Institutionen för Systemteknik

Department of Electrical Engineering

Master Thesis

Multi-View Video Transmission over the Internet

By

Abdullah Jan Mirza

Mahmood Fateh Ahsan

LiTH-ISY-EX - - 10/4409 - - SE

Linköpings universitet
SE-581 83 Linköping, Sweden

Linköping universitet
581 83 Linköping

Master thesis

Multi-View Video Transmission over the Internet

By

Abdullah Jan Mirza

Registration no: 781206-3399

Mahmood Fateh Ahsan

Registration no: 800815-1774

LiTH-ISY-EX - - 10/4409 - - SE

Examiner: Robert Forchheimer

Advisor: Ingemar Ragnemalm

Abstract

3D television using multiple views rendering is receiving increasing interest. In this technology a number of video sequences are transmitted simultaneously and provides a larger view of the scene or stereoscopic viewing experience. With two views stereoscopic rendition is possible. Nowadays 3D displays are available that are capable of displaying several views simultaneously and the user is able to see different views by moving his head.

The thesis work aims at implementing a demonstration system with a number of simultaneous views. The system will include two cameras, computers at both the transmitting and receiving end and a multi-view display. Besides setting up the hardware, the main task is to implement software so that the transmission can be done over an IP-network.

This thesis report includes an overview and experiences of similar published systems, the implementation of real time video, its compression, encoding, and transmission over the internet with the help of socket programming and finally the multi-view display in 3D format. This report also describes the design considerations more precisely regarding the video coding and network protocols.

Key Words: - Multi-view Coding, Stereoscopic, H.264/AVC, Video Compression, Moving Picture Experts Group (MPEG), MPEG-4, 3DTV, Sockets, Streaming Server

Acknowledgment

We would like to thank our Prof Robert Forchheimer for providing us an opportunity to write a thesis with the image coding group at the Electrical Engineering department at Linköping University and providing us hardware along with his guidance and support throughout our work.

We would also like to thank our examiner Prof Ingemar Ragnemalm for his support, guidance and suggestions throughout the thesis work. We appreciate him for giving us time, up-gradation of hardware components, suggestions and interesting technical discussions to keep us on the right track. The work has completed in Bildkodning Laboratory.

We would also like to thank Mr. Mudassar Majeed for his valuable suggestion during the thesis work.

Finally we can never forget to talk about our beloved parents and Government of Sweden who provided us such a decent opportunity for doing a research work and getting the master's degree from Linköping University in Sweden.

Presentation Date 2010-04-09 **Publishing Date (Electronic version)** 2010-07-04	**Department and Division** ISY and Image Coding Group
URL, Electronic Version http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-57903	

Language [✓ English] Other (specify below) **Number of Pages** 73	**Type of Publication** Licentiate thesis [✓ Degree thesis] Thesis C-level Thesis D-level Report Other (specify below)	**ISBN (Licentiate thesis)** **ISRN:** LiTH-ISY-EX - - 10/4409 - - SE **Title of series (Licentiate thesis)** **Series number/ISSN (Licentiate thesis)**

Publication Title
Multi-View Video Transmission over the Internet

Author(s)
ABDULLAH JAN MIRZA , MAHMOOD FATEH AHSAN

Abstract
3D television using multiple views rendering is receiving increasing interest. In this technology a number of video sequences are transmitted simultaneously and provides a larger view of the scene or stereoscopic viewing experience. With two views stereoscopic rendition is possible. Nowadays 3D displays are available that are capable of displaying several views simultaneously and the user is able to see different views by moving his head.
The thesis work aims at implementing a demonstration system with a number of simultaneous views. The system will include two cameras, computers at both the transmitting and receiving end and a multi-view display. Besides setting up the hardware, the main task is to implement software so that the transmission can be done over an IP-network.
This thesis report includes an overview and experiences of similar published systems, the implementation of real time video, its compression, encoding, and transmission over the internet with the help of socket programming and finally the multi-view display in 3D format. This report also describes the design considerations more precisely regarding the video coding and network protocols

Keywords
Multi-view Coding, Stereoscopic, H.264/AVC, Video Compression, Moving Picture Experts Group (MPEG), MPEG-4, 3DTV, Sockets, Streaming Server

Table of Contents

Table of Figures

1

Introduction

Multi-view video transmission is an emerging multimedia technology. The multi-view video content delivery systems are capable to stream multiple simultaneous views of the same scene at a time. The span of the scene depends upon the number of cameras so that end user is able to see a larger view of the scene. The system typically requires several cameras, encoders, decoders, streaming server and a high processing capability.

The multi-view video provides a larger view of the scene. While using multi-view video approach we have multiple video streams of the same scene at user's location and therefore we can also use these video streams to generate stereoscopic 3D video. The stereoscopic 3D video can be watched on 3D displays or with the use of a special color anaglyph glasses.

The demand of multi-view video is increasing in the area of multimedia technology. This technology can be used for coverage of sports events, broadcasting of educational seminars and in medical field etc. Recently many world leading television manufacturers launched 3D TVs and also renowned satellite broadcaster are planning to launch their dedicated 3D Channels. Therefore we can predict that a 3D video technology will replace the 2D technology.

1.1 Background

In the recent years streaming of multi-view video content has received much attention as it enhances users viewing experience. In Multi-view video transmission several video sequences are transmitted simultaneously with slightly different views of the same scene. Stereoscopic viewing experience makes this technology more attractive for the viewers. The display mechanism is capable of displaying simultaneous views of a scene so that the user is able to see a larger view of the scene. The larger view means that redundant area from the scene is removed and views are combined to make one larger view of the scene. A single video camera cannot capture the whole scene, therefore we can use several cameras to cover a particular scene and by removing redundancy in a scene we can make one larger view.

Multi-view coding (MVC) involves the streaming of multi-view video. In MVC we exploit the redundancy from the captured views. In this thesis work our aim was to implement a system which will be capable to capture simultaneous views of a scene in real time with help of several video cameras mounted on specific position and shows on a stereoscopic 3D display or to show a larger view of the scene. More precisely the simultaneous view video will be encoded and streamed out over the IP network. On the receiving end this streamed video will be decoded and hence displayed on a 3D display.

Multi-view video provides a new viewing experience with a high degree of freedom. However, it also brings challenges to data delivery due to the huge data amount that is needed to be transmitted. In figure 1.1 a typical multi-view video system is presented with several cameras. On the receiving end a High-Definition TV with a 3D Display can be used.

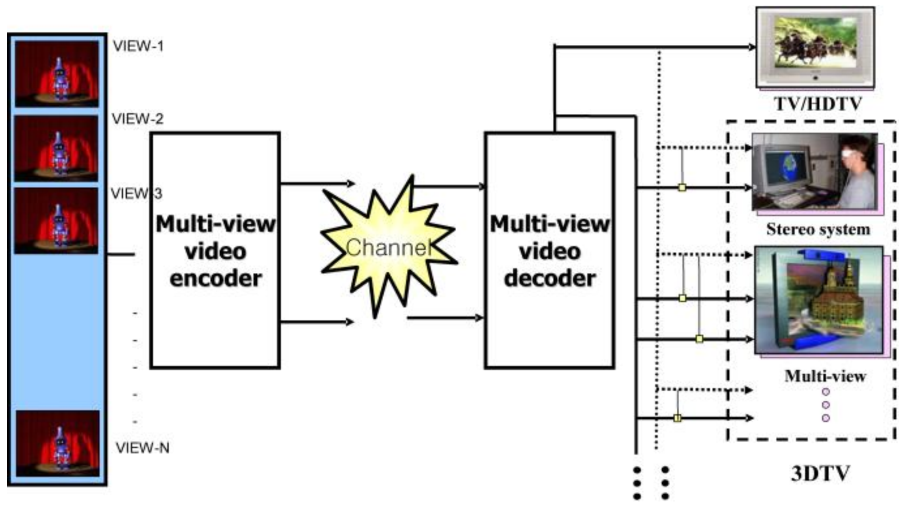


Figure 1.1 Multi-view System. Courtesy from MERL[1]

[1] Mitsubishi Electric Research Laboratories is the North American arm of the Corporate R&D organization of the Mitsubishi Electric Corporation

1.2 Purpose

This thesis work aims to implement a demonstrating system in which several simultaneous video streams can be transmitted over the internet and an end user can be able to watch a larger view of the scene or a 3D video. The main challenges are efficient transmission of huge data over a public network where we have many limitations e.g. bandwidth. We also include an overview and experiences of similar published systems, a description of the design considerations particularity regarding the video coding and network protocols.

1.3 Limitations

The main focus of the thesis is the transmission of the simultaneous views over the internet by installing several video cameras. We performed our experimental work with two cameras only because of unavailability of more cameras.

In hardware the stereoscopic 3D display is also not available therefore we have to show results on a normal LCD screen therefore we used special anaglyph glasses (Red, Green) for analyzing the result on a screen.

1.4 Methodology

This work is consisted of the literature studies, practical evaluations of existing open source software and analysis of existing multi-view systems. During the literature studies we focused on multi-view coding, stereoscopic viewing, multi-view video compression and its transmission, anaglyph video and a 3D video technology.

OpenCV and FFMPEG are used to acquire the video streams and video manipulation. Java Programming language is used for the client server communication using Java’s built-in network classes.

Most of the definitions are taken from RFC’s, books and Internet. Research papers are also consulted to compare the current systems and for the general understanding of the work to be performed.

1.5 Structure

Chapter one is an introduction of the thesis work. In chapter two we present an introduction of multi-view coding and its coding approaches. We also presented a brief introduction to stereoscopic 3D technology in this chapter. In chapter three we discuss about the video encoding, decoding and compression within context of multi-view video transmission environment. In chapter four we present the method of a video streaming over an IP network. Further in this chapter we analyzed which protocol is suitable for multi-view video transmission. In chapter five we present our implementation part of the thesis work with the help of programming code in OPENCV, FFMPEG and Java. In chapter six we discuss about future work that can be added as an extension to our thesis work. At the end of each chapter we include a summary so that the reader is able to have quick look at content of each chapter.

2
Multi-View Coding

2.1 Introduction

Multi-view coding (MVC) is a technique to encode the multiple video sequences which are simultaneously captured from several cameras. MVC provides a wider view of the scene depending upon number of cameras. MVC technique is used for stereoscopic videos, 3D television and a free viewpoint television. In MVC we exploited redundancy in video sequences to save maximum bandwidth in case of video streaming. Streaming of multi-view video content over the internet requires higher bandwidth and therefore different coding techniques are used to efficiently stream multi-view content over the network. At the receiving side users can view a wider scene view or watch it on a stereoscopic 3D display.

In this chapter MVC and 3D stereoscopic technologies are discussed.

2.2 Overview

In recent years multi-view video viewing has been an emerging multimedia technology. In general, the multi-view video processing requires encoders and decoders as many as the number of cameras and thus the processing complexity results in difficulties of practical implementation. With the progress of information technologies and the demand of consumers, a variety of multimedia contents have been served to the consumers. Recently beyond 2D and stereoscopic video, multi-view video acquired from multiple cameras had been a newly emerging media and can provide a wider view to the consumers with immersive perception of natural scenes [1].

In contrast to traditional single view-point video, the multi-view point video requires a much higher bandwidth for transmission. Hence, it is critical that advanced compression techniques are used for coding of multi-view video data in terms of exploitation of redundancies in temporal domain as well as exploitation of inter-view redundancies. Certain types of proposed multi-view video coding algorithms include motion compensation based prediction from inter-view and temporal references in a multi-view video set. The already built in motion estimation and motion compensation functions of the modern video codec's, especially of H.264/AVC, are used to remove temporal redundancies inside a certain view itself and as well as to remove spatial redundancies between neighboring views after slight modifications. The MVC method proposed in [2] based on hierarchical B frame prediction in both temporal and spatial dimension using H.264/AVC gives the best compression performance so far among other reference frame based proposal [3].

Streaming of multi-view video content has received much attention as it enhances a user viewing experience. The multi-view video coding (MVC) systems capture video streams using multiple cameras located at different angles and positions. Unlike single view video coders which perform spatial and temporal decorrelation, MVC schemes also exploit the redundancy among adjacent views. The MVC schemes, as the one presented in [4], can use synthesis prediction and multi-view reference picture management. However, the generated MVC coded streams are sensitive to channel errors and few errors may cause significant degradation in video quality. Thus the design of a reliable multi-view transmission system involves sophisticated selection of employed error resilient tools and channel coding techniques [5].

In figure 2.1 a multi-view video streaming system is presented which is capable to stream five simultaneous views at a time.

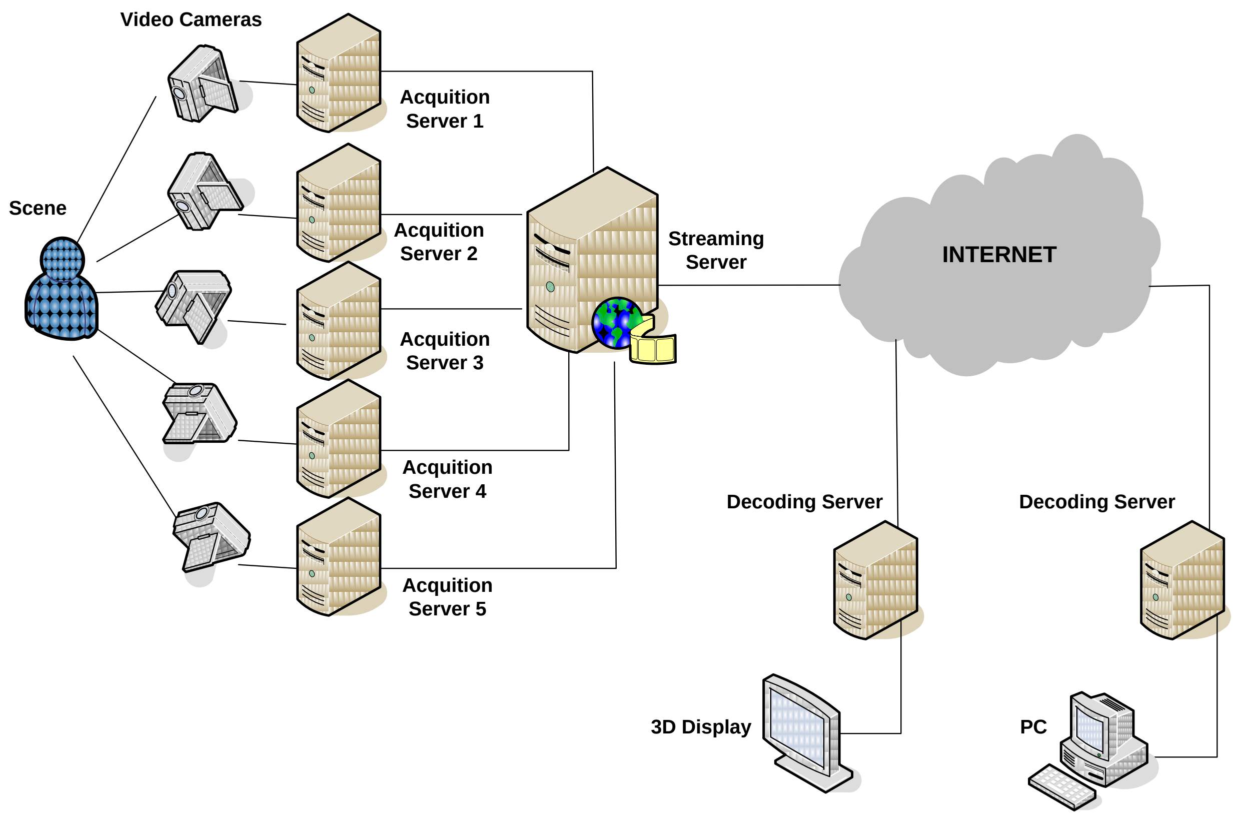


Figure 2.1 Proposed Multi-view System

In the proposed system five cameras acquire the live streams of a scene and send the streams to encoders. The encoders will encode these five streams independently and send them to a streaming server. At the streaming server these streams will be compressed and put together in one packet before streaming over the Internet. At the receiving end these streams will be split again and delivered to a display system. The viewer can watch multi-view video with wider view or a stereoscopic 3D video.

Typically in multi-view coding two or more cameras can be used to capture views. These views are acquired by an acquisition server and then all views are encoded and decoded independently and then displayed simultaneously. In some cases one single pc can be act as an acquisition server, an encoder and the streaming server. This requires a machine with a sufficient processing power.

In figure 2.2 shows the entire process about proposed system in which videos are acquired by two cameras. The encoding and compression are applied on acquired video. The transmission part involves the streaming of video over the internet. At receiving side decoder decodes the receiving videos and then videos are displayed.

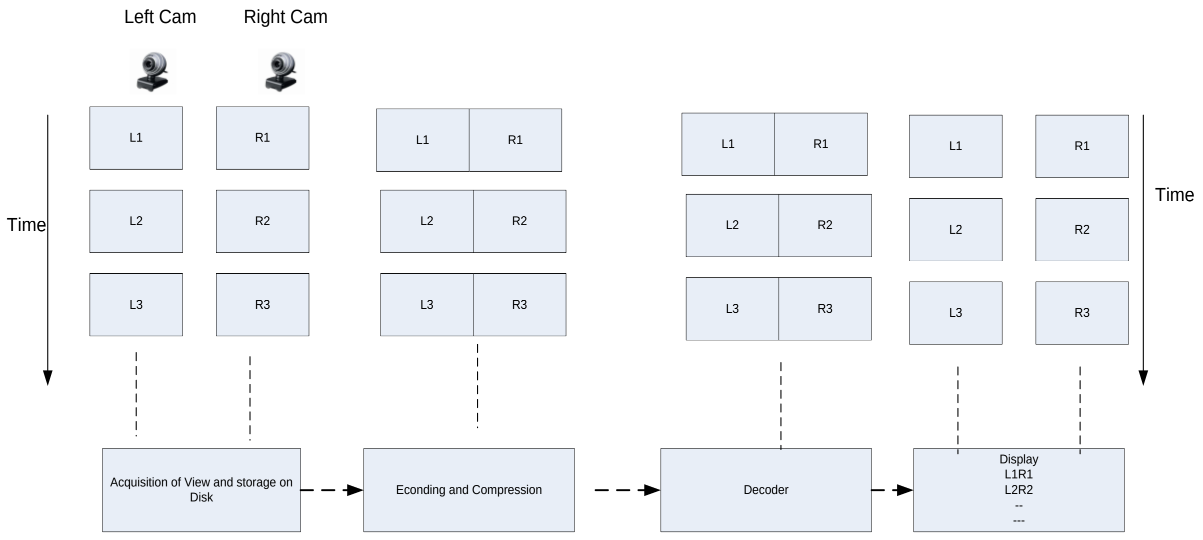


Figure 2.2 The processing of left and right views

The main parts of proposed system are presented below:

Acquisition

The Logitech webcams are used for the video acquisition purpose. These cameras are directly attached to acquisition server through USB ports. The cameras are set on 25fps and different frame rates are also tested. The time for each video for capturing views are set to 30 seconds at both cameras and the videos are stored on hard disk. These cameras are mounted on particular location and their focusing is manually adjusted. The cameras used in the project are shown in Appendix “A”.

Encoding

The open source software FFMPEG and Open Computer Vision (a library of programming functions for real time computer vision) are used to encoding the video. While acquiring the video the raw video is converted to required format. The format of video is selected according to bandwidth requirement and quality of video.

Transmission

In multi-view video coding bandwidth management is always an issue. Due to multiple video streams excessive amount of data is generated and therefore we need robust transmission technique so that data can be transmitted efficiently. Before transmitting the packet the left and right view videos are merged into one large packet. After composing of each large packet the JAVA socket programming along with TCP is used to send the packets over the network. The streaming server is responsible to stream data over the internet and at client side client computer is responsible to decode the stream.

Decoding

Before streaming the packets over the Internet, left and right views are merged into one larger packet and streamed over the internet. Upon receiving the video stream at the decoding server these larger packets are decompressed and split again into left and right view.

Display

NEC multi sync LCD is used to view anaglyph videos generated from left and right views. The anaglyph glasses are used to watch these videos and these glasses are shown in appendix “A”.

2.3 Overview of existing Stereoscopic Video Streaming System

“End-to-End Stereoscopic Video Streaming System” [6] is selected project for the overview and comparison with proposed system. This system is presented by Selen Pehlivan, Anial Aksay, Cagdas Bilen, Gozde Bozdagi Akar, M. Reha Civanlar. The main parts of the system are acquisition, encoding, transmission, decoding and display.

For acquisition purpose Bumblebee stereoscopic camera is used to capture stereoscopic video sequences. This camera consists of two Sony progressive scan CCD sensors. The camera can capture up to 30fps.

In encoder implementation part MMRG multi view codec is used which is based on H.264. The codec is used in two camera-stereoscopic modes with standard compatibility option enabled. In this mode, left frames are predicted only from left frame so just using left NAL Units, left video can be decoded with a standard AVC decoder. Right frames are predicted from both left and right frames to reduce the bandwidth. For decoding of right video both left and right NAL units are required.

The transmission parts consist of the sender side and the client side. At the sender side the bit streams are encoded by MMRG multi-view codec that contain NAL units for both left and right views. The packet format is implemented based on the RTP payload format for H.264 video. The timestamp carried on the RTP header are used to arrange the decoding order of the frames. The RTP timestamps are used to synchronize the frames. H.264 parameters set are used for video coding.

At the receiver side the open source software VideoLAN client (VLC) is modified. The VLC is only used as a player and its stream receiver is modified because VLC does not support raw H.264 streaming over RTP. The modified VLC handles packets in separate threads for left and right views. The corresponding decoder for H.264 coded data is opened by the player. At the decoder the open source H.264 decoder implementation inside the FFMPEG library is used. The data is buffered in order to synchronize related left and right frames. The decoder decodes these streams and sends the decoded picture to the video output modules. The video output units visualize the left and right frames in a synchronized manner by using the time information in the RTP timestamps.

At decoder side the MMRG multi-view codec is developed based on JM Reference software of H.264. Due to limitations the received stream is not decode in real-time.

The display system consists of two sharp MB-70X DLP projectors and a silver screen. The lights from projectors are polarized using circular polarized filter. One filter polarizes light of one projector in right circular way and other filter polarizes light of the second projector in the left circular way. Both projectors projects into silver screen. The users wear glasses which guarantee to provide left and right eyes the corresponding left and right images reflected from the screen using filters matching with the projectors filters. Left views from one projector is shown on the left half of the desktop and right views projected from other projector are displayed on the right half of the desktop. Each projector shows one half of the extended desktop and then two overlapped frames see as 3D images by the users.

In concluding remarks authors mentioned that they have implemented an end-to-end stereoscopic video streaming system using open source components with minor modifications. Author's aims to use a hardware encoder to achieve real-time encoding and streaming of live video in future. Authors wish to investigate the system under network conditions with considerable packets losses. They also plan to increase streaming speed by defining new file format for multi-view video files. Audio support of the system is also under consideration of the authors.

In our system we used two separate cameras to acquire the video streams and in above presented system they used Bumblebee stereoscopic camera which have built in two progressive scan sensors. We used Simulcast stereoscopic coding technique for encoding and they used compatible stereoscopic coding technique. We used MPEG-4 as an encoder and decoder for video compression and they used MMRG multi-view codec based on H.264 and as decoder MMRG multi-view codec is developed based on JM Reference software of H.264. We used normal LCD for stereoscopic video using color glasses. They used two DLP projectors and a sliver screen to display stereoscopic video and user can view the video with the help of color glasses.

2.4 Coding approaches

There are several approaches that are used for stereoscopic video coding. These techniques are used to efficiently manage the bandwidth and maintain the quality of the video. In these approaches we try to exploit redundancy between two videos to achieve better compression. Most of the approaches are based on the motion-compensated DCT coding structure. Some of the approaches are presented below.

It will be beneficial if we write here some basics about disparity between views. Since the left and right views forming a stereoscopic video under the constraints described by the human visualization system, these views are very much correlated. These views are expected to be input to the human visualization system and are made by two fairly separated cameras. Special visual effects can be generated only such as independently coded signal and the right view as the dependently coded signal.

2.4.1 Simulcast Stereoscopic Coding

One simple solution to stereoscopic video coding is the simulcast stereoscopic coding technique. In this technique each layer of video representing a resolution or quality is coded independently. The left and right views are coded independently and can be decoded by a single layer decoder. Hence predictions for the right channel are made on the basis of motion in the right channel itself, and similarly a same phenomenon is applied on the left channel. The available bandwidth is portioned depending on the required quality for each independently layers (left and right views) that needs to be coded. This technique is least efficient in terms of compression efficiency. In this approach it is assumed that independent decoders would be used to decode each layer. This approach involves an independent coding of left and right views. Thus any of two views can be decoded and shown on a normal TV as the compatible signal from a 3D TV [7]. Structure of reference frames in simulcast coding is shown in figure 2.3.

In simulcast, the images of left and right views are encoded as I-Pictures at every second and the rest images are encoded as P-Pictures. While talking about Simulcast coding in the temporal prediction, the three closest preceding frames are referenced as shown in figure 2.3.

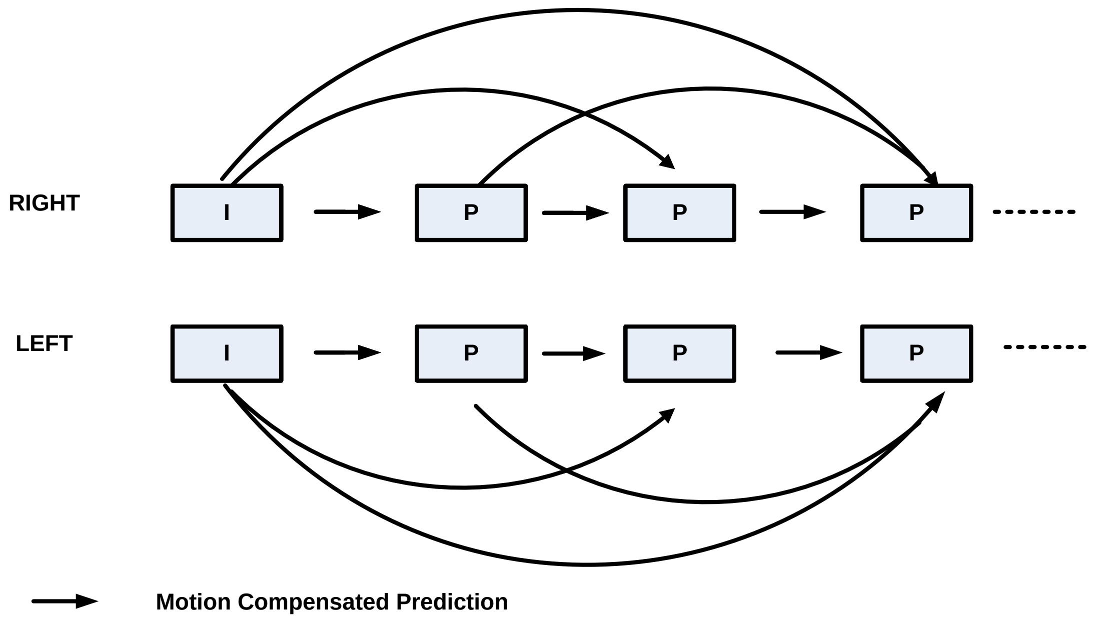


Figure 2.3 Structure of reference frames in Simulcast

In conventional stereoscopic encoding technique, left-view images are encoded as I-frames at every second, the remaining images and all right-view images are encoded as P-frames [8]. In other words, the coding methodology for left-view images is exactly the same like simulcast. The structure of reference frames of P-frames is shown in figure 2.4. The coding of right-view images includes temporal prediction from the two closest previous frames and at the same time spatial prediction from one left view frame. This is the methodology for constructing right-view images in the case of stereoscopic video coding as shown in the figure 2.4.

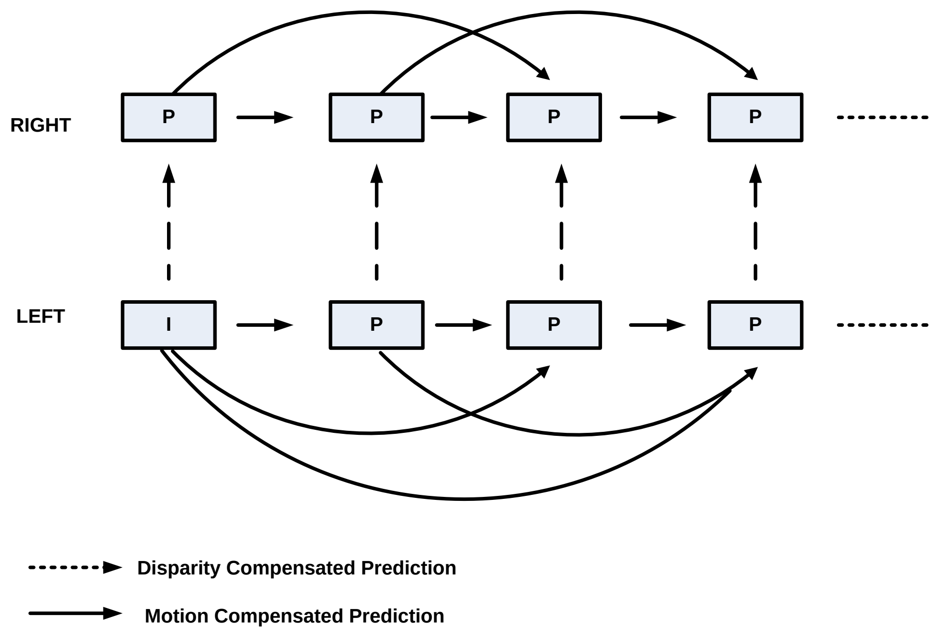

RIGHT
LEFT
P
P
P
P
I
P
P
P
Disparity Compensated Prediction
Motion Compensated Prediction

Figure 2.4 Structure of reference frames in Stereoscopic Video Coding

2.4.2 Compatible Stereoscopic Coding

There are several techniques for compatible stereoscopic video coding. Most of the methods are based on motion-compensated DCT coding structure. Generally in this approach left view is coded independently and the right view is then coded with respect to the independently coded left view. Thus only the independently coded view can be decoded and shown on normal TV as the compatible signals from 3D TV. Compatible stereo coding approach is shown in figure 2.5.

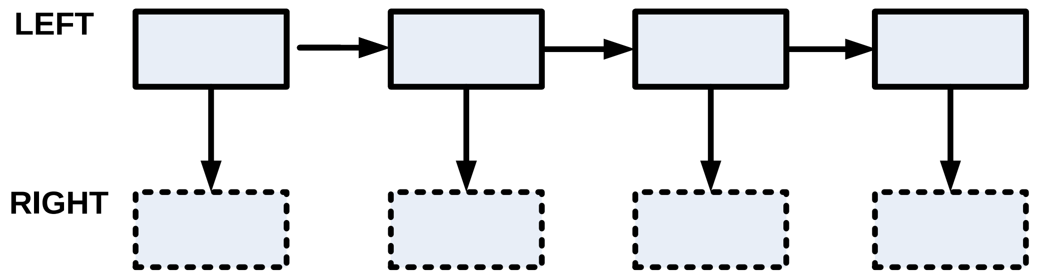


Fig. 2.5 Compatible or Stereo Compensation

In this approach independently encoders encodes only one view, exploiting the correlation between the two views for coding the other view dependently. This approach provides higher compression efficiency and better video quality than simulcast stereoscopic coding.

Figure.2.6 illustrates the compatible stereo video coding method in more detail. The characteristics of this method, if the used group of pictures is I B B P B B P B B, are [9].

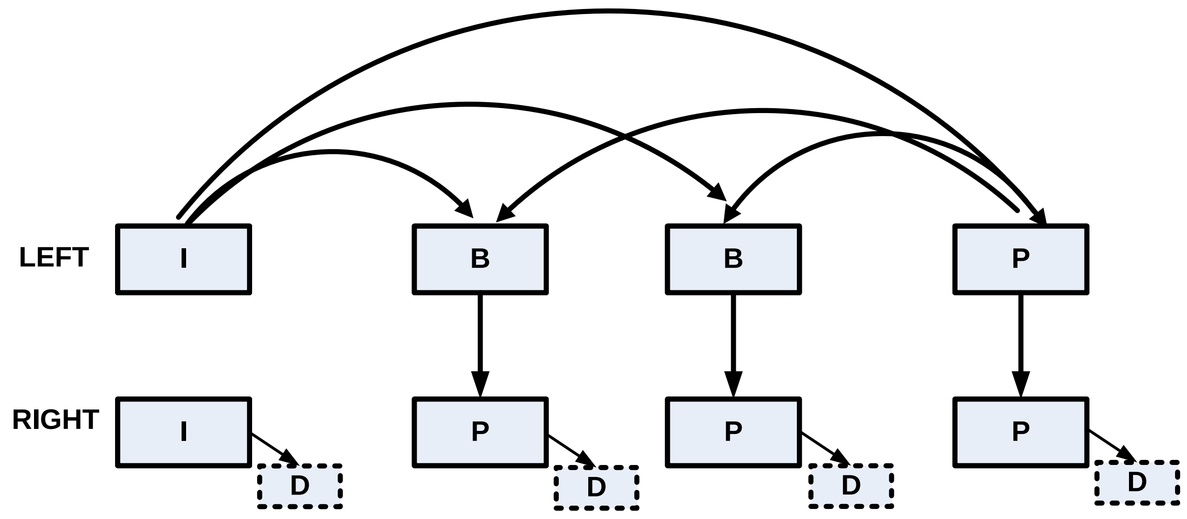


Figure.2.6 Compatible method of Stereoscopic coding

The main stream is coded like in MPEG standard. The auxiliary stream frames are coded by calculating the difference in predicted frames from the corresponding ones of the main stream. Hence the compatible stereoscopy, compresses the auxiliary channel by considering the spatial redundancies between the two channels.

2.4.3 Joint Stereoscopic Coding

In this approach both left and right views are coded together. Since both views are coded jointly and normally neither the left view nor the right view may be coded by itself as the compatible signal. Implementing the double layer structure as shown in the figure 2.7, the base layer can manage the left sequence while the enhancement layer manages the prediction of the right sequence. However, there may be exception where by imposing some restrictions, a subset of a 3D TV signal can be decoded and shown on normal TV as a pseudo-compatible signal. This approach is most efficient approach in terms of video compression. This approach does not involve any independent coding [10].

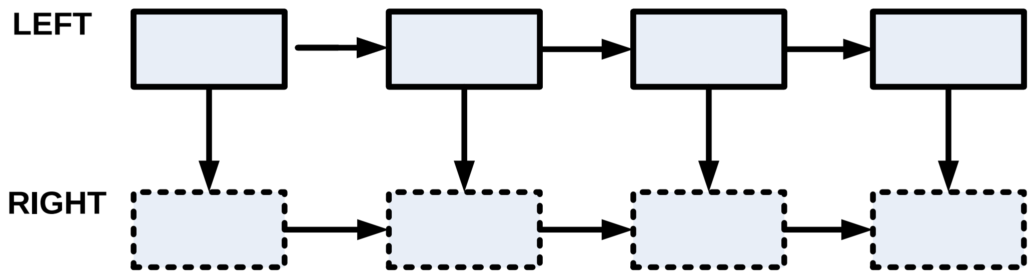


Figure 2.7 Joint Stereoscopic coding

For practical reasons compatible stereoscopic coding offers good solution as it provides a reasonable compromises between the conflicting requirements of higher compression and compatibility with normal TV. This also allows for the possibility of gradual introduction of stereoscopic 3DTV services without making millions of existing TV receivers obsolete.

2.5 Stereoscopic 3D Technology

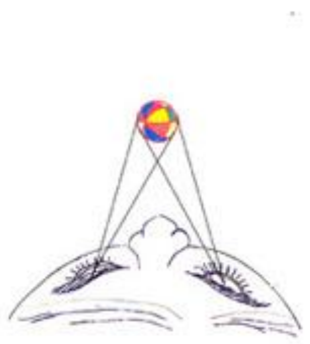
The word “stereo” is derived from the Greek term “stereos” which can be translated by “solid” or “hard”. Stereo visualization thus refers to the visual perception of the solid three-dimensional (3D) properties of some objects. Stereo-scoping is a technique capable of recording three-dimensional visual information or creating the illusion of depth in an image. Sir Charles Wheatstone (1802- 1875) started research on stereo visualization in 1838. He stated that Binocular vision is the concept of two slightly different views of the same object seen with the help of both human eyes.

Nowadays, 3D video and imaging are commonly used technologies in different fields e.g. medical, gaming and entertainment. Previously 3D videos were displayed on 2D visualization systems and users watch them by using special glasses. The choice of selection of these graphics depends on colors of images and videos. Currently different manufacturers are producing different kinds of 3D displays which are available on the market to show 3D content directly (without use of anaglyph glasses). The multi-view stereoscopic system displays simultaneous images at the same time and a viewer can see them from different positions. In this chapter we discuss about stereoscopic 3D technologies and their use.

2.5.1 Stereo Vision

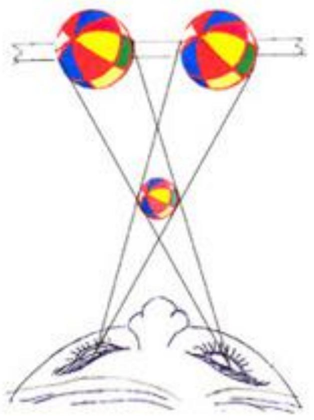
Stereo vision or “stereopsis” is the process of estimating the depth of a scene point from its change in positions between two images. This is done effortlessly by the human visual system, which translates the difference between the views from two eyes into a three dimensional impression of the scene [11]. The stereoscopic vision refers to the ability that human have to see the same scene with both eyes in slightly different ways. It results in our ability to visually perceive depth and distances. Normally two eyes see different images and our brain manipulate the images together to generate a stereoscopic view point. In stereoscopic vision, the scene is presented with slightly different view to produce three-dimensional vision so that human eye manipulates it as one scene. In computer vision different techniques are used to produce stereoscopic view point e.g. Anaglyph Stereo and Crystal Eyes Stereo. In most of the cases you need special glasses to view a stereoscopic video. Currently there are few manufactures that are making the 3D displays which do not require the use of an anaglyph glasses and viewers can watch a 3D video directly on their screens [12].

(A)



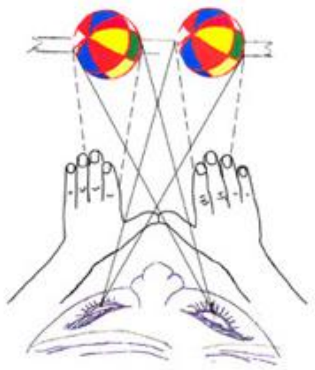
The observer sees the ball in front of his eyes.

(B)



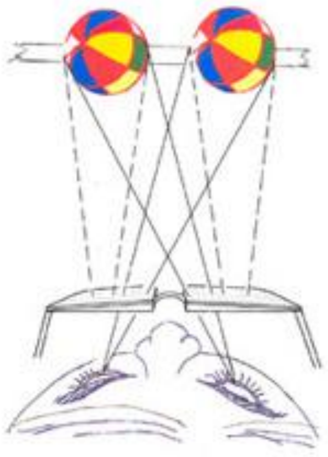
A picture of the ball is drawn on the screen by extending the lines from the eyes to the ball

(C)



The observer sees two balls on the screen, and there are no stereoscopic effects as yet. To produce a stereoscopic effect, we must find a way to eliminate the views represented by the dashed lines. The simplest although not the most practical way is to block the dashed line paths by extended hands.

(D)



For a more practical method, polarized light is used. Polariser glasses are worn to block the dashed line paths yet pass the solid line paths.

Figure 2.8 Principle of Stereoscopy, Source: http://individual.utoronto.ca

2.5.2 Anaglyph Video/Image

Anaglyph image provides stereoscopic 3D effect and we can watch it with special two color glasses. The glasses can be combination of two colors e.g. (Red and Green or Red and blue). Anaglyph image is stereo pair of two images in which each image is shown using a different color. We play with RGB color space of two slightly different images and our brain process these two separate images as a one image. In daily life human brain process two different views (left and right) and make them one view. We use this principle to generate anaglyph video. To make anaglyph image we take two slightly different images and convert them into gray scale. Then we assign red color to one image and green color to other image (we used green color because we want to view the images with red and green anaglyph glasses. You can assign colors according to your requirements). After assigning these colors depending on color of glasses we overlap these two images. Our brain thinks that it is seeing two separate left and right images and so does what it always does that combines them into one picture. These basic 3D images are called anaglyphs and work best with special color anaglyph glasses. An anaglyph image and an anaglyph video are shown at appendix "A".

Many TV manufacturers launched auto stereoscopic 3D displays at low price and they aim to replace currently being used 2D viewing system and also aiming to replace watching an anaglyph video with special glasses.

2.5.3 Auto Stereoscopic 3D Displays

An Auto stereoscopic display provides binocular depth perception. This technology has existed for many years and used to provide stereoscopic vision in research environments [13] . Binocular means using both of the eyes at the same time. In this technology, two cameras are placed at some slightly different positions (angles) and then two video streams are generated. Two different streams are generated and by using different techniques these streams are mixed up and by using different color combination the anaglyph video is generated. This anaglyph video can be viewed by using glasses with colored transparencies or polarization filters. Since auto-stereoscopy is a method of displaying three-dimensional images. These images can be viewable without any kind of special glasses. So it is more user friendly and a person can adopt this method more happily as there is no need to wear the glasses first.

2.6 Free View Point TV

Free Point TV (FTV) is an upcoming technology which gives a freedom to user to interact with broadcasted video. Multiple video streams are broadcasted to users and users can view their desired view-point and can switch between view-points with full control. In FTV the multiple video streams are broadcasted to end users through number of cameras, IP network and streaming servers.

This technology will enhance user experience to view multiple-views of the same scene at a time with full control on selection of scenes.

2.7 YouTube and 3D

YouTube is a video sharing website. Any computer user having an internet connection can view a video free of cost. Registered users can create their own channel for upload. YouTube is widely used for entertainment purpose. YouTube has become very popular among people of several professions. Several user channels are available for news, song and educational content. With change of technology YouTube upgrades accordingly. For example you can watch HD movies on it. The 3D video technology is gaining popularity very rapidly and therefore YouTube has introduced pages for viewing 3D videos. You can watch 3D video on it and you have to use anaglyph glasses. Before playing the video you have to choose between varieties of glasses' color options e.g. (red or blue, red or cyan etc). The 3D video in YouTube is shown with support of 3D in Appendix "A".

Our work is related to stereoscopic 3D video generation and streaming. As we mentioned above YouTube is very popular video sharing website and now it is providing the facility of watching 3D videos. With the provision of this facility on YouTube for 3D video lover shows that in near future 3D content delivery system may be dominated technology in entertainment industry.

Summary

Demand for Multi-view video content is growing rapidly and research on robust multi-view system is carried on in the industry and academia. Different algorithms are proposed for coding of multi-view content. Several encoders and decoders are designed for efficient delivery of multiple views to the users on their legacy displays or a 3DTV.

Stereoscopy is a technique that provides three-dimensional illusion of depth in images. Two slightly different images of the same object are overlapped with combination of different colors so that human brain manipulates them as one image. We make anaglyph images or video by using this technique. Special anaglyph glasses are used to watch such video. Newly introduced stereoscopic 3D displays are capable to display anaglyph video directly thus eliminating the need of special glasses to watch 3D content on screen.

3

Video Encoding And Compression

3.1 Introduction

Most of the technologies are growing rapidly with the passage of time and the technology of multi-view coding is one of those technologies which is growing rapidly. Many people are watching video over the internet and the trend of watching video on mobile devices is increasing too. Having this situation, the objective is to provide a good user experience in the form of fast download speed of data and a decent video quality. For fulfilment of these objectives, a good compression algorithm is required. So that the minimum bandwidth can be used from the total available bandwidth and a decent picture quality could also be maintained.

In this chapter we start with encoding and compression algorithms and define different types of video compression techniques and elaborate how these types can affect the final image quality. We discuss the mechanism of lossy data compression in more detail and define those components which are involved in this mechanism. Then we compare different codecs and their usage. In the end of the chapter we discuss video compression in the context of multi-view video coding.

3.2 Encoding

Encoding is the process of transforming information from one format into another one. The opposite operation is called decoding. Uncompressed information is too large and is not a practical format for Internet delivery. The purpose of encoding is to convert the available information from one format to the other one which is more compatible according to the available resources or requirements. For instance MP4 files are commonly used in mobile players and other devices (iPod, iPhone), but they are not supported by Windows Media Player. To display a MP4 video on a Windows Media player, we need to convert this file into AVI or MPG format which are more suitable for Windows Media Player. If we convert an AVI file which we just encoded, to the format MP4 again, this process is called decoding. This is often used in many digital devices for efficient data transfer. Usually an uncompressed video requires a high data rate and the need for huge storage space. Video compression technology is being efficiently used in many multimedia technologies. To obtain a higher picture quality we encode the data with high data rate.

Before encoding of a video it is necessary that you decide what the video is for (compact Disc, High Definition TV or some other source). Then it will be easy to decide the right type of compression to apply on it. It is also important to know the duration of the video as this will affect the level of compression applied.

The next stage is to analyse the video properly. This is especially important when a low data rate is required, for instance when the video needs to be viewed over a slow internet connection. In such a situation zooms, fast cuts, lots of motion, irregular lighting and excessive detail can make the video harder to encode and increase distortion (make the image look blocky) [14].
Hence an encoding process considers the followings:

Format (QuickTime, Real, MPEG etc)

Data Rate

Frame Rate

Window Size

Streaming method (progressive or real-time)

Hence striking the right balance among compression type, duration of compression applied, data rate, frame rate etc will ensure an optimal user experience.

3.3 Video Compression

As uncompressed video consists of an immense amount of data, such sequences require a huge bandwidth for transmission over the internet and a lot of memory space as well. Due to these expensive problems a video should be compressed gently and it requires a smart algorithm to do that. A very logical way of measuring how well a compression algorithm compresses a given set of data is to look at the ratio of the number of bits required to represent the data before compression to the number of bits required to represent the data after compression [15]. A video consists of rapid succession of still images or frames. This is too fast for the eye to judge and imagine an individual frame. We actually see 25 frames per second, our brain converts them into motion. But anything above around 10 fps will appear as smooth motion. This is because it takes 1/16th of a second for a picture to fade off the back of our eye. To overcome the spatial and temporal redundancies a selection of suitable algorithm(s) is an important matter in compression. A suitable algorithm can be evaluated by analysing how much memory space it requires, how fast it can work and under how much clock rate, how much amount of data it can compress and how much the reconstructed image resembles the original one.

A video consists of a sequence of frames which are related to each other in the temporal dimension and the spatial dimension. The principle of compression is that it removes redundant images and reuses the similar sections so that the information can be stored efficiently. These tasks are achieved by using the temporal and spatial compression.

3.3.1 Spatial and Temporal Redundancies

Spatial compression is sometimes called intra-frame compression. It looks for similarities in an individual frame. This algorithm works on adjacent pixels. It considers the pixels of the same colours or quite similar colour as one pixel rather than two or more pixels. Similarities are to be removed by applying this process on each existing frames. Spatial redundancy can be removed with various transforms, such as a discrete cosine transform (DCT), discrete wavelet transform (DWT), or sub-band decomposition [16].

Temporal compression is sometimes called inter-frame compression. It works in a similar way to spatial compression but the difference is that it looks at the similarities and differences between one frame and the next one and so on.

In temporal compression the encoded pictures are classified into three types as I, P and B pictures:

- **I-frame/ picture**: Intra-coded frame; coded independently of all other frames. If a video drastically changes from one frame into next e.g., fast cut, then such type of frame should be coded by using I-frame.
- **P-frame/ picture**: predictably coded frame; coded based on previously coded frame. Coded with forward prediction from references made from previous I and P pictures or possibly intra coded.
- **B-frame/ picture**: Bidirectional predicted frame, coded based on both previous and future coded frames.

3.4 Types of Compression Techniques

There are two main types of compression techniques:

- Predictive Coding VS Transform Coding
- Lossless VS Lossy Compression

3.4.1 Predictive Coding vs Transform Coding

In predictive coding, future values are predicted by using the already available information. Then the difference among the available information is coded. Since this is done in the image or spatial domain so it is relatively an easy method. Differential Pulse Code Modulation (DPCM) is an example of predictive coding method. Differential pulse code modulation (DPCM) is a procedure of converting an analog signal into a digital signal in which an analog signal is sampled and then the difference between the actual sample value and its predicted value (predicted value is based on previous sample or samples) is quantized and then encoded resulting a digital value. Transform coding, on the other hand, transforms the image from its spatial domain into another type of domain by using some transform method like discrete cosine transform (DCT) method. Then the transformed values (coefficients) are coded. This method provides higher data compression than the predictive coding with the cost of involvement of lot of computations.

3.4.2 Lossless vs. Lossy Compression

Lossless data compression makes use of the algorithm in such a way that after compression it looks the same as the original one. It is numerically identical with the original image. An example of usage of lossless data compression is ZIP file format. In lossy data compression, the reconstructed image is not identical to the original image, which is the case in lossless data compression. The most influencing reason is that it completely removes the redundant information. So by using lossy compression technique more data can be compressed. When we have a view of the reconstructed image by using lossy data compression scheme, interestingly it looks like data is not compressed (visually lossless).

A picture of typical lossy image compression system is described below. It consists of three components which are dependent on the previous component(s) progressively. This encoder system consists of a pre-processing, the Quantizer and an entropy encoder.

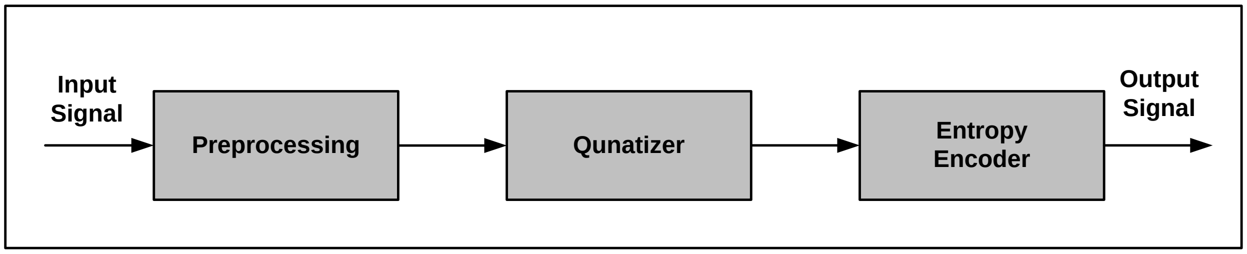


Figure 3.1 Typical Lossy Signal/Image encoder

3.4.2.1 Preprocessing

The preprocessor is used to bring out the significant information from the source. This data is sent into the quantizer as an input. Discrete Cosine Transform (DCT) is an example of preprocessing used in image compression. There are some other preprocessors like Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) and many more which have their own pros and cons e.g., Unlike DFT, DCT is real-valued and provides a better approximation of a signal with fewer coefficients.

3.4.2.2 Quantizer

Quantizer is the main source of compression for an image. It receives the data from preprocessor. It reduces the number of bits which are used to store the transformed coefficients by using the precision method. As it takes the precision values among many values so lot of data is lost as well. But due to this characteristic it is a main source of compression.

3.4.2.3 Entropy Encoder

After quantization the data is inserted into an entropy encoder. It further compresses the values received from quantizer without any loss. An example of entropy encoder is the Huffman encoder. Huffman code is the optimal prefix code in terms of shortest expected length for a given distribution. It works on the basis of probability statistics and the resulting output bit rate is smaller one than the original input.

3.5 Comparison of Video Codecs

A video codec is responsible for cramming a huge amount of data into a relatively smaller space. During the process of compression a lot of data is lost due to temporal and spatial similarities. The aspect of the quality of the image during the process of encoding and decoding cannot be ignored. The basic concept of using the different types of codecs is to maximize constraint compression of the original image having that reconstructed image should be as similar to the original one as possible. Different codecs have different characteristics which are defined below.

Data rate is the speed at which data (video) is transferred (bits/ bytes per second).

Compressing video for CD-ROM is usually done by MPEG-1 (Moving Picture Experts Group-1). In MPEG-1 video's data rate is low but it provides a medium image quality at CD-ROM. It includes both audio and video compression. One of the major problems of MPEG is that it demands very high playback requirements. To overcome this problem either a dedicated MPEG decoder card is required or a higher-capacity CPU is required for playing the software. The most popular use of MPEG-1 is the VCD (Video CD).

MPEG-2 is a compression standard which is related to MPEG-1 and produces broadcast quality video. Compressing video for DVD is done by MPEG-2. When we watch such video it is just similar with the original video. MPEG-2 provides excellent NTSC (National Television System Committee, analog television system) quality. However, playback is even more difficult than MPEG-1.

MPEG can be compatible with several other formats including QuickTime. Software takes long time for compressing a video. The algorithm used for MPEG is Discrete Cosine Transform (DCT) with Motion Prediction based on H.261. It is not well suited to video on web.

MPEG-4 is a standard which is used for compressing data (Audio and Video) for web. It is more than a single codec and it includes specification for audio, video, compression, delivery and interactivity of a multimedia content. MPEG-4 enables multimedia content usage at relatively low bitrates, it is an important characteristic for use on portable and mobile devices.

Low motion video is an ideal source material for MPEG-4. It compresses the source material with medium speed. MPEG-4 is a combination of different algorithms which are used in a multimedia application. It is optimized for delivery of video at Internet data rates.

One implementation of MPEG-4 video is included in Microsoft's NetShow and shown below in the figure 3.2.

Figure 3.2 Microsoft NetShow Player

Sorenson 3 (Used in older versions of QuickTime) and H.264/AVC/MPEG-4 Part 10 (Advanced Video Coding) are also used as codecs. Sorenson 3 is older and more widely compatible, while H.264 is modern, efficient (files tend to be surprisingly small), slightly prettier than ordinary Sorenson 3, and less widely supported. Only users with Quick-time 7 or later can view H.264 files. The professional Sorenson Squeeze is still the best of all, and the best choice is often Sorenson 3, but if you are not worried about compatibility with older systems and you need that extra compression edge, H.264 is your selected choice. H.264/AVC is the most advanced video coding standard available today, reaching approximately 50% bit rate saving when compared to previous standards like MPEG-4 and MPEG-2 and offers higher resolution quality picture with increased frame rates. You can record longer periods at higher quality settings as compared to MPEG-2 or MPEG-4. Since it offers reduced image size, it required less bandwidth. The most recent version of MPEG-4 is now incorporating H.264 and is increasingly replacing MPEG-2 for broadcasting of HDTV.

There are wide range of codecs which can be used according to the requirements and available resources. MPEG-1 and MPEG-2 are used for VCD and DVD respectively. MPEG-4 contains several functionalities and is used for web applications generally. H.264/AVC/MPEG-4 Part 10 provides excellent compression results with high resolution quality video.

3.6 Multi-view Video Compression

Multi-view video compression involves many streams but at least two streams are required. It is very interesting task for reconstruction of the scene from that stream which contains more than one stream. As multi-view video consists of multiple 2D video sequences, at least two video streams are transmitted or stored. To achieve that goal a massive amount of data has to be handled.

Multi-view video compression algorithms should reduce the redundant images among all of the used streams as much as possible to reduce the data volume and for better image quality.

FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. It includes libavcodec which is the leading audio and video codec library [17]. Since we used FFMPEG as an encoder that have a powerful built-in quantizer inside. It uses the technique of lossy data compression. With these attributes of FFmpeg we are compressing a much amount of data to send over the Internet efficiently. It is also an important task that a data which is going to be compressed should be in a good visual quality when it is being decompressed. In other words, when a viewer will view a decompressed video in 3D, he/ she should feel the same visual quality of the data presented in the same time frame. The compression algorithms should provide strictly low delay for real time applications and good negotiable delays for non real time applications. Delays will occur while encoding, compressing, saving the compressed data into hard disk, accessing the data from the hard disk, transmission over the internet and decoding.

Summary

Compression techniques play a major role for saving the bandwidth over the Internet and saving the space on storage media. Severall algorithms/ codecs are used for fulfilment of this task. These algorithms should be used according to the requirements. These requirements can contain the disk space, bandwidth, compression volume, speed of algorithm on a specific system and finally the image quality. Each compression encoder has its own benefits and limitations but H.264 is the best one among all of them because it compresses a lot of data by carrying a good image quality of the reconstructed image. This is the most basic reason of image compression especially in a multi-view transmission over the internet which consumes a higher bandwidth than an ordinary video transmission.

4 Video Streaming

4.1 Introduction

Streaming is a technology for transmission of live running events in real time over the internet. Events can be a lecture, medical training, sports competitions etc. These events can be watched by using computers, 3G mobile phones, 3D displays, hand-held wireless devices etc. Delivering video over the internet is not an easy task especially when we talk in the domain of real time. Normally a video file contains huge amount of data which can take several hours to deliver successfully over the internet. There are many factors which are obstacle in delivering a video with a higher speed like current speed of the internet varies, current available bandwidth varies, current congestion and load of the network also varies.

Computer technology is growing from that day on which computer was invented. The system of internet, codecs, compression, streaming servers and multimedia technology was not developed at that time which could be used as primary components for streaming. But with the passage of time, there are some basic improvements and additions in the components which are written below due to which streaming has became prominent and improved.

- Encoding and Compression algorithms
- Different Streaming Servers
- Higher speed of Internet and cable modems
- Use of standard protocols such as TCP/IP

It is also true that these three parameters can also become an obstacle in the way of doing the streaming as it is not possible for every one to get all these three parameters defined above. Streaming can be done by several methods.

4.2 Types of Streaming

There are two types of streaming, real time streaming and progressive streaming.

4.2.1 Real Time Streaming

In real time streaming, you start watching a video before it is fully downloaded, as it takes time for downloading the entire video even if you have a good internet download speed. Real time streaming media typically refers to audio and video that is directly played on the computer and do not download on the hard disk of the computer to view it at some later time. If a client needs for a specific video, it will requests to the server again. With the help of Real-Time Streaming Protocol/ Real-Time Transport Protocol (RTSP/ RTP), no file is ever downloaded at the viewer's hard disk. The movie is played in media player but not stored. Special streaming servers are used for the successful streaming of the data instead of traditional web servers. These special streaming servers have the capability to manage the real time streaming (data rate) according to the current network situation.

Real time streaming is an intelligent streaming in which it adopts the data rate with respect to the speed of viewer's internet connection. Furthermore the duration of the transmitting and receiving streams must be the same.

4.2.2 Progressive Streaming

In progressive streaming an entire movie must be first downloaded in the hard drive from a conventional server e.g., Apache, and then it can be viewed by a viewer. A movie's part which is not downloaded cannot be viewed by the viewer. No special servers are required in this type of streaming. Also it has no capability to detect the internet's connection speed. Progressive download offers high quality picture playback regardless of users internet connection speed but the users having low internet's connection speed will have to wait longer before the media player starts playing the movie. Furthermore, downloaded media can occupy large amounts of client storage space.

Recommendation for progressive streaming is that it should have a higher minimum download data rate than real time streaming because the progressive streaming must be downloaded fully first before play-out. It should use for short (durations) movies while real time streaming should use for longer (durations) movies and there should no delay of downloading it. Real-Time

Streaming is recommended for delivery of long-form video and 24/7 Internet TV/ Radio Channels. An example of 24/7 TV channel is NASA TV that broadcasts live events and mission's coverage.

4.3 Bandwidth Requirements

Transportation of packets over the Internet needs to have some specific requirements w.r.t bandwidth and protocol. Bandwidth is required according to the user's type of connection and protocol is required according to the data type that needs to be transmitted. e.g., if we want to transmit a simple XYZ.doc file securely over the internet having the fact that a user is using 56 K modem, for successful transmission of XYZ.doc file, safe bit-rate must not be less than 8 Kbit/sec and used protocol will be TCP. Otherwise the Internet does not provide any guarantees for the packets to reach the destination without any kind of losses. It might be possible that the sent packets have reached safely to the destination but they are not in the proper sequence of order. Different protocols like TCP can be considered from OSI model for in time and reliable packet transfer. Actual throughput and safe bit rate for different user connections are presented in table 4.1 appendix “B” table

4.4 An Architecture of Media Streaming

The architecture of media streaming can be divided into several parts as shown in the figure 3.3 shown below.

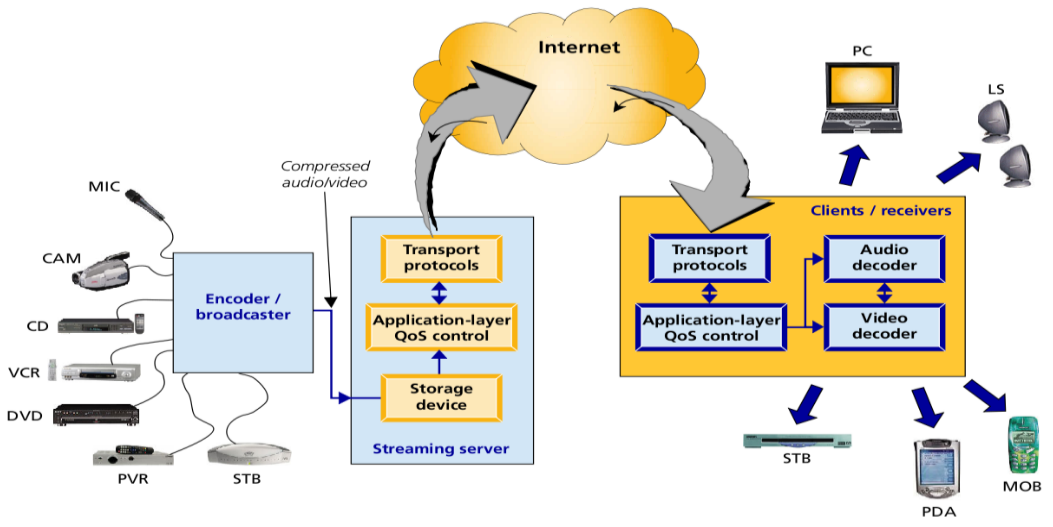


Figure 4.1 Multimedia Streaming. [18]

4.4.1 Media Compression and Encoding

It is important for good streaming that the data rate of the transmitted data is less than the user's internet connection speed. A trade off between compression algorithms and reconstructed picture quality and other issues related to compression and encoding have been discussed in chapter 3. For live streaming an application program encoder/ broadcaster is needed which takes an input signal from different devices such as web camera, video camera etc. It compresses these sources on the fly for transmission at the desired data rate, and creates a real-time audio-video stream. The broadcaster forwards its signal to the streaming server and streaming server offers the facility of real-time streaming and/ or progressive streaming according to the received request(s) from client(s) computer(s).

4.4.2 Streaming Servers

Streaming servers are one of the major elements in the process of streaming over the internet. It streams out the data in synchronous fashion. For instance, a user should listen at the same time synchronously with the motion of lips. It also provides the interactive control functionality like Play, Pause, forward, rewind etc. It can provide the facility of live webcast and pre recorded webcasts. It efficiently uses the available bandwidth, when the data packet get lost during the transmission between a client and a server, only the lost packet(s) is being sent again, resulting in low usage of bandwidth and opening the rooms for the other packets to travel efficiently. One streaming server can serve thousands of streams simultaneously. Streaming Server waits for an input signal from a user who requests for a specific media file, that task is being completed by the RTSP request to the streaming server. Streaming server then looks into its specific folder for a hinted media. If the requested file is found it is streamed out to the user by using RTP streams.

4.4.3 Protocols for Streaming

Streaming protocols are used for making the possibility for the communication between the streaming servers and the client computers/ devices. IP addressing (IP), transportation of packets (UDP) and session control (RTSP) are managed by streaming protocols.

4.4.3.1 Real-time Streaming Protocol (RTSP)

The Real Time Streaming Protocol (RTSP) is an application-level protocol for control over the delivery of data with real-time properties. It enables control on the request of audio/ video streams. These streams can be real-time or static. It is a state full protocol that uses RTP as underlying data delivery protocol. It handles the request for stream(s) and as a result these stream(s) are streamed out by using the RTP. RTSP is not used for transmission of streaming data but we can say that it is being used with RTP for successful transmission of streaming data. It offers VCR-like control: play, pause, forward etc. RTSP helps the server with control of bandwidth of the network so that the congestion problem can be solved successfully, if it exists. RTSP is similar to HTTP/1.1 with respect to the operation but differs in several ways too for instance in RTSP both client and server can generate a request during interaction while in HTTP/1.1 only the client can generate a request for document(s).

4.4.3.2 Real-time Transport Protocol (RTP)

RTP provides an end-to-end delivery service for data with real-time characteristics such as interactive audio and video [19]. RTP header includes different fields which are used for successful transmission of the packet(s). A sequence number is used for keeping track of successful packet transmission and incremented by one on every successful transmission of each packet. Sequence number is used by the player which keeps track of the packet loss, re-order out of order packets and to delete duplicate packets. Time-stamp is used for the synchronization and jitter control. Source identifier keeps track of the synchronization of the RTP multiple streams for instance an RTP stream can have several mixed media streams which are requested by the different users.

All media stream(s) are encapsulated into the RTP packets. RTP and UDP make a wonderful combination as RTP runs on UDP and uses its checksum and multiplexing methodology. As it uses UDP it does not re-transmit lost and damaged packets but on the other side by using UDP, RTP transmits packets in real time fashion.

4.4.3.3 Real-time Control Protocol (RTCP)

RTCP is a sister protocol to RTP. It is used to maintain the Quality of Service (QoS) between service provider and user(s) on RTP session(s) individually. It uses TCP for bi-directional client-server connection. RTCP provides the information about the number of packet lost, packets arrive later or earlier in time than the expected time (jitter control). Such information is being used for higher level applications in order to maintain the flow of stream(s) in real time format.

4.5 Sockets

Sockets provide an interface for programming networks at the transport layer [20]. A pair consisting of an IP address and a port number specifies a socket. Sockets are a service offered by transport layer and this layer provides end to end data delivery services by using TCP or UDP. Sockets include a set of primitives to enable a bi-directional communication between two/ several systems. Each client has 65536 ports. Some ports are well known ports, some are registered ports and are dynamic ports. The port on which we try to connect the server must be greater than 1023 because the ports between [0-1023] are well known ports. In the figure 4.2

shown below there are packets of a process traveling from a source socket towards a destination socket and the other way around. This work is done at the transport layer of OSI model.

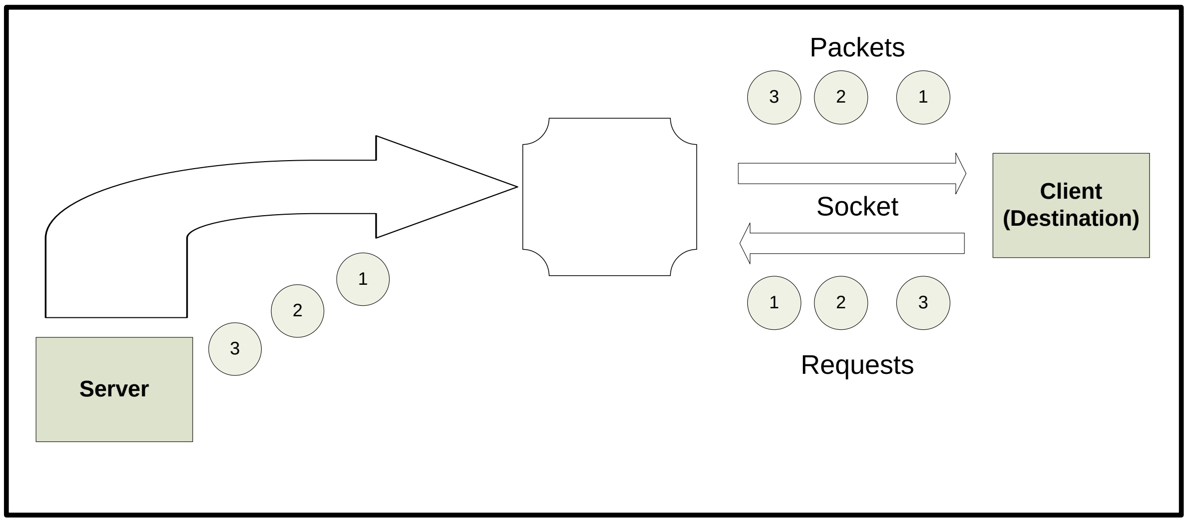


Figure 4.2 Transportation of Segments of Packets via Socket

4.6 Socket Types

Sockets are classified according to the communication protocols. Processes usually talk with each other under the same type of protocol between the sockets. However, if the underlying communication protocols support the communication, different types of sockets can also communicate. There are some famous types of sockets.

4.6.1 SOCK_STREAM

This is a type of internet socket. TCP is used as an underlying protocol in such type of socket. SCTP can also be used for stream socket. It transmits the packets with respect to the speed of the internet. It provides the reliability, in order delivery, connection oriented, unduplicated bidirectional flow of data. Stream sockets are implemented on top of TCP so that applicaitons can run across any network using TCP/IP protocol.

4.6.2 SOCK_DGRAM

This type of sockets is connectionless. It provides data-grams of fixed maximum length. The underlying protocol is UDP/IP. It is normally used for short messages where the reliability, in-order delivery and duplication of messages are not required but on the other hand it is faster than SOCK_STREAM socket. It provides bi-directional data flow like SOCK_STREAM socket.

4.6.3 SOCK_RAW

Raw sockets allow an application to have direct access to lower level communication protocols with the help of upper level communication protocol. RAW sockets are used for those users who want to add more functionality on top of an existing protocol so that the ultimate objective can be achieved with the help of two protocols. RAW sockets are datagram-oriented originally but they behave in total with both of the underlying and upper layer protocol.

4.6.4 SOCK_CONN_DGRAM

This is the type of sockets which provides connection oriented datagram service. It is a bidirectional data flow socket which provides inorder and unduplicated data transfer. On the other hand it is not reliable.

4.7 Client-Server Model

Computer processes that offer application ervices are called servers. Server creates socket(s) which is in listening state initially and waiting for client(s) to provide service. Whenever a client connects with the server, a bidirectional communication get started until and unless they close their sockets. A TCP server can serve several clients concurrently by creating different independent threads. These threads are also known as child processes. Whenever a request comes from a client to a server, server makes a new thread for that client (process) immediately and enters into listening state to provide service to the next client (process) and so on. A TCP link is created between a specific client and a child process/ thread to provide a duplex communication. A server can establish several TCP sockets with the same port number and IP address but operating system considers every new process as a new socket since these requests are reaching the server from different clients having different IP addresses and port numbers. The communication between client(s) and server can continue until and unless sockets are open. Whenever a socket is closed from either side of the communicating link, communication will also close.

The TCP header is defined here for describing the fields which are specifically used for sockets. TCP data is encapsulated in an IP datagram. The figure 3.5 shows the format of the TCP header. Regular size of a TCP header is 20 bytes unless options are present. Fields are as defined below.

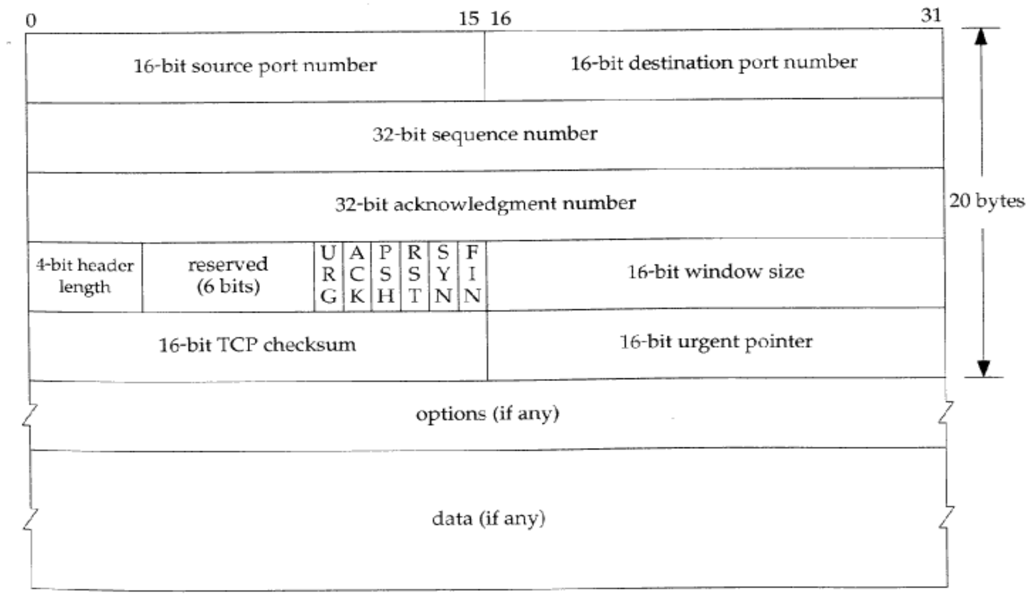


Figure 4.3   TCP Header [21]

The sixteen 16-bit source port number and sixteen 16-bit destination port number identifies the source and destination ports. These two fields and source and destination IP addresses, combine to uniquely identify each TCP connection.

Socket = <IP address+ Port number>

TCP Connection = <Source Socket> + <Destination Socket>

A TCP header contains several fields which can be defined in detail but it will be sufficient to note here that other fields used by a TCP header are for the reliable and in order delivery of packets in a secure format.

4.7.1 Client-Server Socket's Flow Graph (Connection Oriented)

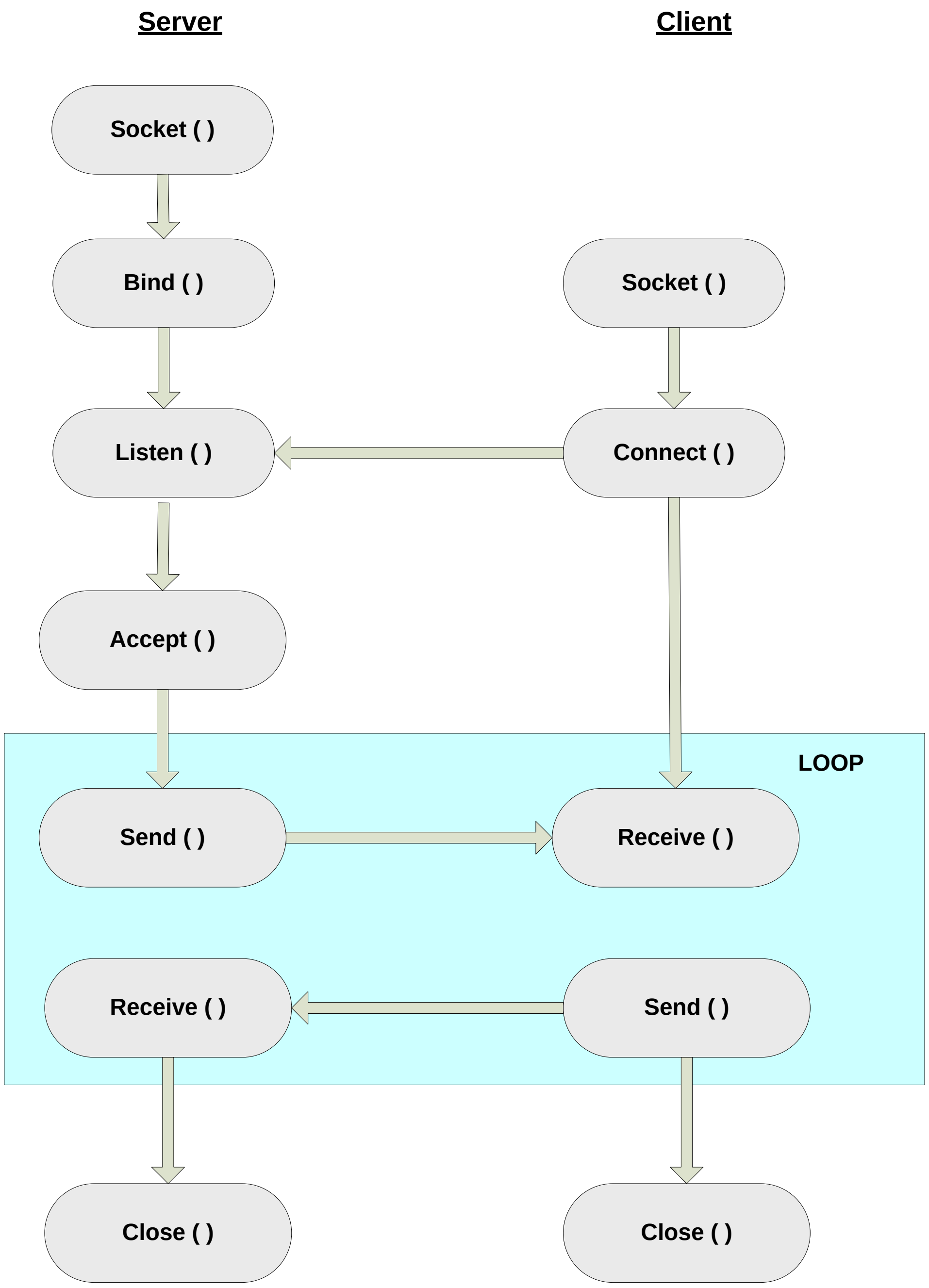


Figure 4.4 Client-Server Communication using TCP

Some description of sockets is provided here which is used in client-server model using TCP as an underlying protocol so that data can be transmitted in-order and with reliability. The steps involved in establishing a socket on the **client side** are as follows:

- Create a Socket with the Socket () system Call and allocate system resources to it.
- Connect the Socket to the address of the server using the Connect () system call. Client needs to specify IP address and port number of the server.
- Send and receive data using Send () and recv (), Or Write () and read (), or sendto () and recvfrom().
- Close () system call is used to terminate the connection.

The steps used for establishing a socket on **Server side** are as follows:

- Create a Socket with the Socket () system call and allocate system resources to it.
- Bind the socket to an address using the bind () system call. This address is the port number of the host machine over the Internet.
- Listen () system call is used to bound TCP socket to enter into listening state. It listen for connections
- Accept() System call is used for accepting a client connection. This call typically blocks until a client connects with the Server.
- Send and receive data.
- Close () system call is used to terminate the connection.

Summary

Bi-directional streaming between a server and a client can be done with different techniques. Different kinds of servers are available for the process of streaming like Darwin Streaming Server, Quick Time streaming server and FFserver etc. For listening of music or news, radio stations are available. Many TV channels are available and all of them are transmitting their signals 24/7 but the streaming technology can defeat or come parallel with all of them in some way like you can have the functionalities of play, pause, rewind, fast rewind for the running program and a good picture quality over the Internet. If you are late in time to watch your favorite program, you are not late indeed. You are welcome to start watching it now via on demand request to the streaming server as it is stored for you in the hard disk of the server which can handle up to hundreds and even thousands of users at the same time. The major protocols which are involved in the process of streaming are RTSP and RTP. Sockets are also used for client-server communication. Since the components of a streaming server are platform dependent, use of sockets is a good solution as socket programming is not platform dependent and works with reliability with the speed of the Internet on any kind of operating system.

5

Implementation And Experiences

This chapter will contain piece of code and our findings during the thesis work. First part of chapter will cover how we acquired simultaneous streams from video cameras. Second part is about encoding of video streams using OpenCV with libraries of FFMPEG. Third part will be covering video streaming over the IP network. Part four will be about decoding of video streams and 3D display.

In this thesis work the video signals from two (left and right) cameras are encoded, compressed, combined (Packet = <first camera + second camera>) and stored on the hard disk of the server. The video from each camera (left and right) is captured for 30 seconds each and merged into one larger packet of 60 seconds. The construction of these packets (60 seconds each) could be continued as long as you want. This work is done with the help of OpenCV and FFMPEG. Additionally we can watch these video streams on the server in the form of a 3D or multi-view. Here two approaches can be used to send these packets over the Internet and to view them on some remote client machine ultimately. These approaches include the use of a streaming server or computer sockets defined more precisely in section 5.2.

5.1 Encoding and Video Acquisition

We used OpenCV and FFMPEG to acquire video from the cameras.

i) **FFMPEG command for video acquisition**

We used FFMPEG to acquire video from camera. The command is given below with different switches used to acquire the video.

```
$ FFMPEG

-t  hh:mm:ss  <Duration of time>

-f <Force format>

-s <Size/ Resolution>

-r <Frame rate>

-vcodec <codec used>

-i <Input file name> or <device location>

<Output File name>
```

Description

Duration of time (-t):

Restrict the captured video sequence to the time duration specified in seconds.

If we do not follow the exact format of time which is like –t hh:mm:ss, then we cannot record a movie exactly for a given time duration. For recording of a movie for exactly 30.00 seconds, we initially wrote: -t 30 but when we played out this recorded movie, it was ended up at 30.04 seconds which was because we do not provided the parameters of time in the format of –t hh:mm:ss. But when we provided the parameter of time like –t 00:00:30, the recorded movie was played-out exactly for 30.00 seconds. So it is an important feature in the real time context Force Format.

Video4linux2:

Video4linux2 is a video capturing interface for Linux. Its second version is denoted by video4linux2 which can fix some of the design bugs. It can take input from several USB webcams, TV tuners etc. Video capture interface is denoted by /dev/video0. Several videos can be captured by using different interfaces.

Size/ Resolution (-s):

The simple concept of resolution applies here. We can play with display size by giving different sizes and we know that if we want to change the size (i.e., resolution) of the video to standard VGA, we have to give the parameter like: **-s 640x480**

Coding Decoding (-vcodec):

For coding and decoding this function is used. You can change several codec to other codec depends on requirement.

Frame rate (-r):

Some imaging device produces unique consecutive images called frames. Frame rate or frequency specifies the number of frames taken per second.

Input file, Output file:

Input and output files are the simple files which you take as an input and produces output accordingly.

Example:

```
$ FFMPEG -t 30:00:00 -f video4linux2 -s 640x480 -r 25 -i /dev/video1 test100.mpeg
```

This command captured 25 frames per second from video device 1 for 30 seconds and save it as output.mpeg. As we mentioned above FFMEPG supports many codec therefore the output file can be with different extension as per requirement.

i) OpenCV syntax to Grab video from cameras or file

`cvCaptureFromCAM(device);` this method is used to capture frames from camera.
`cvCacputerFromFlie (filename);` this method is used to capture frames from file.

Description

OpenCV supports capturing images from video file or camera. We have to mention from which interface we start capturing the frames.

Initializing capture from a camera:

```
CvCapture* capture = cvCaptureFromCAM (n);
```

Where n= 0, 1, 2, 3,…..n. Each installed camera getting device ID in Linux so that we have to mention the device number.

Initializing capture from a file:

```
CvCapture* capture = cvCaptureFromFile();
```

Capturing a frame:

```
IpImage* img=0;
cvGrabFrame(capture);  //capture each frame
img = cvRetrieveFrame(capture);   //retrieve the captured frame
```

To obtain images from several cameras simultaneously, we first grab an image from each camera and then retrieve the captured images after the grabbing is complete.

The syntax of code is taken form http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html

5.2 Video Streaming

Quick Time streaming server is used in windows and MAC operating system and Darwin Streaming Server is the open source streaming server equivalent to Quick Time Streaming Server. Live movies are commonly streamed over the Internet as they happen with the assistance of broadcasting software, such as Quick Time broadcaster. The broadcasting software encodes a live source, such as video from a camera in real-time and delivers the resulting stream to the server. The server then serves the movie to the clients.

The Quick Time broadcaster is required for encoding a live source (in our case stream from camera) but a problem is that Quick Time broadcaster is compatible with Windows or Mac operating system. It cannot be installed on Linux operating system. We did not find appropriate open source broadcaster so we used FFMPEG instead of Quick Time broadcaster for encoding a live stream. Also Quick Time player is only available as a part of Quick Time on Windows and Mac Operating system. Open Source players such as VLC is available which may be able to play certain Quick Time contents on client side. We did try to stream-out some movies with the help of VLC and Darwin Streaming Server but did not succeed. As we used Linux operating system which is an open source and we cannot use Quick Time broadcaster for encoding on Linux and hence avoided to narrow our scope with respect to operating system. So we decided to use another approach in which data can be streamed out by using sockets.

The approach used for our working system is to take two streams of 30 seconds each from two different cameras, combine them into one packet of 60 seconds, save it on the hard disk of local computer (server) and stream-it out over the Internet according to the request of a certain client(s). At the client machine, this packet is saved on the hard disk, divided into two parts, and displayed on the screen in the form of a 3D or multi-view video. OpenCV with FFMPEG is used for encoding, compression, 3D and multi-view display and sockets are used for client-server communication so that multimedia packets can be transported from server to the client successfully. The client machine must have installed OpenCV, FFMPEG and Java and may use any kind of operating system.

We used the Java programming language and its built-in packages for Client-Server communication. All the classes related to sockets are in the java.net package. All the input and output stream classes are in the java.io package. As we need a secure, in-order and reliable connection oriented communication, so we used TCP protocol which is working as an underlying protocol in this mechanism. Basic functions of source code for this application are written as follows.

5.2.1 Open a Socket

At the **client side**, create an object of class Socket. Pass the parameters (Machine name, port number):

```
 Socket sock = null;
 try {
    System.out.println ("Connecting ...");
    sock = new Socket (IP, port);
}
catch (IOException e) {
   System.out.println ("Cannot connect, may be server is not running");
exit (1);
}
```

- Machine name/ IP is the machine name/ IP we are trying to open a connection to. i.e., <Server's IP address>

- Port number is the number of that port on which we want to connect to i.e., <Server's port number> (Server must be in the running state when a client wants to connect with it)

- Information of ports are as follows:

  Well known ports: 0 – 1023
  Registered ports: 1024 – 49151
  Dynamic/ private ports: 49152 – 65535

The Registered Ports are listed by the IANA and on most systems can be used by ordinary user processes or programs executed by ordinary users [19]. So for our client-server communication model we can select one of the registered ports.

At the **server side** we open a socket with the followings:

```
ServerSocket servsock;
try{
    servsock = new ServerSocket(PortNumber);
}
catch (IOException e) {
  System.out.println(e);
}
```

- Port number is the port number on the server machine.
- When we open a socket on server side, we also require a socket object by using that object we can listen and accept connections from the client side as follows:

```
  ServerSocket servsock = new ServerSocket(portNumber);
  Socket clientsock = null;
  try {
    clientsock = servsock.accept();
  }
  catch (IOException e){
    System.out.println(e);
  }
```

5.2.2 Sending and Receiving the Data-Stream

For sending an input stream we have the following code on the **client side**:

```
         public boolean sendFileRequest(String filename, Socket sock) throws Exception{

           OutputStream os = sock.getOutputStream();
            byte[] bytes = filename.getBytes();
            os.write(bytes,0,bytes.length);
            os.flush();
            return true;
            }
```

On the **server side,** we use the followings to receive an input from the client:

```
public String receiveFileRequest() throws Exception {

  InputStream is = sock.getInputStream();
  int bytesRead = -1;
  byte [] mybytearray  = new byte [];
  bytesRead = is.read(mybytearray,0,mybytearray.length);
  if(bytesRead <= 0)
  return "";
  String filename = new String(mybytearray);
  return filename;}
```

- *Class InputStream has a method read which takes the parameters* read(byte[]b, int off, int len) *Reads up to* `len` *bytes of data from the input stream into an array of bytes.*

  On the **server side,** we use the followings to send the file to the client:

```
     public boolean sendFile(String filename) throws Exception{

        File myFile = new File (filename);
        myFile = new File (filename);
     }
        byte [] mybytearray  = new byte [(int)myFile.length()];
        FileInputStream fis = new FileInputStream(myFile);
        BufferedInputStream bis = new BufferedInputStream(fis);
        bis.read(mybytearray,0,mybytearray.length);
       OutputStream os = sock.getOutputStream();
       os.write(mybytearray,0,mybytearray.length);
    }
```

For receiving an output stream we have the following code on **client side:**

```
  public boolean receiveFile(String filename,Socket sock) throws Exception{

    int bytesRead=0;
    int current = 0;
    byte [] mybytearray  = new byte [filesize];
    InputStream is = sock.getInputStream();
    FileOutputStream fos = new FileOutputStream(filename);
    BufferedOutputStream bos = new BufferedOutputStream(fos);
    Do
          {
          bytesRead = is.read(mybytearray,0,mybytearray.length);
          bos.write(mybytearray, 0 , bytesRead);
          }
```

```
}
```

5.2.3 Socket Closing

We always close the output and input stream before we close the sockets. After closing of streams on client and server side, we always close the sockets as well so that some other process can utilize that socket in the future.

- **On the client side:**

```
try{
    output.close();
    input.close();
    MyClient.close();
}
catch (IOException e) {
   System.out.println(e);
 }
```

- **On the server side:**

```
 try{
   output.close();
   input.close();
   Clientsocket.close();
   MyService.close();
}
 catch (IOException e) {
   System.out.println(e);
}
```

5.3 Decoding

After initializing of video writer you can mention different codec according to your requirement.

Initializing a video writer

```
CvVideoWriter *writer = 0;
int isColor =  1;
int fps     =   25;
int frameW  =  640;
int frameH  =  480;
writer=cvCreateVideoWriter("out.avi",CV_FOURCC('P','I','M','1'),fps,cvSiz(frameW
,frameH),isColor);
```

Other possible codec codes:

```
CV_FOURCC ('P','I','M','1')    = MPEG-1 codec
CV_FOURCC ('M','J','P','G')    = motion-jpeg codec
CV_FOURCC ('M', 'P', '4', '2') = MPEG-4.2 codec
CV_FOURCC ('D', 'I', 'V', '3') = MPEG-4.3 codec
CV_FOURCC ('D', 'I', 'V', 'X') = MPEG-4 codec
CV_FOURCC ('U', '2', '6', '3') = H263 codec
CV_FOURCC ('I', '2', '6', '3') = H263I codec
CV_FOURCC ('F', 'L', 'V', '1') = FLV1 codec
```

5.4 Display

5.4.1 Anaglyph video

First we created three images and you can declare them by using

```
IplImage * image_name;
```

Assign left view to first image and right view to second image. For third image which is empty use `CvCreateImage ();` This image will be used for container of final anaglyphic video.

```
CvCreateImage (cvSize (640,480), IPL_DEPTH_8U, 0);
```

For left view and right we created six more images by using `CvCreateImage ()` function as described above. We used first three images for left view to handle RGB color channel separately and used another three images for right view to handle RGB color channel.

We used `cvSplit()` function to separate RGB color channels from main left and right view images and put them in newly created three images for each left and right view.

We removed color channels from images by using `cvZero()` function.

We merged images after removing color by using `cvMerge ()` function.

After getting desired left and right view with required color combination we overlap left and right view in third empty image which we created in first step by using `cvAddWeighted()` function.

We used cvShowImage () function to display the video on screen.

5.4.2 Multi-View Video Display

At the client side you received the video from server and stored them on hard disk to separate left and right view. In first step we captured frames from stored video by using cvCapturedFromFile ().

We created two new images by using CvVideoWriter () and assigned codec to these images by using cvCreateVideoWriter ().

We used cvGrabFrame () to capture frame from source of each left and right view. The source will be a stored in a video file on hard disk or a live stream. After capturing these frames we retrieve these frames cvRetriveFrame (). After retrieving we wrote these frames to left and right view by using cvWriteFrame ().

We determined left and right view by obtaining total number of frames from one combine video (the combine video contained both left and right view for specific time) and then write the two video in two different files. We used cvShowImage () to display these left and right view.

6 Conclusion And Future Work

In this master thesis we have implemented a stereoscopic multi-view system. The system is proficient in transmitting of multiple view video streams over the internet and also capable to display these video streams simultaneously. We have implemented this system by using open source softwares such as OpenCV, FFMPEG. The Java socket programming is used for client server communication. OpenCV and FFMPEG are used for video grabbing from cameras. The encoding and decoding is also achieved by using FFMPEG with combination of OPENCV. We have implemented a client server model. Server is capable to acquire multi video streams from number of cameras and also act as an encoder and stream out the encoded video over the Internet with efficient compression techniques. The client is capable to receive the multiple video streams and also act as a decoder. The decoded streams are sent to the displaying mechanism which includes a 3D display or anaglyph video system.

In future, audio support can be added to this system. The efficient multi-view coding techniques can be studied and adopted to reduce the bandwidth requirements. Different codecs can be studied for time saving during video acquisition from multiple cameras. Buffering of video on server and client can be added to improve video playing delay between a numbers of video streams. A mechanism can also be defined in case of delay in a video transmission due to packet loss or congestion in the network.

The thesis work can be extended to Free View Point TV (FTV) so that a user has the freedom to choose between different views of the same scene in a real-time with full control. Switching between different views in real time can be an interested topic of research.

We used sockets for transportation of packets between the client-server communications. These packets contain the video taken from the left and right cameras and on the client side; these video are then being played-out. These videos signals are streamed out with the help of socket programming which uses TCP as an underlying protocol. We learned that TCP is not a good choice for the real-time applications because of different issues as one of the issues of additive increase multiplicative decrease (AIMD) could happen in the case of congestion in the network and hence cut the congestion window in half after loss of a packet. In this case where we experienced congestion in the network our suggestion is to use TCP-RTM (Using TCP for Real Time Multimedia Applications) instead of TCP as it provides a new mode for TCP and it supports real-time applications with higher performance than a basic datagram service such as UDP which is an unreliable protocol but a faster one.

For streaming of packets, if we think about using of a streaming server: Streaming server offers VCR-like control: play, pause, forward etc. RTSP helps the server control of bandwidth of the network so that the congestion problem could be solved successfully, if it exists.

At the server side, we are combining two packets. Each packet contains left and right video captured from two cameras. These packets are also saved on the hard disk of the server and client machine. This whole process takes some time. We can save time by using the streaming server with some suggestions is as follow: client machine should have a media player which should have a capability of a multi-view display. It connects with a streaming server to get some multimedia file initially and at the next step it could be able to show multi-view video (Left and right view). Also instead of saving these videos on the hard disk of the client or server computer, this video should only be stored on the hard disk of the streaming server. These video must be available at the same time on the hard disk of the streaming server so that the synchronization of the video frames can be managed and video could be played-out in the form of a multi-view synchronously at the client machine. By adopting this approach we can save a lot of time. Hence the video could be played-out in a real-time truthfully.

We did analysis between sockets and streaming servers for a client-server communication and we conclude that sockets do not provide the facilities which a streaming server can provide. If we want to use socket programming for the transportation of segments between the client-server communications, we should use TCP-RTM instead of a simple TCP. To make an application a real time truthfully along with the current standards of the multimedia technology, we suggest the designing of a media player which can support multi-view video so that it can connect with the streaming server and play out the movies in the form of a multi-view.

References

[1] Haksoo Kim, Manbae Kim, "The Design of Single Encoder and Decoder for Mulit-view Video", Springer, 2006.

[2] K.Muller and et.al.,"Multi-video Coding Based on H.264/AVC Using Hierarchical B-Frames", PCS 2006, China 2006.

[3] E.Ekmekcioglu, S.T.Worrall, A.M.Kondoz, "Multi-View Coding Via Virtual View Generation", Picture Coding Symposium (PCS 2007), November 2007, Lisbon, Portugal.

[4] E.mARTINIAN, a.Behrens, J.Xin, A.Vetro, and H.Sun, "EXtentions of H.264/AVC for multi-viewVideo Compression", in IEEE Int.Conf.on Image Processing,Oct 2006, Atlanta, USA.

[5] S.Argyropoulos, A.S.Tan, N.Thomos, E.Arikan, M.G.Strintzis, "Robust Transmission of MultiView Video Streams Using Flexible Macroblock Ordering and Systematic LT Codes", 3DTV Conference, 2007.

[6] Selen Pehlivan1, Anil Aksay2, Cagdas Bilen2, Gozde Bozdagi Akar2, M. Reha-Civanlar1, " End-To-End Stereoscopic Video Streaming System",cCollege of Engineering, Koç University, Istanbul, Turkey, Electrical and Electronics Engineering Department, Middle East Technical University, Ankara, Turkey.

[7] Barry G. Haskell, Atul Puri, Arun N. Netravali, "Digital video: an introduction to MPEG-2",Chapman and Hall, New York, United States, 1997.

[8] H.M.Ozaktas,L.Onural "Three-Dimensional Television: Capture, Transmission, Display", Springer, 2008.

[9] J. N. Ellinas, M. S. Sangriotis,”Stereo Video Coding Based on Interpolated Motion and Disparity Estimation”,University of Athens, Department of Informatics, Ilissia, 157 84 Athens, Greece.

[10] Barry G. Haskell, Atul Puri, Arun N. Netravali, “Digital video: an introduction to MPEG-2”,Chapman and Hall, New York, United States, 1997.

[11] Daniel Scharstein, " View Synthesis Using Stereo Vision", Springer,1999.

[12] http://www.ks.uiuc.edu/Research/vmd/vmd-1.7.1/ug/node96.html.

[13] http://www.3d-forums.com/autostereoscopic-displays-t1.html.

[14] Why Encode? Pigeon Ltd.registered in England no.4461294, www.pinkpigeon.net.

[15] Khalid Sayood "Introduction to Data Compression", Second Edition, Academic Press, San Diago, CA, United States, 2000.

[16] Kregimir DuraEiC, Hrvoje Mlinarid, Mario KovaE,” Optimization Methods For Mpeg-4 Algorithms In Multimedia Applications Running On PDA”, EC-VIP-MC 2003.4th EURASIP Conference focused on Video I Image Processing and Multimedia Communications, 2-5 July 2003. Zagreb, Croatia Faculty of Electrical Engineering and Computing Unska 3, Zagreb, Croatia.

[17] http://ffmpeg.org.

[18] Franc Kozamernik "Media Streaming over the Internet - an overview of delivery technologies", EBU Technical Review, October 2002.

[19] Audio-Video Transport Working Group, Lawrence Berkeley National Laboratory, January 1996, RFC 1889.

[20] IANA Rajkumar Buyya, S Thamarai Selvi and Xingchen Chu “Object Oriented Programming with JAVA, Essentials and Applications: Chapter 13 Socket Programming Tata McGraw Hill Education Private Limited, First Edition June 2009.

[21] Reprinted from TCP/IP Illustrated, Volume 1: The Protocols
by W. Richard Stevens, Copyright 1994 by Addison-Wesley Publishing Company, Inc.

Glossary

AIMD	Additive Increase Multiplicative Decrease
AVC	Advanced Video Codec
DVD	Digital Video Disk
FPS	Frame Per Second
FTV	Free View Point TV
HDTV	High Definition Tele Vision
IANA	Internet Assigned Number Authority
IEEE	Institute of Electrical and Electronics Engineers
LCD	Liquid Crystal Display
MERL	Mitsubishi Electric Research Laboratories
MPEG	Moving Picture Experts Group
MVC	Multi-view Coding
OPENCV	OPEN Computer Vision
PAL	Phase Alternating Line
RFC	Request for Comments
RTP	Real Time Protocol
RTSP	Real Time Streaming Protocol
TCP-RTM	Transmission Control Protocol-Real-Time Mode
3D	Three Dimensional
TV	Tele Vision
UDP	User Datagram Protocol
VGA	Video Graphic Adaptor

Appendix A

Figure A1: Logitech Cameras used for Video Acquisition

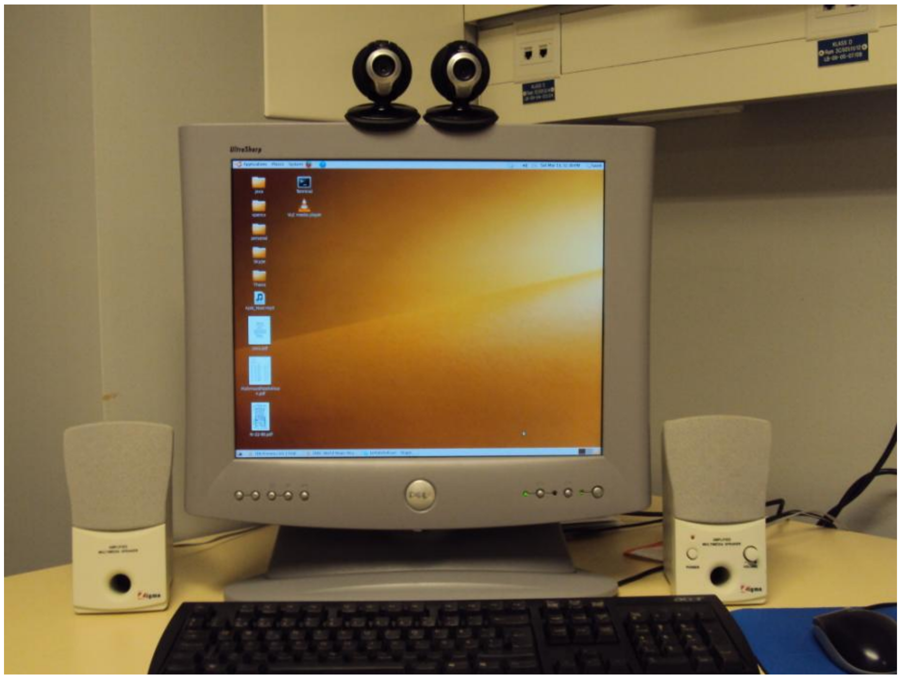


Figure A2: Setup of cameras with Acquisition Server

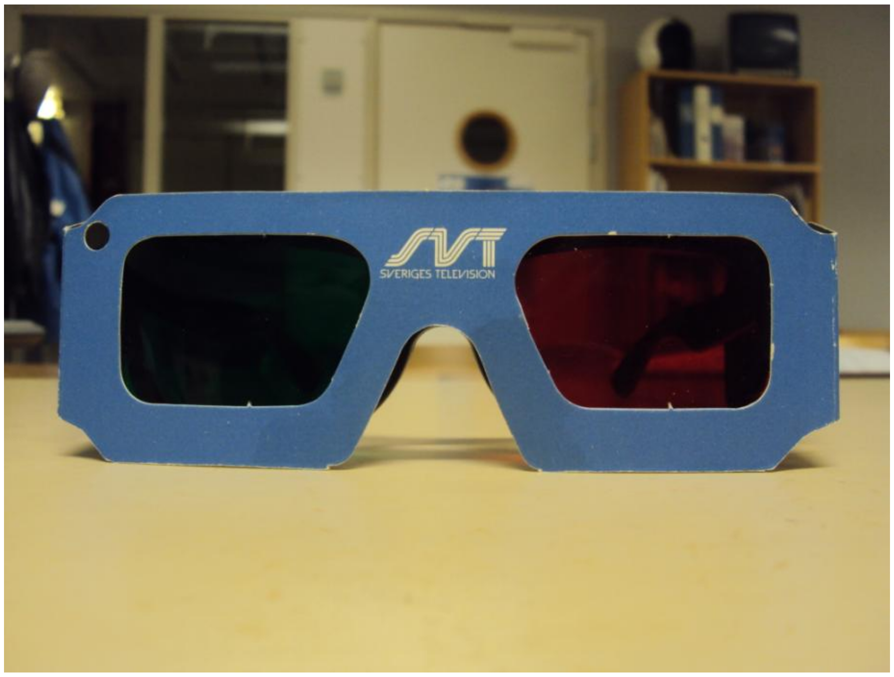


Figure A3: **Anaglyph Glasses (Green and Red)**

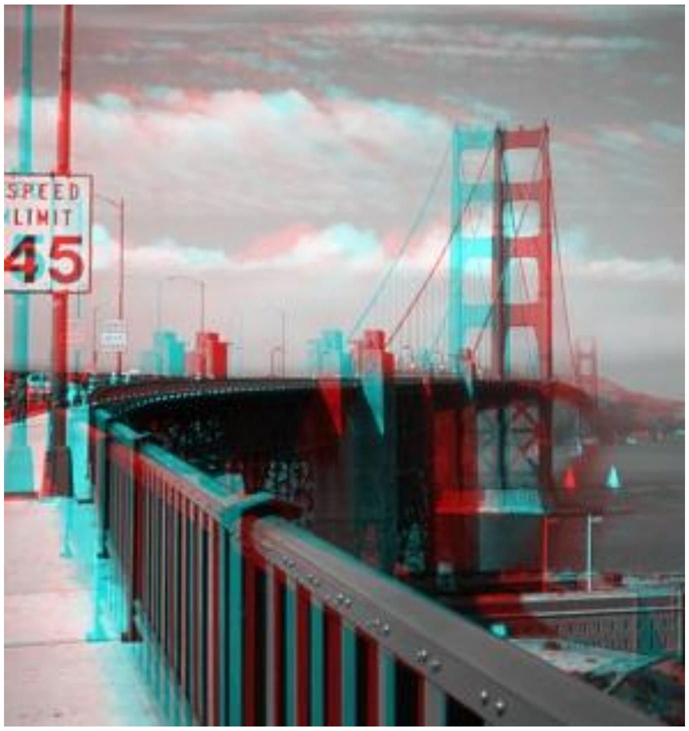


Figure A4: **Anaglyph Image**

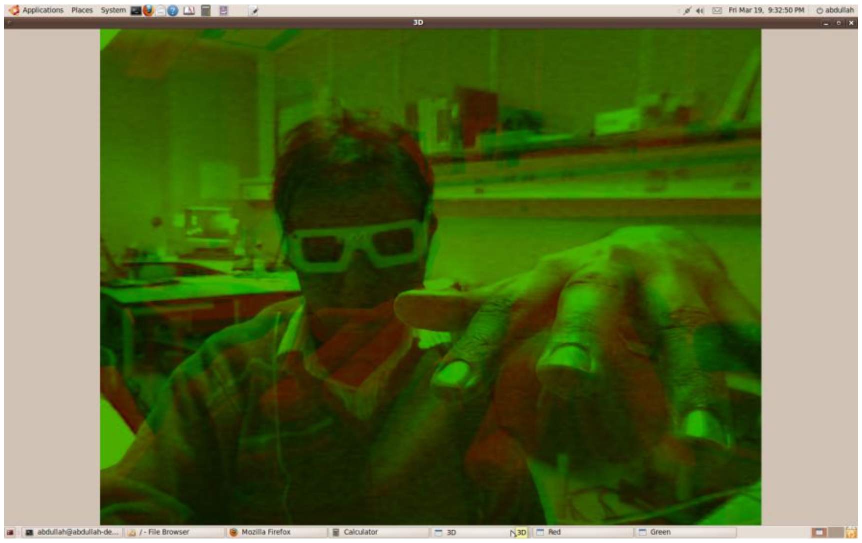


Figure A5: Generated Anaglyph Video

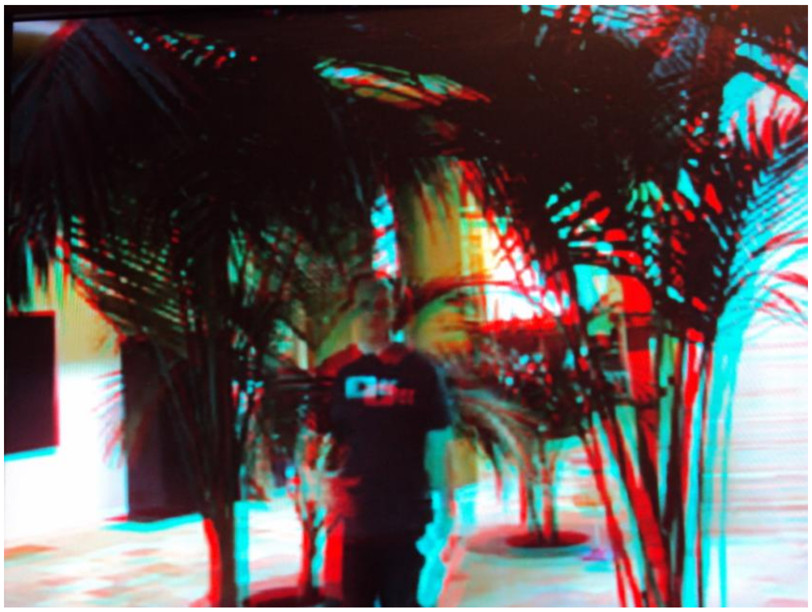


Figure A6: 3D on YouTube

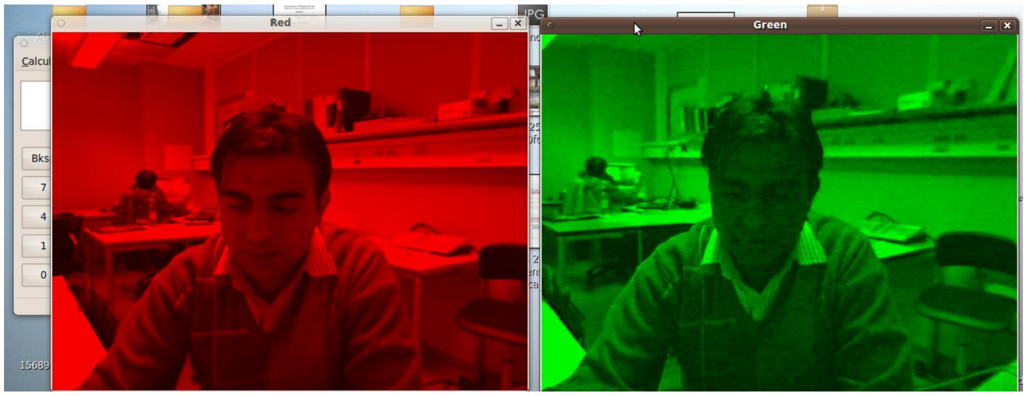
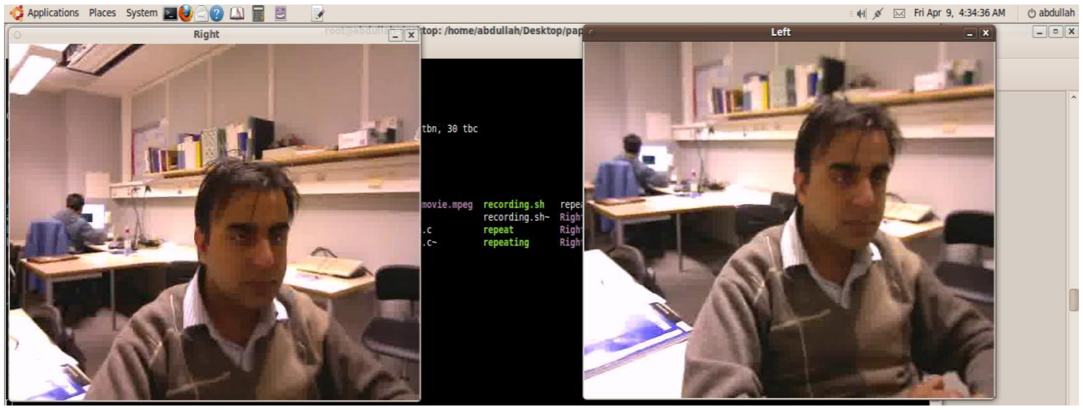


Figure A7: Generated two different color channels from left and right views



```
tbn, 30 tbc

movie.mpeg   recording.sh    repe
             recording.sh~   Righ
.c           repeat          Righ
.c~          repeating       Righ
```

Figure A8: Generated left and right simultaneous views

APPENDIX B

User connection	Rated bandwidth	Actual throughput	Safe bit-rate
14.4K modem	14.4 kbit/s	9.6 kbit/s	8 kbit/s
28.8K modem	28.8 kbit/s	19.2 kbit/s	16 kbit/s
33.6K modem	33.6 kbit/s	24 kbit/s	20 kbit/s
56K modem	56 kbit/s	40 kbit/s	32 kbit/s
Single ISDN	64 kbit/s	64 kbit/s	64 kbit/s
Dual ISDN	128 kbit/s	128 kbit/s	128 kbit/s
DSL downlink	384 kbit/s	300 kbit/s	240 kbit/s
E-1	2.048 Mbit/s	2.048 kbit/s	2 Mbit/s
Cable modem	6 Mbit/s	2.4 Mbit/s	300 to 1000 kbit/s
Intranet/LAN	10 Mbit/s	3.0 Mbit/s	2 to 3 Mbit/s
100Base-T LAN	100 Mbit/s	10 Mbit/s	6 to 10 Mbit/s

Table 4.1 Actual throughputs and safe bit-rates for different user connections