EE6782

Assignment 2: JARVIS

Parth Bansal
21D170028

Contents

1 Introduction

The AI Guard Agent is a fully autonomous, offline security assistant designed to monitor and protect a hostel room. It integrates robust face recognition, speech-based command interpretation, and structured intruder handling to provide reliable real-time monitoring.

Key highlights:

- **Offline Operation:** No LLMs or internet dependency. Operates without calls to online models like Gemini.
- **Autonomous and Edge-Friendly:** Runs on standard laptops without GPU acceleration.
- **Multi-Modal Security:** Combines vision and audio cues for robust monitoring.
- **Structured Escalation:** Multi-tier intruder handling prevents false alarms and ensures security.
- **Logging and Robustness:** Snapshot capture, temporary unknown tracking, and thread-safe audio handling.

2 Milestone-Based System Architecture

2.1 Milestone 1: Wake-Word and Command Detection

- Continuously listens for wake words such as "Jarvis" like "Siri".
- Detects guard activation commands like "guard my room" or "start monitoring".
- Separation of wake-word and command detection improves reliability.
- Ensures standby mode until a valid activation is received.

2.2 Milestone 2: Face Detection and Recognition

- Detects faces and computes embeddings for identification.
- Matches detected faces with trusted user database.
- Tracks individuals across frames to maintain identity during occlusions.
- Flicker tolerance and re-greeting logic prevent repetitive prompts.
- Accurate even under sub-optimal lighting or partial visibility.

2.3 Milestone 3: Trusted User Interaction and Unknown Handling

- Trusted users are greeted politely and monitored without interruption.
- Unknown individuals trigger a structured escalation:
  1. **Yes/No Verification:** Confirms authorization.
  2. **Password Challenge:** Requests pre-defined passphrase.
  3. **Persistent Intruder Handling:** Snapshots, audible alerts, and logging.
- Reduces false alarms from transient detections.

2.4 Milestone 4: Logging and Robustness Enhancements

- **Automatic Snapshot Storage:** Persistent intruders are captured via `cv2.imwrite()` with timestamped filenames for audit and logging purposes.
- **Temporary Unknown Tracking:** Unknown individuals are temporarily stored with embeddings and frame counts to avoid repeated false alerts.
- **Robust Face Recognition:**
  - **Flicker Tolerance:** Faces are only acted upon after stable detection across multiple frames to prevent false positives.
  - **Centroid-Based Tracking:** Tracks are matched across frames using centroid distances, handling occlusions and temporary misdetections.
- **Keyword Robustness:** The system uses a carefully curated small vocabulary for wake words and guard commands to reduce accent related mis-recognition of words and ensure reliable offline recognition.
- **Thread-Safe Audio Handling:** Ensures TTS prompts do not overlap, using `queue.Queue()` and `threading.Thread()`.
- **CPU-Friendly Design:** All computations are lightweight, ensuring smooth real-time monitoring on edge devices without GPU.

3 Integration Challenges and Solutions

- **Real-Time Synchronization:** Ensured audio and visual modules run concurrently without delay using threading and non-blocking queues.
- **False Positives:** Flicker tolerance and temporary unknown enrollment reduced mislabeling.
- **Resource Constraints:** Optimized CPU-friendly models to run on standard laptops without GPU acceleration.

4 Ethical Considerations and Testing Results

- **Privacy:** Only stores snapshots for persistent intruders; trusted user data handled locally.
- **Transparency:** Audible alerts inform individuals when monitored.
- **Testing:** Successfully detected known users and managed unknown intruders in varied lighting and partial occlusion scenarios.

5 Instructions to Run the Code

1. Ensure Python libraries are correclty installed to prevent conflicts.
2. Add trusted users' face embeddings to the database.
3. System waits for wake word, then follows milestone-based workflow automatically.

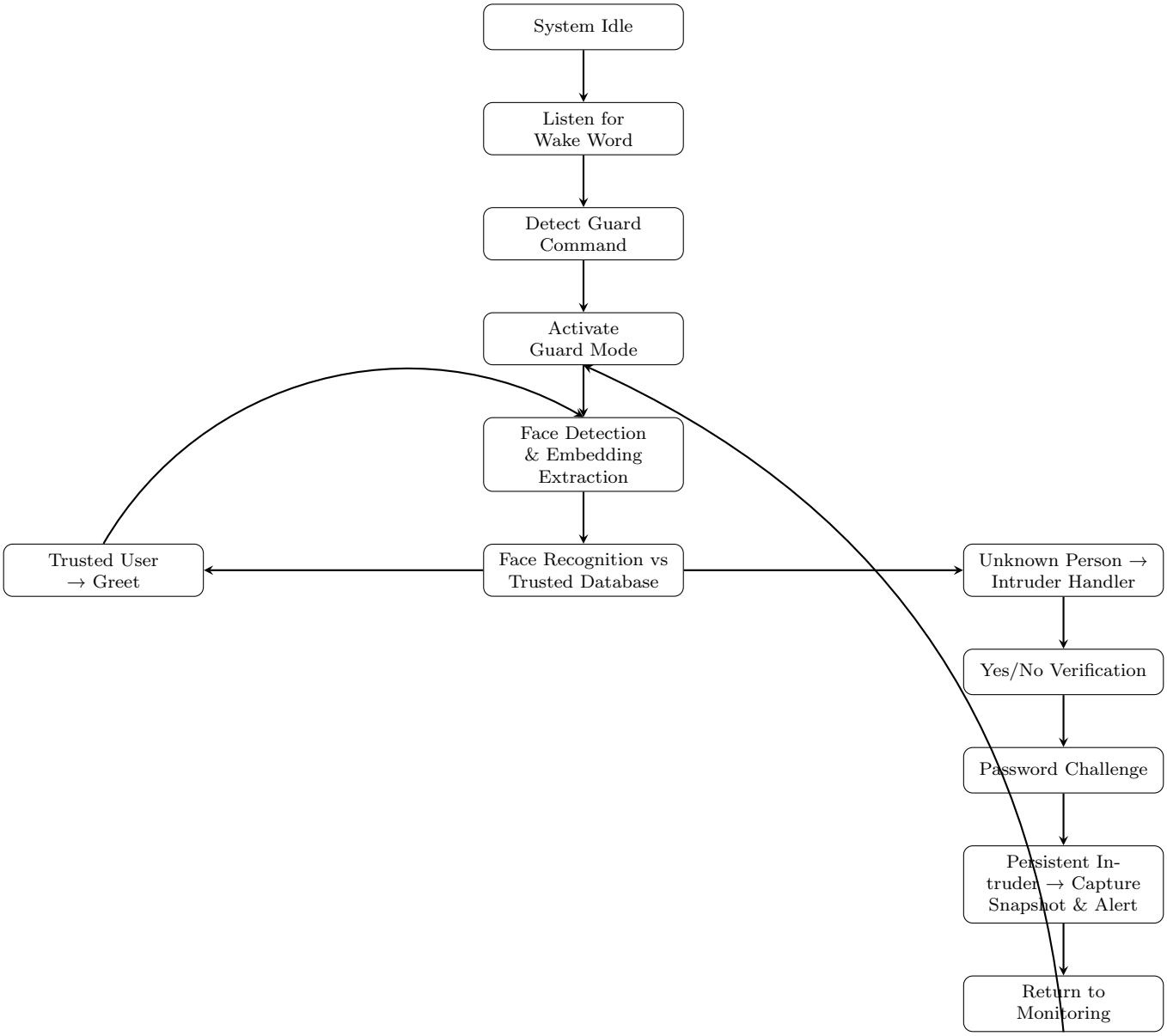

Figure 1: Milestone-Based Workflow of AI Guard Agent: Wake-Word Activation, Face Recognition, Intruder Handling, and Escalation.

6 Links and Resources

- Chatgpt was used in this assignment
- Code Repository: https://github.com/ParthBansal100/Jarvis-AI-Guard-Agent.git
- Video Link/Code Link: https://drive.google.com/drive/folders/1kP4iSDYZ05B5N1jiZtptpeQSHidKIhd7?usp=sharing