Uzun Mesafeli Aktarım İçin Görüntü Sıkıştırma

Ahmet Demir

YÜKSEK LİSANS TEZİ

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Eylül 2021

Video Compression for Long Range Transmission

Ahmet Demir

MASTER OF SCIENCE THESIS

Department of Electrical and Electronics Engineering

September 2021

Video Compression for Long Range Transmission

Ahmet Demir

A thesis submitted to Eskişehir Osmangazi University
Graduate School of Natural and Applied Sciences in partial
fulfillment of the requirements for the degree of Master of Science
in Discipline of Telecommunication-Signal Processing
of the Department of Electrical and Electronics Engineering
by

Ahmet Demir

Supervisor: Prof. Dr. Hakan Çevikalp

This thesis was supported by ESOGU BAP within the framework of the project "202015026"

September 2021

ETHICAL STATEMENT

I hereby declare that this thesis study entitled “Video Compression for Long Range Transmission” has been prepared in accordance with the thesis writing rules of Eskişehir Osmangazi University Graduate School of Natural and Applied Sciences under academic consultancy of my supervisor Prof. Dr. Hakan ÇEVİKALP. I hereby declare that the work presented in this thesis is original. I also declare that, I have respected scientific ethical principle and rules in all stages of my thesis study, all information and data presented in this thesis have been obtained within the scope of scientific and academic ethical principles and rules, all materials used in this thesis which are not original to this work have been fully cited and referenced, and all knowledge, documents and results have been presented in accordance with scientific ethical principles and rules. 03/09/2021

Ahmet Demir

Signature

ÖZET

Dar bir bant üzerinden gerçek zamanlı, yüksek kaliteli, kablosuz video aktarımı ticari, askeri ve sivil alanlar gibi bazı alanlarda artan iletişim ihtiyacından dolayı araştırmacılar için güncel bir çalışma alanı konumundadır. Gerçek dünyada karşılaşılan haberleşme sistemi problemlerini yazılım problemlerine dönüştüren Yazılım Tanımlı Radyo (YTR) sistemleri, araştırmacılar tarafından haberleşme sistemlerinin tasarımı sırasında zaman, maliyet ve iş gücü tasarrufu sağlayan çok iyi bir teknoloji olarak tanımlanmaktadır. Bu YTR sistemlerini gerçekleyen bazı ortamlar geliştirilmiştir. GNU Radio bu ortamlardan bir tanesidir ve esnek YTR sistemlerinin geliştirilebilmesi için bazı sinyal işleme blokları içerir. YTR sistemleri işlenen sinyallerin radyo frekansında gönderilip alınması için bazı donanımlara ihtiyaç duyar. Araştırmacılara yüksek kaliteli haberleşme sistemlerinin geliştirilmesi fırsatını veren USRP cihazları bu donanımlardan bir tanesidir.

Bu tez çalışmasında, GNU Radio ve USRP cihazlarını içeren YTR sistemlerini kullanarak İHA (İnsansız Hava Aracı) cihazı ve bir merkez istasyonu arasında gerçek zamanlı bir video aktarımı amaçlanmaktadır. İHA kamerasından alınan ham bir video Gstreamer ve VLC programları kullanılarak yeni bir video kodlama yöntemi olan H.264 ile sıkıştırılır. Sıkıştırılmış video sinyali GNU Radio tarafından yapılan paket kodlama ve GMSK modülasyonu işlemleri ile kablosuz aktarım için hazırlanır. Son olarak, elde edilen video sinyali gerçek zamanlı yüksek bant genişliğine sahip olan bir USRP B210 cihazı kullanılarak kablosuz aktarım için radyo frekansına yükseltilir. Bu işlemlerin tersi alıcı tarafı olarak bilinen merkez istasyonunda yapılır. Sonuç olarak, alınan video merkez istasyonda Mplayer video oynatıcısı ile oynatılır. Önerilen sistemin gerçek hayatta gerçeklemesi ile elde edilen deneysel sonuçlar, yüksek kaliteli ve gerçek zamanlı bir videonun çok küçük bir bozulma ile bir alıcı ve bir verici cihazı arasında aktarılabileceğini göstermiştir. Bu tez çalışmasında hedefimiz analog video aktarımı sırasında görülen bozulmaları azaltmak ve İHA cihazından merkez istasyonuna faydalı bilgiler aktarmaktır.

Anahtar Kelimeler: YTR; GNU Radio; H.264; Video Aktarımı; Kablosuz Aktarım

SUMMARY

Real-time, high quality, wireless video transmission over a narrow band has been a popular study field for some researchers due to increased connectivity needs in some fields like commercial, military, and civilian. Software Defined Radio (SDR) systems converting the real world communications system problems into software problems are defined as a great technology by the researchers to save time, cost, and effort in the design process of the transmission systems. Some environments realizing these SDR systems have been developed. GNU Radio is one of these environments, and it contains some signal processing blocks for the development of some flexible SDR systems. SDR systems need some hardware to transmit and receive the processed signals in radio frequency. Universal Software Radio Peripheral (USRP) device giving high quality communication system developing opportunity to researchers is one of these hardwares.

In this thesis study, it is aimed to transmit a real-time video between an Unmanned Aerial Vehicle (UAV) device and a center station by using SDR systems containing GNU Radio and USRP devices. A raw video taken from a UAV camera is compressed with a new video coding technique named as H.264 by using Gstreamer and VLC programs. Compressed video signal is prepared for the wireless transmission by GNU Radio with packet encoding and Gaussian Minimum Shift Keying (GMSK) modulation operations. Lastly, obtained video signal is raised to the radio frequency for wireless transmission by using USRP B210 device with a high real-time bandwidth. Reverse of these operations are done on the center station known as receiver side. Lastly, the received video is displayed by Mplayer video player in the center station. Experimental results obtained by the real environment implementation of the proposed system suggest that a high quality and real-time video can be transmitted between a transmitter and a receiver device with little distortion. Our aims in this thesis study are to diminish the distortions seen in analog video transmission, and transmit some useful information from the UAV device to the center station.

Keywords: SDR; GNU Radio; USRP; H.264; Video Transmission; Wireless Transmission

LIST OF CONTENTS

Page

LIST OF CONTENTS (continue)

Page

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (continue)

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF ABBREVIATIONS AND SYMBOLS

Symbols	Descriptions
W	Watt
Hz	Hertz
dB	Decibel
k	Kilo
G	Giga
M	Mega
m	Mili
S	Sample
s	Second

Abbrevitions	Descriptions
YTR	Yazılım Tanımlı Radyo
İHA	İnsansız Hava Aracı
SDR	Software Defined Radio
USRP	Universal Software Radio Peripheral
UAV	Unmanned Aerial Vehicle
GMSK	Gaussian Minimum Shift Keying
VCO	Voltage Controlled Oscillator
PLL	Phase Locked Loop
SNR	Signal to Noise Ratio
BER	Bit Error Rate
FSS	Free Software Foundation
CR	Cognitive Radio
DVB-T	Digital Video Broadcating-Terrestrial
MER	Modulation Error Ratio
OFDM	Orthogonal Frequency Division Multiplexing
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation

LIST OF ABBREVIATIONS AND SYMBOLS (continue)

Abbrevitions	Descriptions
DSA	Dynamic Spectrum Access
SSIM	Structural Similarity Index
RSSI	Received Signal Strength Indicator
PU	Primary User
SU	Secondary User
UDP	User Datagram Protocol
JVT	Joint Video Team
CABAC	Context-Adaptive Binary Arithmetic Coding
CAVLC	Context-Adaptive Variable Length Coding
AVI	Audio Video Interleave
VOD	Video on Demand
MSK	Minimum Shift Keying
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
FPGA	Field Programmable Gate Array
DUC	Digital Up Converter
DDC	Digital Down Converter
IF	Intermediate Frequency
RF	Radio Frequency
USB	Universal Serial Bus
MIMO	Multiple Input Multiple Output
UHD	USRP Hardware Driver
SWIG	Simplified Wrapper and Interface Generator
bps	Bit per Second

1. INTRODUCTION AND PURPOSE

Wireless communication systems have become an evolving and the most studied research area in recent years due to advanced demand for wireless connectivity in some fields like commercial, military, and civilian. Transfer of data, images, and videos which are requiring a wide bandwidth is tried to be realized on the same medium which is used for voice transmission normally. Wireless communication performance is aimed to be increased by some techniques which are applied in simulations. However, real world implementation of these techniques with some hardware requires too much time, effort, and a high cost, and so it is seen as a challenging task (Nguyen, 2013). Radio systems using traditional hardwares like filters, modulators, demodulators, and converters are expensive, and these hardwares have low compatibility with other components. Easy integration of different components and escaping from the usage of dedicated hardware are some of the reasons for using the evolving technology of SDR.

The main goal of SDR is to turn the problems encountered in hardware into some software problems. SDR is a radio system where some or all of the traditional hardware functions can be defined as software functions. For instance, adaptations of different modulators and demodulators in SDR can be managed by changing some software blocks easily (Chen et al., 2010). Therefore, a number of different radios can be acquired by using the same hardware components for a variety of applications. This is the reason why a low cost and flexible system can be created by using these SDR systems. Researchers using SDR are free to try new protocols, methods or techniques. Some environments such as GNU Radio, Virtual Radio, OSS, Microsoft's Sora, IRIS can be used to implement the SDR systems (Nimmi et al., 2014).

GNU Radio containing some signal processing blocks is a free and open-source SDR development environment, and it is licenced under GNU General Public Licence with copyright from Free Software Foundation (FSS). GNU Radio experiments are realized by the blocks, and these blocks can be added to the working area of GNU Radio by dragging and dropping them. C++ is used to create the library of signal processing blocks, and

Python is used to connect these blocks. There are some hardwares using SDR to implement communication related experiments. USRP which has the ability of transmitting and receiving radio frequency signals over a frequency range is one of these hardwares. Personal computers are given the ability of acting as high bandwidth SDRs by USRP (Zhang, 2013). Therefore, an improved communication system can be easily built by developers by using these devices.

In this thesis study, a real-time video taken from a UAV camera is aimed to be transmitted wirelessly to a center station by following some steps. In this system, operations done on the UAV side are known as the transmitter side operations, and operations done on the center station side are known as the receiver side operations.

Transmitter side operations consist of taking the raw video from UAV camera, compressing, scaling, and pipelining it by using Gstreamer or VLC programs, creating a GNU Radio flow graph for encoding and modulating this pipelined video, converting the digital video signal taken from GNU Radio to analog signal, and raising it to the radio frequency for wireless transmission by using USRP devices.

Receiver side operations consist of taking the radio frequency signal transmitted from the transmitter side, converting it to a digital signal to transmit to GNU Radio, demodulating and decoding this signal by GNU Radio, and lastly displaying the obtained video signal by Mplayer.

Some simulations of real-time video streaming operation are also implemented in this study. Since signal is not transmitted on air, an artificial noise source is used to represent the noise added to the transmitted signal on air. Sensitivity of different modulation techniques are obtained by measuring bit error quantities of each of the techniques by using different magnitudes of noises. Moreover, the effect of Multiply Const value on SNR is examined under a constant noise amplitude.

This thesis study consists of 4 parts in total. These are Literature Review, Materials and Method, Results and Discussion, and Conclusions and Recommendations parts respectively.

Firstly, all of the different studies that are showing history of video transmission operations realized by SDR systems in the real-time video streaming field are given with details in the **Literature Review** part.

Second, all the video transmission system steps followed in this study, H.264 video compression method which is used to obtain higher quality videos, Gstreamer and VLC programs which are used to compress and stream real-time video, GMSK modulation which is a method to get modulated digital video signals, USRP which is a hardware for wireless radio frequency transmission, and lastly GNU Radio which is used to prepare the digital signal for transmission are explained in details in **Materials and Method** part.

Third, all the operations done on the transmitter and receiver sides for the real environment implementation of wireless video transmission and some simulations containing all the transmitter and receiver side signal processing operations are given in **Results and Discussion** part.

Lastly, in the **Conclusions and Recommendations** part of this thesis study, obtained experimental results and some future works for the researchers who want to realize a higher quality video transmission are discussed.

2. LITERATURE REVIEW

Dhruv and Vishal (2017) transmit a good quality, real-time video with little data loss by using USRP B210. This research creates a pathway for real-time video transmission by using pipelines created between Gstreamer and USRP devices. One pipeline is created for the transmitter and one pipeline is created for the receiver. Gstreamer is used for H.264 encoding and H.264 decoding of taken video, and lastly creating the pipelines. GNU Radio is used for all of the signal processing operations in transmission process. In this study, a constant source is transmitted firstly. For this aim, both the simulation showing transmission of a constant source and the transmitter-receiver pairs of transmission are given. Secondly, transmitter-receiver pairs for the real-time video taken from a camera are given.

Patil et al. (2017) do some analysis to manage real-time video transmission with a GMSK system by using a GNU Radio flowgraph. Test of this system is done on some software programmed hardware transceiver SDR-LAB kits which are allowing the design and implementation of SDR systems easily. The main goal of this study is to show how the modulation techniques would effect the video transmission in Cognitive Radio (CR) environment. CR finds the frequency band that is not used by the primary users at any moment, and allows the secondary users to use that frequency band for the aim of efficient usage of radio spectrum. The experiments done in this study suggests that GMSK is a good method for the real-time video transmission task.

Crespi et al. (2012) broadcast a real-time video by using the Digital Video Broadcasting - Terrestrial (DVB-T) digital transmission standard and the GNU Radio platform over USRP2 board. Since only a small part of x86 instructions are used, this study enables the users to use low power Central Processing Units (CPUs) in the real-time video transmission process. Moreover, some vital parts of encoding and modulation operations are faster by the extensive usage of memory. Modulation Error Ratio (MER) obtained by the application of the method mentioned in this study is measured as 44 dB. Another test done on this study is the CPU usage test. Three cases are tried in total for it. In the first

case, C++/python is used for time-domain resampling and reached the worst CPU usage. In second case, resampling is done in frequency-domain by using C++/python and reached a better usage of CPU compared to the first case. In the last case, only C++ is used for the implementation of frequency-domain resampling and reached the best CPU usage compared to first two cases. Windowing effect is also examined in this study with a test, and better spectral characteristics are reached by using a raised cosine window to filter the signal.

Ghani et al. (2017) use television white space to transmit a video captured by a program having video capturing and displaying capabilities. This system is based on Orthogonal Frequency Division Multiplexing (OFDM), and it uses SDR and USRP B200. Indoor environment video transmission under ultra high frequency is managed by using two types of modulations which are Phase-shift keying (PSK) and Quadrature amplitude modulation (QAM) in this study. Transmitter and receiver parts of this video communication task are created by using GNU Radio layers. The allocated system bandwidth values used in this study are 0.5 MHz, 1 MHz and 2 MHz, and the best performance is reached in 1 MHz frequency case. Performance metrics used in this study are packet delivery ratio and end to end delay. When the experiments realized at 1 MHz system bandwidth are examined, medium and fast motion videos are transmitted with a better performance with the usage of BPSK, and slow motion videos are transmitted with a better performance with the usage of QPSK modulation.

Debashri et al. (2020) aim to create a dynamic spectrum access network for HD and 360 degrees video stream. They study on video encoding and channel selection adaptation subjects. Encoding parameters are selected according to some parameters such as received signal strength indicator (RSSI) of received signal, received video quality at receiver, and pathloss of the signal in transmission process. Moreover, some channel parameters like center frequency and channel bandwidth are selected according to the transmission activities of the primary users by the radio transmitter. In this study, the best encoding bit rates are found by a mechanism which is based on multilevel threshold. Moreover, they propose another threshold based mechanism to find optimal channel between transmitter and receiver. These proposed methods are tested on a real-time video taken from a webcam and a 360 degrees VR camera. USRPs and GNU Radio are used for transmission in an

indoor environment. Gstreamer is used to stream the H.264 encoded video. Omni-directional antennas, two B210s for transmitter and receiver, and one B210 for channel energy level sensing are used. Transmitter and receiver signal processing blocks are created in GNU Radio. Received video quality is determined by looking at two parameters: PSNR (Peak Signal to Noise Ratio) and Structural Similarity Index (SSIM). As a result of the test process of this study some results are reached. Firstly, even if the RF conditions change, video encoder and USRPs can adapt to new conditions. Secondly, video quality is increased after the application of adaptation schemes. Lastly, it is seen that USRPs can change their active channels fast enough. A simulation showing the applicability of the proposed method to larger networks is also presented in this study.

Nimmi et al. (2014) propose a real-time video streaming method which is using Gstreamer and GNU Radio. Gstreamer is used for frame capturing from a camera, compressing the captured frame, encoding the compressed data and lastly muxing the resultant data into a stream. GNU Radio is used for the design of transmitter and receiver blocks. A pipeline created by Gstreamer is used to transmit the video taken from a basic camera to GNU Radio. Mplayer is used for real-time displaying of received video streams in this study. Furthermore, they buffer received video for the aim of storage. This study shows that reliable transmission of real-time video can be achieved by using USRP and RTL-SDR devices.

Tahir et al. (2012) apply Dynamic Spectrum Access (DSA) in their study to achieve a better video transmission performance. Their aim is to show if a system using DSA does high bandwidth video transmission with a better quality than the systems which do not use DSA. They use GNU Radio and Ettus Research USRP N200 in their study. They have two primary users (PU) and two secondary users (SU). One of these PUs is set to a constant 2.4 GHz frequency, and it uses OFDM modulation. Second PU starts scanning a predefined frequency range except 2.4 GHz after ten seconds of transmission. Moreover, one of the SUs is used to transmit the video signal, and the other SU transmits the signal containing randomly generated bits. Data transmitting SU uses GMSK modulation. SU devices are programmed to work independently. Spectrum sensing operation is done by scanning some predefined spectrum range. When a frequency having less energy than a threshold is found, the transmitter is informed, and it is required to pass this frequency. Transmitter switches

its frequency to this value only when its current operating frequency has higher energy than the energy of this frequency. In addition to these USRPs, one more USRP is used to receive the video signal. GNU Radio Companion is used to create all the necessary blocks for video transmission. Video transmission operation has a few steps in this study. Firstly, H.264 encoded video stream is transmitted to GNU Radio by using a UDP (User Datagram Protocol) port. Then, all the signal processing operations on this stream are done by GNU Radio, and GNU Radio transmits this processed data to USRP for transmission at the predefined 2.4 GHz frequency. This frequency is changed according to the spectrum sensing requests. As a result of this experiment, although there are some packet losses in systems using DSA, it is shown that a system using DSA performs video transmission operation with a better quality compared to the systems using non-DSA.

Shavanthi et al. (2018) do video transmission tests and simulations by using different modes of transmission, modulations and code rates. They implement the real-time video transmission task by using USRP B210 devices and GNU Radio. They design a Digital Video Broadcasting-Terrestrial/Satellite (DVB-T/T2/S/S2) for the aim of video transmission and reception. DVB having some standards is a digital video transmitter. This study uses a stored video file in the transmission process, and GNU Radio is used for the modulation and processing of this video. USRP is used to transmit the signal using DVB-T/T2/S/S2 standards. These standards are seen in the digital video and television signal sending operations. DVB requires less power for transmission compared to other standards. DVB standards differ from each other by some parameters like used data rate, modulation scheme, and channel bandwidth. This study shows that a successful transmission of a stored video file can be achieved by using USRP B210 and GNU Radio. This study tests transmission with and without an antenna. Different observations are acquired by using different sample rates (2M, 1M), transmission modes (2K, 8K), FFT Lengths (1024, 2048, 4096, …), modulation schemes (QPSK, 16QAM, 64QAM), and lastly code rates (1/2, 2/3, 3/4, …). Although a latency in real-time transmission of the video is seen, a successful implementation of different standards in video transmission is shown in this paper.

3. MATERIALS AND METHOD

Video streaming task using USRP devices is aimed in this study, and Figure 3.1 shows all the steps followed in this study. A camera is used to capture the video frames, and they are H.264 encoded and pipelined by using either VLC or Gstreamer. This processed digital video data is taken by the transmitter NVIDIA Jetson Nano device, and some operations like modulation and encoding are done on it by GNU Radio software. Output of this device is transferred to the transmitter USRP for streaming. Some signal processing operations on the received signal are done by the receiver side USRP. Output data is transferred to a laptop, and some operations like demodulation and decoding are done on it. Lastly, processed data is transferred to Mplayer to display. All the components creating this system are explained in details below.

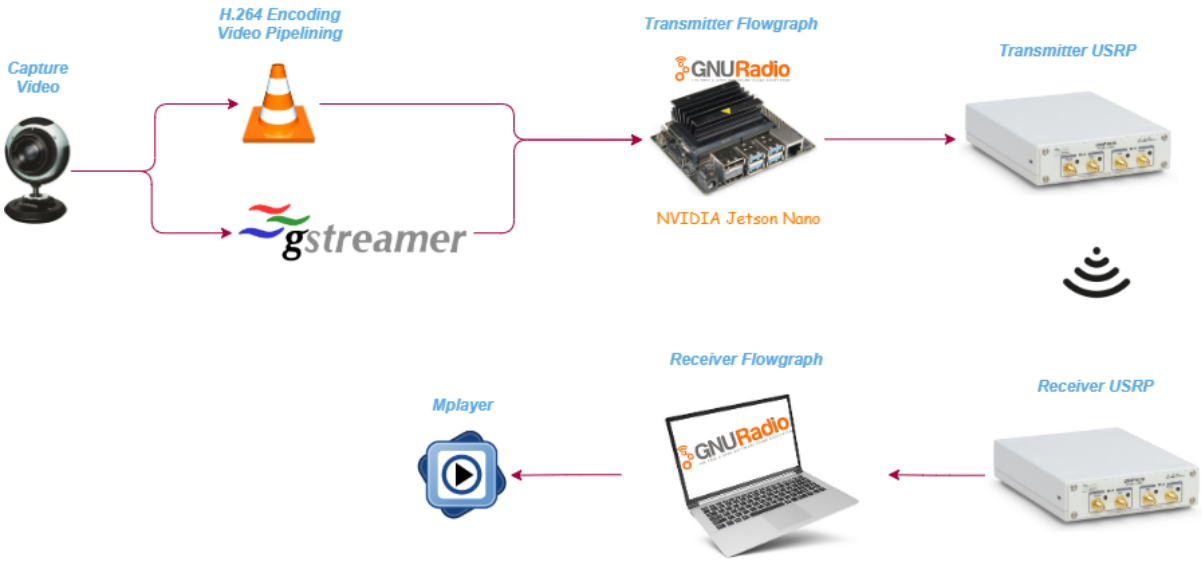

Figure 3.1. Real-Time Implementation of Video Streaming Using USRP and GNU Radio

3.1. H.264

Compression of raw digital video is vital for the visual information transmission. Video compression methods transmit some information about any video frame instead of transmitting the raw video frame. Video frame predicting method and predicted difference

values are some information which are transmitted for any video frame. The reason why a video is compressed before sending is to remove the information which is not necessary for the reconstruction of that video on the receiver side. Video compression is used widely in some different areas such as the transmission of medical videos and 3D medical images in low bandwidth networks (Yu et al., 2005). Since some transmission media offers lower data rates, coding efficiency has a great importance for the transmission of higher quality videos. Figure 3.2 shows some coding methods which make possible higher quality video transmission on lower bandwidth networks.

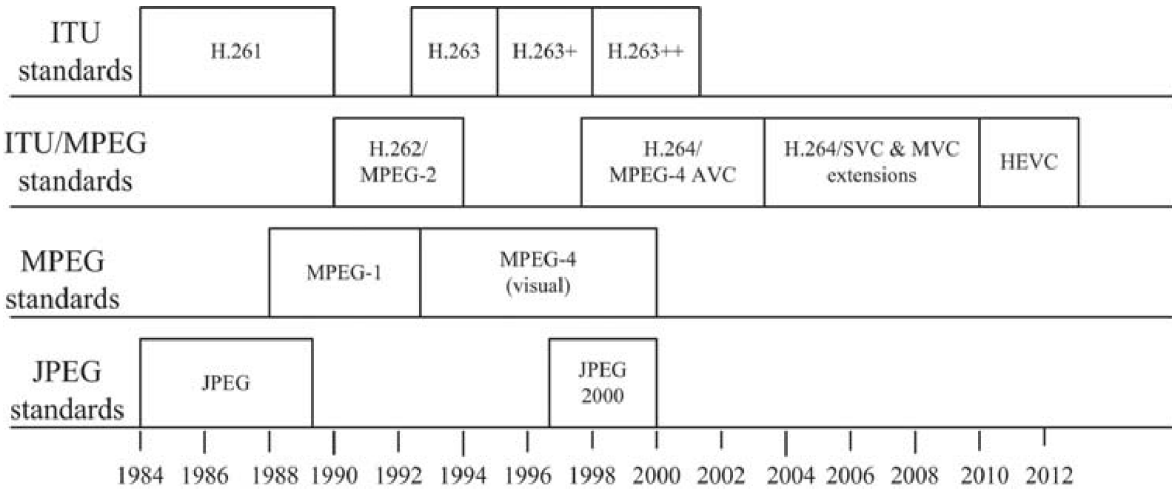

Figure 3.2. Coding standards

H.261 was invented in 1990 with the aim of combining speech and data on the same line. H.261 does not work well over TCP/IP Internet. Macroblock which is a video processing unit is firstly seen in H.261. Digital television systems were facilitated in 1994 by the H.262 standard that is also known as MPEG-2. H.262 enables us DVD storage of standard definition videos, and standard definition and high definition TV signal transmission. H.263 was invented in 1995, and it can be used in a wide range of bandwidths. H.263 uses a video coding algorithm grounding to H.261 standard. H.263 is used in video conferencing and cell phone codec today. After H.263, some projects like H.263+ and H.263++ are studied to have better performance on video compression.

ITU-T Video Coding Experts Group and ISO/IEC Moving Picture Experts Group joined with the name Joint Video Team (JVT) to create H.264 video coding standard as a

result of the need for a new video coding technique (Wiegand, 2003). Main goals of them were to obtain an improved video compression method and provide network-friendly representation of video. Although H.264 uses similar coding with the previous standards, rate distortion efficiency is greatly enhanced, and intercoding flexibility is increased in H.264 standard compared to the previously used video compression standards. According to a study, H.264 reaches a coding gain of 50% compared to MPEG-2, and 47% coding gain compared to H.263 (Kamaci and Altunbaşak, 2003). Although H.264 gives great results in some applications like network video transmission, videoconferencing, streaming of video, HD broadcasting and hard disk storage, it is too complex with respect to the previous standards.

H.264 has some elements in the video coding process. These elements form an encoder as shown in Figure 3.3. Encoder's aim is to select between intra-coding and inter-coding firstly. There are two components in every video frame which are luma and chroma. Encoder takes video frames as macroblocks and the size of these macroblocks for luma is 16×16, and 8×8 for chroma. A macroblock can be divided into blocks having sizes ranging from 4×4 to 16×16 for the aim of motion-compensated prediction.

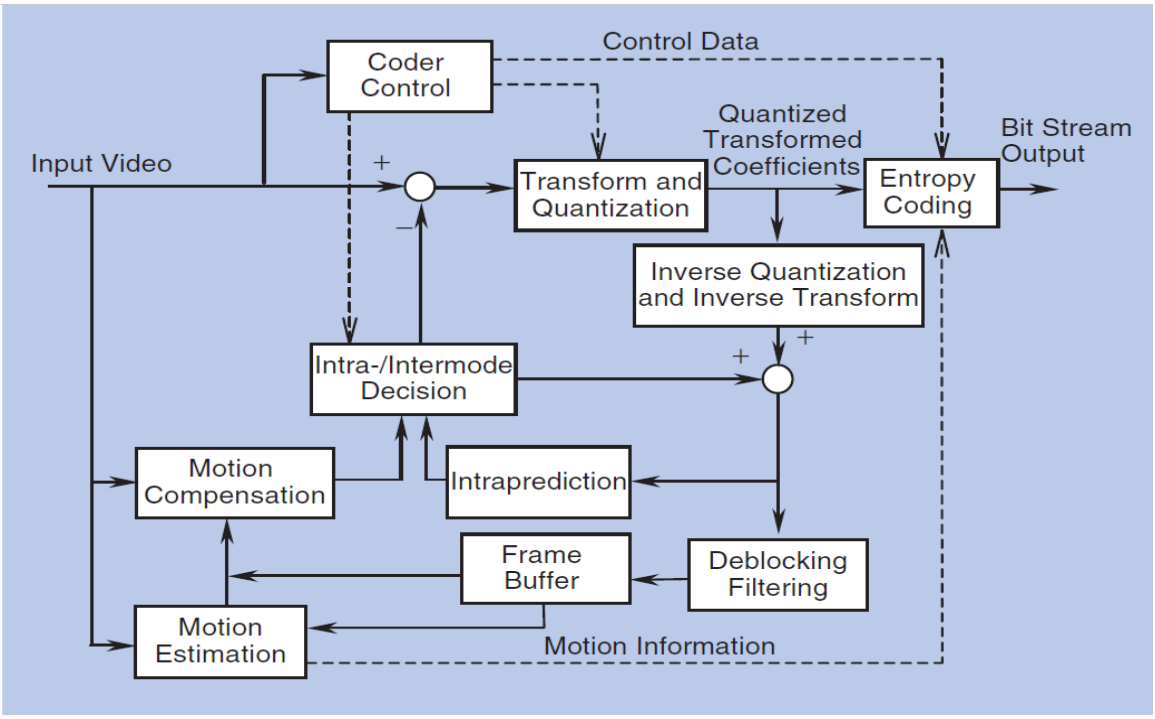

Figure 3.3. Encoder structure of H.264 standard

Intra coding which is firstly used in H.264 standard is done by coding a picture without looking at any other previously decoded pictures. It uses only previously decoded blocks of the picture for intra prediction. Intra coding gives detailed information about decoding of the encoded pictures correctly. Aim of intra coding is to diminish spatial redundancy in a video frame. Some spatial prediction modes are used for this aim. For luma components, there are 3 types of subblocks having sizes 4×4, 8×8, and 16×16. For the chroma component, only 8×8 sized blocks are used. The number of mode directions differs with respect to the size of these subblocks and being luma or chroma block. The number of direction modes for 4×4 and 8×8 luma blocks is 9. Moreover, 4 direction modes are used for 16×16 luma blocks. All chroma components use 4 prediction mode directions. These prediction mode directions are named with numbers like mode 0, mode 1, mode 2, etc. Figure 3.4 shows some prediction mode directions for 4×4 luma subblocks. As can be seen in the figure, there are 9 prediction mode directions in total. Mode 2 which is not seen in this figure takes the average of samples to find predicted samples, and so it is known as the DC component.

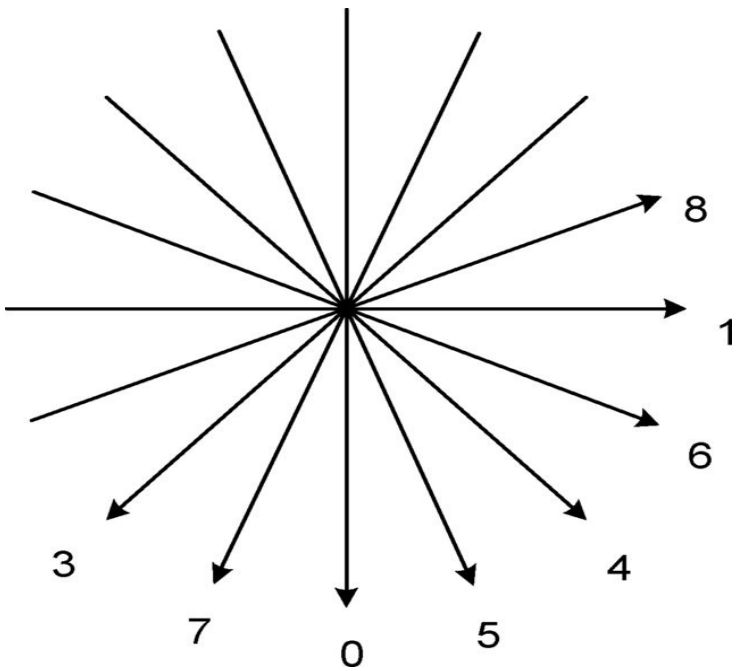

Figure 3.4. Nine prediction mode directions for 4×4 luma subblocks

Intra coding uses gray values of coded pixels to find residual data (Wang, 2011). Higher compression ratios than previous coding standards can be reached by compressing video frames both in time domain and in space domain with intra coding. Prediction model for a macroblock is selected according to deviation quantity between current macroblock and previously decoded and reconstructed pixel value which is on the edge of current macroblock. A cost function is selected to obtain the best of these prediction models.

On the other hand, intercoding uses previously decoded video frames to predict each block value in the current picture. More than one reference picture can be used in intercoding of a frame. Aim of intercoding is to reduce the temporal redundancy by using correlations among different pictures, and this results in bit rate reduction. Inter Prediction is done in the intercoding step, and motion vectors are used for this aim. Different sized macroblocks ranging from 4×4 to 16×16 are used in inter frame prediction. Smaller block sizes are preferred for the video frame areas containing some details.

Deblocking filtering is another element used in H.264 encoder structure. This filtering is done after the quantization step to diminish the blocking artifacts resulting from the quantization step at the block boundaries. Before a block is quantized, a transform on prediction residual is applied to obtain a more compressed prediction residual. The last step of the encoding process is to combine quantized transform coefficients, motion vectors or intra prediction modes, control data coming from coder control, and using entropy coding to obtain bit stream output. H.264 uses context-adaptive binary arithmetic coding (CABAC) and context-adaptive variable length coding (CAVLC) as entropy codes (Kalva, 2006).

In this study, two video recordings are done to see the coding performance of H.264 encoding. One of these recordings is in uncompressed packed YUV format, and the other is in ITU H.264 (High 4:2:2 Profile) format. Every recording is done in 30 seconds by using below Gstreamer commands. Figure 3.5 and Figure 3.6 show raw video file properties and H.264 encoded video file properties respectively.

Uncompressed packed YUV format video recording:

gst-launch-1.0 -v v4l2src device=/dev/video0 ! videoscale ! video/x-raw ,width=800, height=600, framerate=15/1 ! avimux ! filesink location=original.avi

ITU H.264 (High 4:2:2 Profile) format video recording:

gst-launch-1.0 -v v4l2src device=/dev/video0 ! videoscale ! video/x-raw ,width=800, height=600, framerate=15/1 ! videoconvert ! x264enc ! avimux ! filesink location=H264encoded.avi

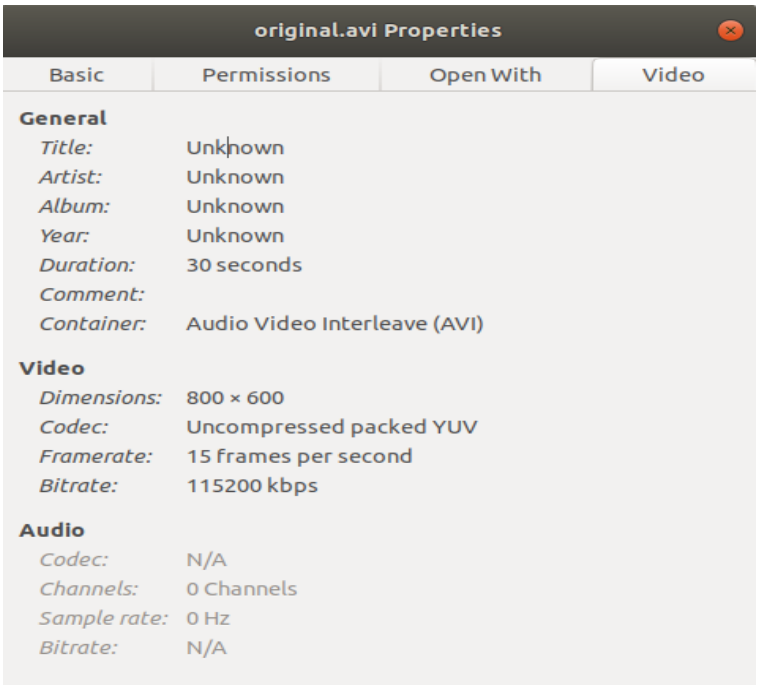

Figure 3.5. Uncompressed video file properties

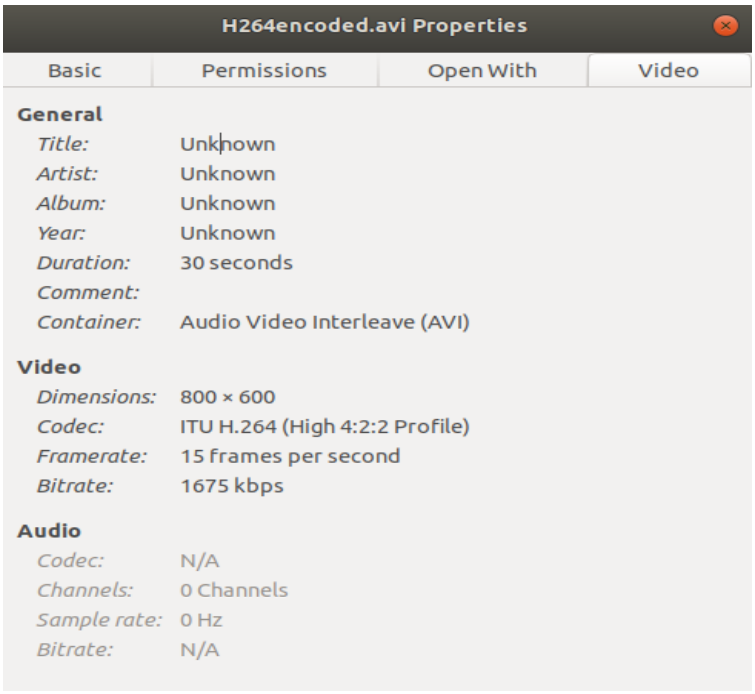

Figure 3.6. H.264 encoded video file properties

Both of the recordings are done by using a dimension of 800×600 pixels, a frame rate of 15 frames per second, and Audio Video Interleave (AVI) container for the true comparison. Figure 3.5 shows that the bit rate of uncompressed video is 115200 kilo bit per second (kbps). On the other hand, Figure 3.6 shows that H.264 encoded video bitrate is 1675 kbps. These results show that bitrate of uncompressed raw video is about 69 times higher than the bitrate of H.264 encoded video. When the file sizes of these two recordings are examined, it is seen that the size of uncompressed video file is 441.6 MB, and the size of the H.264 encoded video file is only 6.4 MB. This result shows that raw video file size is about 69 times higher than the H.264 encoded video file size.

3.2. Gstreamer

Streaming media applications are developed by using Gstreamer framework, and they are written by using C programming language. Gstreamer is compatible with Linux, Solaris and Open Solaris, Microsoft Windows, FreeBSD, NetBSD, OpenBSD, Mac OS X, and OS/400. Gstreamer sticks to GObject, GLib 2.0 object model (Gstreamer, 2021).

Gstreamer makes it possible to write all types of streaming media applications thanks to its development framework. A number of formats such as MP3, Ogg/Vorbis, AVL mod, MPEG-1/2, QuickTime, H.264 are supported by Gstreamer (Taymans et al., 2013). Audio or video or both can be acquired by using Gstreamer. Gstreamer is used by some media players as a core cornel (Gstreamer, 2021). However, it is more complex than these media players as it supplies some functions like streaming changing, recording etc. Its main goal is to create pipelines by mixing and matching some pluggable components.

Plugins form the basic structure of Gstreamer framework, and they serve various codecs and some functionalities. A pipeline in gstreamer framework is created by linking these plugins. Gstreamer has over 250 plugins supplying more than 1000 elements. Gstreamer plugins are created by using protocols handlers, sources, formats, codecs, filters, and sinks.

Elements in Gstreamer execute some basic functions like reading, decoding, and outputting of data. Elements are combined to create the pipelines mentioned above to do some combinational tasks like media capturing and playing. Input and output part of the elements where data is flowing through are called as pads in the gstreamer framework.

A pad can only handle the data that is accepted by the element containing that pad. Pads can be connected to other pads only when data types of them match. Data types are defined as GstCaps, and negotiation of them is done by a process called as caps negotiation (Sundari, 2015). There are two types of pads: source pads and sink pads. Data leaves an element by using one or more source pads, and data enters to an element by using one or more sink pads. A number of elements are combined to create bins.

When a Gstreamer Project is too complex, bins are used to change the state of all the elements simultaneously. When a bin is created, the situations of individual elements are not important for us, we only look at the bins situation. In this way, a complex pipeline can be divided into smaller parts with the usage of these bins. All the bins come together to create pipelines. Task of these pipelines is to ensure synchronization. All the Gstreamer applications must include these pipelines.

Figure 3.7 shows a media flowing diagram containing 2 source outputs, 2 sink inputs, 3 elements and 1 bin. Figure also shows that the combination of all the elements forms a bin, and elements are connected to each other by using their src and sink pads.

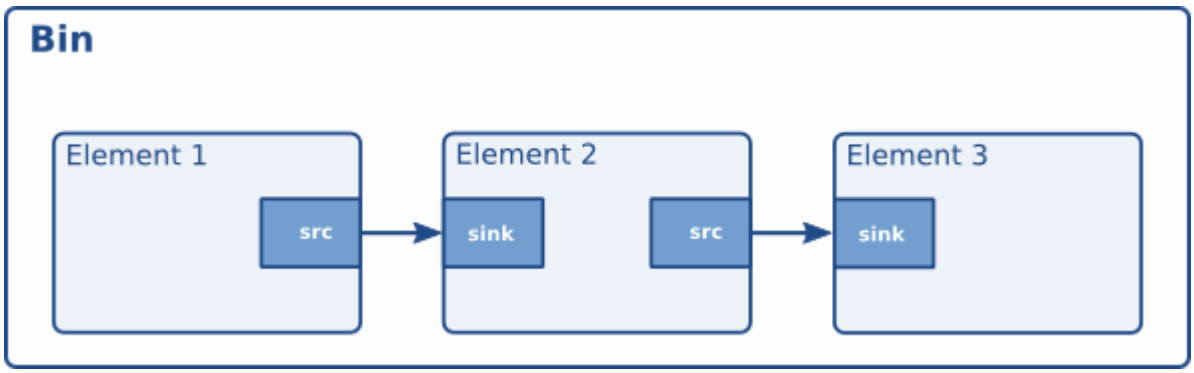

Figure 3.7. Gstreamer media flowing pipeline

3.3. VLC

VideoLan is a software creating organisation to play and stream video files or any other media formats. The VideoLan Project started as a school Project at L'Ecole Central des Paris in 1996 to watch television on PCs (Personal Computers). As a result of some studies, the first stream was served and read in 1998 (VideoLAN, 2020). This Project was made open source in 2001. VideoLan is most notable for its flagship Project, VLC media player.

VLC media player was only a client at first. Today, VLC media player is not only a multimedia client but a server which streams multimedia as well. After this improvement, it is named as VLC media player instead of VideoLAN Client. Besides commercial products like Freebox and Di.com, VLC is used in a number of some other areas. There are some hardwares that VLC media players accept as input like DVD, VCD, SVCD, Acquisition, Encoding cards, Satellite dish, Webcams, DV camcorders, network cameras, DirectShow/DirectX devices (Saman, 2006).

VLC network streams can be created in a few ways such as broadcast, unicast, multicast, and video on demand (VOD). Network broadcast address is used to stream in

broadcast streaming, and different port numbers are used for each multimedia stream. Advantage of this method is that all the clients on a network can access data. On the other hand, one disadvantage is the usage of network bandwidth inefficiently. When a server uses some predefined set of ip addresses to stream, this means that unicast streaming network is used. Different port numbers and ip addresses are used for each stream. This method uses less network bandwidth compared to the previous one. On the other hand, sending streams to only a number of known clients, and not giving the ability of controlling streams to clients are some disadvantages. Streamer server sends data to a multicast group in multicast streaming network case. Video on Demand streams data only when a request occurs, and each viewer has its own stream. Bandwidth is efficiently used in this case. However, server starts a new stream for each client.

3.4. GMSK Modulation

Modulation is a process to prepare a message signal to transmit in air. The most important aim of modulation is to use frequency spectrum efficiently in the transmission process. Modulation techniques use high frequency carrier signals. A message signal is transmitted by changing some physical properties of the carrier signals like frequency, phase, or amplitude according to the amplitude of the transmitted message signal.

GMSK that is widely used in radio communications systems is a digital modulation technique. GMSK modulation method is derived from the Minimum Shift Keying (MSK) modulation (Electronics-notes, 2021). MSK is a widely used modulation method in digital communication systems. In both of these two modulation techniques, the point where the carrier signal value is zero is the point at which carrier frequency changes. Therefore, there is not any phase discontinuities in the MSK or GMSK modulated signals. Data rate of message signal is the factor which is used to determine the difference of logic one and logic zero frequencies. This frequency difference must be equal to half the data rate of message signal to prevent the phase discontinuities. Figure 3.8 shows a message data and its MSK modulated form. Figure shows that frequency of modulated signal changes only at points where the value of carrier signal is zero.

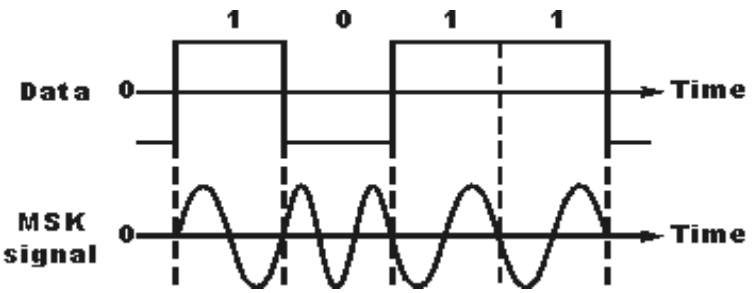

Figure 3.8. MSK modulation

A Gaussian filter is applied to the message signal before the modulation step so as not to let the frequency spectrum of the modulated signal have elements beyond data rate. By applying this filter, GMSK modulated signal is derived from MSK signal. This is the reason why GMSK is better than MSK modulation. Figure 3.9 shows the obtained spectral densities of both MSK and GMSK signals.

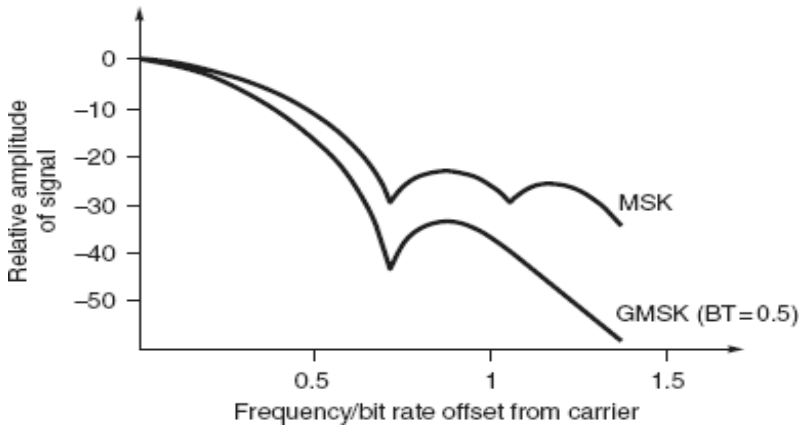

Figure 3.9. Power spectral densities

In general, GMSK modulated signals can be obtained in two ways. First way is to use a Gaussian filter for modulating signal filtering, and then applying it to a frequency modulator having a modulation index of 0.5. A VCO (Voltage Controlled Oscillator) block is used for this aim. This method is not used widely owing to the difficulty of holding the modulation index at a constant value. Figure 3.10 shows how modulation is done with this method.

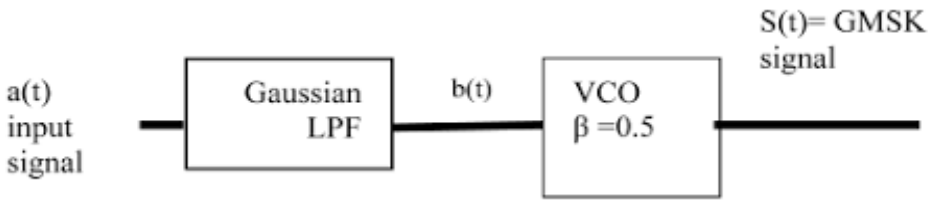

Figure 3.10. GMSK modulator using VCO

Second way of realizing GMSK modulation is to use a quadrature modulator. The reason why the quadrature term is used is that the angle between phases of two signals is 90 degrees. These two signals are known as in-phase and in-quadrature. Due to these two signals, this modulator is known as I-Q modulator. Modulation index is held constant at 0.5 in this modulator type without any extra effort. This is the reason why this method is better than the first way to obtain GMSK modulated signal. Figure 3.11 shows the schematic to obtain a GMSK modulated signal using the second way.

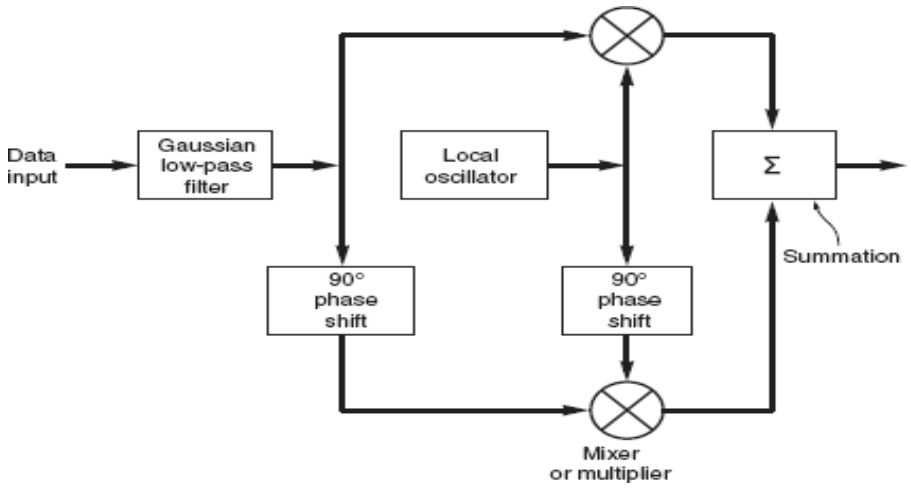

Figure 3.11. GMSK modulator using I-Q modulator

GMSK modulation provides some advantages. Firstly, even if a nonlinear amplifier is used to amplify the GMSK modulated signal, it is not distorted. Because GMSK modulated signal does not carry any information depending on its amplitude changes. This shows also that GMSK modulated signals are more robust to any noise than any other modulation techniques.

3.5. USRP

Personal computers can be used as wide bandwidth software radios by using USRP hardware which are developed by Ettus Research (Ettus Research, 2021). USRPs are developed to support the GNU Radio, and they connect personal computers to the world of radio frequency signals. Different radio frequency bands can be used thanks to the various daughterboards of it. USRP can be used as a transmitter or a receiver in a communication system. It can also transmit and receive signals simultaneously in real-time. It has 4 daughterboards, 2 of which are used as transmitters, and the other 2 are used as receivers. All of these daughterboards are on the motherboard of the USRP (Tucker and Tagliarini, 2009). Figure 3.12 shows the basic structure of a USRP motherboard.

Figure 3.12. Structure of a USRP motherboard

Motherboard of USRP contains 4 Analog to Digital Converters (ADCs) with a precision of 12 bits, and they can obtain 64 MS (Mega Sample) per second. It also contains 4 Digital to Analog Converters (DACs) with a precision of 14 bits, and they can obtain 128 MS per second. Digital data is converted into analog form for transmission by DACs. CPU undertakes all the processing operations done on the USRPs. USRPs contain Field

Programmable Gate Array (FPGA) where all the general purpose operations requiring high speed are realized. Four Digital Up Converters (DUCs) and four Digital Down Converters (DDCs) are provided by the FPGA to obtain the required frequency value from the baseband frequency (Dhar et al., 2006). Figure 3.13 shows the structure of digital up converters.

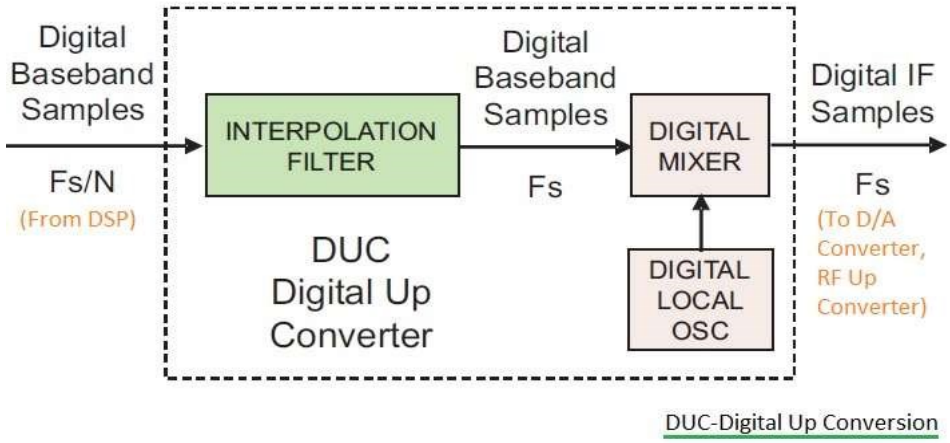

Figure 3.13. Digital up converter

The aim of this converter is to convert the digital baseband samples to digital Intermediate Frequency (IF) samples before the data is sent to the DAC and Radio Frequency (RF) up converter. Figure 3.14 shows the structure of digital down converters provided by FPGAs.

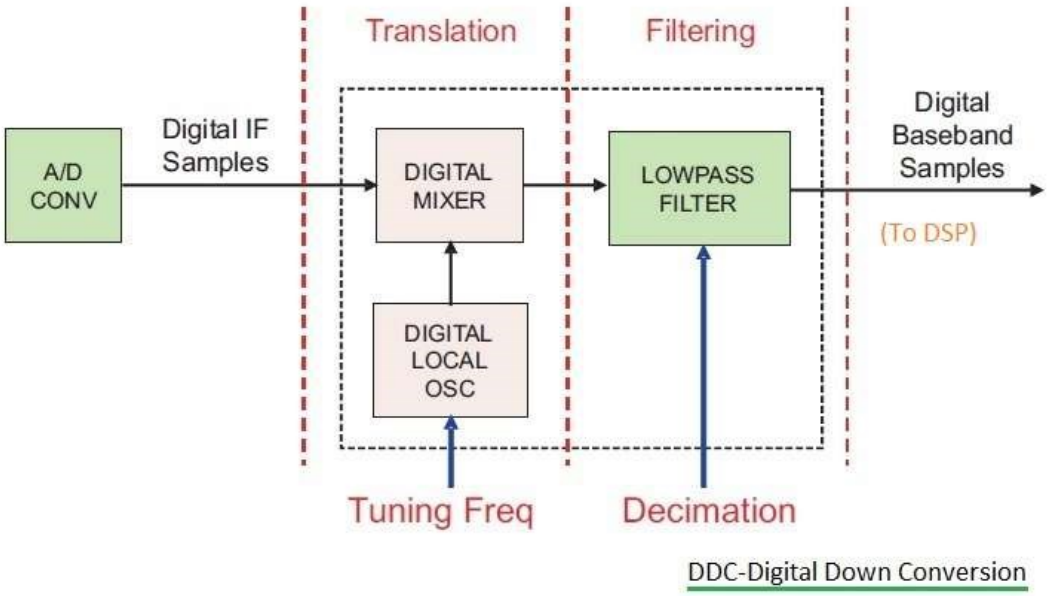

Figure 3.14. Digital down converter

Aim of these digital down converters is to obtain digital baseband samples from digital IF samples. In the decimation process seen in figure, a sample is taken from the coming N samples. In this way, data rate is decreased, and also memory usage is lowered (Rfwireless-world, 2012).

Universal Serial Bus 2 (USB2) or Gigabit Ethernet port is required to connect the USRPs to personal computers, and a 5 VDC adaptor is used to power them. Transfer speed of USRPs is limited with the maximum rate of USB2 (60 MB/sec) when USB connection is used. Multiple Input Multiple Output (MIMO) techniques are used for multi-radio cooperation. Figure 3.15 shows the structure of a USRP device.

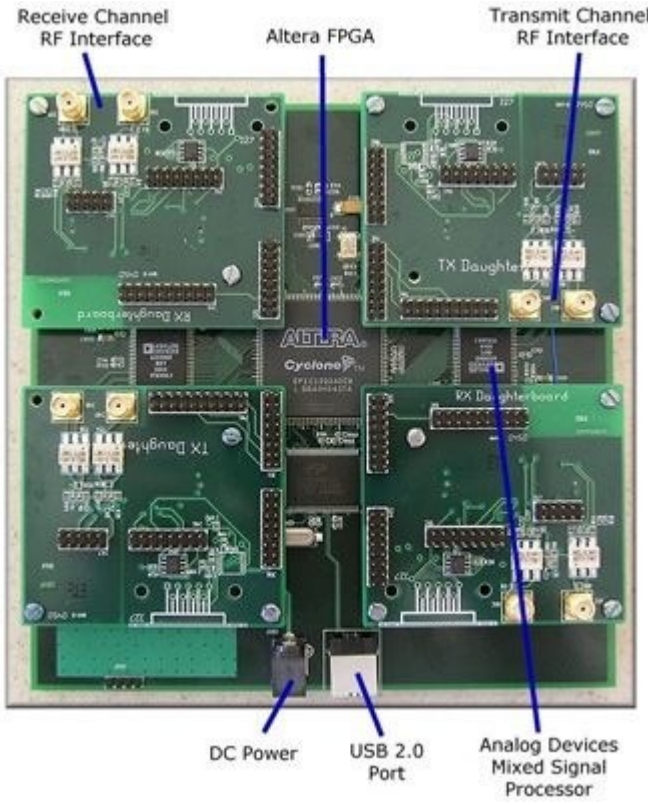

Figure 3.15. Structure of a USRP device

When Figure 3.15 is examined, two TX and two RX daughterboards are seen. In the middle of Figure 3.15, an Altera FPGA is seen. At the bottom of the figure, a DC Power input and a USB 2.0 connection port are seen.

There are some USRP versions having similar architectures but differing in the selection of daughterboards. USRP B210 model is used in both transmitter and receiver sides in this thesis. Figure 3.16 and Figure 3.17 show front and back sides of this device respectively.

Figure 3.16. Front side of USRP B210 device

There are two important inputs seen in Figure 3.16. USB input is used to connect this USRP device to a computer for data transmission purposes. Along with, PWR input is used to connect this device to a power supply by using a power adaptor.

Figure 3.17. Back side of USRP B210 device

Figure 3.17 shows that the back side of a USRP device is used for some antenna connections. In this study, TX/RX input of RF A is used to connect the transmitter side antenna, and RX2 input of RF A is used to connect the receiver side antenna. USRP B210 has some specifications. These are:

- USRP B210 has an operational frequency range of 70 MHz to 6 GHz.
- The range of signals which can be used with USRP B210 is too wide.
- USRP B210 is a single board USRP.
- USRP B210 can reach a real-time RF bandwidth of 56 MHz thanks to its AD9361 transceiver.
- USRP B210 has a sample rate of 64 Ms/s.
- The FPGA model of USRP B210 is Spartan6 XC6SLX150 which is a more than one time programmable FPGA. It provides onboard signal processing and control of the AD9361 mentioned above.
- Data flow of USRP B210 is provided by a USB 3.0 connection.
- USRP Hardware Driver (UHD) software is fully supported by USRP B210 to develop software defined radios using GNU Radio. Thanks to this support, developers can use the designs created before, and develop these designs to integrate to their own studies.
- USRP B210 has MIMO capability.
- USRP B210 has two transmit and two receive channels.
- USRP B210 can be used with GNU Radio and OpenBTS to develop some applications (Ettus Research, 2021).

3.6. GNU Radio

Software defined radios are used nowadays due to some reasons such as having enough processing power, reducing cost, and allowing upgrades in software (Vilches and Dujovne, 2014). Software radios are realized by GNU Radio which is a free toolkit for

software development applications (GNU Radio, 2021). GNU Radio can be used to simulate a radio frequency application, or it can be used with an RF hardware. GNU Radio is a good selection to create Graphical User Interface, and this interface is known as GNU Radio Companion.

3.6.1. GNU radio companion

GNU Radio Companion is used to create software implemented radios instead of hardware implemented radios. Therefore, some operations which are normally done in dedicated circuits like mixers, filters, amplifiers, modulators/demodulators and detectors are implemented in software by using GNU Radio Companion. Thus, some problems encountered in hardware before the usage of software radios can be solved by easily changing the software, and these changes can be realized on-the-fly (Leferman et al., 2010). Some improved versions of GNU Radio provide advanced efficiency, more library elements, and better hardware support. Figure 3.18 shows an empty opening of GNU Radio Companion.

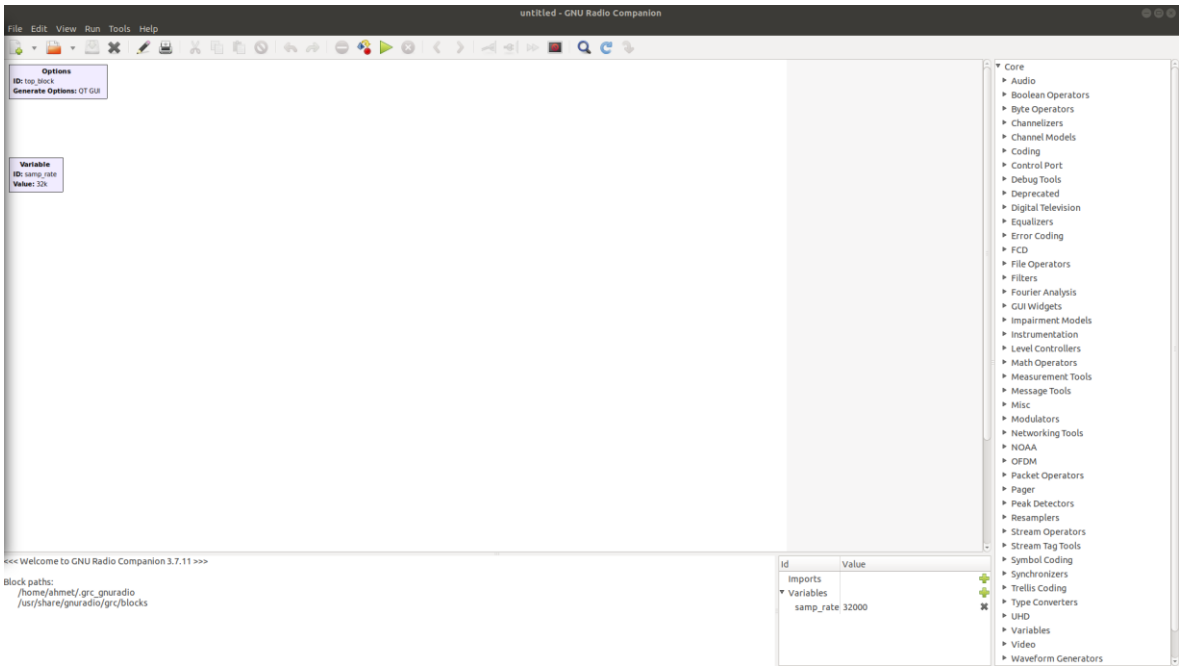

Figure 3.18. GNU Radio Companion first opening

When Figure 3.18 is examined, only Options and Variable blocks come at the first opening of the program. The area having white background and containing these two blocks is known as the working area. Moreover, at the right side of Figure 3.18, some categories are seen. Each of these categories contains some blocks, and these blocks can be seen by clicking to the triangles seen on the left of each category name.

GNU Radio Companion has some blocks and these blocks are connected to each other to create the flowgraphs of software defined radios. They perform all the signal processing operations such as filtering, modulating, demodulating, encoding and decoding on their input signal. Block outputs are used to obtain the processed signals. Block input and output data types may be bits, bytes, vectors and more complex data types, and they can be easily changed in the block properties. GNU Radio offers us to create a block that is not contained in its library and append it to our flow graph. GNU Radio Companion is used to form the flow graph of any operation by using these blocks, and it is not necessary to write the python code of formed flow graph. The only thing that is done is dragging the necessary blocks from the library and dropping them into the flow graph of our software defined radio. Each block in GNU Radio Companion can be used by entering its necessary parameters.

All the operations done when the program is run are shown in the below-left part of Figure 3.18. After all the necessary blocks are added, and parameters of them are set, flowgraph is run by firstly doing Run-Generate from the menu which is seen in the upper part of Figure 3.18. After that, Run-Execute command is done from the same menu to execute the flow graph. The aim of Run-Generate is to create a python file for the flowgraph of our work. This must be done before the flowgraph is executed, and this operation is done even if a python file is created before. In this way, the changes implemented in flowgraph are saved to the previously created python file. This python file is formed in the same folder with GNU Radio Companion file, and it shows the flow graph of the model and how python calls C++ classes through SWIG (Simplified Wrapper and Interface Generator) interface (Vachhani and Mallari, 2015).

3.6.2. GNU Radio softwares

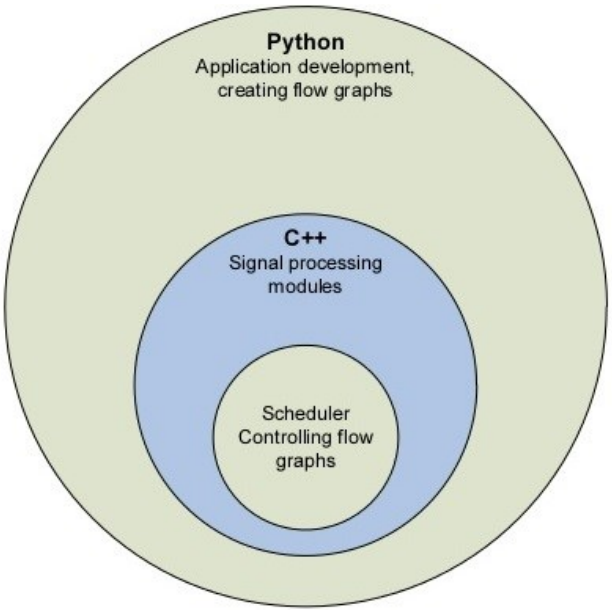

Figure 3.19. GNU Radio software relations

The relations among softwares used in the process of running of a GNU Radio Companion flowgraph are shown in Figure 3.19. Python and C++ programming languages are used together in GNU Radio. Python is responsible for creating the flow graphs from source to sink blocks, and C++ is used for all the performance critical signal processing operations by some predefined block classes. SWIG interface is used to create an interface between these two languages. When a software radio is run by python, corresponding objects and functions are called through the SWIG interface from C++ libraries.

GNU Radio can be used in Linux and Windows operating systems. But, it is better-suited for Linux operating system. It can be installed into the Linux system by typing the command given in (GNU Radio, 2021) to the command line of the Linux.

4. RESULTS AND DISCUSSION

Real-time implementation of video streaming task consists of some transmitter side and receiver side operations in this study. Transmitter side operations are realized by an NVIDIA Jetson Nano device. It has 128-core Maxwell GPU, Quad-core ARM A57 1.43 GHz CPU, 4 GB 64-bit LPDDR4 25.6 GB/s Memory. On the other hand, receiver side operations are realized by a laptop. It has a Third Generation Intel Core i5-3210M 2.50 GHz Processor, 8 GB DDR3 Ram, and an NVIDIA GeForce GT630 2 GB Graphics Card. This laptop device is also used to implement the simulation part of this study. All the transmitter, receiver and simulation part operations of real-time video streaming are explained in details below.

4.1. Transmitter Side of Real-Time Video Streaming

Transmitter side does the necessary operations for the transmission of real-time video taken from an external camera to the receiver side. There are four steps to transmit the video: real-time video pipelining, USRP device connection to the Jetson Nano, executing the transmitter GNU Radio flowgraph and doing transmitter side USRP operations. All of these steps are explained in details below.

4.1.1. Real-time video pipelining

The first step in the transmitter side is to pipeline the external camera video to GNU Radio. Gstreamer or VLC can be used to do real-time video pipelining. These two softwares are compared below with respect to the transmitted video latency values.

4.1.1.1. Using Gstreamer

Gstreamer is a video streaming software, and it is installed using the command given below;

- *sudo apt-get install libgstreamer1.0-0 gstreamer1.0-plugins-base gstreamer1.0-plugins-good gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly gstreamer1.0-libav gstreamer1.0-doc gstreamer1.0-tools gstreamer1.0-x gstreamer1.0-alsa gstreamer1.0-gl gstreamer1.0-gtk3 gstreamer1.0-qt5 gstreamer1.0-pulseaudio*

This command installs Gstreamer and its all necessary dependancies. After this step, pipeline files are created by using the linux command 'mkfifo'. One pipeline file is created for the transmitter side, and one pipeline file is created for the receiver side. Video frames taken from the camera are transferred to the file source block of GNU Radio through the pipeline created at the transmitter side. Transmitter side pipeline file is formed by the command;

- *mkfifo tx.ts*

This command shows that the name of this pipe file is "tx", and its extension is "ts". Files having .ts extension are known as container files, and these files generally contain video streams.

After creating these streaming files, the code given below is run to take real-time raw video from the camera, scale it, convert it to a specified video format and lastly sink it to transmitter side pipeline tx.ts file.

- *gst-launch-1.0 -v v4l2src device=/dev/video0 ! videoscale ! video/x-raw ,width=800, height=600, framerate=15/1 ! videoconvert ! x264enc ! mpegtsmux ! filesink location=tx.ts*

In this code, “gst-launch-1.0” command is used to build and run Gstreamer pipelines. A pipeline has a number of elements, and these elements are separated by using exclamation marks. Some of the properties of elements are added to code in “property = value” format. The command “-v” is used to obtain verbose information about our streaming. Capturing video from the external UAV camera is done with the command “v4l2src device=/dev/video0”. If there are more than one capturing devices connected to Jetson Nano device, then /video1, /video2 commands can be used to activate different camera devices according to the number assigned to a camera by Jetson Nano device. Linux command line command “ls /dev/video*” is used to see the possible assigned number for a connected camera. This command shows all the video capturing devices connected to the Jetson Nano device. The following element in the above code is “videoscale”. This command scales the video frames. Moreover, the commands coming after videoscale is used to obtain a video having 800×600 pixels. Frame rate of video is set to 15 frames per second by the frame rate parameter. Frame rate value can differ for each selected camera. This parameter is very important. If it is selected as higher or lower than the default frame rate of the camera, then the video is displayed as faster or slower than the normal speed of it. Frame rate of any selected camera can be learned by recording a video taken from that camera using below command:

- *gst-launch-1.0 -v v4l2src device=/dev/video0 ! videoconvert ! x264enc ! avimux ! filesink location=test1.avi*

This code has some elements. The element “videoconvert” is used to change the captured video format. “x264enc” element is used to encode the captured raw video in H.264 video format. ITU H.264 High 4:2:2 Profile is the default format for this codec. An MPEG transport stream is created by using “mpegtsmux” element. This element shows

that H.264 encoded video is muxed in a MPEG-TS container in this code. Lastly, “filesink” element is used to write the resultant data to “test1.avi” file.

Properties of the recorded video file shows the frame rate value of the used camera as can be seen in Figure 4.1. In addition to that, video dimensions, video codec, and video bitrate can also be seen in this figure.

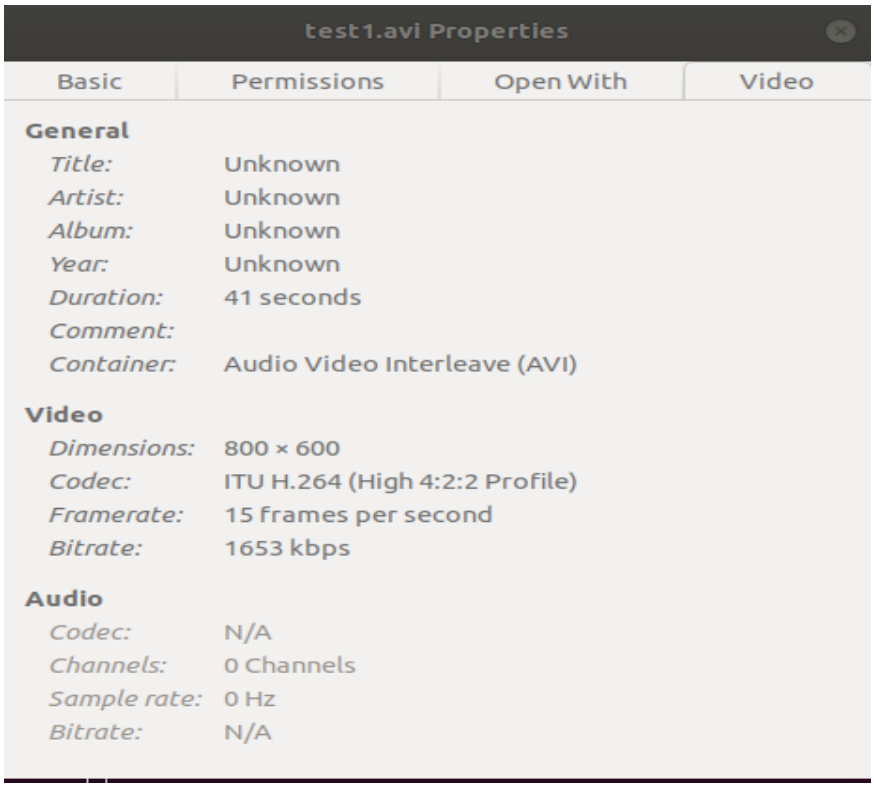

Figure 4.1. Learning camera properties

4.1.1.2. Using VLC

VLC is used to create a pipeline for the real-time video received from an external camera. VLC is installed by using the app store of Linux operating system. After installing VLC, the instructions given in the Gstreamer pipelining part are followed to create .ts files for streaming of data. After creating the .ts files, the code given below is used to capture real-time raw video, scale it, convert it to a specified video format and lastly sink it to the transmitter side pipeline file.

- *vlc v4l2:// :v4l-vdev="/dev/video0" :v4l-norm=3 :v4l-frequency=-1 :v4l-caching=300 :v4l-chroma="" :v4l-fps=-1.000000 :v4l-samplerate=44100 :v4l-channel=0 :v4l-tuner=-1 :v4l-width=640 :v4l-height=480 :v4l-brightness=-1 :v4l-colour=-1 :v4l-hue=-1 :v4l-contrast=-1 :no-v4l-mjpeg :v4l-decimation=1 :v4l-quality=100 --sout "#transcode{vcodec=x264,vb=800,scale=1,ab=128,channels=2,samplerate=44100}:duplicate{dst=std{access=file,mux=ts,dst=/home/ahmet/tx.ts}}"*

In this code, "/dev/video0" is used as the source of video. Video captured from the camera is scaled to 640×480 pixel size. Moreover, "vcodec=x264" is used to encode the obtained video in H.264 standard. Lastly, video is pipelined to the "tx.ts" file.

4.1.2. USRP device connection

USRP device connection with the transmitter side device can be done by using USB or Ethernet interfaces. In this study, USB interface is selected, and this connection is done by using a USB 2.0 cable. After this connection, first step is to download the USRP images to the device by using the command line command;

- *usrp_images_downloader*

After download operation, the following command is used to control if connected USRP device is detected by the Jetson Nano or not;

- *uhd_find_devices*

If the USRP device is detected by Jetson Nano, the output of this command is some information about the connected USRP such as device name and serial number. After the

connection is established, transmitter flowgraph is created to prepare the compressed video for transmission.

4.1.3. Creating transmitter flowgraph using GNU Radio

Figure 4.2 shows the transmitter side flowgraph. It is created by dragging and dropping the necessary blocks to the working area from the right side menu seen in the figure. After creating this flowgraph, it is executed and run. Some explanations about these blocks are given in details below.

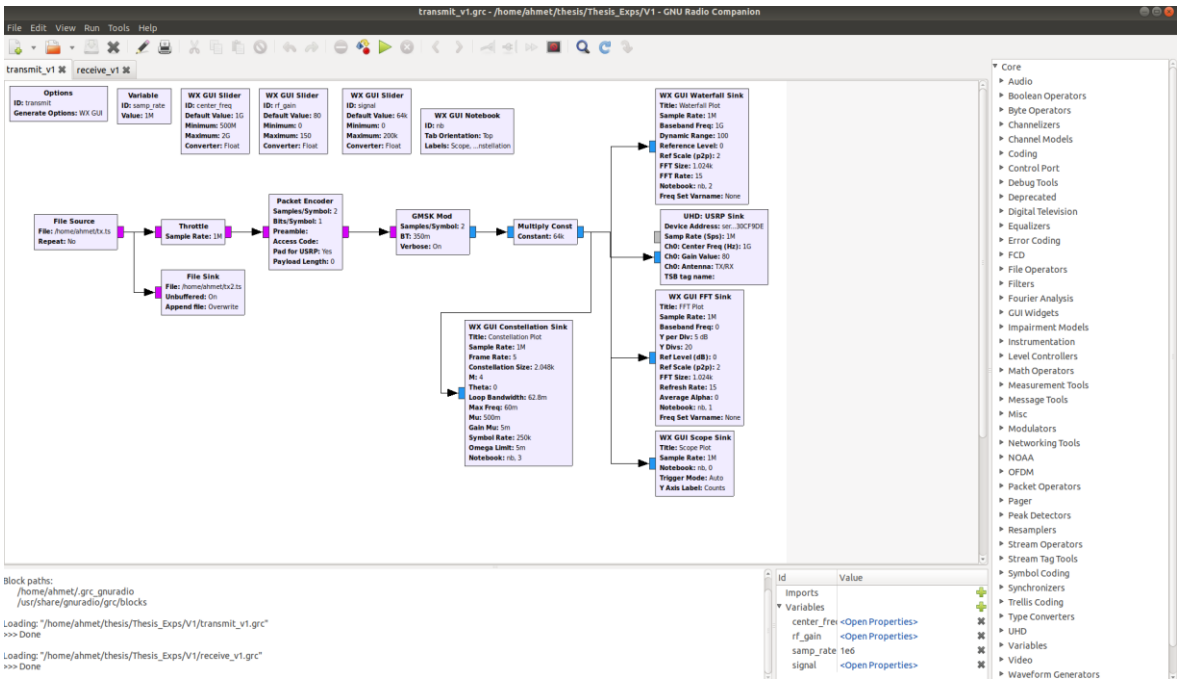

Figure 4.2. Transmitter flowgraph

"Options" block is used to enter some information about the created flowgraph. Every flowgraph file contains this block. It has some parameters. ID parameter is for the aim of identification, and it is also the name of the generated python code file for a flowgraph. Figure 4.2 suggests that the generated python code file for transmitter side of this study is named as "transmit.py". "Generate Options" parameter determines the created code type. On the transmitter side, WX GUI is selected for the Generate Options

parameter. WX GUI blocks are used for the goals of showing some output graphs such as Constellation, Waterfall, FFT, Scope and creating some variables and notebooks.

"File Source" block is used to read any specified file in binary format. This block has some parameters. "File" parameter is used to specify the file path of the file that will be read. "Repeat" parameter determines whether the transmission operation will be repeated when the end of the transmitted file is reached. In this study, since a constant source file is not transmitted, the Repeat parameter is selected as "No". This block is used to read the bit stream sent by Gstreamer in this thesis study.

"Throttle" block is used to limit the rate of input data flow. It is connected to the File Source block. It throttles the samples which are created by File Source block. When this block is not connected to the source block, GNU Radio Companion gives a warning like "UUUUUU…" that is showing the flowgraph underruns. Throttle has only one parameter named as "Sample Rate", and this value is set as the same with the sample rate used in the flowgraph.

"File Sink" block is used to write the stream which is coming from the File Source block to a file in binary format. The output of this block can be used by another File Source block. The file created by File Sink block contains only binary data. This block has some parameters. "File" parameter takes the file path that the video stream will be saved. The file path of "tx2.ts" file in Jetson Nano must be given as input to this parameter. "Unbuffered" parameter determines if the memory buffering of coming data is used or not. If this parameter is selected as "On", the flowgraph may work slower because of the required time to reach the disk. Since this study does not use memory buffering, this parameter is selected as "On". "Append file" parameter is used to determine if a new file is created in every run of flowgraph or not. This parameter is selected as "Overwrite" to create a new file for each run of the transmitter flowgraph. The aim of using this block is to see if the taken bit stream to the GNU Radio has some streaming errors or not.

"Packet Encoder" block creates some data packets by using the bit stream coming from the Throttle block to transmit the video data more safely. These packets contain a header, a preamble, and an access code. Access code parameter can be used to increase the

transmitted data security. “Samples/Symbol” parameter of this block shows the number of samples in each symbol, and it is set to 2 in this study. “Bits/Symbol” parameter determines the number of bits in each symbol. Since GMSK modulation which is used in the transmission process in this study encodes data as every symbol has 1 bit, Bits/Symbol parameter of Packet Encoder block is set to 1. “Payload Length” parameter determines the number of bytes that encoder takes, and its value is set to 0. This specifies that the value of this parameter will be determined by GNU Radio Companion according to the coming data. Since a USRP B210 is used in this study, the “Pad for USRP” parameter is selected as “Yes”.

“GMSK Mod” block is used to obtain a GMSK modulated signal. The input of this block is a byte stream which is coming from the packet encoder. But its output is a modulated complex data in baseband. It has some parameters. “Samples/Symbol” parameter determines the number of samples in every baud and its value must be equal to or bigger than 2. “BT” parameter is determined by the multiplication of Gaussian filter bandwidth by symbol time value. In this study, BT value is set to 350×10^{-3}. Verbose parameter is selected as “On” to see the information about modulation process. The aim of modulation process is to decrease the needed bandwidth to transmit the real-time video data in this study. We transmit the complex signals representing one or more bits when we modulate the data. There are some modulation schemes. Higher quality videos can be transmitted by using the same bandwidth with a more effective digital modulation scheme.

“Multiply Const” block multiplies its input data by a constant value. In this study, GMSK modulated signal is multiplied by 64×10^3, and the resultant signal is transferred to USRP device for the transmission. If the input of this block is a vector, then element-wise multiplication is applied. In this case, “Constant” parameter of this block takes a vector having the same size with its input. Some applications use this block as a switch to allow or block the passing of input signals. In this study, this block is used as a signal amplifier.

“UHD: USRP Sink” block is used to transmit the resultant signal to a USRP device. This block has some parameters.

Device Address parameter is used to specify the serial number of the USRP that will be used in the transmission process. If only one USRP is connected to a system, then this parameter can be left blank. When Figure 4.2 is examined, the serial number of transmitter side USRP is entered to this area.

Samp Rate parameter shows the sampling rate of the transmitter USRP. This parameter is set by using the variable "samp_rate" seen in Figure 4.2. Clock Rate value of the USRP device is equal to this sample rate value. Sample rate determines the bandwidth of used USRP device in terms of Hz. Sample rate value is known as the baud rate of USRP device. If the baud rate of used transmitter side USRP is lower than the baud rate of transmitted real-time video, some latency in the received video is seen. Resolution or fps value of transmitted video can be lowered to decrease the latency. Every USRP device has a maximum sample rate value. This parameter must be assigned according to the specifications of used USRP device.

Ch0: Center Freq (Hz) parameter determines the center frequency of transmitted signal. The assignment of this parameter is done by using a "WX GUI Slider" block. As can be seen in Figure 4.2, the default value of center frequency is set to 1 GHz, and it ranges from 500 MHz to 2 GHz. These values are determined according to the frequency value of used antenna. On the transmitter side USRP, a VERT2450 (2.45GHz) antenna is used. This antenna is tested in some different frequencies, and it is seen that this antenna works well in 1 GHz frequency.

Ch0: Gain Value parameter determines the gain value of transmitter USRP. Absolute gain or normalized gain can be selected by changing the "Ch0: Gain Type" parameter of the block. When normalized gain is used, the gain ranges from 0 to 1, where 0 is the minimum gain value, and 1 is the maximum gain value that the USRP can have. When absolute gain is used, gain changes from 0 to maximum gain of USRP device in dB. In this study, absolute gain is used, and its value is set by using a "WX GUI Slider" block which has an ID of "rf_gain". Default gain value is set to 80 dB for the transmitter side USRP.

Ch0: Antenna parameter is used to determine which antenna of USRP will be used for transmission of signal. “TX/RX” antenna is selected on the transmitter side.

“WX GUI Waterfall Sink” block is used to show the signals on a spectrogram plot. This plot shows the power of signals in every frequency value around the defined baseband frequency. Figure 4.3 shows the transmitter side Waterfall Plot obtained in this study.

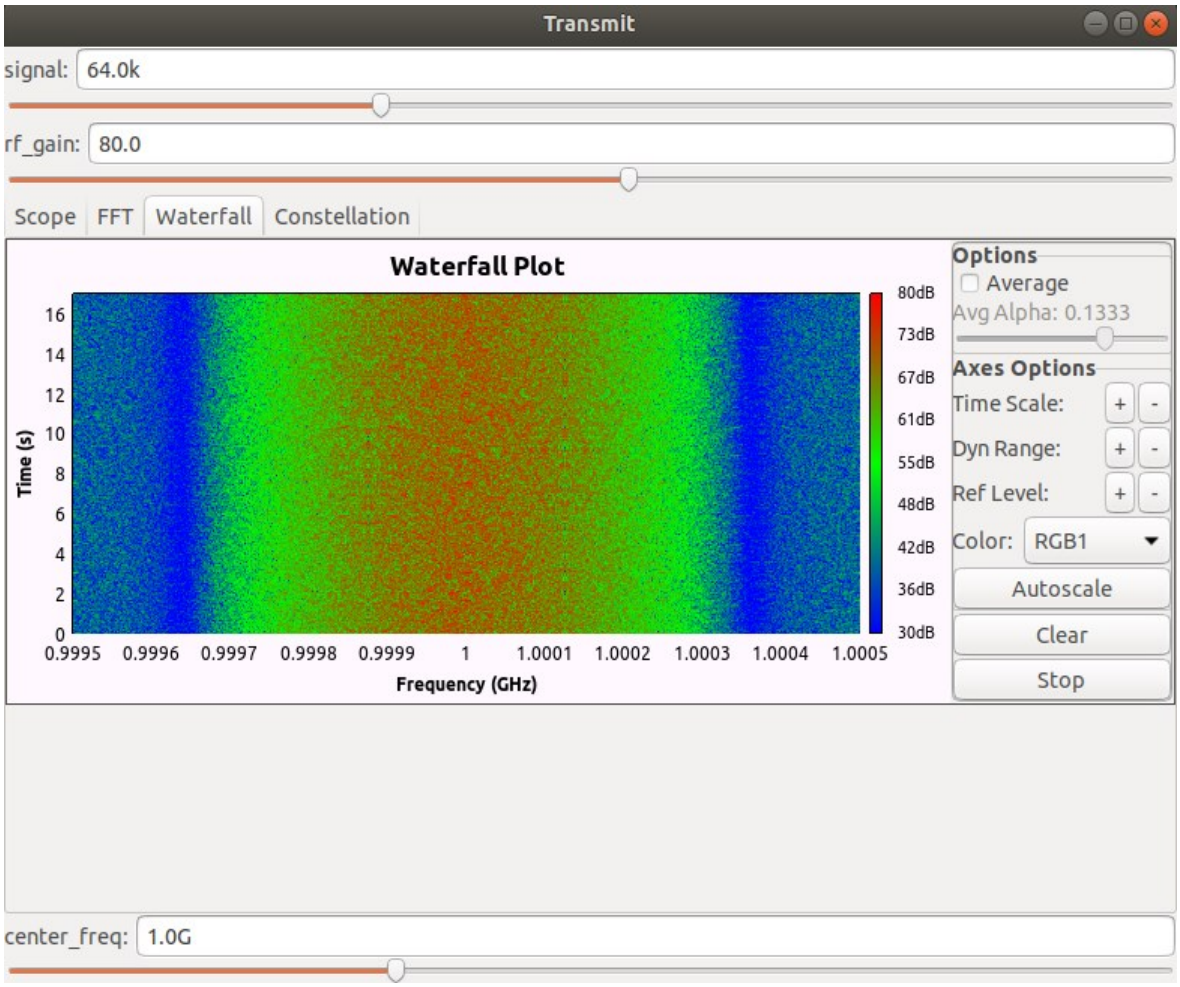

Figure 4.3. Transmitter side Waterfall plot

In this figure, the horizontal axis shows the frequency values of transmitted signal, and the vertical axis shows the time in seconds. Figure shows the power of transmitted signal second by second by using some color codes. In this figure, RGB1 color scheme is used for the determination of colors. Power values seen in figure range from 30 dB to 80 dB. This range is determined by clicking the “Autoscale” button seen in the figure. Since

the center frequency of transmitted signal is set to 1 GHz (baseband frequency in figure), the power of the signal in this frequency is more than the powers seen in other frequencies.

"WX GUI FFT Sink" block is used to get the frequency plot of the transmitted signal. On the transmitter side, the sample rate parameter of this block is set as the same with USRP sample rate. Baseband frequency of this plotting is set to 0 Hz. Figure 4.4 shows the FFT plot of the transmitted signal.

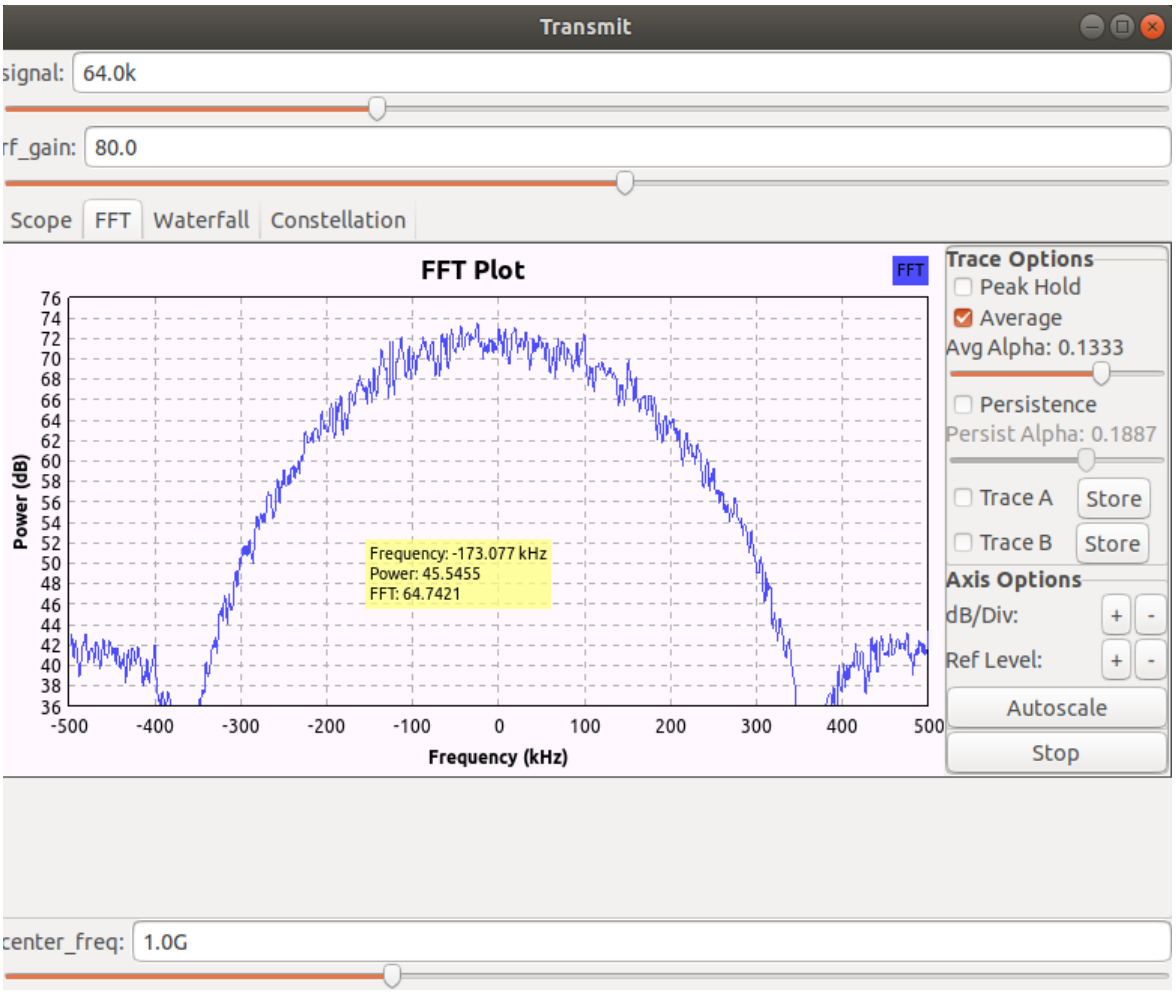

Figure 4.4. Transmitter side FFT plot

This plot shows the average FFT of the transmitted signal. If the FFT of the signal can not be seen at beginning, the Autoscale button is clicked to scale the Power (dB) values seen on the left vertical axis of the plotting. The peak values of the FFT plot are obtained by clicking to the Peak Hold option seen in the figure. Horizontal axis shows the

frequency values, and it ranges from -500 kHz to 500 kHz. This range depends on the bandwidth of our transmitted signal. Vertical axis shows the power values for each frequency. Power of the signal can be changed by changing the “signal” and “rf_gain” parameters seen in the figure.

“WX GUI Constellation Sink” block is used to show Inphase/Quadrature plot of the transmitted signal. These plots give some information about the signal. Signal power is determined by looking at the distance of a point from the origin. Carrier phase shift is determined by looking at the angle between a point and the horizontal axis. Every point in constellation plots represents a symbol used in transmission process. The carrier representing a symbol is created by summing Inphase and Quadrature values of a point in these plots. Inphase component corresponds to cosine, and Quadrature component corresponds to sine component of a carrier signal. This means that each symbol is a complex number. Therefore, the horizontal axis in these figures can be thought of as real axis, and the vertical axis can be thought of as imaginary axis. Figure 4.5 shows the Constellation plot of the transmitted signal in this study.

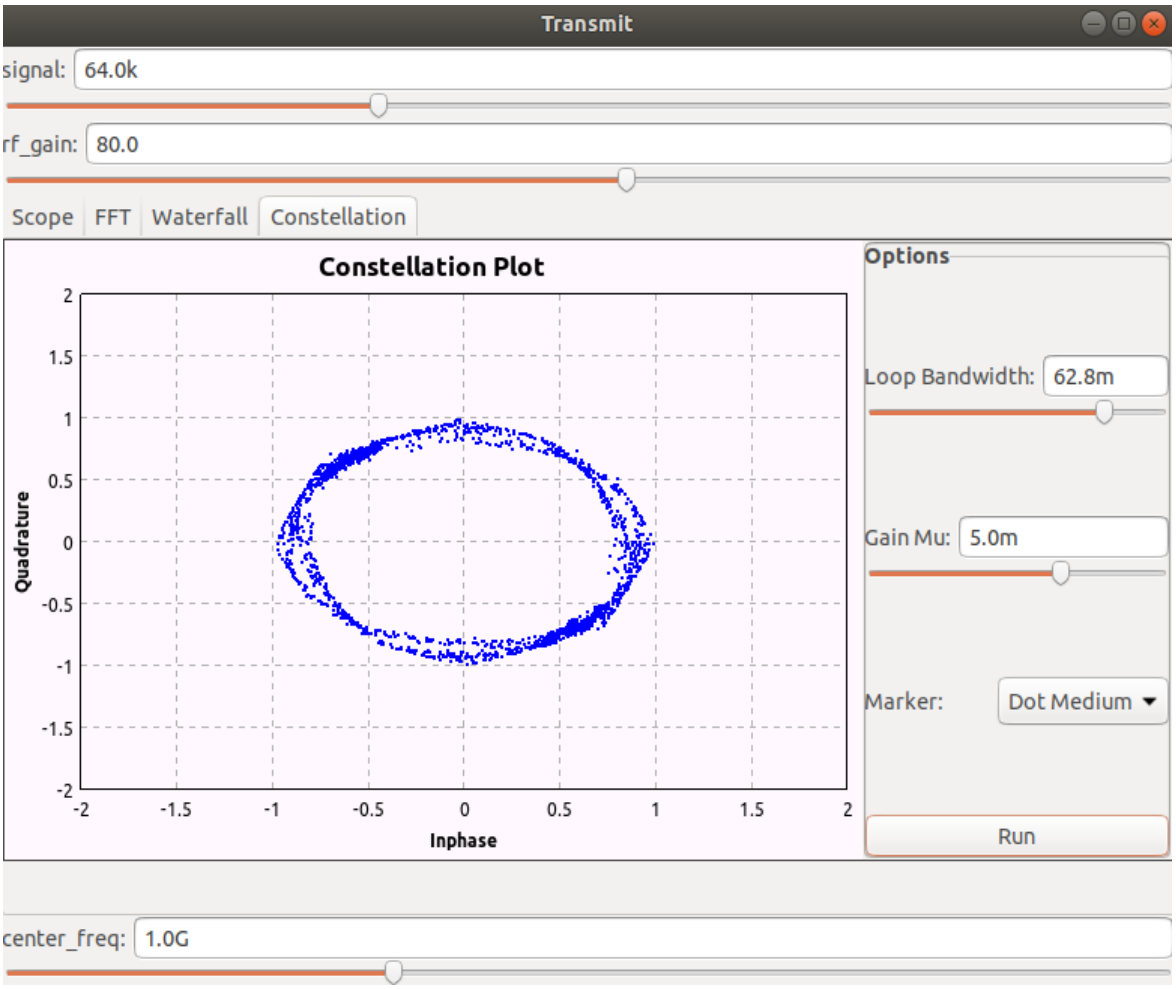

Figure 4.5. Transmitter side Constellation plot

"WX GUI Scope Sink" block is used to see the oscilloscope plotting of the transmitted complex signal. Vertical axis of this plot shows the counts value, and horizontal axis of it shows the time. Figure 4.6 shows the Scope Plot of transmitted signal.

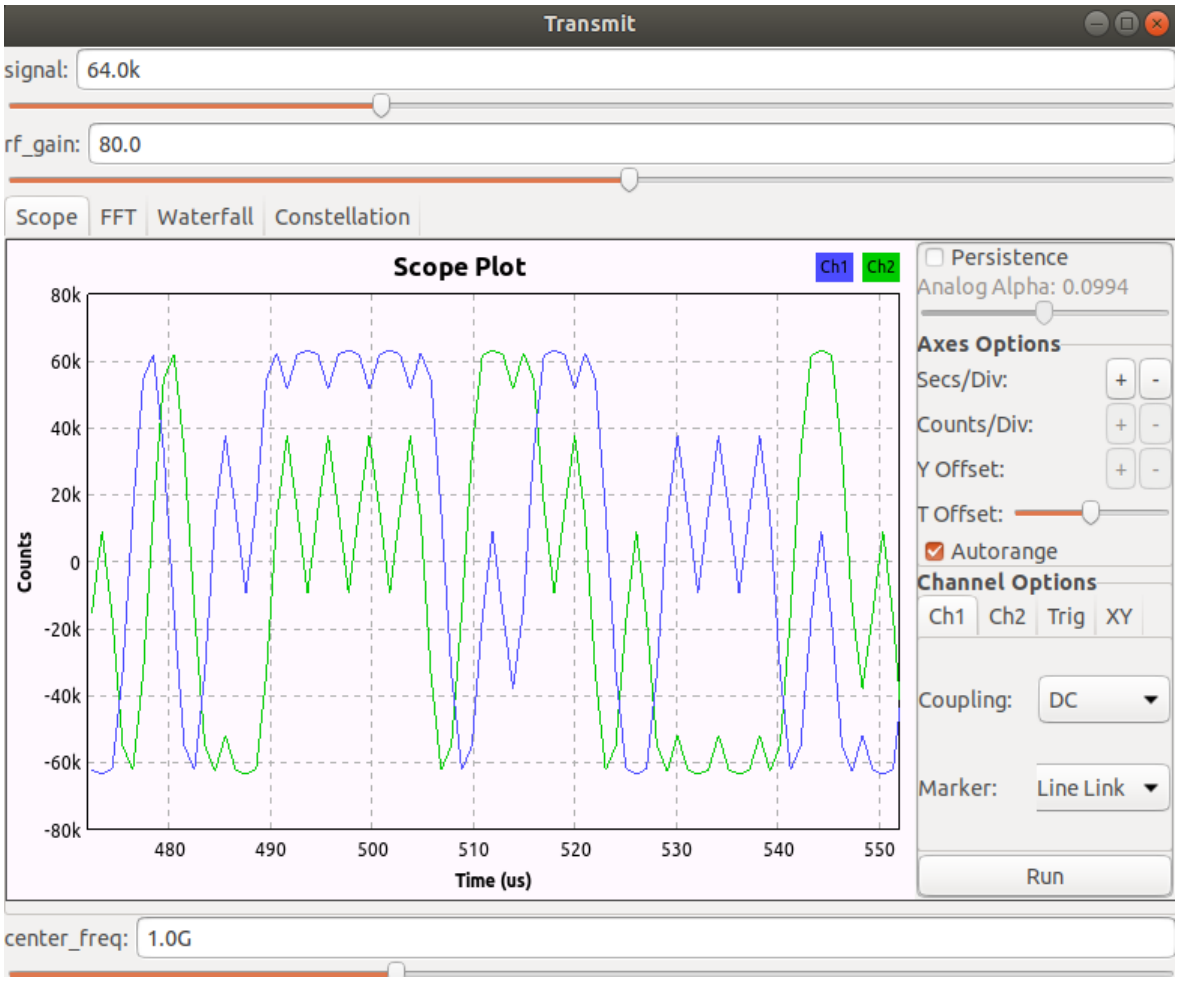

Figure 4.6. Transmitter side Scope plot

The signal seen in the figure can be narrowed or expanded in horizontal axis by changing the “Secs/Div” parameter. In the same manner, the signal can be narrowed or expanded in vertical axis by changing the “Counts/Div” parameter.

“WX GUI Notebook” block is used to combine all the obtained output figures in one window. It has some parameters. “ID” parameter determines the name of this block. “Tab orientation” parameter determines the place of the tabs in output figure, and it is selected as “top” in this study. “Labels” parameter holds the labels of the tabs. “Notebook” parameter of any output block is used to place it into any notebook block. In this study, Constellation, Waterfall, FFT, and Scope plots are combined in one output window by using “nb” notebook. The placements of these outputs are determined according to the number written next to the notebook ID. Since the notebook parameter of Scope Plot is

entered as “nb,0”, it is the first obtained output in output window. In the same manner constellation plot is the last output.

4.1.4. Transmitter side USRP operations

Signal processing operations applied in the transmitter side USRP device are given at the upper part of Figure 4.7. Jetson Nano device used in the transmitter side sends some digital data having In-phase (I) and Quadrature (Q) components to USRP device. Firstly, this digital data goes to Digital Up Converter (DUC) structure which is on the FPGA of USRP. DUC converts the digital baseband samples to the digital intermediate frequency samples. This operation is known as signal interpolation. After that these digital samples are sent to Digital to Analog Converter (DAC) structure which is in USRP motherboard to obtain some analog signals. Then noise and high frequency components of this analog signal are removed by a low-pass filter seen in the figure. Obtained signal is raised to a predefined RF frequency by using a mixer structure. Lastly, acquired signal is amplified by the transmit amplifier and given to air by an antenna. RF Switch component seen in the figure is used to use the same antenna as both transmitter and receiver.

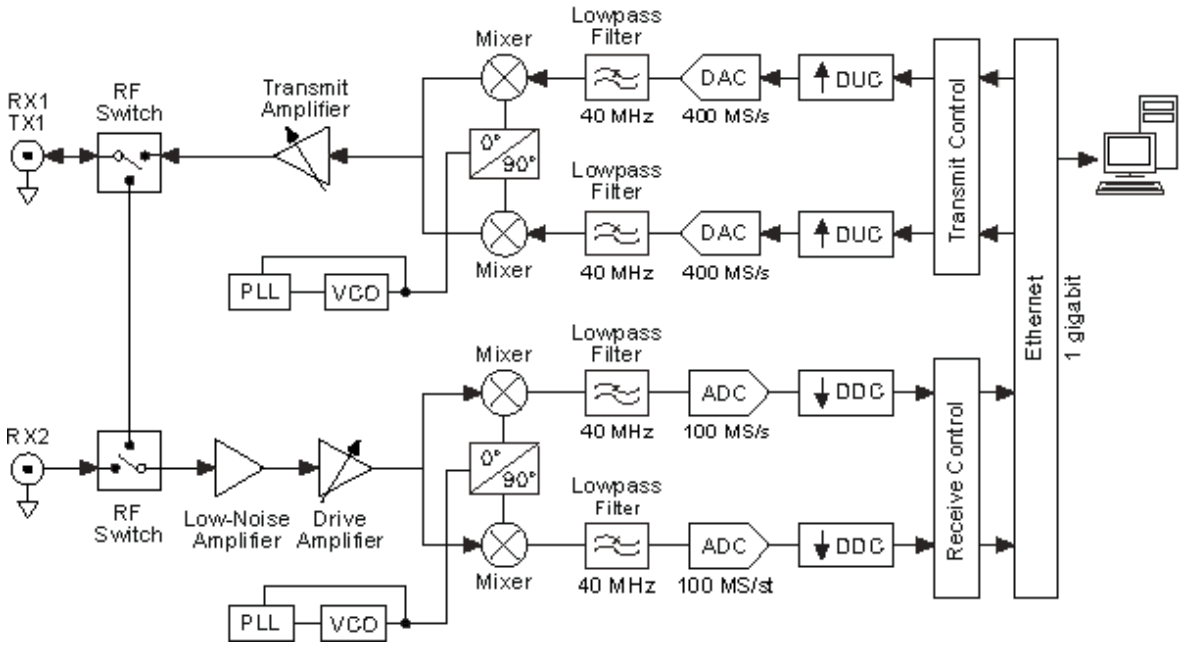

Figure 4.7. USRP system diagram

4.2. Receiver Side of Real-Time Video Streaming

Receiver side does the necessary operations to display the real-time video taken by the receiver side USRP device. A laptop is used to do all the processing operations in receiver side. There are five steps to display the received video. These steps consist of installing necessary softwares, USRP device connection to the laptop, doing receiver side USRP operations, executing the receiver side GNU Radio flowgraph, and lastly displaying the received video. All the steps are explained in details below.

4.2.1. Installing necessary softwares

Laptop device used in the receiver side must be up to date for the new software installations. Therefore, some commands are run in the command line of laptop device. Firstly, each of below linux commands are run separately to update and upgrade the linux operating system of the laptop.

- *sudo apt update*
- *sudo apt upgrade*
- *sudo apt dist-upgrade*
- *sudo reboot*

System reboots and operating system is up to date for new program installations after all the commands given above are run. The next step is to install the GNU Radio software by using the command:

- *sudo apt-get install gnuradio*

This command installs GNU Radio 3.7.11 version. The next software to be installed in receiver side is Mplayer to display the received video frames, and its installation is done by the command given below;

- *sudo apt-get install -y mplayer*

After all these commands are run and softwares are installed, connection between USRP and laptop device is established.

4.2.2. USRP device connection

USRP device connection with laptop must be done by using either USB or Ethernet interface. In this study, USB connection is done by using a USB 2.0 cable. After this connection, USRP images are downloaded to the laptop by using the command line command;

- *usrp_images_downloader*

After that, the following command is run to control if connected USRP device is detected by the laptop or not;

- *uhd_find_devices*

If the USRP device is detected by laptop, the output of this command is some information about the connected USRP such as device name and serial number. Some hardware information about the connected USRP can also be obtained by the command;

- *uhd_usrp_probe*

After the USRP connection is established, receiver side USRP signal processing operations are executed on the received data.

4.2.3. Receiver side USRP operations

Signal processing operations applied in the receiver side USRP device is given at the lower part of Figure 4.7. The first operation seen in the receiver side USRP device is to amplify the analog signal taken by the antenna by using Low-Noise amplifier and Drive Amplifier structures. Phase Locked Loop (PLL) and VCO structures seen in the figure are used to lock the frequency to a predefined value. A mixer structure is used to downconvert the signal to baseband I and Q components. After that a low-pass filter is used to decrease the noise and remove high frequency signal components. This signal is sent to an Analog to Digital Converter (ADC) structure to obtain digital signal from the analog signal. Samp_rate parameter of receiver side USRP block is used in this conversion process. This sample rate value must be selected as higher than or equal to twice of received analog signal bandwidth. Lastly, a Digital Down Converter (DDC) is used for decimation of the signal to a user defined rate. After this last step, obtained data is sent to the receiver side computer by using USB interface to implement the receiver side GNU Radio operations.

4.2.4. Creating receiver side flowgraph using GNU Radio

Receiver side blocks do the required signal processing operations for displaying of real-time video data coming from receiver side USRP. Flowgraph given in Figure 4.8 shows receiver side blocks and their parameters. These blocks are explained in details below.

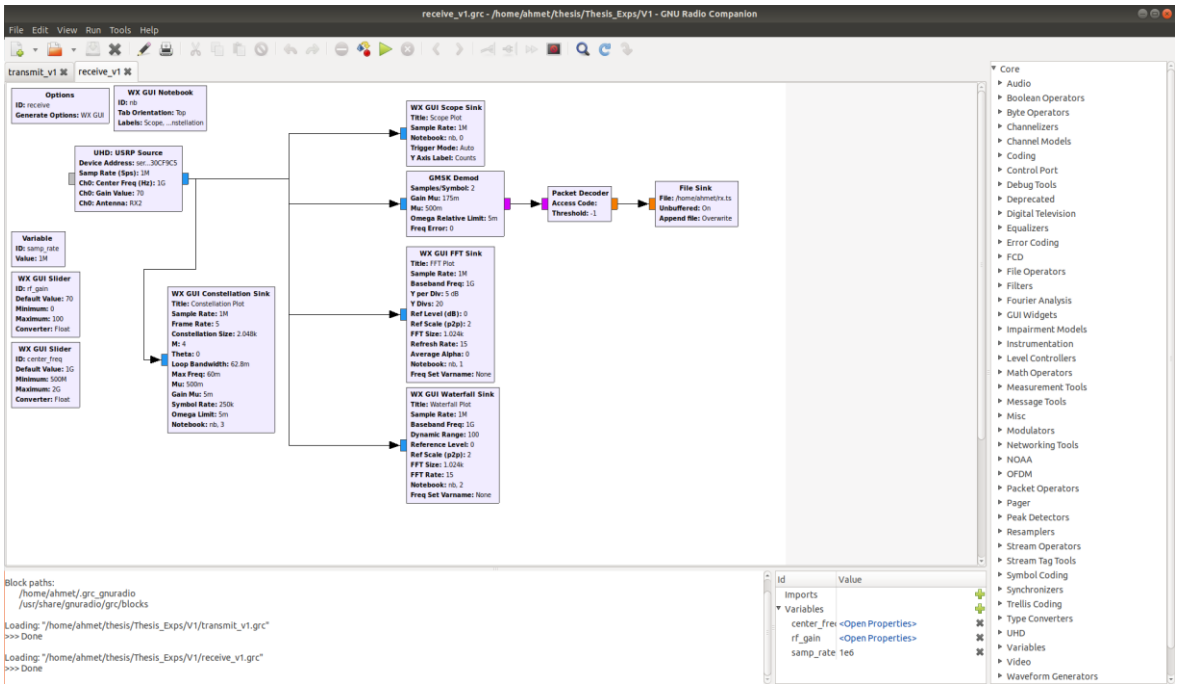

Figure 4.8. Receiver flowgraph

"Options" block is used to enter some information about the flowgraph. Every flowgraph file must contain this block. It has some parameters. "ID" parameter is used for the aim of identification, and its value is the name of the generated python code file for the flowgraph. "Generate Options" parameter determines the created code type for flowgraph. On the receiver side, "WX GUI" is selected as Generate Options. WX GUI blocks are used for the goals of showing some output graphs such as Constellation, Waterfall, FFT, Scope Sinks, and creating some variables and notebooks.

"UHD: USRP Source" block is used to receive the signal coming from USRP device. This block has some parameters.

Device Address parameter takes the serial number of receiver side USRP. If only one USRP is connected to the laptop, then this parameter can be left blank.

Samp Rate parameter shows the sampling rate value of the receiver side USRP. This parameter is set by using the variable "samp_rate" seen in Figure 4.8. Sample rate determines the bandwidth in Hz, and it must be set according to the specifications of used USRP device. This parameter is used in the ADC structure of receiver side USRP.

Ch0: Center Freq (Hz) parameter shows the center frequency of the received signal. The assignment of this parameter is done by using a "WX GUI Slider" block. Figure 4.8 shows that the default value of center frequency is set to 1 GHz, and it ranges from 500 MHz to 2 GHz. These values are determined according to the frequency value of receiver side antenna. A VERT2450 (2.45 GHz) antenna is used on the receiver side USRP in this study.

Ch0: Gain Value parameter holds the gain value of receiver side USRP. Absolute gain or normalized gain can be selected as gain type by changing the "Ch0: Gain Type" parameter. When normalized gain is selected, the gain ranges from 0 to 1, where 0 is the minimum gain value, and 1 is the maximum gain value of receiver side USRP. When absolute gain is selected, gain ranges from 0 to maximum gain of the used USRP device. In this study, absolute gain is selected, and its value is set by using a "WX GUI Slider" block having an ID of "rf_gain". Default gain value is set to 70 dB for receiver side USRP, and this gain value is determined as a result of some tests done by using some different USRP gain values during transmission.

Ch0: Antenna parameter is used to determine which antenna of receiver side USRP will be used, and RX2 is selected as the receiver antenna in this study.

"GMSK Demod" block is used to obtain GMSK demodulated version of the received data. The input of this block is a complex signal, and it is at baseband frequency. However, output of this block is a stream of packed bits. This block has some parameters. "Samples/Symbol" parameter determines the number of samples in every baud and its value must be set as equal to or bigger than 2. "Gain Mu" parameter determines Mu adjustment rate, and its value is set to 175×10^{-3}. Mu is a parameter taking values between 0 and 1, and it is known as Fractional delay. Its value is set to 500×10^{-3}. Omega Relative Limit parameter shows the upper and lower limits of omega, and its value is set to 5×10^{-3}.

"Packet Decoder" block is used to decode the encoded packets coming from the GMSK Demod block. Firstly, payload length is found by reading the header of packets.

Then, payload is extracted by using this header. Lastly, decoder sends extracted payload to the File Sink block.

"File Sink" block writes the stream coming from Packet Decoder block to a file in binary format. The output of this block can be used by a "File Source" block to read the data into GRC flowgraph. This block has some parameters. "File" parameter takes the file path that the video stream will be saved. The path of the "rx.ts" file in the laptop device is given as input to this parameter. "Unbuffered" parameter determines if the memory buffering of coming data is used or not. If this parameter is selected as "On", the flowgraph may work slower because of the time needed to reach the disk. Since memory buffering is not used in this study, this parameter is selected as "On". "Append file" parameter determines if a new file in every run of the flowgraph is created or not. This parameter is selected as "Overwrite" to create a new file in each run of the transmitter flowgraph in this study.

"WX GUI Waterfall Sink" block is used to show its input signals on a spectrogram plot. This plot can be used to see the power of signals in every frequency value around the predefined baseband frequency. Figure 4.9 shows the receiver side Waterfall Plot obtained in this study. Horizontal axis of this figure shows some frequency values of the received signal, and the vertical axis of it shows the time in seconds. Figure also shows the power of received signal with color codes second by second. In this study, RGB1 color scheme is selected for the determination of colors for each power. Power values range from -40 dB to -20 dB. Since the center frequency of received signal is set as 1 GHz (baseband frequency in figure), the power of the signal in this frequency is more than the power at other frequencies. Power distribution changes if the "rf_gain" parameter seen on the top of the figure is changed. When this value is decreased, high power frequency range seen in the figure shrinks.

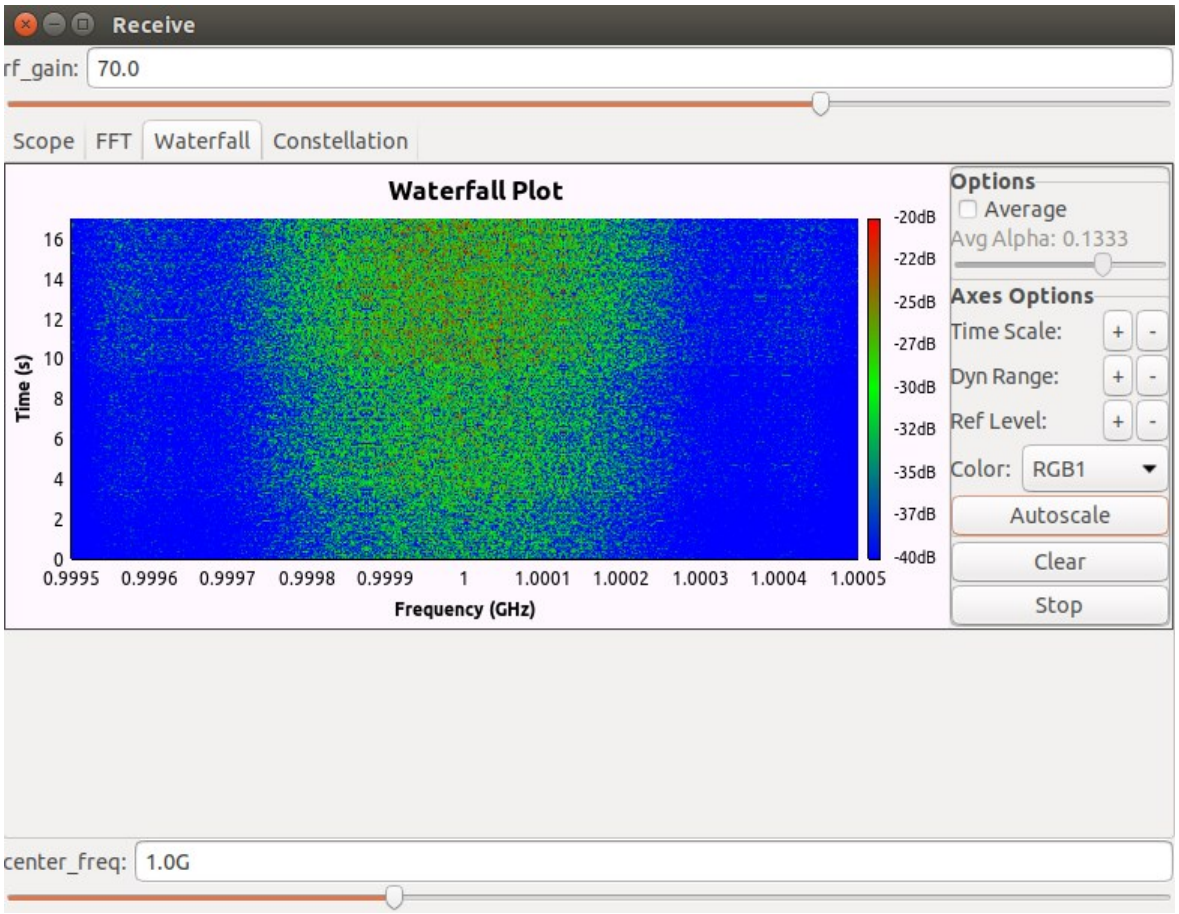

Figure 4.9. Receiver side Waterfall plot

"WX GUI FFT Sink" block is used to display the frequency plot of the received signal. It has some parameters. "Sample Rate" parameter of it is set as the same with the USRP sample rate value. Figure 4.10 shows the receiver side FFT plotting, and baseband frequency of this plotting is set to 1 GHz. This plot shows the average FFT of the received signal. If the FFT of the signal cannot be seen at beginning, the "Autoscale" button is clicked to scale the Power (dB) values seen on the left vertical axis of the plotting. The peak values of the FFT plot is obtained by clicking to the "Peak Hold" seen in the figure. Horizontal axis of this plotting shows the frequency values ranging from 0.9995 GHz to 1.0005 GHz. This range changes according to the bandwidth of our received signal. Vertical axis of it shows the power value for each frequency. Power of the signal can be changed by changing the "rf_gain" parameter seen in the figure.

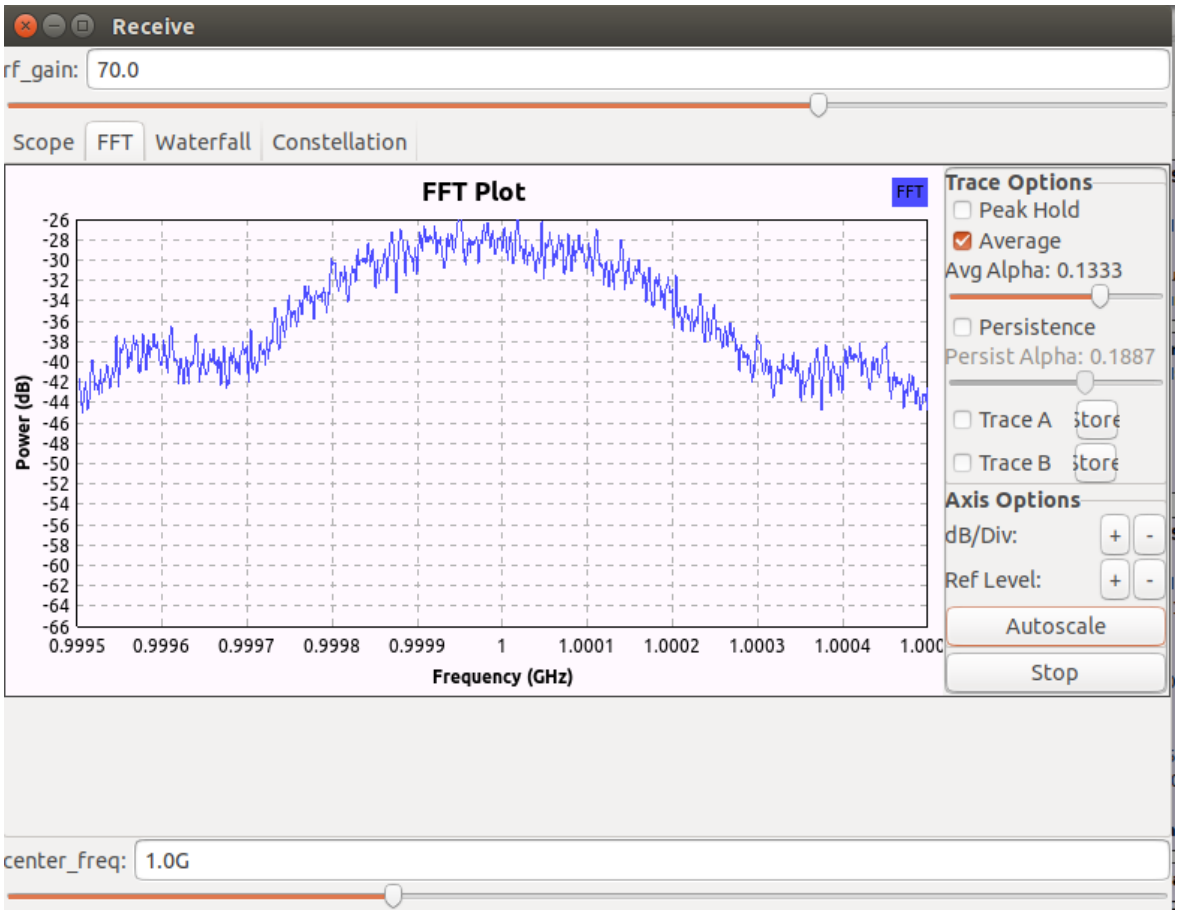

Figure 4.10. Receiver side FFT plot

Maximum FFT power value seen in Figure 4.10 can change according to some communications system parameters. In this study, some tests are done to see the effect of transmitter and receiver side USRP device distances on the maximum FFT power of received signal. Table 4.1 gives the obtained power values, and it shows that FFT power decreases in general once distance is increased. The obtained results come from the behavior of radio waves. Radio waves are near to each other at the output of the antenna and they are further from each other as they go away from the antenna. Therefore, wave power decreases as we are further from the source as a result of energy conservation law. This is known as free space loss. Moreover, some diffraction can be seen when the wave hits an object in the propagation process of it. This may result in decrease in signal power. Passing from some solid like wall or wood may also result in attenuation in signal power.

Table 4.1. Relation between USRP distances and peak receiver FFT power

Distance (meters)	FFT power (dB)
2	-24
3	-36
4	-38
5	-40
6	-38
7	-42

"WX GUI Constellation Sink" block is used to show the Inphase/Quadrature plot of the received signal. These plots give some information about the signal. Signal power is determined by looking at the distance of a point from the origin. Carrier phase shift is determined by looking at the angle between a point and the horizontal axis. Every point in constellation plots represents a symbol used in transmission. Sum of Inphase and Quadrature values of a point creates a carrier in these plots. Inphase component corresponds to cosine, and Quadrature component corresponds to sine component of a carrier signal. This means that each symbol is a complex number. Therefore, the horizontal axis in these figures can be thought of as the real axis, and the vertical axis can be thought of as the imaginary axis. Figure 4.11 shows the Constellation plot of the received signal in this study.

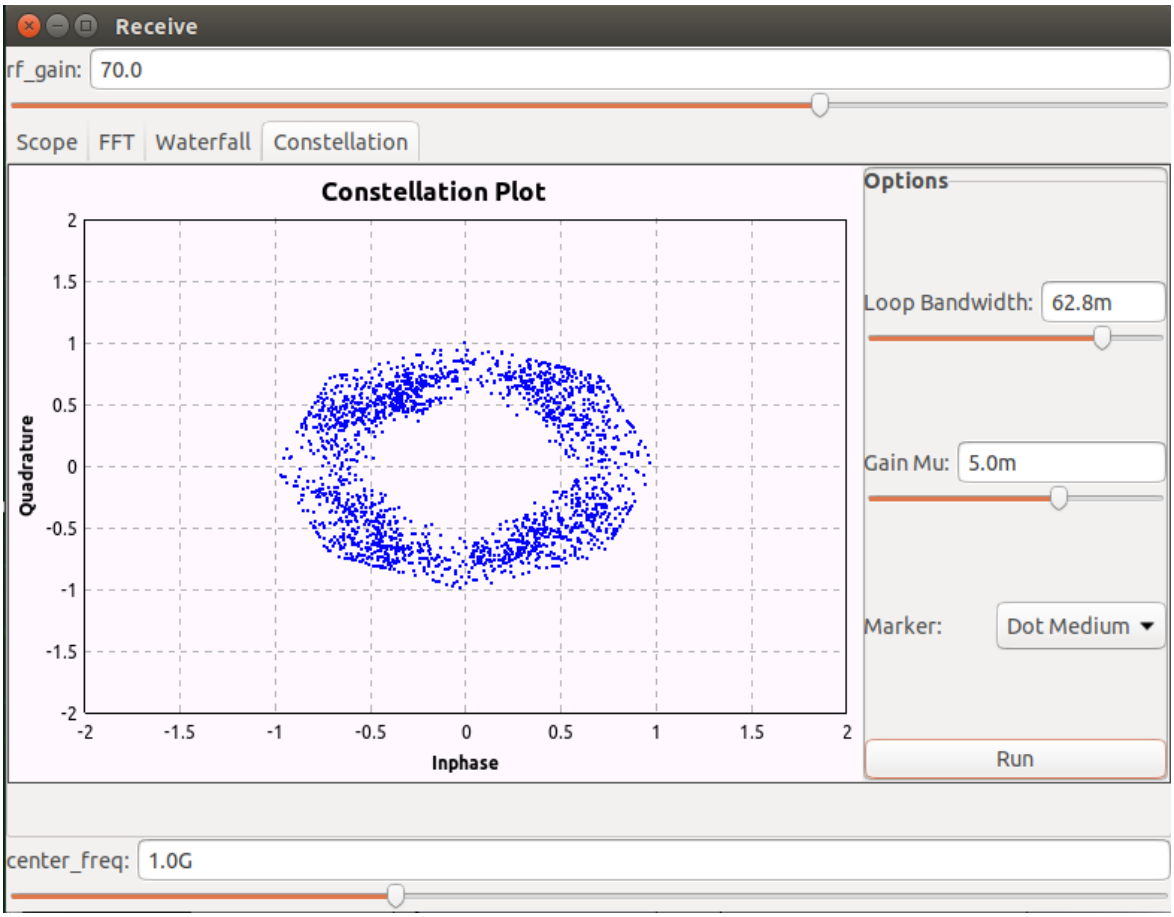

Figure 4.11. Receiver side Constellation plot

"WX GUI Scope Sink" block is used to see the oscilloscope plotting of the received complex signal. Vertical axis of this plot shows the counts value, and the horizontal axis of it shows the time. Figure 4.12 shows the Scope Plot of received signal. The signal can be narrowed or expanded in horizontal axis by changing the "Secs/Div" parameter seen in the figure. In the same manner, the signal can be narrowed or expanded in the vertical axis by changing the "Counts/Div" parameter.

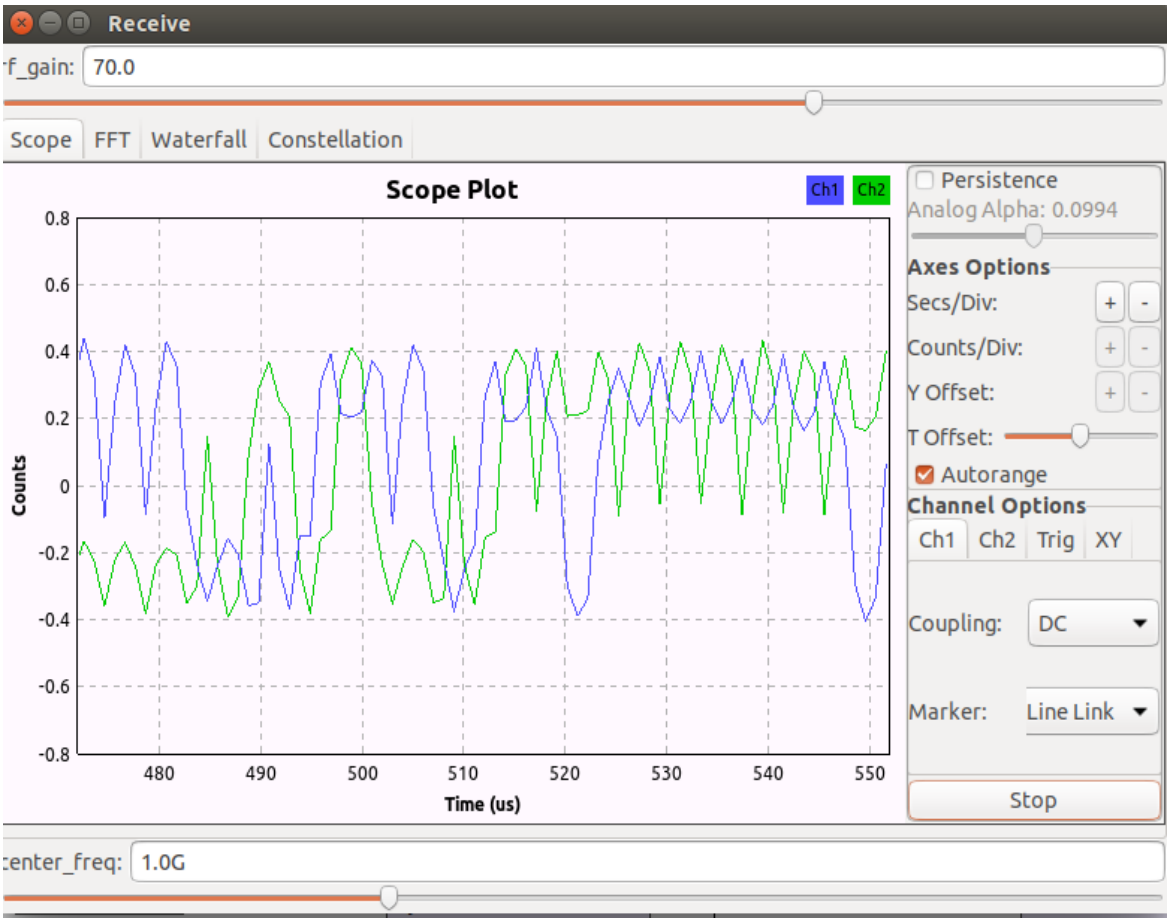

Figure 4.12. Receiver side Scope plot

"WX GUI Notebook" block is used to combine all the obtained output drawings in the same window. It has some parameters. "ID" parameter is the name of the block, and it is entered as "nb" on the receiver side. "Tab orientation" parameter determines the place of tabs in output window, and it is selected as "Top" in this study. "Labels" parameter shows the names of the tabs seen in output window. "Notebook" parameter of any output block can be used to place it into a notebook. On the receiver side, Constellation, Waterfall, FFT, and Scope plots are combined in the same output window by using "nb" notebook. The order of these output plottings are determined according to the number written next to the notebook ID. Since notebook parameter of Scope plot is entered as "nb,0", it is the first output obtained on the receiver side. In the same manner, Constellation plot is the last output.

After creating this flowgraph, it is executed and run. The next step is to display the video created in “rx.ts” pipeline.

4.2.5. Displaying the received video

4.2.5.1. Displaying the video streamed by Gstreamer

The pipeline file created on the receiver side, and named as “rx.ts” is used to display the received video. This video stream is displayed by using Mplayer program by using the command line command:

- *mplayer rx.ts*

Figure 4.13 shows a video frame obtained after playing the received real-time video which is pipelined by Gstreamer. In this real-time video, some latency is seen. A change in transmitter side video is seen in receiver side video with a latency of 2 seconds.

Figure 4.13. Gstreamer pipelined video frame on the receiver side

4.2.5.2. Displaying the video streamed by VLC

On the receiver side, after executing and running the receiver side GNU Radio Companion flowgraph, below command is used to play the received video:

- *mplayer rx.ts*

Figure 4.14 shows a video frame obtained after playing the received real-time video which is pipelined by VLC. In this real-time video some latency is seen. A change in transmitter side video is seen in receiver side video with a latency of 3 seconds.

Figure 4.14. VLC pipelined video frame on the receiver side

4.3. Simulation of Real-Time Video Streaming

In this part of thesis, some simulations using different modulation techniques such as 16QAM, 32PSK, DBPSK and GMSK are done on the laptop computer. Simulations are done by using the GNU Radio and Matlab softwares. Aims of these simulations are to measure the strength of different modulation schemes against some noise and to examine the relation between multiply constant value and SNR. Detailed explanations about the used GNU Radio simulation flowgraph, Signal to Noise Ratio (SNR) estimation method and Matlab Bit Error Analysis method are given below.

4.3.1. GNU Radio simulation flowgraph

GNU Radio simulation flowgraph using GMSK modulation is given in Figure 4.15. This flowgraph contains both the transmitter side operations and receiver side operations of video transmission. The input of this simulation flowgraph is the byte stream which is

encoded and pipelined by Gstreamer software. Flowgraph takes this data as input and does necessary transmitter side operations on it. After that gaussian noises having different amplitudes are added onto the obtained signal to see the noise sensitivity of used modulation technique for different amplitudes of noises. Since we don't give the signal into air in simulations, this noise is used to symbolize the noise added onto the transmitted signal from air in real environment video streaming case. Different modulation schemes can be simulated by changing the GMSK Mod and GMSK Demod blocks seen in Figure 4.15 according to the used modulation type. Since GMSK modulation uses 1 bit/symbol, the value of this parameter in Packet Encoder block must also be changed according to the bits/symbol value of used modulation type. Simulation flowgraph continues with executing receiver side operations on noisy signal. Reverse operations done on the transmitter side of simulation are done on the receiver side. Lastly, acquired data is written to a file and displayed by Mplayer software.

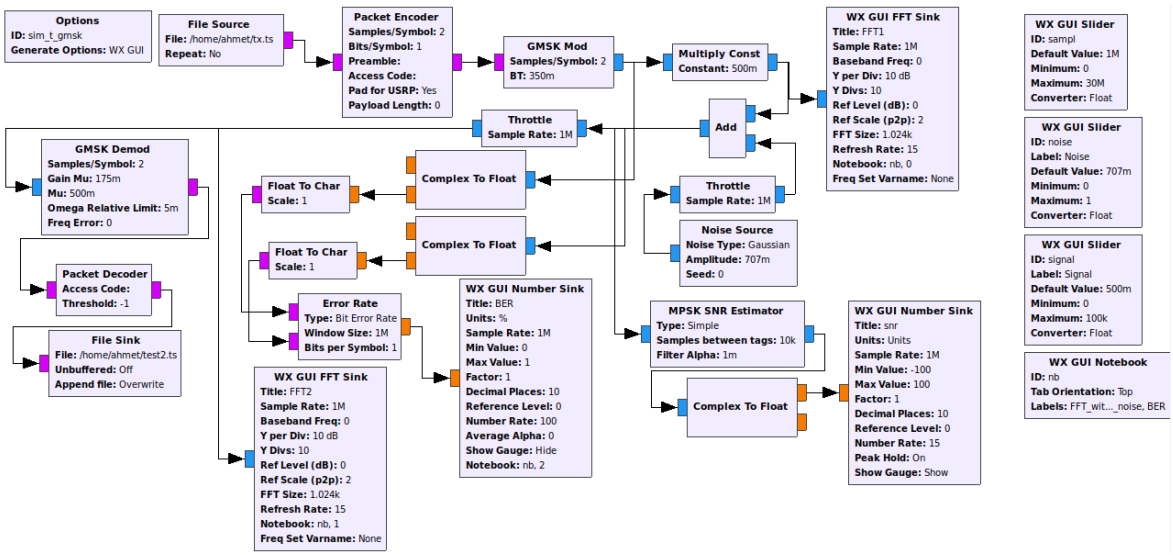

Figure 4.15. GNU Radio simulation flowgraph

4.3.2. SNR estimation using GNU Radio

SNR estimation for different Multiply Const values is an experiment done by using the noisy signal in the simulation flowgraph given in Figure 4.15. Multiply Const value is known as signal amplifier. In this experiment, our aim is to observe how this amplifier

value effects the measured SNR. GMSK modulation and a Gaussian noise with a constant amplitude of 0.707 is used in this experiment. Figure 4.15 shows how this estimation operation is realized. MPSK SNR Estimator block is used for this aim. This block has some parameters. “Type” parameter determines which SNR estimation method will be used. “Samples between tags” parameter shows how many samples will be used to send a tag containing SNR to output port. This value is set to 1×10^3 in our experiment. “Filter Alpha” parameter sets the update rate of internal running average calculations, and it is set to 1×10^{-3} in this experiment. The complex data obtained at the output of this block is sent to Complex To Float block to acquire its real part. Lastly, output of this block is printed onto the screen by WX GUI Number Sink block. Table 4.3 shows estimated SNR values in dB for different Multiply Const values. The estimated SNR values change continually because of real-time video transmission in our study. Therefore, Table 4.2 shows the obtained maximum SNR values for different Multiply Const values. Maximum estimated SNR values increase as Multiply Const value is increased, and increase rate of them is nearly the same.

Table 4.2. Multiply constant and maximum estimated SNR relation

Multiply Const	Max. Estimated SNR (dB)
1	2.273727
2	2.751729
3	4.268905
4	4.944375
5	5.576206
6	6.963826
7	7.805684
8	8.539181
9	9.999021
10	10.338159
100	99.988380
1000	998.684448
10000	9979.349609

4.3.3. Bit error rate analysis using Matlab

Noise sensitivities of different modulation techniques are measured by Bit Error Rate (BER) analysis method. In this method, different Signal to Noise Ratio (SNR) values are applied to the transmission system to obtain the corresponding BER values for the used modulation schemes. SNR is defined as the ratio of the power of a signal to the power of background noise, and expressed in decibels (dB). BER is defined as the bit errors per unit time in a transmission system. It is the ratio of the number of bit errors to the total number of transferred bits. Bit errors could occur because of noise, distortion or interference in a transmission system.

BER analysis is done by using either “bertool” command on the command window of Matlab software or by using some Matlab codes. When bertool command is entered on the command window, a window having some parameters comes on the screen. In this window, SNR range, channel type, modulation type and modulation order parameters are set separately for the MSK, 16QAM, 32PSK, DBPSK modulation schemes in this study. SNR range is set as “0:20” and channel type parameter is set as “AWGN” for all modulation schemes. Lastly, modulation type and modulation order parameters are set according to the used modulation schemes. Since SNR value is set as bigger than 0 dB, signal power is always bigger than noise power in this experiment. Moreover, the ratio of signal power to the noise power increases as SNR goes from 0 dB to 20 dB. Additive White Gaussian Noise (AWGN) is a basic noise model. This noise has some characteristics as its name suggests. The received signal is equal to the sum of transmitted signal and noise in AWGN case. Moreover, AWGN has uniform power across the whole frequency band. Lastly, the probability distribution of the noise samples is Gaussian with a zero mean.

When all these parameters are set for each modulation scheme, some values showing the relation between SNR and BER are obtained. Table 4.3 shows these values. First colomn of the table shows SNR values which are set between 0 dB and 20 dB. Corresponding colomns of the table show the obtained BER values for MSK, 16QAM, 32PSK and DBPSK respectively. When the obtained BER values are examined, they

decrease continuously for increasing SNR values. It is also seen that their decreasing rate speeds up when SNR values are increased.

Table 4.3. Bit error rate analysis of different modulation schemes

	Bit Error Quantity ($\times 10^{-6}$)			
SNR (dB)	MSK	16QAM	32PSK	DBPSK
0	0,144928	0,140982	0,225631	0,183940
1	0,106229	0,118997	0,207331	0,141980
2	0,072199	0,097741	0,189238	0,102485
3	0,044710	0,077453	0,171360	0,067989
4	0,024689	0,058623	0,153812	0,040558
5	0,011837	0,041892	0,136823	0,021165
6	0,004765	0,027871	0,120659	0,009333
7	0,001544	0,016967	0,105523	0,003329
8	0,000382	0,009247	0,091470	0,000909
9	$6{,}725219\times 10^{-5}$	0,004390	0,078405	0,000177
10	$7{,}744186\times 10^{-6}$	0,001754	0,066140	$2{,}269996 \times 10^{-5}$
11	$5{,}226134 \times 10^{-7}$	0,000565	0,054512	$1{,}704223 \times 10^{-6}$
12	$1{,}801202 \times 10^{-8}$	0,000139	0,043495	$6{,}544347 \times 10^{-8}$
13	$2{,}665862 \times 10^{-10}$	$2{,}423379 \times 10^{-5}$	0,033248	$1{,}080577 \times 10^{-9}$
14	$1{,}362037 \times 10^{-12}$	$2{,}763208 \times 10^{-6}$	0,024063	$6{,}165749 \times 10^{-12}$
15	$1{,}824791 \times 10^{-15}$	$1{,}841856 \times 10^{-7}$	0,016266	$9{,}233633\times 10^{-15}$
16	$4{,}534791 \times 10^{-19}$	$6{,}250201 \times 10^{-9}$	0,010100	$2{,}566819\times 10^{-18}$
17	$1{,}351793 \times 10^{-23}$	$9{,}071625 \times 10^{-11}$	0,005642	$8{,}564165 \times 10^{-23}$
18	$2{,}792028 \times 10^{-29}$	$4{,}522309 \times 10^{-13}$	0,002762	$1{,}980800 \times 10^{-28}$
19	$2{,}002147 \times 10^{-36}$	$5{,}874180 \times 10^{-16}$	0,001147	$1{,}591230 \times 10^{-35}$
20	$2{,}088488 \times 10^{-45}$	$1{,}404037 \times 10^{-19}$	0,000387	$1{,}860038 \times 10^{-44}$

The other method of obtaining the relation between SNR and BER values is to use some Matlab commands. BER values for some SNR values for each modulation scheme are obtained by using “berawgn” Matlab command. The resultant BER values are the same with the values given in Table 4.3 when we use this command. These values are plotted with “semilogy” Matlab command. Figure 4.16 shows SNR versus BER values for all the modulation schemes used in simulation part of this study. Drawings are done with different colors and line types for each modulation type. Figure shows that the worst modulation scheme is 32PSK for all SNR values. Moreover, 16QAM modulation is better than

DBPSK and MSK at low SNR values, but it is worse than both of them at higher SNR values.

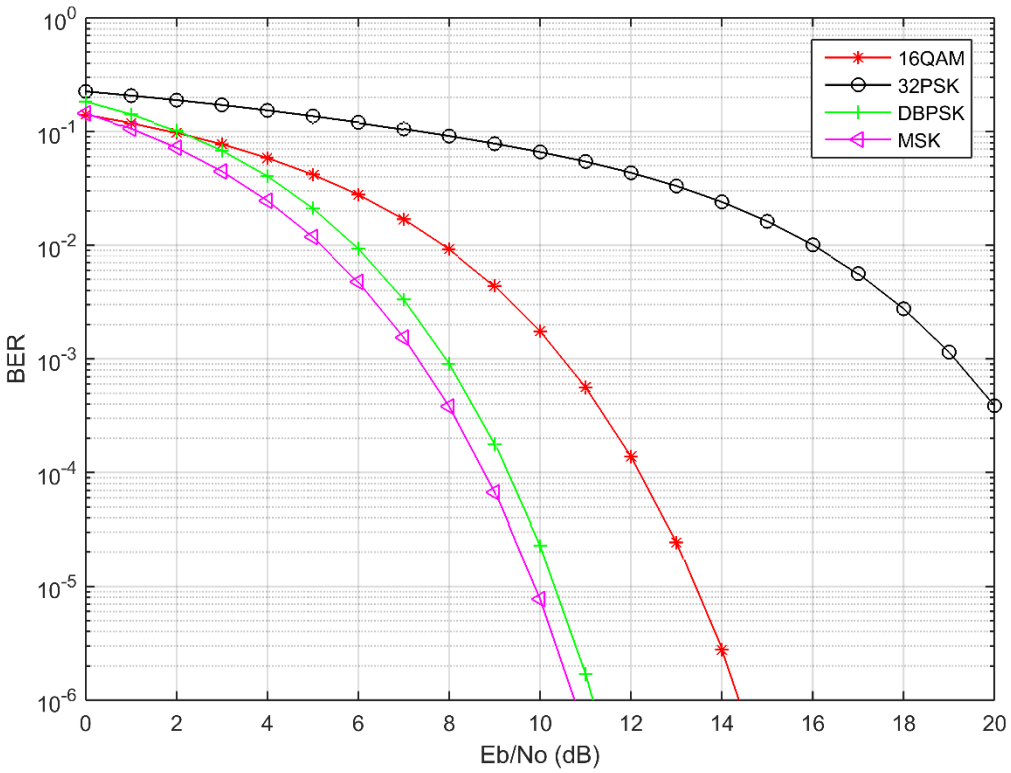

Figure 4.16. SNR versus BER values

5. CONCLUSIONS AND RECOMMENDATIONS

Aim of this thesis study is to transmit a high resolution real-time compressed video obtained from a UAV camera in real and simulation environments by using Gstreamer, VLC, GNU Radio and USRP devices. H.264 video compression method is used to obtain a video having lower bit rate than the raw video, and compressed video is transmitted over a low bandwidth communication network. Compression and pipelining of raw video operations are realized by using Gstreamer and VLC programs. Compressed video is modulated in GNU Radio by using GMSK modulation which is known as a good modulation technique for real-time video transmission purposes. USRP devices which have enough real-time transmission bandwidth are selected for the transmission and reception of video signals.

Some communication system parameters like sample rate and center frequency are tried to be optimized according to the received video quality. Bit error quantity is also an important quality factor in video transmission operations. It determines the number of bits that are transmitted incorrectly. Therefore, some distortions are seen in received video when this error quantity increases. According to the experiments done in this study, error quantity depends on the noise added onto transmitted signal and used digital modulation type. It is also shown that increase rate of error quantity is higher when added noise amplitude is bigger. SNR is another important parameter in video transmission. Experiments done in this study show that this parameter depends on Multiply Const parameter used in GNU Radio flowgraph.

In future works, it is aimed to transmit higher quality videos by changing video signal modulation type and raw video compression method. Modulation type determines coding quantity of digital signals. Simulation part of this study shows that new modulation techniques can safely transmit more data at the same time thanks to their more efficient symbol coding methods compared to GMSK modulation. Since new modulation techniques can create the symbols having more bits compared to some old techniques, lower latency is seen in the received high quality video when these techniques are used. In

addition to this, video compression method is also important in high quality video transmission. Some new video coding techniques can transmit the same video with less data compared to some old coding techniques. This is vital for higher quality video transmission due to limited bandwidth of communication networks. Moreover, received video latency is aimed to decrease by doing all the signal processing operations on FPGAs instead of using laptop CPU. Since video codec programs also cause some latency in video coding and decoding processes, more efficient video codec programs are also aimed to be used to decrease the received video latency.

REFERENCES

Chen, C. Y., Tseng, F. H., Chang, K. D., Chao, H. C. C., Chen, J. L. 2010, Reconfigurable software defined radio and its applications, Journal of Applied Science and Engineering, 13,1, 29-38.

Crespi, F. L., Maglioli, M., Benco, S., Perotti, A., 2012, A real-time video broadcasting system based on the GNU Radio-USRP2 platform, Proc. Karlsruhe Workshop on Software Radios (WSR).

Debashri, R., Tathagata, M., Mainak, C., Eduardo, P., 2020, Adaptive streaming of HD and 360Â° videos over software defined radios. Pervasive and Mobile Computing, 67, DOI: 10.1016/j.pmcj.2020.101215.

Dhar, R., George, G., Malani, A., Steenkiste, P., 2006, Supporting integrated MAC and PHY software development for the USRP SDR, 1st IEEE Workshop on Networking Technologies for Software Defined Radio Networks, 68-77, IEEE.

Dhruv, D., Vishal, S., 2017, Wireless video transmission using USRP and GNU Radio, International Journal of Research in Modern Engineering and Emerging Technology (IJRMEET), 5,11.

Electronics-Notes, 2021. What is GMSK Modulation - Gaussian Minimum Shift Keying, https://www.electronics-notes.com/articles/radio/modulation/what-is-gmsk-gaussian-minimum-shift-keying.php access date: 03.06.2020.

Ettus Research Products, 2021, USRP B210, https://www.ettus.com/all-products/ub210-kit/ access date: 01.05.2020

Ettus Research Products, 2021, https://www.ettus.com/products/ access date: 09.01.2021.

Ghani, M. F. A., Latiff, N. A., Yusof, S. S., Rashid, R. A., Malik, N. N. A., et al. 2017, Video transmission based on Orthogonal Frequency Division Multiplexing (OFDM) utilizing television white space, 2017 IEEE 13th Malaysia International Conference on Communications (MICC), 152-157.

GNU Radio, 2021, About GNU Radio, https://www.gnuradio.org/about/ access date: 18.02.2021.

Gstreamer Open Source Multimedia Framework, 2021, What is Gstreamer? , https://gstreamer.freedesktop.org/ access date: 13.09.2020.

Kalva, H., 2006, The H. 264 video coding standard, IEEE Annals of the History of Computing, 13, 4, 86-90.

REFERENCES (continue)

Kamaci, N., Altunbasak, Y., 2003, Performance comparison of the emerging H. 264 video coding standard with the existing standards, 2003 International Conference on Multimedia and Expo. ICME'03. Proceedings, 1, 1-345, IEEE.

Leferman, M. J., Pu, D., Wyglinski, A. M., 2010, GNU radio for cognitive radio experimentation, Cognitive Radio Communications and Networks, 507-538. DOI: 10.1016/B978-0-12-374715-0.00018-6.

Nguyen, D. T., 2013, Implementation of OFDM systems using GNU Radio and USRP, Master of Engineering, Research thesis, School of Electrical, Computer and Telecommunications Engineering, University of Wollongong. http://ro.uow.edu.au/theses/3977

Nimmi, S., Saranya, V., Gandhiraj, R., 2014, March, Real-time video streaming using GStreamer in GNU Radio platform, 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE), 1-6, IEEE.

Patil, R. B., Kulat, K. D., Gandhi, A. S., 2017, GMSK based real time video transmission on SDR platform, International Journal Appl. Eng. Research. (IJAER*)*, 12,15, 5089-5093.

Rfwireless-world, 2012, DUC vs DDC, Difference between DUC and DDC, https://www.rfwireless-world.com/Terminology/DUC-vs-DDC.html access date: 21.11.2020.

Saman, J., 2006, Streaming networks with VLC, NLUUG.

Shavanthi, L., Nagamani, K., Sandya, H. B., 2018, May, Implementation of DVB Standards using Software-Defined Radio, 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 860-864, IEEE.

Sundari, G., Bernatin, T., Somani, P., 2015, March), H. 264 encoder using Gstreamer, 2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015], 1-4, IEEE.

Tahir, M., Mohamad, H., Ramli, N., Jarot, S. P., 2012, July, Experimental implementation of dynamic spectrum access for video transmission using USRP, 2012 International Conference on Computer and Communication Engineering (ICCCE), 228-233, IEEE.

Taymans, W., Baker, S., Wingo, A., Bultje, R. S., Kost, S., 2013, Gstreamer application development manual (1.2. 3)", Publicado en la Web.

Tucker, D. C., Tagliarini, G. A., 2009, Prototyping with gnu radio and the usrp-where to begin, IEEE Southeastcon 2009, 50-54, IEEE.

REFERENCES (continue)

Vachhani, K., Mallari, R. A. 2015, August, Experimental study on wide band FM receiver using GNURadio and RTL-SDR, 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 1810-1814, IEEE.

VideoLAN, 2020, https://wiki.videolan.org/VideoLAN/ access date: 19.09.2020.

Vilches, T., Dujovne, D., 2014, GNUradio and 802.11: Performance evaluation and limitations", IEEE Network, 28,5, 27-31.

Wang, Y., 2011, August, A Improved Image Edge Detection Algorithm Based on H. 264 Intra Prediction, 2011 International Conference on Intelligence Science and Information Engineering, 450-453, IEEE.

Wiegand, T., Sullivan, G. J., Bjontegaard, G., Luthra, A., 2003, Overview of the H. 264/AVC video coding standard, IEEE Transactions on Circuits and Systems for Video Technology, 13,7, 560-576.

Yu, H., Lin, Z., and Pan, F., 2005, Applications and improvement of H. 264 in medical video compression, IEEE Transactions on Circuits and Systems I: Regular Papers, 52, 12.

Zhang, L., 2013, Implementation of wireless communication based on software defined radio, Unpublished Thesis, Technical University of Madrid, Madrid.