APPENDICES

APPENDIX A - Source Code

```
import cv2
from random import randrange
import os

# Define media and algorithm paths (adjust based on the online compiler's file upload
mechanism)
media_path = "./media"  # Adjust this based on where the media files are uploaded
algorithm_path = "./algorithm"  # Adjust based on where the classifier XML is uploaded

# Load the car image (ensure this file exists in the 'media' folder)
car_img = cv2.imread(os.path.join(media_path, "car_image.jpg"))
if car_img is None:
    print("Error: Image file not found.")
    exit()

# Convert the image to grayscale
grayscaled_car_img = cv2.cvtColor(car_img, cv2.COLOR_BGR2GRAY)

# Load the pre-trained car classifier (ensure the classifier file exists in the 'algorithm'
folder)
car_tracker = cv2.CascadeClassifier(os.path.join(algorithm_path, "car_detection.xml"))
if car_tracker.empty():
    print("Error: Cascade Classifier file not found.")
    exit()
```

```
# Detect cars in the image
car_coordinates = car_tracker.detectMultiScale(grayscaled_car_img)

# Draw rectangles around detected cars
for (x, y, w, h) in car_coordinates:
    cv2.rectangle(car_img, (x, y), (x + w, y + h), (randrange(256), randrange(256),
randrange(256)), 2)

# Save the result as an image (since online compilers might not support GUI windows)
output_image_path = "detected_cars_image.jpg"
cv2.imwrite(output_image_path, car_img)
print(f"Processed image saved as {output_image_path}")

# --- Car and Pedestrian Detection from Video ---

# Load the video file (ensure this video file exists in 'media')
sample_video = cv2.VideoCapture(os.path.join(media_path, "tesla_dashcam.mp4"))
if not sample_video.isOpened():
    print("Error: Cannot open video file.")
    exit()

# Prepare the output video writer
frame_width = int(sample_video.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(sample_video.get(cv2.CAP_PROP_FRAME_HEIGHT))
output_video_path = "processed_video.avi"
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter(output_video_path, fourcc, 20.0, (frame_width, frame_height))

frame_count = 0
while True:
```

```
    successful_frame_read, frame = sample_video.read()
    if not successful_frame_read:
        break  # End of video

    # Convert to grayscale for detection
    grayscaled_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Detect cars in the current frame
    car_coordinates = car_tracker.detectMultiScale(grayscaled_frame)

    # Draw rectangles around detected cars
    for (x, y, w, h) in car_coordinates:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (randrange(256), randrange(256),
randrange(256)), 2)

    # Write the processed frame to the output video
    out.write(frame)

    # Optionally, you can break the loop after a few frames for testing purposes
    frame_count += 1
    if frame_count == 10: # Limiting to 10 frames for testing purposes
        break

# Release the video capture and writer objects
sample_video.release()
out.release()

print(f"\nProcessed video saved as {output_video_path}\n")
```

APPENDIX B - Screenshots