MOGOL TORI VISION TASK

SSH

- SSH stands for Secure Shell.
- Used to access remote machines securely via terminal.
- SSH generally works on port 22 by default.
- Uses public-private key authentication or passwords.
- I achieved an SSH connection through my macOS as a client and Ubuntu (from vm) as a server.

Process

Accessing Ubuntu IP address:
ip a

From my zsh terminal:
ssh servername@ip
Password

Later on I created a config file for faster connectivity:

Using nano i created a file in .ssh folder

```
nafizahmed@Nafizs-MacBook-Air .ssh % cat config
Host pizza
        Hostname 192.168.64.3
        Port 22
```

Gstreamer setup:

GStreamer is a powerful multimedia framework to process audio and video streams. Using Python bindings (PyGObject), you can easily create pipelines for video capture, processing, and display.

macOS

For installation of Homebrew,

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

For gstreamer devtools,

```
brew install gstreamer gst-devtools gst-plugins-{base,good,bad,ugly} python@3 pygobject3
```

MOGOL TORI VISION TASK

Video concepts:

-video is basically frame by frame image
-has different containers containing metadata(.mp4)

Importing libraries & initializations:

```
import gi
import numpy as np
import cv2
from gi.repository import Gst, GstApp

gi.require_version("Gst", "1.0")
gi.require_version("GstApp", "1.0")
Gst.init(None)
```

Pipeline constructions:

```
pipeline = Gst.parse_launch("avfvideosrc ! videoconvert ! video/x-raw,format=BGR ! appsink name=sink")
appsink = pipeline.get_by_name("sink")
pipeline.set_state(Gst.State.PLAYING)
```

Avfvideosrc = macOs specific webcam source
Videoconvert = video format conversion
video/x-raw,format=BGR = opencv compatible video format
appsink name=sink = sinking of video

Changes I wanted to make but couldn't yet:

Changeable bitrate:

Putting this command in pipeline x264enc bitrate=2048
We can write a function for changeable bitrate.

Making grid for multiple cameras:

We can get multiple sources and resize frames for different output grid.

MOGOL TORI VISION TASK

Pipeline workflow summary:

- This pipeline captures a webcam stream
- converts format
- pulls raw BGR images to Python
- processed with OpenCV.

Different mode for changing filters, flip, rotate:

- r: Rotate
- c: Canny edge
- g: Grayscale
- h: HSV
- l: LAB
- n: Normal

Frame processing:

sample = appsink.try_pull_sample(Gst.SECOND)

Asks the appsink to give you the next video frame (sample), waiting up to Gst.SECOND (1 second).

buffer=sample.get_buffer()
caps=sample.get_caps()

Gets the capabilities (metadata) of the sample, such as width, height, and pixel format.

width = structure.get_value("width") and height = structure.get_value("height") extract the frame's resolution so the raw pixel buffer can be reshaped into an image.

```
    width=structure.get_value("width")
    height=structure.get_value("height")
```

```
try:
  while True:
    sample=appsink.try_pull_sample(Gst.SECOND)
    if sample is None:
       continue
```

```
    buffer=sample.get_buffer()
    caps=sample.get_caps()
    structure=caps.get_structure(0)
    width=structure.get_value("width")
    height=structure.get_value("height")

    success, map_info=buffer.map(Gst.MapFlags.READ)
    if not success:
        continue

    frame=np.frombuffer(map_info.data, np.uint8).reshape((height, width, 3))
    buffer.unmap(map_info)

    if mode=="r":
        frame=np.rot90(frame)
    if flip:
        frame=cv2.flip(frame, 1)
    elif mode=="c":
        gray=cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        frame=cv2.Canny(gray, 75, 150)
    elif mode=="g":
        frame=cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    elif mode=="h":
        frame=cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    elif mode=="l":
        frame=cv2.cvtColor(frame, cv2.COLOR_BGR2Lab)
```

Gstreamer exit of openCV exit:

pipeline.set_state(Gst.State.NULL)
cv2.destroyAllWindows()