Colab Notebook to run the codes

In [1]:

```
from tqdm import tqdm
import os
import cv2
from PIL import Image
import imageio.v2 as imageio  # Explicitly use v2 to avoid deprecation warnings
```

In [2]:

```
gpu_info = !nvidia-smi
gpu_info = '\n'.join(gpu_info)
if gpu_info.find('failed') >= 0:
  print('Not connected to a GPU')
else:
  print(gpu_info)

from psutil import virtual_memory
ram_gb = virtual_memory().total / 1e9
print('Your runtime has {:.1f} gigabytes of available RAM\n'.format(ram_gb))

if ram_gb < 20:
  print('Not using a high-RAM runtime')
else:
  print('You are using a high-RAM runtime!')
```

```
Mon Dec  9 18:04:01 2024
+---------------------------------------------------------------------------------------
-+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------
-+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC
|
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M.
|
|=========================================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0
|
| N/A   45C    P8              12W /  70W |      0MiB / 15360MiB |      0%      Default
|
|                                         |                      |                  N/A
|
+-----------------------------------------+----------------------+----------------------
-+


+---------------------------------------------------------------------------------------
-+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory
|
|        ID   ID                                                             Usage
|
|=======================================================================================|
|  No running processes found
|
+---------------------------------------------------------------------------------------
-+
Your runtime has 13.6 gigabytes of available RAM

Not using a high-RAM runtime
```

Load drive

In [ ]:

```
# Load the drive and check
from google.colab import drive
drive.mount('/content/drive')
!ls /content/drive/My\ Drive/'ECE Parallel Programming'/Aravind/
# Copy the backup.zip file to the current directory
!cp /content/drive/My\ Drive/'ECE Parallel Programming'/Aravind/backup.zip /content/
# Unzip the backup file
!unzip backup.zip
```

In [29]:

```
# Clear the src/temp/ directory
!rm -rf src/temp/
!mkdir src/temp/
```

Backup to drive

In [ ]:

```
# To save the labeled output to make a video

# If using colab
# !zip -r labeled_output.zip /content/collab_backup/combined_steps/labeled_output
# !rm -rf /content/src/temp/labeled_output
# !mkdir /content/src/temp/labeled_output

# Else:
!rm -rf src/temp/labeled_output
!mkdir src/temp/labeled_output
```

In [ ]:

```
!ls
```

```
5b_upscale  backup.zip data  drive  include  models  performance_metrics  src tests
```

In [ ]:

```
# Zip the folder
all_files = ['ece1747.ipynb','data/','include/',
             'models/','performance_metrics/',
             'src/','tests/']
# Zip everything in above all_files
!zip -r backup.zip {' '.join(all_files)}
```

In [32]:

```
!du -sh backup.zip
```

```
945M backup.zip
```

In [33]:

```
!cp backup.zip /content/drive/MyDrive/ECE\ Parallel\ Programming/Aravind/
!ls /content/drive/MyDrive/ECE\ Parallel\ Programming/Aravind/
```

```
backup_v1.zip  backup_v2.zip  backup.zip
```

Upscaled Images

Generate the upscaled images

In [5]:

```
# Create folders 1b/1x 1b/2x 1b/3x/ 1b/4x and make different versions of the upscaled ima
ge
subfolder_parent_upscale = "5b"
folder_parent_upscale = "upscale"
folder_parent_upscale = f"{subfolder_parent_upscale}_{folder_parent_upscale}"
folder_names = ["1x", "2x", "3x", "4x"]
folder_names = [f"{folder_parent_upscale}/{folder_name}" for folder_name in folder_names
]

for folder_name in folder_names:
    os.makedirs(folder_name, exist_ok=True)


def upscale(image, factor):
    height, width = image.shape[:2]
    new_height = int(height * factor)
    new_width = int(width * factor)
    image_upscaled = cv2.resize(image, (new_width, new_height), interpolation=cv2.INTER_
CUBIC)
    return image_upscaled

# Run through each image in 5b
files = os.listdir(f"/content/data/OSUdata/{subfolder_parent_upscale}")
for file in tqdm(files):
    for i in range(1, 5):
        image = cv2.imread(f"/content/data/OSUdata/{subfolder_parent_upscale}/{file}")
        image_upscaled = upscale(image, i)
        cv2.imwrite(f"{folder_parent_upscale}/{i}x/{file}", image_upscaled)
```

```
100%|██████████| 2031/2031 [01:05<00:00, 31.01it/s]
```

In [ ]:

```
# !rm -rf /content/5b_upscale
```

Get the upscaled images' background

In [9]:

```
!nvcc -c tests/gmm.cu -o tests/gmm.o


!g++ tests/get_background.cpp \
    tests/gmm.o -o \
    tests/get_background \
    -I/usr/include/opencv4 -I/usr/local/cuda/include \
    -L/usr/lib -L/usr/local/cuda/lib64 -lopencv_core \
    -lopencv_imgproc -lopencv_highgui -lopencv_imgcodecs \
    -lopencv_features2d -lcudart -lcufft
```

In [10]:

```
!./tests/get_background /content/5b_upscale
```

```
Free memory: 14999 MB, Total memory: 15102 MB
Processing folder: /content/5b_upscale/4x
Number of frames: 508
Kernel execution started. Processing pixel (0, 0).
Background generation kernel started. Processing pixel (0, 0).
Background saved to background_4x.png
Time taken for /content/5b_upscale/4x: 134412 ms
Processing folder: /content/5b_upscale/3x
Number of frames: 508
Kernel execution started. Processing pixel (0, 0).
Background generation kernel started. Processing pixel (0, 0).
Background saved to background_3x.png
Time taken for /content/5b_upscale/3x: 77915 ms
Processing folder: /content/5b_upscale/2x
Processing folder: /content/5b_upscale/2x
Number of frames: 508
Kernel execution started. Processing pixel (0, 0).
Background generation kernel started. Processing pixel (0, 0).
Background saved to background_2x.png
Time taken for /content/5b_upscale/2x: 34681 ms
Processing folder: /content/5b_upscale/1x
Number of frames: 508
Kernel execution started. Processing pixel (0, 0).
Background generation kernel started. Processing pixel (0, 0).
Background saved to background_1x.png
Time taken for /content/5b_upscale/1x: 9689 ms
```

Metrics Calculation

HOG CPU and GPU

In [13]:

```
!cd tests/hog_tests && make
```

```
g++ -std=c++17 -I. -I/usr/include/opencv4 -c hog_descriptor_test.cpp -o hog_descriptor_te
st.o
g++ -o hog_descriptor_test hog_descriptor_cpu.o hog_descriptor_test.o hog_descriptor.o -L
/usr/lib -L/usr/local/cuda/lib64 -lopencv_core -lopencv_imgproc -lopencv_highgui -lopencv
_imgcodecs -lcudart
```

In [14]:

```
!./tests/hog_tests/hog_descriptor_test tests/hog_tests/test.png
```

```
HOG Descriptor Test Results:
GPU Version Time: 8.342 ms
CPU Version Time: 125.821 ms
```

FOURIER CPU and GPU

In [17]:

```
!cd tests/fourier_tests && make
```

```
g++ -std=c++17 -I. -I/usr/include/opencv4 -I/usr/local/cuda/include -c fourier_descriptor
_test.cpp -o fourier_descriptor_test.o
g++ -std=c++17 -I. -I/usr/include/opencv4 -I/usr/local/cuda/include -c fourier_descriptor
_cpu.cpp -o fourier_descriptor_cpu.o
nvcc -std=c++17 -I. -I/usr/include/opencv4 -diag-suppress=611 -c fourier_descriptor.cu -o
fourier_descriptor.o
g++ -o fourier_descriptor_test fourier_descriptor_test.o fourier_descriptor_cpu.o fourier
_descriptor.o -L/usr/lib -L/usr/local/cuda/lib64 -lopencv_core -lopencv_imgproc -lopencv_
highgui -lopencv_imgcodecs -lcudart -lcufft
```

In [18]:

```
!./tests/fourier_tests/fourier_descriptor_test tests/bounding_images/binary_bounding_1.p
ng
```

```
Fourier Descriptor Test Results:
CPU Version Time: 8.25898 ms
GPU Version Time: 0.216808 ms
```

GMM CPU and GPU

In [19]:

```
!cd tests/gmm_tests && make
```

```
nvcc -std=c++17 -I. -I/usr/include/opencv4 -diag-suppress=611 -c gmm_test.cu -o gmm_test.
o
```

```
g++ -o gmm_test gmm_cpu.o gmm_test.o -L/usr/lib -L/usr/local/cuda/lib64 -lopencv_core -lo
pencv_imgproc -lopencv_highgui -lopencv_imgcodecs -lcudart -lcufft
```

In [ ]:

```
!./tests/gmm_tests/gmm_test tests/gmm_tests/test_imgs/1x
```

```
GMM Test Results:
CPU Version Time: 45848 ms
```

In [ ]:

```
!./tests/gmm_tests/gmm_test tests/gmm_tests/test_imgs/2x
```

```
GMM Test Results:
CPU Version Time: 168161 ms
```

In [ ]:

```
!./tests/gmm_tests/gmm_test tests/gmm_tests/test_imgs/3x
```

```
GMM Test Results:
CPU Version Time: 374805 ms
```

In [ ]:

```
!./tests/gmm_tests/gmm_test tests/gmm_tests/test_imgs/4x
```

```
GMM Test Results:
CPU Version Time: 669370 ms
```

If any nvidia issues:

In [20]:

```
!nvidia-smi
```

```
Mon Dec  9 18:19:37 2024
+---------------------------------------------------------------------------------------
-+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------
-+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC
|
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M.
|
|=========================================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0
|
| N/A   41C    P8              11W /  70W |      0MiB / 15360MiB |      0%      Default
|
|                                         |                      |                  N/A
|
+-----------------------------------------+----------------------+----------------------
-+


+---------------------------------------------------------------------------------------
-+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory
|
|        ID   ID                                                             Usage
|
|=======================================================================================|
|  No running processes found
|
+-----------------------------------------------------------------------------------------
-+
```

Convert images to .gif

In [ ]:

```
def create_gif_with_ffmpeg(image_folder, output_name, fps):
    # Sort images alphabetically
    images = sorted([os.path.join(image_folder, f) for f in os.listdir(image_folder) if
f.endswith(('png', 'jpg', 'jpeg'))])
    if not images:
        print(f"No images found in the folder: {image_folder}")
        return

    # Load images into a list
    frames = []
    for image_path in images:
        frames.append(imageio.imread(image_path))   # Use imageio.v2.imread explicitly

    # Output GIF path
    output_path = os.path.join(f"{output_name}_{fps}fps.gif")
    # Write GIF using ffmpeg
    imageio.mimsave(output_path, frames, format="GIF", fps=fps)
    print(f"GIF saved at {output_path}")
```

In [ ]:

```
# If enabled in src folder only
# image_folder = "./src/temp/labeled_output/"
# # If colab, use the following path instead
# # image_folder = "/content/src/temp/labeled_output/"

# # Target size for resizing (optional, adjust as needed)
# resize_to = (80, 60)   # Width, Height. Set to None to skip resizing.

# # Create GIFs for different frame rates
# for fps in [30, 60, 90]:
#     create_gif_with_ffmpeg(image_folder, "animated", fps, resize_to=resize_to)
```

Run main code

In [24]:

```
!cd src/ && make
```

```
g++ -std=c++17 -I/usr/include/opencv4 -I/usr/local/cuda/include -I../include -c main.cpp
-o build/main.o
g++ -o human_detection build/main.o build/fourier_descriptor.o build/gmm.o build/hog_desc
riptor.o -L/usr/lib -lopencv_core -lopencv_imgproc -lopencv_highgui -lopencv_imgcodecs -l
opencv_ml -lopencv_objdetect -lopencv_features2d -lopencv_flann -L/usr/local/cuda/lib64 -
lcudart -lcufft
```

In [ ]:

```
!cd src/ && ./human_detection
```