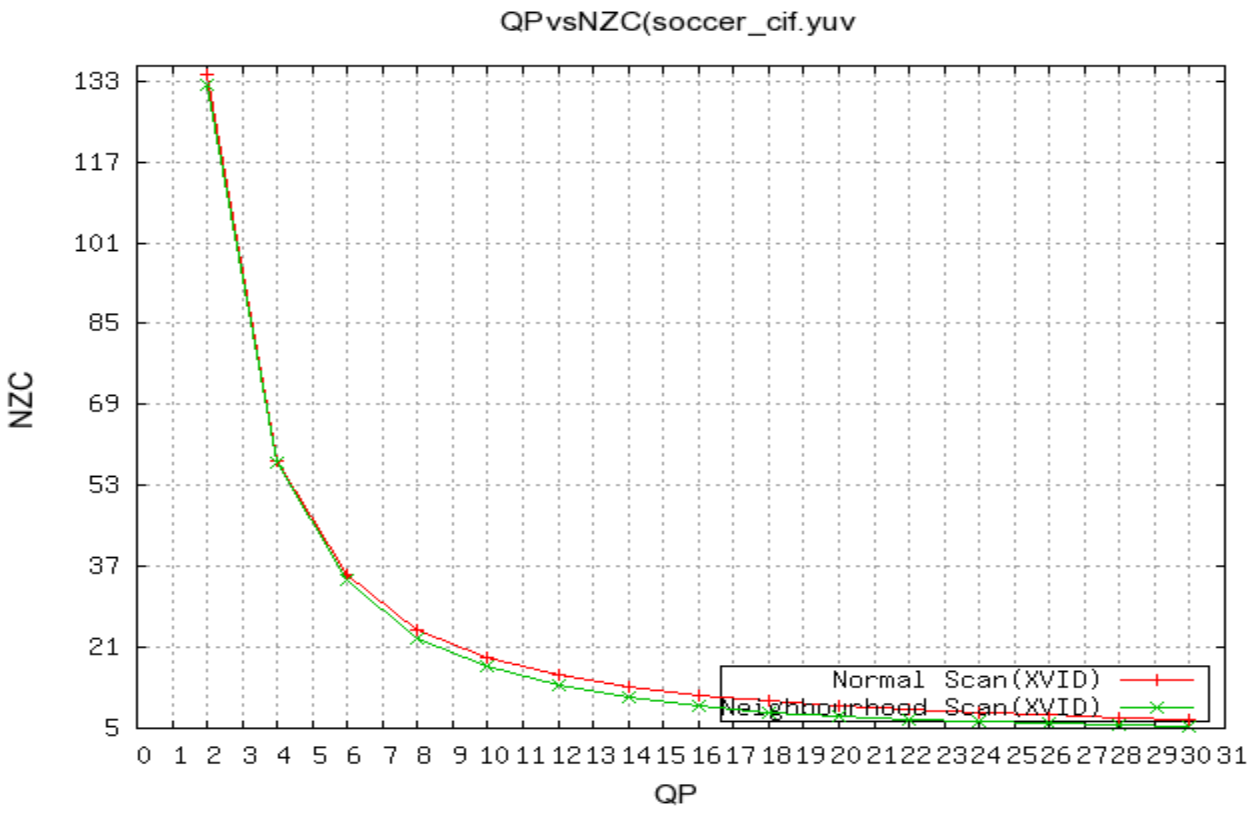


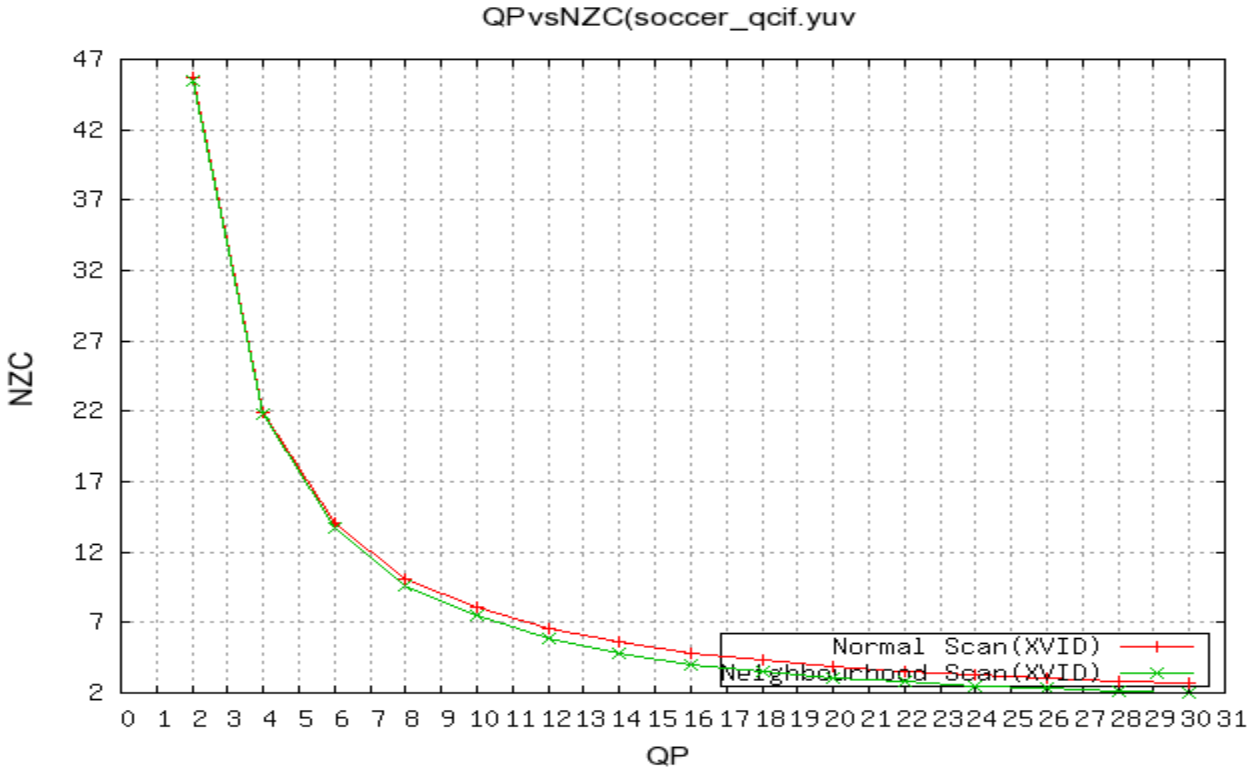

QPvsNZC(flower_cif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)
322
283
244
205
166
127
88
49
10
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31


QPvsNZC(flower_qcif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)
74
65
56
47
38
29
20
11
2
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

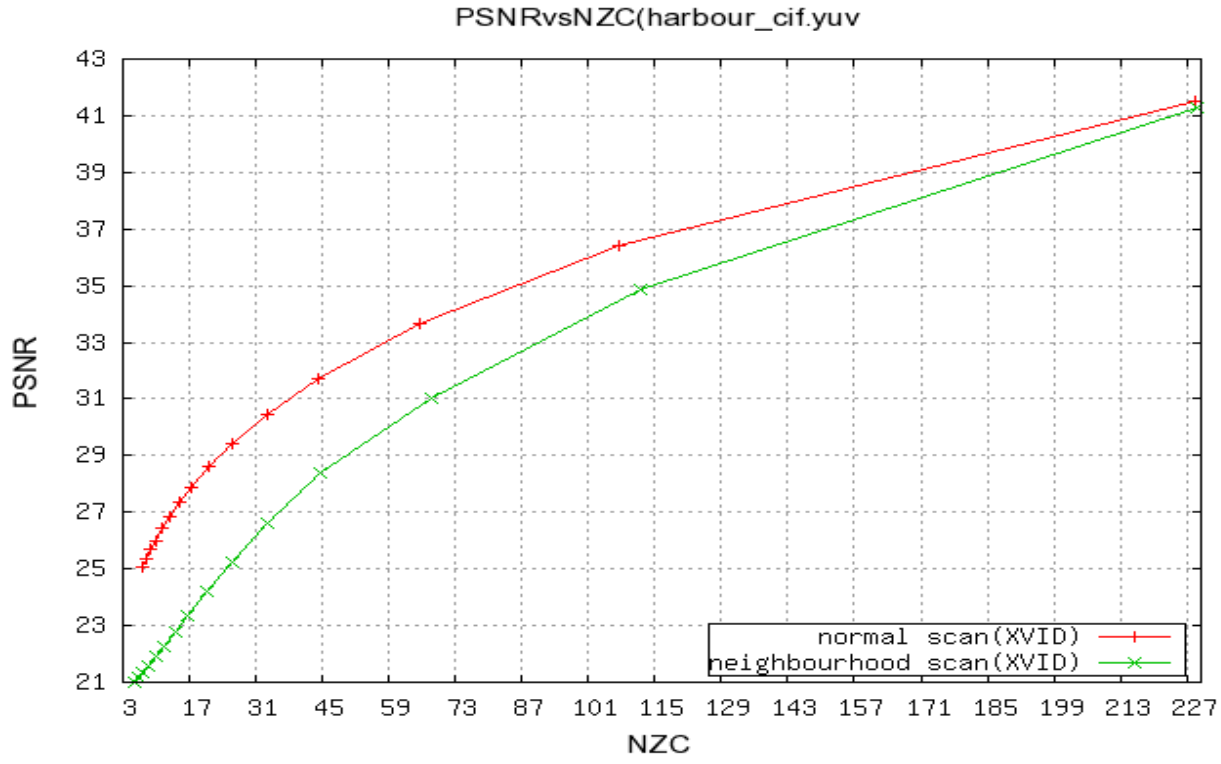

QPvsNZC(city_cif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)

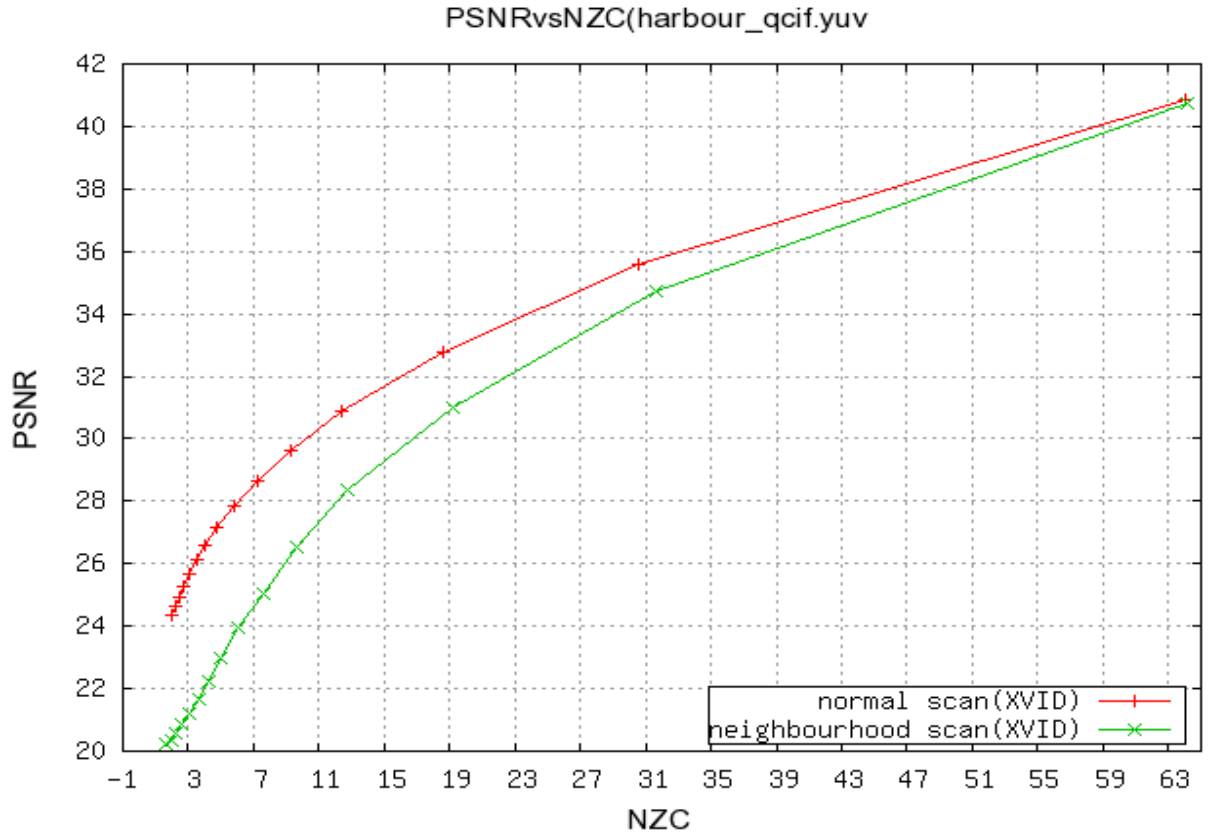

QPvsNZC(city_qcif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)

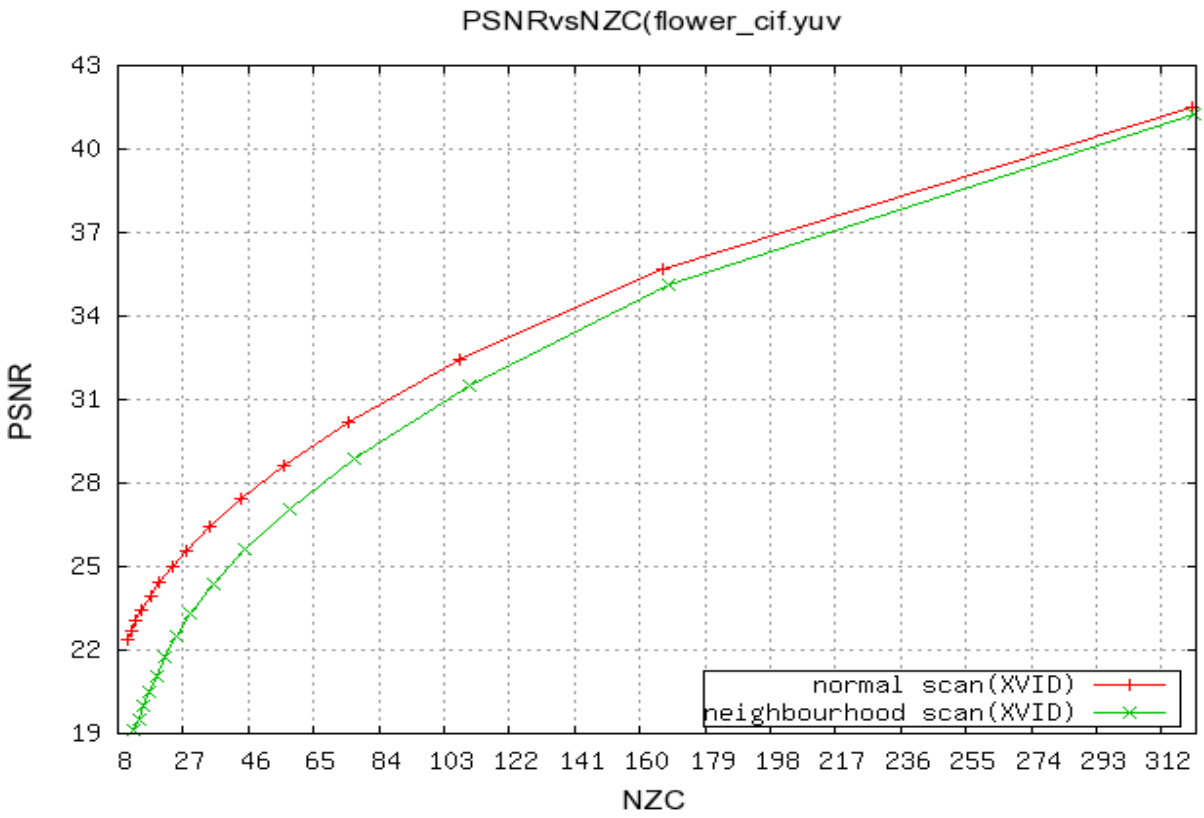

QPvsNZC(soccer_cif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)

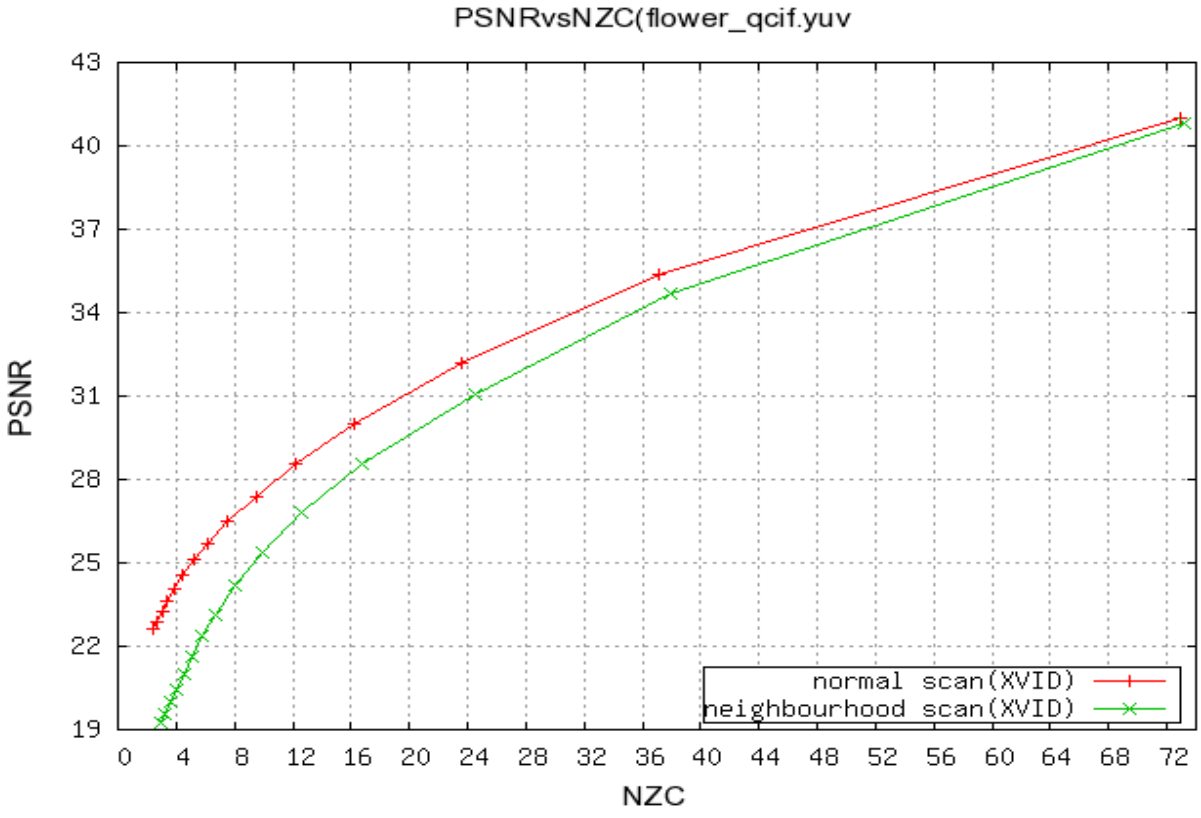

QPvsNZC(soccer_qcif.yuv
NZC
QP
Normal Scan(XVID)
Neighbourhood Scan(XVID)

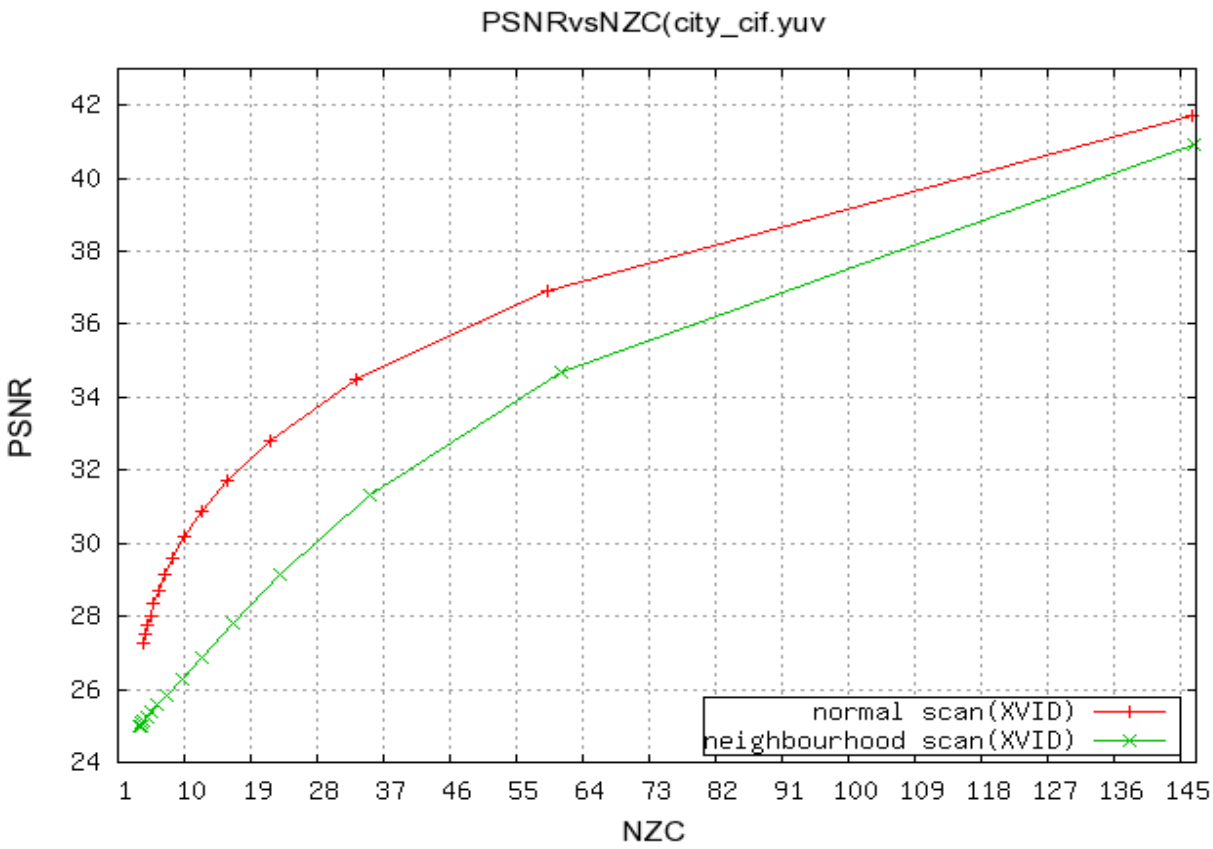
A.10 RD CURVES (XVID IMPLEMENTATION)

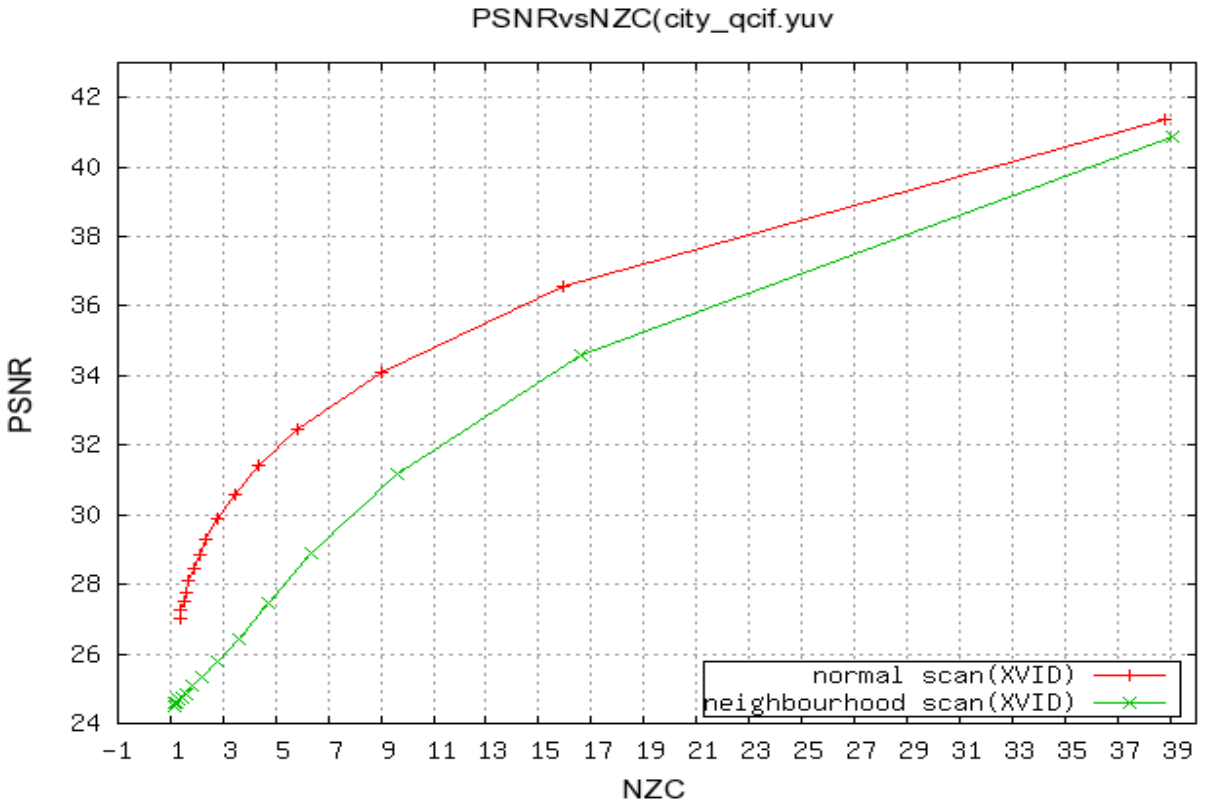


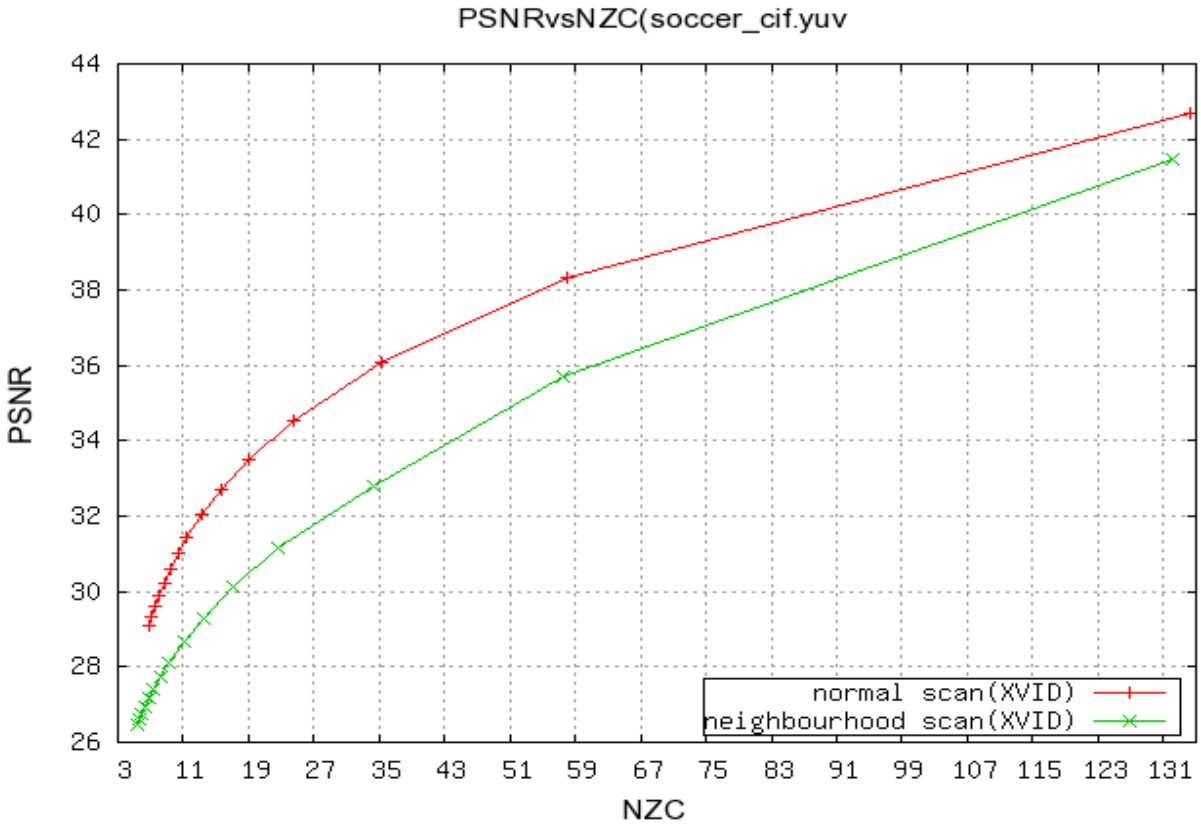


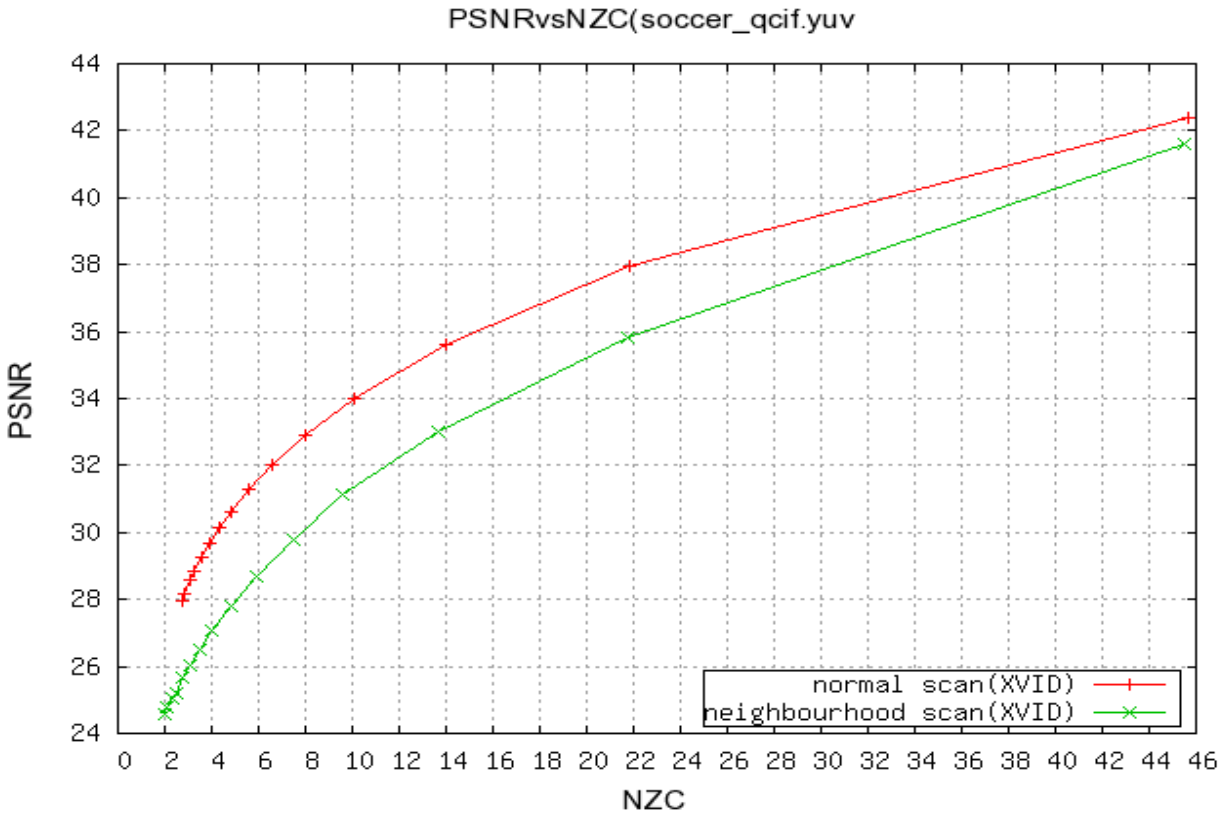

PSNRvsNZC(flower_cif.yuv
PSNR
NZC
43
40
37
34
31
28
25
22
19
8
27
46
65
84
103
122
141
160
179
198
217
236
255
274
293
312
normal scan(XVID)
neighbourhood scan(XVID)


PSNRvsNZC(flower_qcif.yuv
PSNR
NZC
43
40
37
34
31
28
25
22
19
0
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
normal scan(XVID)
neighbourhood scan(XVID)


PSNRvsNZC(city_cif.yuv
PSNR
NZC
normal scan(XVID)
neighbourhood scan(XVID)
42
40
38
36
34
32
30
28
26
24
1 10 19 28 37 46 55 64 73 82 91 100 109 118 127 136 145


PSNRvsNZC(city_qcif.yuv
PSNR
NZC
normal scan(XVID)
neighbourhood scan(XVID)
42
40
38
36
34
32
30
28
26
24
-1 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39


PSNRvsNZC(soccer_cif.yuv
PSNR
NZC
normal scan(XVID)
neighbourhood scan(XVID)
44
42
40
38
36
34
32
30
28
26
3
11
19
27
35
43
51
59
67
75
83
91
99
107
115
123
131


PSNRvsNZC(soccer_qcif.yuv
PSNR
NZC
normal scan(XVID)
neighbourhood scan(XVID)
44
42
40
38
36
34
32
30
28
26
24
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
44
46

APPENDIX B:- XVID MPEG-4 ENCODER

B.1 OVERVIEW

Xvid is an open-source research project focusing on video compression and is a collaborative development effort. All code is released under the terms of the GNU GPL license.

The Xvid video codec implements MPEG-4 Simple Profile and Advanced Simple Profile standards. It permits compressing and decompressing digital video in order to reduce the required bandwidth of video data for transmission over computer networks or efficient storage on CDs or DVDs. Due to its unrivalled quality Xvid has gained great popularity and is used in many other GPLed applications, like e.g. Transcode, MEncoder, MPlayer, Xine and many more.

B.2 GOALS

Xvid has been created to promote the adoption of open standards, namely MPEG-4 video, and to permit the interoperable creation and exchange of digital video between different software applications and a wide range of devices.

A major goal of the project is to further trigger the creation of GPLed applications and to support the Free Software movement. Hence, the Xvid source code is published under the terms of the GNU General Public License, which requires that combined and derived works must be distributed as a whole under the GPL again.

B.3 FEATURES

Since being founded in 2001, the Xvid project focuses on MPEG-4 video compression. Early Xvid versions (0.9.x) implemented MPEG-4 simple profile (SP) de- and encoding. The Xvid 1.0 source tree introduced MPEG-4 advanced simple profile (ASP) compression including all advanced coding tools like B-frames, quarterpixel motion compensation and GMC. The upcoming development branch Xvid 2.0 adds support for MPEG-4 advanced video coding (AVC) de- and encoding up to High Profile and dramatically advances upon the compression performance of earlier Xvid versions.

The most outstanding feature of Xvid is the excellent picture quality it provides even at high compression ratios. A major emphasis of the project is put on research to achieve highest possible picture quality and novel algorithms have been developed that enable outstanding detail-richness and image fidelity. The

Xvid codec is especially optimized towards offline, multi-pass compression for storage and archival purposes and has been found to deliver superior quality over any other MPEG-4 part 2 based codec by independent third parties.

Xvid is primarily developed for PC (Linux, Windows) but also other popular platforms are supported such as Mac/iMac. Due to extensive algorithmic optimizations and specific support for latest PC processor extensions like SSE2/SSE3 SIMD instruction sets or dual-core/hyper threading technology, Xvid provides highest performance video de- and encoding on modern PCs. The Xvid ASP and AVC de- and encoder provide highest performance on PC and enable HD resolution video processing on today's hardware. Hence, the Xvid codec library is frequently used in GPLed software players or encoder applications on Windows and Linux such as MPlayer or Transcode.

APPENDIX C:- XVID STATISTICS

Here the statistics for the sequence Ice are presented as mentioned in chapter 4.

Table C1 Last, Run, Level statistics (Normal scan)

LastRunLevel	Count	Probability	Code	Length
0$0$1	79674	0.333147	4	3
0$1$1	29204	0.122113	13	4
0$0$2	24704	0.103297	31	5
0$2$1	12129	0.050716	28	5
0$0$3	11492	0.048052	43	7
1$0$1	7120	0.029771	15	5
0$0$4	6779	0.028346	46	8
0$3$1	5757	0.024072	27	6
0$1$2	5295	0.02214	40	7
0$4$1	5054	0.021133	24	6
0$5$1	4533	0.018954	22	6
0$0$5	4113	0.017198	63	9
0$0$6	2557	0.010692	75	10
1$1$1	2540	0.010621	30	7
0$6$1	2213	0.009253	38	7
1$2$1	2167	0.009061	29	7
0$1$3	2138	0.00894	61	9
0$7$1	1932	0.008078	36	7
0$0$7	1830	0.007652	73	10
1$3$1	1633	0.006828	26	7
1$4$1	1620	0.006774	25	7
0$8$1	1439	0.006017	35	7
0$0$8	1246	0.00521	67	11
1$5$1	1217	0.005089	39	8
1$6$1	1150	0.004809	36	8
0$9$1	1137	0.004754	32	7
1$7$1	1069	0.00447	35	8
0$2$2	1022	0.004273	58	9
1$8$1	985	0.004119	33	8
0$0$9	962	0.004022	65	11
0$1$4	790	0.003303	30	11
0$0$10	680	0.002843	15	12
1$9$1	593	0.00248	53	9
1$11$1	550	0.0023	48	9
0$0$11	550	0.0023	12	12
1$12$1	534	0.002233	47	9
0$10$1	534	0.002233	45	8
0$11$1	507	0.00212	42	8
0$3$2	495	0.00207	71	10

0$12$1	454	0.001898	41	8
1$10$1	449	0.001877	50	9
1$13$1	448	0.001873	44	9
0$0$12	393	0.001643	65	12
0$1$5	387	0.001618	67	12
0$2$3	379	0.001585	29	11
0$4$2	371	0.001551	68	10
1$14$1	360	0.001505	42	9
0$0$13	351	0.001468	52	11
0$0$14	278	0.001162	223	13
0$3$3	247	0.001033	27	11
0$1$6	233	0.000974	161	13
0$0$15	218	0.000912	811	15
1$16$1	182	0.000761	38	9
1$15$1	176	0.000736	41	9
0$1$7	170	0.000711	109	12
0$0$16	169	0.000707	1582	16
0$13$1	166	0.000694	56	9
0$5$2	156	0.000652	24	11
0$0$17	150	0.000627	3134	17
0$6$2	136	0.000569	22	11
0$2$4	135	0.000564	162	13
1$17$1	134	0.00056	49	10
1$0$2	133	0.000556	51	10
0$14$1	131	0.000548	55	9
0$7$2	118	0.000493	21	11
0$0$18	107	0.000447	6218	18
1$18$1	98	0.00041	47	10
0$0$19	95	0.000397	6216	18
1$19$1	93	0.000389	45	10
0$2$5	87	0.000364	221	13
0$1$8	87	0.000364	809	15
0$0$20	80	0.000335	12355	19
1$20$1	74	0.000309	43	10
1$21$1	73	0.000305	41	10
0$8$2	73	0.000305	18	11
0$4$3	72	0.000301	165	13
1$22$1	71	0.000297	38	10
1$24$1	65	0.000272	35	10
0$0$21	65	0.000272	12353	19
0$16$1	64	0.000268	65	10
1$23$1	58	0.000243	37	10
0$0$22	54	0.000226	24590	20
0$15$1	53	0.000222	67	10
0$9$2	51	0.000213	17	11
0$18$1	49	0.000205	60	10
0$17$1	47	0.000197	62	10

0$0$24	47	0.000197	24641	20
0$5$3	45	0.000188	167	13
0$19$1	44	0.000184	59	10
0$0$23	44	0.000184	24589	20
1$26$1	37	0.000155	13	11
0$1$9	37	0.000155	3132	17
0$0$26	37	0.000155	31465525	30
1$0$3	36	0.000151	10	12
0$6$3	36	0.000151	169	13
1$1$2	35	0.000146	9	12
0$20$1	35	0.000146	56	10
0$3$4	33	0.000138	411	14
0$4$4	31	0.00013	409	14
0$22$1	30	0.000125	52	10
1$25$1	29	0.000121	15	11
0$21$1	26	0.000109	55	10
1$27$1	25	0.000105	10	11
0$0$28	23	0.000096	31473609	30
1$29$1	20	0.000084	72	12
0$1$10	20	0.000084	12319	19
1$30$1	19	0.000079	75	12
0$0$25	19	0.000079	31465523	30
0$2$6	18	0.000075	3131	17
0$0$29	18	0.000075	31465531	30
0$0$27	18	0.000075	31473611	30
1$28$1	17	0.000071	9	11
0$5$4	16	0.000067	406	14
0$0$30	16	0.000067	31465533	30
0$10$2	14	0.000059	170	13
0$7$3	12	0.00005	804	15
0$0$31	11	0.000046	31465535	30
1$33$1	10	0.000042	177	13
1$32$1	10	0.000042	79	12
0$1$12	10	0.000042	49313	21
0$0$34	10	0.000042	31473597	30
1$37$1	9	0.000038	184	13
1$31$1	9	0.000038	77	12
0$24$1	9	0.000038	70	12
0$23$1	9	0.000038	68	12
0$0$33	9	0.000038	31465539	30
0$0$32	9	0.000038	31473601	30
0$8$3	8	0.000033	802	15
0$3$6	8	0.000033	12314	19
0$0$38	8	0.000033	31465549	30
0$0$35	8	0.000033	31473595	30
1$6$2	7	0.000029	1572	16
1$5$2	6	0.000025	1575	16

1$4$2	6	0.000025	792	15
1$3$2	6	0.000025	794	15
1$2$2	6	0.000025	796	15
0$11$2	6	0.000025	1579	16
0$0$37	6	0.000025	31473591	30
1$9$2	5	0.000021	3125	17
1$7$2	5	0.000021	1570	16
1$39$1	5	0.000021	188	13
1$38$1	5	0.000021	186	13
1$34$1	5	0.000021	179	13
0$6$5	5	0.000021	12311	19
0$6$4	5	0.000021	806	15
0$25$1	5	0.000021	172	13
0$1$11	5	0.000021	24642	20
0$27$1	4	0.000017	117	12
0$0$49	4	0.000017	31473567	30
0$0$45	4	0.000017	31465563	30
0$0$41	4	0.000017	31465555	30
1$42$1	3	0.000013	1823	16
1$40$1	3	0.000013	190	13
1$35$1	3	0.000013	181	13
1$11$2	3	0.000013	3121	17
1$10$2	3	0.000013	3122	17
0$9$3	3	0.000013	800	15
0$28$1	3	0.000013	237	13
0$26$1	3	0.000013	174	13
0$2$8	3	0.000013	49315	21
0$0$39	3	0.000013	31473587	30
0$0$36	3	0.000013	31465545	30
1$50$1	2	0.000008	7221	18
1$49$1	2	0.000008	3616	17
1$48$1	2	0.000008	3619	17
1$44$1	2	0.000008	1819	16
1$41$1	2	0.000008	463	14
1$12$2	2	0.000008	3119	17
1$0$4	2	0.000008	206	13
0$4$6	2	0.000008	49317	21
0$39$1	2	0.000008	3624	17
0$33$1	2	0.000008	1831	16
0$32$1	2	0.000008	919	15
0$29$1	2	0.000008	476	14
0$0$46	2	0.000008	31465565	30
0$0$43	2	0.000008	31465559	30
0$0$42	2	0.000008	31473581	30
0$0$40	2	0.000008	31465553	30
1$63$1	1	0.000004	14375	19
1$55$1	1	0.000004	7211	18

1$53$1	1	0.000004	7214	18
1$52$1	1	0.000004	7217	18
1$46$1	1	0.000004	3622	17
1$43$1	1	0.000004	1821	16
1$36$1	1	0.000004	182	13
1$12$7	1	0.000004	32718835	30
1$1$3	1	0.000004	799	15
0$6$6	1	0.000004	49321	21
0$41$1	1	0.000004	7222	18
0$4$5	1	0.000004	6212	18
0$37$1	1	0.000004	3628	17
0$34$1	1	0.000004	1829	16
0$31$1	1	0.000004	921	15
0$30$1	1	0.000004	923	15
0$3$5	1	0.000004	6215	18
0$2$7	1	0.000004	12316	19
0$10$3	1	0.000004	1581	16
0$1$29	1	0.000004	31481915	30
0$1$27	1	0.000004	31481911	30
0$0$82	1	0.000004	31473501	30
0$0$72	1	0.000004	31473521	30
0$0$64	1	0.000004	31465601	30
0$0$54	1	0.000004	31465581	30
0$0$52	1	0.000004	31473561	30
0$0$48	1	0.000004	31473569	30
0$0$47	1	0.000004	31473571	30
0$0$44	1	0.000004	31473577	30

Table C2 Last, Run, Level statistics (Neighbourhood scan)

LastRunLevel	Count	Probability	Code	Length
0$0$1	70426	0.306901	4	3
0$1$1	27453	0.119634	13	4
0$0$2	21790	0.094956	31	5
0$2$1	12294	0.053574	28	5
0$0$3	11075	0.048262	43	7
0$0$4	6687	0.02914	46	8
1$0$1	6364	0.027733	15	5
0$3$1	6045	0.026343	27	6
0$4$1	5457	0.02378	24	6
0$5$1	4885	0.021288	22	6
0$1$2	4725	0.02059	40	7
0$0$5	4072	0.017745	63	9
0$0$6	2512	0.010947	75	10
0$6$1	2432	0.010598	38	7
1$1$1	2401	0.010463	30	7

0$7$1	2313	0.01008	36	7
0$1$3	2081	0.009069	61	9
1$2$1	1941	0.008458	29	7
0$0$7	1886	0.008219	73	10
0$8$1	1806	0.00787	35	7
0$9$1	1704	0.007426	32	7
1$3$1	1612	0.007025	26	7
1$4$1	1592	0.006938	24	7
0$0$8	1244	0.005421	67	11
1$5$1	1189	0.005181	39	8
1$6$1	1099	0.004789	36	8
1$7$1	1074	0.00468	35	8
0$0$9	994	0.004332	65	11
0$2$2	990	0.004314	58	9
1$8$1	899	0.003918	33	8
0$10$1	827	0.003604	45	8
0$1$4	769	0.003351	30	11
0$12$1	724	0.003155	40	8
0$11$1	724	0.003155	42	8
0$0$10	665	0.002898	15	12
1$9$1	589	0.002567	53	9
1$10$1	572	0.002493	51	9
0$0$11	543	0.002366	12	12
1$12$1	536	0.002336	47	9
1$11$1	525	0.002288	48	9
1$13$1	502	0.002188	44	9
0$3$2	426	0.001856	71	10
1$14$1	397	0.00173	43	9
0$2$3	389	0.001695	29	11
0$1$5	380	0.001656	67	12
0$0$12	375	0.001634	65	12
0$0$13	368	0.001604	52	11
0$4$2	343	0.001495	68	10
0$13$1	328	0.001429	56	9
0$14$1	311	0.001355	54	9
1$15$1	298	0.001299	41	9
1$16$1	271	0.001181	39	9
0$0$14	245	0.001068	223	13
0$1$6	221	0.000963	161	13
0$3$3	211	0.000919	27	11
0$0$15	205	0.000893	811	15
0$1$7	193	0.000841	109	12
0$0$16	179	0.00078	1582	16
1$17$1	172	0.00075	49	10
0$17$1	170	0.000741	63	10
0$5$2	169	0.000736	24	11
0$16$1	166	0.000723	65	10

1$18$1	165	0.000719	47	10
0$20$1	165	0.000719	57	10
1$20$1	162	0.000706	43	10
0$19$1	161	0.000702	58	10
0$18$1	158	0.000689	60	10
1$19$1	152	0.000662	44	10
0$21$1	152	0.000662	54	10
0$22$1	147	0.000641	52	10
0$15$1	146	0.000636	66	10
0$0$17	146	0.000636	3134	17
1$22$1	138	0.000601	38	10
1$0$2	137	0.000597	45	8
1$24$1	135	0.000588	34	10
0$6$2	133	0.00058	22	11
0$7$2	132	0.000575	21	11
1$23$1	130	0.000567	36	10
1$21$1	123	0.000536	41	10
0$2$4	122	0.000532	162	13
0$0$18	115	0.000501	6218	18
0$0$19	97	0.000423	6216	18
1$25$1	93	0.000405	14	11
0$2$5	86	0.000375	221	13
1$27$1	84	0.000366	11	11
1$30$1	78	0.00034	74	12
1$26$1	77	0.000336	13	11
1$28$1	71	0.000309	9	11
0$4$3	69	0.000301	165	13
0$0$21	69	0.000301	12353	19
1$29$1	68	0.000296	72	12
0$1$8	68	0.000296	809	15
0$8$2	63	0.000275	18	11
0$27$1	63	0.000275	117	12
0$5$3	59	0.000257	167	13
0$0$20	59	0.000257	12355	19
0$24$1	58	0.000253	70	12
1$34$1	56	0.000244	179	13
1$31$1	55	0.00024	77	12
1$32$1	54	0.000235	79	12
0$0$24	52	0.000227	24641	20
0$0$22	51	0.000222	24591	20
1$33$1	50	0.000218	177	13
1$1$2	50	0.000218	9	12
0$9$2	49	0.000214	17	11
0$23$1	45	0.000196	69	12
0$1$9	45	0.000196	3132	17
0$0$23	40	0.000174	24589	20
1$38$1	39	0.00017	187	13

1$37$1	39	0.00017	184	13
0$26$1	39	0.00017	175	13
0$3$4	37	0.000161	411	14
1$35$1	36	0.000157	180	13
0$28$1	36	0.000157	236	13
1$40$1	35	0.000153	191	13
1$36$1	34	0.000148	183	13
0$6$3	34	0.000148	169	13
0$4$4	34	0.000148	408	14
0$0$28	33	0.000144	31473609	30
1$39$1	31	0.000135	188	13
1$0$3	30	0.000131	10	12
1$41$1	28	0.000122	463	14
0$25$1	26	0.000113	173	13
0$0$26	26	0.000113	31465525	30
1$43$1	22	0.000096	1821	16
1$42$1	21	0.000092	1823	16
0$2$6	21	0.000092	3131	17
0$0$29	21	0.000092	31465531	30
0$0$27	21	0.000092	31473611	30
0$0$25	21	0.000092	31465523	30
0$32$1	20	0.000087	918	15
0$1$10	18	0.000078	12319	19
1$48$1	17	0.000074	3618	17
1$49$1	16	0.00007	3617	17
0$34$1	16	0.00007	1828	16
0$30$1	16	0.00007	923	15
1$44$1	15	0.000065	1819	16
1$45$1	13	0.000057	1817	16
0$31$1	13	0.000057	920	15
0$29$1	13	0.000057	476	14
0$0$32	12	0.000052	31473601	30
0$36$1	11	0.000048	1825	16
0$0$30	11	0.000048	31465533	30
1$46$1	10	0.000044	3622	17
0$8$3	10	0.000044	803	15
0$33$1	10	0.000044	1830	16
0$1$11	10	0.000044	24642	20
1$51$1	9	0.000039	7218	18
1$50$1	9	0.000039	7220	18
0$35$1	9	0.000039	1827	16
0$0$31	9	0.000039	31465535	30
1$53$1	8	0.000035	7214	18
1$52$1	8	0.000035	7216	18
0$41$1	8	0.000035	7222	18
0$0$35	8	0.000035	31473595	30
1$47$1	7	0.000031	3620	17

0$7$3	7	0.000031	804	15
0$5$4	7	0.000031	407	14
0$3$5	7	0.000031	6215	18
0$10$2	7	0.000031	171	13
0$0$33	7	0.000031	31465539	30
1$59$1	6	0.000026	14383	19
1$57$1	6	0.000026	7207	18
1$56$1	6	0.000026	7208	18
1$55$1	6	0.000026	7211	18
1$1$3	6	0.000026	799	15
1$0$4	6	0.000026	206	13
0$6$4	6	0.000026	806	15
0$40$1	6	0.000026	7225	18
0$4$5	6	0.000026	6212	18
0$0$34	6	0.000026	31473597	30
1$7$2	5	0.000022	1570	16
1$54$1	5	0.000022	7213	18
1$5$2	5	0.000022	1574	16
0$6$5	5	0.000022	12311	19
0$42$1	5	0.000022	14402	19
0$39$1	5	0.000022	3624	17
0$11$2	5	0.000022	1579	16
0$0$45	5	0.000022	31465563	30
0$0$37	5	0.000022	31473591	30
1$9$2	4	0.000017	3125	17
1$6$2	4	0.000017	1572	16
1$2$2	4	0.000017	796	15
0$47$1	4	0.000017	14393	19
0$1$12	4	0.000017	49313	21
0$0$42	4	0.000017	31473581	30
0$0$39	4	0.000017	31473587	30
1$8$2	3	0.000013	1568	16
1$61$1	3	0.000013	14378	19
1$60$1	3	0.000013	14381	19
1$4$2	3	0.000013	792	15
0$38$1	3	0.000013	3627	17
0$37$1	3	0.000013	3629	17
0$12$2	3	0.000013	1577	16
0$0$41	3	0.000013	31465555	30
0$0$36	3	0.000013	31473593	30
1$58$1	2	0.000009	14385	19
1$3$2	2	0.000009	795	15
1$0$6	2	0.000009	24587	20
0$9$3	2	0.000009	800	15
0$7$4	2	0.000009	12308	19
0$49$1	2	0.000009	14388	19
0$44$1	2	0.000009	14399	19

0$4$6	2	0.000009	49316	21
0$2$8	2	0.000009	49315	21
0$10$3	2	0.000009	1580	16
0$0$51	2	0.000009	31473563	30
0$0$50	2	0.000009	31473565	30
0$0$49	2	0.000009	31473567	30
0$0$38	2	0.000009	31465549	30
1$62$1	1	0.000004	14376	19
1$11$2	1	0.000004	3120	17
1$10$6	1	0.000004	32686069	30
0$8$4	1	0.000004	12306	19
0$52$1	1	0.000004	114861	22
0$51$1	1	0.000004	57415	21
0$45$1	1	0.000004	14397	19
0$3$6	1	0.000004	12315	19
0$2$7	1	0.000004	12316	19
0$2$27	1	0.000004	31498295	30
0$1$31	1	0.000004	31489987	30
0$1$29	1	0.000004	31481915	30
0$0$82	1	0.000004	31473501	30
0$0$71	1	0.000004	31473523	30
0$0$55	1	0.000004	31465583	30
0$0$52	1	0.000004	31465577	30
0$0$47	1	0.000004	31473571	30
0$0$43	1	0.000004	31465559	30