CovaCare (Computer Vision Accident Care) Project Requirements

Brydon Herauf (200454546) & David Kim (200405213)

Faculty of Engineering & Applied Science, University of Regina

ENSE 400

December 6, 2024

Table of Contents

1. Introduction

1.1. Document Purpose

The purpose of this document is to define the requirements for a computer vision-based accident detection system. It will provide a detailed description of the project's scope, purpose, functional and non-functional requirements, and interfaces. This document will be used by the developers of this project, as well as any stakeholders involved.

1.2. Executive Summary

This project focuses on the development of a computer vision-based accident detection system aimed at enhancing safety for the elderly and disabled within residential settings. This system will use advanced computer vision and machine learning technology to detect slips, falls, and prolonged inactivity in real time. Upon detection of an incident, the system will promptly notify designated emergency contacts, ensuring timely intervention and reducing the risk of severe injuries or fatalities.

1.3. Project Scope

This project is intended to be a general accident detection system for homes. Due to time constraints, we will be focussing on falls and prolonged inactivity. The capabilities of our system could be expanded in the future. This project will also include an interface that allows users to enter emergency contacts, receive alerts, and customize the model. Customization options will likely be in the form of turning on/off parts of the model within specific hours of the day (ex. Turning off inactivity detection during sleeping hours). Finally, this project will not be exploring the hardware infrastructure required to bring this system to life in an actual home (cameras and local computing). It will be more of a proof of concept, showing that we can detect accidents from live video input and send alerts in a timely manner.

1.4. References

Surveillance report on falls among older adults in Canada – Public Health Agency of Canada, 2020. (Link)

2. Overall Description

2.1. Problem Statement

According to the Public Health Agency of Canada, falls are the leading cause of injury-related hospitalizations and deaths among individuals aged 65 and older in Canada. Approximately 52% of these falls occur within household residences, and many incidents go unnoticed at first. If there were a way to quickly and reliably notify caregivers after an accident took place, people could get the timely help they need. Current solutions, such as wearable devices, suffer from limitations including user compliance and restricted functionality, highlighting the need for a more reliable and comprehensive monitoring system. We plan to build such a system, with fall and prolonged inactivity detection, as well as customization options that other computer vision-based solutions do not offer.

2.2. User Classes and Characteristics

2.2.1. Individuals Being Monitored

The first class of users for this system are the individuals who will be monitored by it. This will primarily be elderly and disabled people, but is not limited to any particular group. In many cases, these users will simply be monitored by our system, and will not have to interact with it at all. If desired, these individuals will be able to configure things in the application.

2.2.2. Caregivers/Emergency Contacts

The second class of users for this system are the caregivers, and/or people receiving the emergency alerts. These users will likely require full permission for configuring the model and setting up contact information. They will also need to be prepared for alerts if they come in.

2.3. Operating Environment

Our system is designed to operate in homes, but could also be installed in facilities or other locations where people are often alone. We plan to design our system in a way that can run locally, so no private video ever leaves a personal network. Cameras would need to be installed to cover all rooms and angles that a person would like to monitor. Those cameras could communicate over Wi-Fi or some other network to a local computing device running our incident detection algorithms. This local computing device would require hardware capable of processing video in real time. Finally, this device would need to be connected to the Internet, so that when an incident is detected, alerts could be sent to the smart phones of emergency contacts.

2.4. Dependencies

2.4.1. Hardware Dependencies

- Multiple Wi-Fi cameras capable of recording at approximately 30 fps and 720p resolution. For our demo, we will only need one camera.
- A local computing device with sufficient processing power to handle real-time video processing and analysis (exact specifications to be determined). For our demo, we will probably just use a virtual server running on a laptop.

2.4.2. Software Dependencies

- Python
- React Native (Expo) for developing the mobile application.
- OpenCV for video capture and frame preprocessing.
- TensorFlow for machine learning.
- MoveNet, MediaPipe, or another model for pose estimation.
- YOLO for object detection if necessary.
- Numpy and Pandas for data manipulation and preprocessing.
- Flask for API development.
- UI libraries such as Bootstrap (if necessary).
- Conda for managing project dependencies and environments.
- An SMS API such as Twilio for sending alerts to emergency contacts.

2.4.3. Data Dependencies

- Pre-trained pose estimation models (e.g., MoveNet or MediaPipe).
- Fall datasets for training an LSTM on pose-estimated fall sequences (URFD, GMDCSA24, etc.).

2.4.4. Network Dependencies

- A stable Wi-Fi network for communication between components (cameras, computing devices, and mobile app).
- Internet access for sending alerts.

3. System Features

3.1. Slip, Trip, and Fall Detection

3.1.1. Functional Requirements

- The system shall distinguish between falls and normal activities.
- The system shall process video feeds from multiple cameras simultaneously.
- The system shall perform this detection in real-time or near real-time.
- The detection algorithm should have some resistance to changes in lighting, resolution, and frame rate.

3.1.2. Technical Details

Slip, trip, and fall detection is the primary safety mechanism that our system supports. Our intended approach is as follows:

1. Process live camera data using OpenCV to capture frames.
2. On each frame, extract key features by either:
    a. Running the image through a fast pose estimation model such as Mediapipe or MoveNet to extract body position data.
    b. Running the image through a CNN, trained to extract key features important to fall detection.
3. Using the key extracted input from the previous step, run a sequence of frames (ex. 30) through an LSTM trained to detect fall sequences.

Frame rate will be standardized so we are always processing the same duration of time. Sequence length will be long enough to capture most, if not all, full falls. The LSTM will be trained on a variety of data to improve robustness.

If this approach fails, we can employ a simpler solution, by inspecting the orientation of the body, and seeing if it rapidly changes from vertical to horizontal, suggesting a potential fall. This solution will likely not be as robust, and more prone to false-positives.

3.2. Prolonged Inactivity Detection

3.2.1. Functional Requirements

- The system shall distinguish minimal movement and no movement.
- The system shall keep track of how long an individual has not moved, and flag it as concerning if it has exceeded a certain threshold.
- The system shall process video feeds from multiple cameras simultaneously.
- The system shall perform this detection in real-time or near real-time.
- The detection algorithm should have some resistance to changes in lighting, resolution, and frame rate.

3.2.2. Technical Details

Prolonged inactivity detection is a feature of our system that may detect other emergency situations, and catch missed falls. Our intended approach is as follows:

1. Process live camera data using OpenCV to capture frames.
2. On each frame, extract body position by either:
    c. Running the image through a fast pose estimation model such as Mediapipe or MoveNet to extract exact body position data.
    d. Running the image through a CNN, trained to approximate body position.
3. Using the key extracted input from the previous step, compare this frame’s body position with last frame’s body position and calculate the difference.
4. If the difference calculation is lower than a certain threshold, increase the time passed since last movement.

5. Once the time since the last movement exceeds a certain duration, flag this as prolonged inactivity.

3.3. Alerting

3.3.1. Functional Requirements

- The system shall send alerts to designated emergency contacts via SMS.
- Alerts should include information on type of incident, time of incident, location (if applicable), and an image (if explicitly allowed by a monitored individual).
- The system shall ensure alerts get to the emergency contact by re-sending until it successfully goes through.

3.3.2. Technical Details

SMS notifications will be sent from a local computing device which is running our machine learning algorithms. We will likely utilize an API like Twilio to send alerts. The computing device will have to be connected to the Internet for this purpose alone. Information from the detected incident will be captured and sent along with the message. If possible, the system will wait for message acknowledgement, but this may not be feasible.

3.4. Emergency Contact Configuration

3.4.1. Functional Requirements

- The system shall allow users to add, edit, and remove emergency contacts through a user interface.
- Users should be able to set emergency contacts to active or inactive.
- Users should be able to send a test alert to a specified contact.

3.4.2. Technical Details

Emergency contact information will be stored locally on the device that is processing video and sending alerts. Users will be able to configure contacts through a React Native mobile application that connects to the local computing device through an API while on the same network.

3.5. Model Customization

3.5.1. Functional Requirements

- The system shall allow users to turn on/off fall detection and prolonged inactivity detection during specific hours of the day.
- The system shall allow users to enable/disable fall detection and prolonged inactivity detection for specific cameras.
- The system shall allow users to configure sensitivity and time thresholds for prolonged inactivity detection.

3.5.2. Technical Details

System settings will be stored locally on the device that is processing video and sending alerts. Users will be able to configure these settings through a React Native mobile application that connects to the local computing device through an API while on the same network.

3.6. Camera Configuration

3.6.1. Functional Requirements

- The system shall allow users to add, edit, and remove cameras.
- The system shall allow users to view camera feeds to ensure they are working and capturing the full room.

- The system shall allow users to name and set cameras to active/inactive.

3.6.2. Technical Details

Camera details will be accessible through a React Native mobile application that connects to the local computing device through an API. The entire application will be strictly limited to devices on the same password protected network, so there is no risk of unauthorized individuals viewing the live feeds. Camera settings will be stored locally on the server. Camera detection will either be done automatically, or by entering specific device IPs.

4. External Interface Requirements

4.1. Users

- The system shall provide a mobile application for users to interact with.
- The application will allow users to configure emergency contacts, cameras, and model parameters.
- The application will only be accessible when on the same network as the server that is hosting it.
- The application shall feature a simple and intuitive modern design.
- The mobile application shall run seamlessly on both iOS and Android devices.

4.2. Cameras

- The system shall integrate well with various types of cameras, so long as they support a minimum resolution and frame rate.
- Cameras on the same network shall be able to communicate directly with the server by sending video data over Wi-Fi.
- The system will either automatically detect cameras, or allow users to set them up by specifying an IP address or other necessary information.

- Communication with cameras should be implemented in a secure way.

4.3. Notifications

- The system shall be capable of sending SMS notifications to both iOS and Android devices.
- The system must integrate with an API like Twilio, and sends alerts over the Internet.
- The alerts shall be sent in a concise, readable format.

4.4. External Machine Learning Models

- The system shall integrate with pre-trained machine learning for tasks such as pose-estimation.
- Any pre-trained model should be downloaded and stored directly on the device, so that there is no need to connect with this model over the Internet.
- The system will need to utilize efficient models that can run in real-time or near real-time.

5. Non-Functional Requirements

5.1. Performance

Performance is an important part of this system. It must process video frames in real-time or near real-time to ensure timely incident detection and response. Accidents should be detected, with alerts sent out, within one minute. If necessary, frame rate and/or resolution of the input can be reduced to improve response time. Frames should be dropped sparingly, if at all, to ensure no incidents are missed.

5.2. Reliability

Reliability is essential in any safety-critical system. Although our solution is designed to complement, not replace, other monitoring methods, maintaining dependability is still a very high priority. False-positives, and especially false-negatives should be minimized. Our goal is to have a system uptime of at least 99%, and a false-negative rate below 5%. Most missed falls should be caught by prolonged inactivity detection.

5.3. Privacy

To protect user privacy, all video processing must be done locally. Any image or video data leaving the home through alert messages will be done with explicit permission from the individual. Video processing, and access to the application, will be entirely dependent on a local computing device hosting our system. The only link to external networks will be for sending out alerts over the Internet.

5.4. Usability

The system will feature a simple, user-friendly interface for configuring emergency contacts, setting up cameras, and adjusting model settings. The application does not require an abundance of features, so usability can be made a priority. An API hosted by the local computing device should be accessible from a mobile application while on the same network as the system.

5.5. Security

Security is an important aspect of this system, as it is responsible for processing sensitive video data. The entire system shall run on a general or isolated local Wi-Fi network. Any video or API access will be strictly limited to devices on the same network. We will employ industry standards for ensuring secure API communication and camera data transmission.

5.6. Compatibility

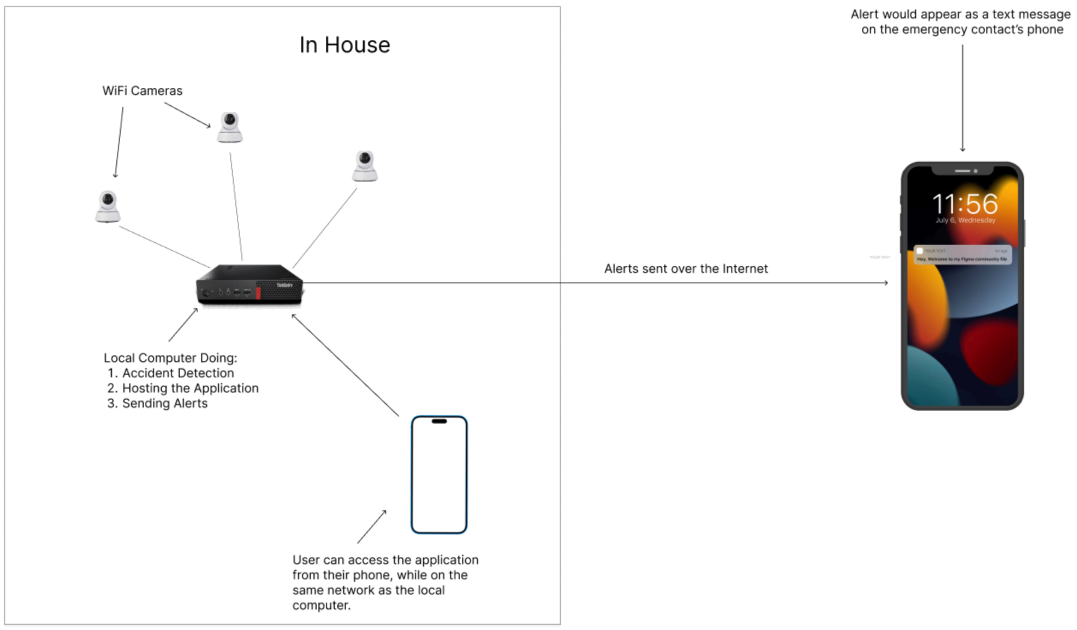
Compatibility is not as important as some of the other non-functional requirements of this system, since it would likely be sold along with specialized hardware. However, our mobile application should still work on both iOS and Android devices, and the system should be able to integrate with a variety of camera types.

5.7. Maintainability

This system will be designed in a modular fashion, adhering to agile principles. The code will be well written and structured so that it is easy to locate bugs, improve the model, and expand functionality in the future.

6. Appendix

A simple diagram of our system architecture:


In House
WiFi Cameras
Local Computer Doing:
1. Accident Detection
2. Hosting the Application
3. Sending Alerts
User can access the application from their phone, while on the same network as the local computer.
Alerts sent over the Internet
Alert would appear as a text message on the emergency contact's phone
11:56
July 6, Wednesday