WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI PRIVACY VIEWER – An AI-Powered Visual Privacy Protection System

Baby Kalpana Y[1],Dharun Vengadesh N[2], Dhiyaneshwar M[3], Divya Sri M[4], Elamparithi P[5]

[1]Professor, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, India.
[2]Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, India.
[3] Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, India.
[4] Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, India.
[5] Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, India.

ABSTRACT :

Personal and academic information is frequently viewed on laptop and desktop screens in classrooms, offices and public spaces. When another person looks at a screen without permission, confidential information may be exposed through shoulder surfing. This paper presents AI Privacy Viewer, an AI-based visual privacy protection system designed to use camera input and computer-vision techniques to support screen privacy. The proposed workflow identifies the authorized user, analyses facial and gaze information, and aims to keep the region being viewed clear while blurring other screen areas. An alert mechanism is also intended to notify the user when an unauthorized viewer is detected. The prototype uses a web interface with a Python-based backend and computer-vision tools such as OpenCV and MediaPipe. The current development focuses on connecting the user interface, camera access and privacy-processing workflow. Face and gaze detection reliability and end-to-end selective blurring require further verification under different lighting and viewing conditions. Therefore, this paper describes the system design, development approach, limitations and evaluation plan without making unsupported quantitative claims. Future work includes improving detection accuracy, adaptive privacy settings, event logging and performance.

Keywords: AI Privacy Viewer, Visual Privacy, Shoulder Surfing, Face Detection, Gaze Tracking, Computer Vision, Screen Blurring, Privacy Alert.

I. INTRODUCTION

Laptops and desktop computers are widely used for education, communication and professional tasks. These devices often display private information such as personal messages, academic records, project files and account details. In shared environments, another person may view this information by looking at the screen, a privacy risk commonly known as shoulder surfing. Traditional privacy methods depend on the user manually locking the device, hiding the screen or using a physical privacy filter. Such methods may not adapt to the user's viewing direction or respond immediately when another person approaches.

The problem addressed in this project is the risk of visual information exposure when a user works on a computer in a shared environment. Conventional screen protection has several limitations: it may require manual action, does not adapt to the user's gaze, cannot dynamically protect selected screen regions, and may not alert the user when someone else is viewing the display. A software-based solution using a camera and computer-vision methods may help provide an additional layer of visual privacy.

This paper presents **AI PRIVACY VIEWER** – An AI-Powered Visual Privacy Protection System. The main objectives are:

1. To design a visual privacy application that uses camera input and computer-vision techniques to support screen protection.
2. To explore face detection or recognition for distinguishing an authorized user from another viewer.
3. To use gaze information to identify the screen region that the authorized user is looking at, where reliable tracking is available.
4. To design a privacy display that keeps the selected viewing region clear and blurs other areas when the required processing is implemented.
5. To provide an alert mechanism for suspected unauthorized viewing and create a modular foundation for future improvements.

The system is organized around a user interface, camera input, computer-vision processing and a privacy display. The user opens the application and starts the camera through the interface. The processing

workflow is intended to analyse the available face and gaze information and determine the privacy response. If an unauthorized viewer is detected by a functioning detection module, the system can display an alert. The exact behaviour depends on the modules successfully integrated in the final prototype.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes the proposed methodology and architecture. Section IV discusses system implementation. Section V summarizes the advantages. Section VI describes development status, limitations and evaluation. Sections VII and VIII present the conclusion and future work.

II. LITERATURE SURVEY

Research on gaze estimation, face landmarks and visual privacy provides a foundation for the AI Privacy Viewer project. Existing work studies how gaze can be estimated from images, how facial landmarks can be detected, and how visual interfaces can protect information from nearby observers.

Gaze Estimation. Gaze estimation methods analyse eye and facial appearance to estimate the direction in which a person is looking. Deep-learning approaches have been reviewed and benchmarked for their accuracy and generalization across different conditions [1], [2]. These studies show that gaze estimation is an active computer-vision research area, but accuracy can vary with lighting, camera position and user movement.

Deep Learning for Gaze Analysis. Survey research summarizes methods for gaze analysis and discusses the challenges of applying them across users and environments [2]. For a privacy application, gaze estimation must be sufficiently stable to select a useful screen region without making the interface distracting.

Shoulder-Surfing Protection. Visual privacy research has explored methods for preventing nearby observers from reading sensitive content on mobile devices and other displays [3]. These approaches establish the need to consider not only digital access but also information visible to people physically near a device.

Gaze-Based Privacy. EyeSpot investigated using gaze to protect private text content on mobile devices from shoulder surfing [4]. Its focus is related to the present project because both approaches consider the user's viewing direction as part of visual privacy protection.

Face Landmarks and Camera Processing. MediaPipe Face Landmarker provides face landmark and facial information that can be used in computer-vision applications [5], [7]. Such tools can support the development of face and eye-related processing, although a landmark detector alone does not guarantee identity recognition or reliable gaze estimation.

Gaze Target Estimation. Recent work such as Gaze-LLE investigates gaze-target estimation using learned visual encoders [6]. This line of research illustrates the complexity of identifying where a person is looking and the importance of testing models in the intended operating environment.

Research Gap. The reviewed work covers gaze estimation, face landmark detection and shoulder-surfing protection. AI Privacy Viewer applies these concepts to a computer screen, aiming to combine camera-based user observation, selective screen privacy and alerts in one interface. The main practical challenges are reliable detection, low processing delay, correct screen-region selection and handling cases where the camera or model cannot provide valid results.

III. PROPOSED METHODOLOGY

AI Privacy Viewer follows a modular architecture in which the frontend provides the controls and visual feedback, the camera supplies frames, computer-vision components process face and gaze information, and the privacy layer applies the intended screen response. A Python backend can coordinate processing and application status. The system must clearly distinguish a successfully detected user or gaze position from an unavailable or uncertain result.

System Architecture. The proposed architecture contains four logical layers. The presentation layer provides the interface, camera status and privacy controls. The application layer handles start and stop requests and coordinates processing. The computer-vision layer uses available face-landmark and gaze-estimation methods. The privacy-response layer is intended to display a focused region, blur other regions and show an alert when the relevant detection is available. The actual capabilities depend on successful integration and testing of each module.

Workflow. The user opens the application and grants camera permission. The user starts the privacy process through the interface. Camera frames are supplied to the computer-vision component, which attempts to locate the face and estimate viewing direction. If a reliable gaze region is available, the privacy layer can use it to determine which screen area should remain visible. When unauthorized viewing is detected by an implemented detector, the interface can show an alert or sound a warning. If camera access or detection fails, the application should communicate the status instead of implying that protection is active.

Camera Input and Face Processing. The camera module requests permission and captures frames from the user's webcam. Face-processing methods may locate facial landmarks and support user identification when an appropriate recognition component is available. Face landmark detection and identity recognition are different tasks; the implementation must not treat a detected face as proof that the person is authorized.

Gaze Tracking. The gaze component analyses facial or eye-related information to estimate the user's viewing direction. The estimated direction may be mapped to a region of the display. Accuracy can be affected by camera placement, lighting, glasses, head movement and model limitations, so the region should be tested before it is used to control privacy.

Selective Screen Privacy. The intended privacy mechanism keeps the region of interest visible and blurs other screen areas. This requires a reliable mapping between gaze coordinates and the displayed content, together with a rendering method that can update the screen without excessive delay. A static blur demonstration alone does not prove that gaze-based selective privacy works end to end.

Logical Data Model. The prototype can maintain application state such as camera status, privacy mode, detection status, alert messages and optional event records. If event logging or user profiles are added, the data structure should define the fields and access rules clearly. Persistent database storage should only be described as implemented if a database is actually connected to the application.

Security and Validation. Camera permission must be requested through the browser or operating system. The application should handle denied permissions, unavailable devices, missing model files and invalid processing results. Alerts and privacy status should accurately reflect the current state. Any stored face data or event logs require appropriate consent, access restrictions and retention rules.

IV. SYSTEM IMPLEMENTATION

The project is structured as a web-based interface connected to a Python-oriented computer-vision workflow. The following modules describe the intended organization of the prototype and should be read alongside the features that have been verified in the actual implementation.

Development Environment. The project uses HTML, CSS and JavaScript for the user interface, Python for backend or processing tasks, and computer-vision libraries such as OpenCV and MediaPipe where configured. Development and debugging can be performed using Visual Studio Code and a modern browser. Camera access depends on browser permissions and the availability of the required model and library assets.

User Interface Module. The interface provides controls to start or stop the privacy process and displays camera or processing status. Labels and feedback messages should help users understand whether the camera is active, whether detection is available and whether privacy processing is running. The interface should not indicate successful protection when the required modules have failed.

Camera Access Module. The camera module uses the browser's media-device interface or a configured backend capture process to obtain frames. It must handle permission denial, a missing camera and capture errors. Camera access is an input step and does not, by itself, implement face recognition, gaze tracking or screen blurring.

Face and Gaze Processing Module. Face landmarks and gaze-related features are processed using the available computer-vision components. The module should return a valid detection state and, when possible, an estimated viewing region. Face recognition requires a separate identity-matching mechanism if the application must distinguish an authorized user from other people.

Privacy Display Module. This module represents the intended screen-protection behaviour: the selected viewing area remains clear while other areas are blurred. The rendering approach must be connected to the gaze result and updated as the user moves. If this integration is not operational, the interface should be reported as a prototype rather than a completed adaptive privacy system.

Backend Integration. A Python backend may coordinate processing requests, return the current privacy status and provide an alert endpoint. Frontend and backend communication should be checked for valid responses, error conditions and correct state transitions. An
endpoint that returns a sample status is not evidence that real-time AI detection has been integrated.

Event and Record Handling. The application may later record alert time, detection status and other relevant events for review. The current project should only claim persistent storage if an actual database or file-based logging mechanism is implemented. Any records should avoid storing unnecessary personal or biometric information.

Error Handling. The application should handle camera permission failures, missing model assets, unavailable libraries, backend errors, uncertain detections and network interruptions. Clear messages help the user identify whether the camera, processing component or privacy display is unavailable.

Testing Methodology. The following checks can be used to evaluate the prototype. They are test cases to perform and document, not a claim that every test has passed.

Open the application: the initial interface loads correctly.

Start the camera: permission is requested and the camera preview appears when access is granted.

Stop the camera: camera capture stops and the interface reflects the stopped state.

Face processing: a face is detected consistently under supported lighting conditions.

Gaze processing: the estimated viewing region responds appropriately to changes in gaze direction.

Privacy display: the selected region remains clear and other regions are blurred when the feature is integrated.

Unauthorized-viewer response: the alert appears only when the implemented detection logic reports a relevant event.

Missing model or library: the application displays an understandable error instead of silently failing.

Camera permission denied: the application explains that camera access is unavailable and does not claim active protection.

V. ADVANTAGES

1. **Visual Privacy Support**. The proposed system aims to reduce exposure of sensitive screen content when the user works in a shared environment.
2. **Gaze-Aware Display.** When reliable gaze tracking is available, the display can adapt the visible region to the user's viewing direction.
3. **Reduced Manual Effort**. Automated privacy responses may reduce the need for the user to manually hide the screen whenever another person approaches.
4. **Immediate Feedback.** Camera and privacy-status messages can help users understand whether the protection process is active.
5. **Modular Design**. Separating the interface, camera input, vision processing and privacy response makes the system easier to maintain and improve.
6. **Expandability**. Future modules can add configurable privacy levels, improved identity verification and event logging when required.
7. **Potential for Multiple Environments**. The concept may be useful in classrooms, offices, libraries and other shared spaces where screen confidentiality matters.
8. **Privacy-Focused Development**. Clear permission handling and limited retention of sensitive data can help guide responsible development. These are intended benefits; their effectiveness must be established through testing.

VI. RESULTS AND ANALYSIS

This section describes the prototype status and the evaluation needed to assess the AI Privacy Viewer. Reliable numerical performance results should only be reported after repeatable testing. The presence of a camera preview or interface alone does not demonstrate successful face recognition, gaze tracking or adaptive screen blurring.

9. **Development Status**. The project has an interface and a planned camera-based workflow for visual privacy. The camera preview, face-processing component, gaze tracking, selective blur and alert mechanism should each be verified separately. Any feature that is not consistently working must be identified as a limitation in the final report.
10. **Functional Analysis**. Functional testing should confirm that the interface loads, camera permissions are handled, start and stop controls behave correctly, and processing status is reported accurately. Face detection should be checked with different user positions and lighting conditions. Gaze-based privacy should only be considered functional when the visible region follows the user's gaze reliably.
11. **Detection Analysis**. Face and gaze processing should be tested with the authorized user, no face in view, multiple people in view and changes in head position. False detections and missed detections should be recorded. Face landmark detection alone should not be treated as successful identity recognition.
12. **Performance Analysis**. Processing delay should be measured from frame capture to detection and display update. Repeated tests can identify delays caused by camera capture, model inference, rendering or backend communication. Performance claims should be based on recorded measurements rather than visual impressions.
13. **Limitations.** (i) Face and gaze detection may be affected by lighting, camera angle and movement. (ii) Reliable gaze estimation requires suitable processing and calibration. (iii) Selective blurring depends on correctly mapping gaze coordinates to the screen. (iv) Camera permission and model-file availability can prevent operation.

(v) An alert is only dependable when the detection logic is implemented and validated. (vi) Privacy and identity data must be handled carefully. (vii) Overall effectiveness cannot be quantified without systematic tests.

14. **Proposed Evaluation Protocol.** Each component should be evaluated with repeatable test cases and documented observations.

Interface functionality: successful loading and correct operation of the start and stop controls.
Camera access: successful preview when permission is granted and correct feedback when denied.
Face detection: detection success and failure cases across different lighting and positions.
Gaze tracking: consistency of the estimated viewing region when the user looks at different screen areas.
Privacy display: correct clear and blurred regions during gaze changes.
Alert handling: correct alert behaviour for the cases supported by the implemented detection logic.
Error handling: appropriate response to missing libraries, model files, camera access or backend services.
Performance: measured delay and stability over repeated runs.

15. **Discussion**. AI Privacy Viewer addresses visual privacy by combining camera-based observation with a proposed gaze-aware screen response. Its practical value depends on reliable face and gaze processing and on the correct integration of the privacy display. A careful evaluation must distinguish a working user interface from a fully functioning visual privacy system. Testing under realistic

lighting, viewing distances and user movement is necessary before drawing conclusions about its effectiveness.

VII. CONCLUSION

This paper presented AI PRIVACY VIEWER – An AI-Powered Visual Privacy Protection System, a project designed to support screen privacy through camera input and computer-vision methods. The proposed workflow uses face-related processing and gaze estimation to identify the user's viewing region and aims to blur other parts of the display. An alert mechanism is also considered for unauthorized viewing.

The project provides a modular foundation for developing adaptive visual privacy on laptops and desktop computers. Its success depends on reliable detection, correct screen-region mapping, responsive rendering and clear error handling. Further testing is required to verify each feature and measure performance before the system can be considered a validated privacy solution.

VIII. FUTURE WORK

Improved Face Detection. Face processing can be strengthened by testing reliable models under different lighting, camera positions and user movements, with clear handling of uncertain results.

Accurate Gaze Estimation. Gaze estimation can be improved through calibration, more robust models and better mapping between eye direction and screen coordinates.

Adaptive Screen Blurring. Future versions can update the clear region smoothly and blur other areas with minimal delay, while avoiding distracting changes when gaze estimates fluctuate.

Unauthorized Viewer Detection. A more reliable method can be added to detect the presence of additional viewers and trigger alerts only when supported by validated detection logic.

Customizable Privacy Levels. Users could choose the strength of blurring, the size of the visible region and the response to uncertain gaze information.

Audio and Visual Alerts. Optional sound and on-screen notifications could inform users of potential privacy risks, with user-controlled alert settings.

Event Logging. A privacy-conscious log could record alert timestamps and system status without unnecessarily storing face images or biometric data.

Security and Consent. Future development should include clear camera-permission explanations, secure handling of any stored information and controls for disabling data collection.

Testing and Optimization. Systematic testing can measure detection accuracy, processing delay, stability and usability across different devices and environments before wider deployment.

REFERENCES :

1. Cheng, Y., Wang, H., Bao, Y., & Lu, F. (2024). Appearance-Based Gaze Estimation With Deep Learning: A Review and Benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence, 46(12), 7509–7528.
2. Ghosh, S., Dhall, A., Hayat, M., Knibbe, J., & Ji, Q. (2024). Automatic Gaze Analysis: A Survey of Deep Learning Based Approaches. IEEE Transactions on Pattern Analysis and Machine Intelligence, 46(1), 61–84.
3. Privacy Protection Against Shoulder Surfing in Mobile Environments. (2023). Lecture Notes in Computer Science. https://doi.org/10.1007/978-3-031-34776-4_8
4. EyeSpot: Leveraging Gaze to Protect Private Text Content on Mobile Devices from Shoulder Surfing. (2018). Multimodal Technologies and Interaction, 2(3), Article 45.
5. Google AI Edge. MediaPipe Face Landmarker Guide. https://developers.google.com/edge/mediapipe/solutions/vision/ face_landmarker
6. Ryan, F., Bati, A., Lee, S., Bolya, D., Hoffman, J., & Rehg, J. (2025). Gaze-LLE: Gaze Target Estimation via Large-Scale Learned Encoders. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 28874–28884.
7. Google AI Edge. MediaPipe Face Landmarker for Python. https://developers.google.com/edge/mediapipe/solutions/vision/ face_landmarker/python