Report: Q4 – Face Blurring in Video Feeds

Author: Syed Rahmath Ullah Hussaini

Guide: Dr. Ahmed Rimaz Faizabadi

Institution: Delloyd R&D

---

1. Problem Statement

The goal of this project was to develop a real-time face blurring system for live video feeds from webcams or CCTV cameras. The system should:

1. **Capture live video feed.**
2. **Detect faces in each frame.**
3. **Apply blur to detected faces to protect privacy.**
4. **Display the processed feed.**
5. **Include functionality to record video clips on demand.**

---

2. Tools and Technologies Used

- **Python 3.x – Primary programming language.**
- **OpenCV – For video capture, image processing, and GUI display.**
- **MediaPipe – For high-accuracy real-time face detection.**
- **NumPy – For numerical operations and frame manipulation.**
- **Haar Cascade (OpenCV fallback) – Backup face detection method if MediaPipe fails.**

Optional features like overlays and textured blur were implemented using OpenCV drawing and filtering functions.

---

3. Implementation Details

3.1 Video Capture

- Captured live feed using cv2.VideoCapture(0) for webcam input.
- Supports dynamic adjustment of FPS, resolution, and recording parameters.

3.2 Face Detection

- Primary: MediaPipe Face Detection (min_detection_confidence=0.5) for accurate detection.
- Fallback: OpenCV Haar Cascade (haarcascade_frontalface_default.xml) if MediaPipe detects no faces.
- Multiple faces can be detected in a single frame.

3.3 Face Blurring

- Each detected face region is extracted and blurred using a Gaussian texture blur (cv2.GaussianBlur) with enhanced kernel size.
- Optional textured blur overlay adds a pixelated pattern for a professional effect.

3.4 Visual Enhancements

- Bounding Box: Pink rectangle drawn around detected faces for better visualization.
- FPS Counter: Live frames-per-second displayed on the top-left corner.
- Face Count: Number of detected faces displayed dynamically.
- Recording Indicator: Red “REC” circle shown when recording is active.

3.5 Recording Functionality

- Press r to start/stop recording.
- Video saved as blurred_faces_output.avi with all overlays included.
- The recorded video preserves the live processed feed with blurred faces and annotations.

3.6 User Interaction

- q – Quit the application.
- r – Start/Stop recording.

- **Optional: Could add snapshot functionality (s) to save single frames.**

---

4. Challenges Faced

- **Ensuring real-time performance while applying high-quality blur.**
- **Maintaining face tracking accuracy during rapid movements.**
- **Synchronizing MediaPipe detection with OpenCV frame updates for smooth display.**
- **Handling variable lighting and occlusions to prevent detection failure.**

---

5. Results

- **Successfully detected and blurred faces in real-time from webcam feed.**
- **Multiple faces in a single frame were detected and blurred accurately.**
- **FPS counter averaged ~18–25 FPS depending on system load.**
- **Recording functionality worked seamlessly, producing a playable .avi video with all processed effects intact.**
- **Pink bounding boxes and textured blur enhanced visual clarity for demonstration.**

---

6. Conclusion

The project demonstrates a fully functional real-time face blurring system suitable for privacy-sensitive applications such as surveillance, video streaming, or public displays. The implementation:

- **Uses advanced computer vision techniques (MediaPipe + OpenCV).**
- **Provides live user feedback (FPS, face count, bounding boxes).**
- **Supports on-demand recording with all processed effects preserved.**

The system is robust, interactive, and visually informative, making it suitable for both academic demonstration and practical deployment.

---

7. Future Enhancements

- **Add snapshot capture for single frames.**
- **Improve blur with custom filters or mosaics.**
- **Add CCTV/RTSP support for networked camera feeds.**
- **Implement face tracking across frames to reduce flickering bounding boxes.**
- **Add GUI buttons for starting/stopping recording and taking snapshots.**

---

8. References

1. **OpenCV Documentation – https://opencv.org/**
2. **MediaPipe Face Detection – https://developers.google.com/mediapipe**
3. **Python NumPy Library – https://numpy.org/**

Author: Syed Rahmath Ullah Hussaini

Guide: Dr. Ahmed Rimaz Faizabadi

Institution: Delloyd R&D