HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF ELECTRONICS AND TELECOMMUNICATIONS

PROJECT REPORT

Video Coding, Communication and Evaluation with libxvid and ac3

Multimedia Data Compression and Coding course

Instructor:	Dr. rer. nat. Pham Van Tien	
Student:	Bui Van Binh	20224301
	Dang Nguyen Hai Anh	20224295
	Vu Truong Giang	20224308
	Bui Hoang Giang	20224307
	Le Thanh Long	20224325

HANOI, 5-2025

Table of work resume

Name	Work
Bui Van Binh	- Use FFmpeg to measure **PSNR and SSIM** for each video. - Make statistical tables, analyze the results of each attempt. - Research and learn about video and audio codecs- Write report content: comparisons, reviews, comments
Dang Nguyen Hai Anh	- Video recording of each member - Cut, edit, standardize videos - Insert overlays if done before transfer (drawtext, drawbox) - Save the standard original.mp4 as a source of comparison
Bui Hoang Giang	- Use FFmpeg to stream video via UDP - Different bitrate settings (5 tests) - Encode using registered codecs - Record configuration for each transfer
Le Thanh Long	- Use FFplay or FFmpeg to receive and record videos - Name the file according to the test case (e.g., recv_bitrate500_loss10.mp4)
Vu Truong Giang	- Write and run random Python discard packet scripts according to the rates: 0%, 10%, 20%, 30%, 40% - Connection between the sender and receiver via an intermediate script - Log lost packets

Contents

I. Reasons for choosing codecs

In the list of FFmpeg-supported codecs (https://ffmpeg.org/ffmpeg-codecs.html), our team has selected:

- **Video codec:** libxvid
- **Audio codec:** ac3

1. libxvid (Video codec):

libxvid is an open-source library for encoding/decoding videos according to **the MPEG-4 Part 2 (ASP)** standard. It is a fairly old but stable codec, widely supported, and easily compatible across multiple platforms.

Reasons to choose:

- FFmpeg supports libxvid well and is easy to use.
- Flexible adjustable bitrate (CBR or VBR), in accordance with the bitrate limit requirement of the topic.
- Fast, lightweight encryption, suitable for student devices that are not too strong.
- Easy to check the quality after dropping the packet because there is no complex recovery feature like H.264.

2. ac3 (Audio codec)

ac3 (Dolby Digital) is an ATSC **A/52** compressed audio codec. It is commonly used in television, DVDs, and streaming.

Reasons to choose:

- Easy to use, built-in in FFmpeg, no additional installation required.
- Keeps the quality stable at good compression (~128–192 kbps), not too heavy when transmitted over the network.
- Good handling of stereo or multi-channel (5.1) sound.
- Does not cause VLC compatibility errors when transmitting via UDP.

II. Project Objectives

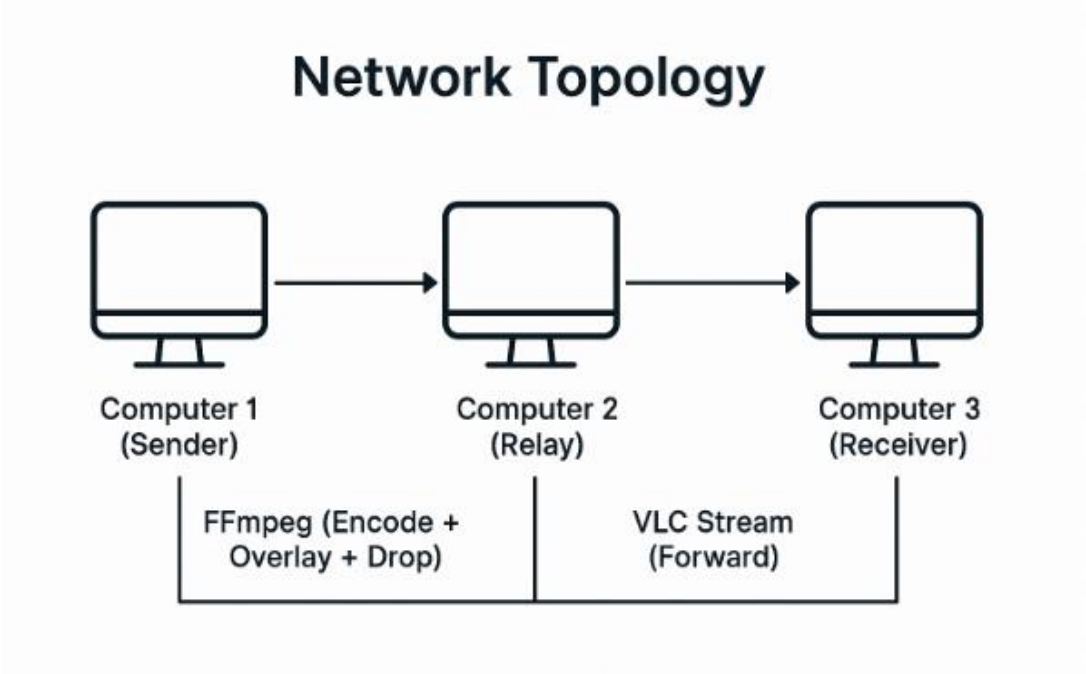
Stream real-time video through **3 Wi-Fi connected computers** using FFmpeg & VLC.

Apply techniques such as:

1. Compress video/audio using a codec of your choice.
2. Overlay nội dung động (blocks, text, animation).
3. Limit the bitrate at the transmitter.
4. Random drop packet simulates packet loss.
5. Evaluate the quality of the video obtained using **PSNR and SSIM**.

III. System diagram and Technology used:

1. System Diagram:



1. Technology used:

Ingredient	Technology
Video codec	Libxvid
Audio Codec	Ac3
Streaming	FFmpeg + VLC
Overlay	FFmpeg filtergraph
Drop Packet	Python script + FFmpeg UDP manipulation
Quality Assessment	ffmpeg-quality-metrics, Python (scikit-image
Transmission Protocol	UDP over Wi-Fi

IV. Implementation process:

4.1. Video Recording

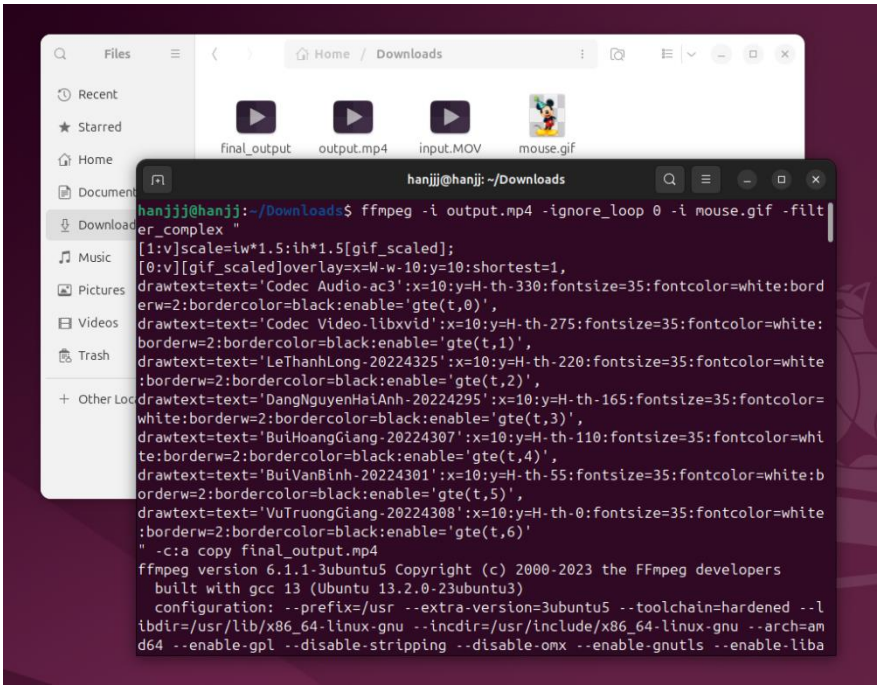
- Video duration: 3 minutes

- Video name : video_output.mp4

4.2. Encode + Overlay:

- **Overlay the mouse.gif animated GIF** on the original video output.mp4, with the size scaled up to 1.5 times, with the upper right corner (x=W-w-10:y=10).

- **Insert dynamic text (drawtext) one line at a** time on the video, including the group nameplate, video/audio codec, name of each member, each text appears every 1 second (enable='gte(t,X)').



4.3. Drop packet script:

This part will be performed by the intermediate machine (machine 2) when it receives the original package from machine 1 and transmits it to machine 2, in the process of which interference is added.

The middleman runs a UDP proxy written in Python. This proxy simulates packet loss by randomly discarding several UDP packets before forwarding them to the receiver. This helps the team assess the codec's fault tolerance when actual data loss occurs.

This section will be saved in the *noise.py file.*

```python
import socket
import random
import argparse

# Tham số dòng lệnh
parser = argparse.ArgumentParser(description="UDP Proxy with Packet Dropping")
parser.add_argument('--listen_port', type=int, default=5000)
parser.add_argument('--forward_ip', type=str, required=True)
parser.add_argument('--forward_port', type=int, default=1234)
parser.add_argument('--drop_rate', type=float, default=0.1)
args = parser.parse_args()

# Khởi tạo socket
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind(('', args.listen_port))

print(f"📡 Proxy running on port {args.listen_port}")
print(f"Forwarding to {args.forward_ip}:{args.forward_port} | Drop rate: {args.drop_rate * 100:.0f}%")

# Biến đếm
total_packets = 0
dropped_packets = 0
forwarded_packets = 0

try:
    while True:
        data, addr = sock.recvfrom(65536)
        total_packets += 1
        if random.random() > args.drop_rate:
            sock.sendto(data, (args.forward_ip, args.forward_port))
            forwarded_packets += 1
        else:
            dropped_packets += 1
            print("💥 Packet dropped")

except KeyboardInterrupt:
    print("\n🛑 Proxy stopped by user.")
    print(f"📦 Tổng số gói nhận:     {total_packets}")
    print(f"📤 Gửi thành công:       {forwarded_packets}")
    print(f"❌ Gói bị loại (drop):   {dropped_packets}")
```

4.4. Transmit & Receive:

Process:

- Machine 1 (Sender): transmit original video using FFmpeg

- Machine 2 (Middle Guy): jamming/dropping the packet with a Python script (proxy drop packet) and transmitting it to machine 3

- Machine 3 (Receiver): Receive from machine 2 and save to the machine with vlc.

a. Network setup (definitely Wi-Fi devices)

- Machine 1 (Sender): 192.168.1.101
- Máy 2 (Middle Proxy): 192.168.1.102
- Máy 3 (Receiver): 192.168.1.103

b. On Machine 2: run the Python drop packet file (named: udp_proxy.py)

python3 udp_proxy.py --listen_port 5000 --forward_ip 192.168.1.103 --forward_port 1234 --drop_rate 0.2

c. On Machine 1: send video to Machine 2 (port proxy)

ffmpeg -re -i input.mp4 -c:v libxvid -b:v 1M -c:a ac3 -b:a 128k -f mpegts udp://192.168.1.102:5000

d. On Machine 3: receive video

ffmpeg -i udp://@:1234 -c copy received.avi

With 5 times bitrate and Drop Rate adjustment with different adjustment values.

Saved videos: *1M-35%.mp4, 1M5-20%.mp4, 2M-10%.mp4, 2M5-0%.mp4, 500K-50%.mp4.*

V. Statistics and evaluation:

5.1. PSNR Measurement Command Description

We use ffmpeg to compare the quality of the transmitted video with the original video via the PSNR index:

ffmpeg -i 1M-35%.mp4 -i final_output.mp4 -lavfi psnr=stats_file=psnr_logfile.txt -f null –

- *1M-35%.mp4* is the video received after transmission with a bitrate of 1Mbps and a packet loss rate of 35%.
- *final_output.mp4* is the original video, with dynamic overlays and codec information, group names.
- The PSNR filter calculates the PSNR value between 2 videos, the result is recorded in the psnr_logfile.txt file.
- *-f null* - means that the video is not exported to a new file, only calculated.

5.2. Description of SSIM Measurement Command

Similar to SSIM, we use the command:

ffmpeg -i 1M-35%.mp4 -i final_output.mp4 -lavfi ssim=stats_file=ssim_logfile.txt -f null -

- The SSIM metric measures the similarity in visual structure between the received video and the original video.
- Recorded results *ssim_logfile.txt.*

5. 3. Kết quả thí nghiệm

Bitrate	Package Loss Rate (%)	PSNR trung bình (dB)	Average SSIM
500K	50	12.31	0.71
1M	35	12.54	0.66
1.5M	20	12.68	0.69
2M	10	13.07	0.689
2.5M	0	15.04	0.785

5.4. Analytics and comments

- **A low PSNR (< 15 dB)** and **an SSIM below 0.8** indicate poor video quality, a lot of frame loss, and a noisy image.
- As the bitrate increases and the packet loss rate decreases, both PSNR and SSIM increase, meaning that the resulting video is closer to the original video, with a clearer and smoother image.
- The **2.5 Mbps lossless configuration** gives the best results with a PSNR of 15.04 dB and an SSIM of 0.785, although it does not reach the "very good" level (SSIM > 0.95) but is quite good, suitable for practical applications.
- The remaining configurations (low bitrate, high packet loss) lead to a marked deterioration in video quality, affecting the viewer experience.

Observe: Adopting bitrate control and reducing packet loss rates is critical to maintaining video quality in transmission over unstable Wi-Fi networks. Through the PSNR and SSIM indicators, we can accurately assess the degree of quality degradation due to factors such as encoding, packet loss, and interference during video transmission.

VI. Attached Files:

Attached Files:

- *output.mp4* :Original video of group recording (name reading + MSSV)
- *final_output.mp4* : Video overlay full name, codec, animation
- *500K-50%.mp4 :* 500Kbps in Bitra - Drop Rate 50%
- *1st-35%.mp4 :* 1Mbps in Bitra - Drop rate 35%
- *1M5-20%.mp4* : 1.5Mbps in Bitra - Drop rate 20%
- *2m-10%.mp4 :* 2Mbps in Bitra - Drop rate 10%
- *2M5-0%.mp4* : Bitrate 2.5Mbps – No drop
- *Noise.py :* Proxy UDP drop packet
- *overlay.sh*: FFmpeg overlay filter script tạo final_output.mp4

Full source code and documentation available at:

https://github.com/Chu-vit-Huster-ngao-ngo-2k4/libxvid-and-ac3-communication

VII. References

[1] *FFmpeg Codecs Documentation*. https://ffmpeg.org/ffmpeg-codecs.html. The official page lists and describes the supported codecs, including libxvid (video codec) and ac3 (audio codec). This is the standard source for learning details about parameters, capabilities, and how to use codecs.
[2] *Xvid Codec Official Site*. https://www.xvid.com/. Xvid is a popular, open-source, MPEG-4 video codec, which is used to compress videos with good quality and small file size. Suitable for streaming and storage.
[3] *Dolby AC-3 Audio Codec Overview.* AC-3 (Dolby Digital) là codec âm thanh đa kênh phổ biến, được dùng trong truyền hình, DVD, và streaming, cung cấp âm thanh vòm chất lượng cao.