|  |  |  |
| --- | --- | --- |
| **Fall Detection System** | | |
| **Team Information** | |  | | --- | | **Gavin Burris,** [**gbb123@live.missouristate.edu**](mailto:gbb123@live.missouristate.edu) | | **Kyle Lepine,** [**kml2008@live.missouristate.edu**](mailto:kml2008@live.missouristate.edu) | | **Logan Kilfoy,** [**lck1699@live.missouristate.edu**](mailto:lck1699@live.missouristate.edu) | | **Logan Koch,** [**koch170@live.missouristate.edu**](mailto:koch170@live.missouristate.edu) | | **Mason Sanborn,** [**sa99@live.missouristate.edu**](mailto:sa99@live.missouristate.edu) | | |
| **Document Type: Software Requirement Specification v1.0** | | ***Last updated on*: 11-24-2020** |
|  |  |  |

**Document Approval**

**This Software Requirements Specification has been accepted and approved by the following stakeholders:**

|  |  |  |
| --- | --- | --- |
| **Printed Name** | **Title** | **Signature and Date** |
| **S. CHAUDHARI** | **Stakeholder** |  |
| **R. IQBAL** | **Customer/Course Instructor** |  |

**Table of Contents**

**1. Introduction………………………………...……….……………………………………….1**

1.1 Purpose........................................................................................................................................... 1

1.2 Scope............................................................................................................................................... 1

1.3 Definitions, Acronyms, and Abbreviations....................................................................................... 1

1.4 References....................................................................................................................................... 1

1.5 Overview.......................................................................................................................................... 1

**2. General Description................................................................................................... 2**

2.1 Product Perspective.......................................................................................................................... 2

2.2 Product Functions............................................................................................................................. 2

2.3 User Characteristics......................................................................................................................... 2

2.4 General Constraints.......................................................................................................................... 2

2.5 Assumptions and Dependencies...................................................................................................... 3

**3. Specific Requirements.............................................................................................. 3**

3.1 Functional Requirements................................................................................................................. 3

3.2 Non-Functional Requirements......................................................................................................... 5

3.3 Design Constraints........................................................................................................................... 7

3.4 External Interface Requirements...................................................................................................... 7

3.5 Logical Database Requirements...................................................................................................... 9

3.6 Other Requirements......................................................................................................................... 9

**4. Change Management Process.................................................................................. 9**

**5. Appendencies.......................................................................................................... 10**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to provide an architectural layout for a fall detection system that accurately detects falls in an in-home environment. This project’s primary focus will be to create a system that receives video input and accurately detects falls. This project is useful for individuals who are prone to falls and require in-home fall detection. This document is intended as a reference point for the development team to use while developing the software. This SRS is also an agreement between the stakeholders and the development team that this software meets the expectations of the stakeholders.

**1.2 Scope**

The Fall Detection System will be a system that consumers can use to detect falls in the environment where the system is placed. The system will be designed to operate in an in-home environment that may or may not include obfuscated views.

The Fall Detection System provides the benefit of fall detection. With fall detection, the system will know when a consumer has fallen and send out a boolean value, true or false. The Fall Detection System will provide the alert within two seconds after the fall has occurred. This system will provide the customer with the assurance that a reliable system will be able to send an alert when a fall has occurred. This system will work to eliminate the fear and dangers of falling scenarios.

**1.3 Definitions, Acronyms, and Abbreviations**

1. SRS - Software Requirement Specifications.
2. CLI - Command Line Interface
3. GUI - Graphical User Interface
4. USB - Universal Serial Bus
5. Stakeholder - Any person with interest in the project who is not a developer.
6. User - The person, or persons, who operate or interact directly with the software.
7. Scrum Master - An ordinary member of the development team who has been elected for a given time period to represent the team members. The scrum master will have the added responsibility of communicating to the stakeholders on behalf of the development team.
8. Customer - The person or persons who pay for the product and usually decide the requirements.
9. Environments - The surroundings or conditions in which the subject is in.
10. Partially Obstructed - When the subject is less than 50% in view.

**1.4 References**

“Circle of Care Helps Clients Build Strength to Reduce Falls.” *Circle of Care*, 12 June 2019, [www.circleofcare.com/circle-of-care-helps-clients-build-strength-to-reduce-falls/](http://www.circleofcare.com/circle-of-care-helps-clients-build-strength-to-reduce-falls/).

“LiveCV” <http://livecv.dinusv.com/>

“Deep Learning Haar Cascade Explained” <http://www.willberger.org/cascade-haar-explained/#:~:text=Haar%20Cascade%20is%20a%20machine,of%20Simple%20Features%22%20in%202001>.

**1.5 Overview**

Descriptions of all functionalities, user characteristics, constraints, and dependencies in their entirety will be examined in the following pages of this SRS document. All information regarding the software for the Fall Detection System will be discussed appropriately. After the general description of the aspects involved in the Fall Detection System, there will be a complete listing of all requirements divided into their appropriate groupings, such as functional requirements, non-functional requirements, design constraints, external interface requirements, and logical database requirements. Following the exhaustive list of requirements will be a description of the change management process, which discusses who can submit changes to this SRS document and how these changes will be implemented. All appendices are listed at the end of this SRS document.

**2. General Description**

**2.1 Product Perspective**

The product that we will be producing will consistently and accurately detect when a person falls by means of machine learning and templating techniques. Using machine learning algorithms, the software will go through a training phase by learning what to look for and be able to differentiate a person who is acting normally versus a person who has fallen down. Pre-made templates will help with the fall detection by comparing the subject to a template database. Other fall detection products that are already in use can be divided into two categories, sensor-based detection, and vision-based detection. With our machine learning algorithm, we hope to follow more along the side of vision-based detection products by creating a program that works efficiently and accurately to detect and respond to people falling.

**2.2 Product Functions**

The Fall Detection System will be capable of many functionalities involving the detection of a fall. Firstly, the system must be capable of classifying objects from a live video feed as “human.” The system must also be capable of distinguishing between the different states humans might be in while in the line of sight of the webcam, such as lying down, sitting, and standing up. Finally, the system must be capable of detecting when a human falls. This should be achievable in various lighting conditions and environments, as well as when the line of sight of the webcam is partially obstructed.

**2.3 User Characteristics**

The user interface of the Fall Detection System will be CLI-based, with some aspects of the system represented in a GUI format. The user will be able to view the live video feed of the webcam on a window panel displayed on the monitor the system is running on. If the system detects a fall, the system will output “Fall detected.” in the command prompt.

**2.4 General Constraints**

The Fall Detection System will require an RGB USB webcam to obtain the required information to determine falls. The system must also require Python to run. The system must support the Windows operating system.

**2.5 Assumptions and Dependencies**

The Fall Detection System assumes that the user will have access to a computer with enough processing power to take in a constant stream of video from a webcam and perform fall detection techniques throughout the stream.

**3. Specific Requirements**

**3.1 Functional Requirements**

**FR.1 The system must detect and classify a human as a “human” object.**

*Source:* Project Description.

*Priority:*

*Introduction:* To design a system that can accurately detect if a human falls, the system must be capable of detecting humans that appear within its line of sight.

*Inputs:* A live video feed.

*Processing:* The system, when given a live video feed, will process each frame, identifying and classifying a human as a “human” object when it appears visually on screen.

*Outputs:* A list of indices that exist in the source image’s array of pixels, which contain pixel values shaping the “human” object.

*Error Handling:* This will be the default state of the system that will be reverted to should there be a processing error or should the program become stuck in a loop.

**FR.2 The system must alert the user when an object classified as a “human” falls in an indoor environment.**

*Source:* Customer requirement.

*Priority:*

*Introduction:* The system must support unique, indoor home environments. Having only one layout that the system can detect falls in would not be beneficial to future users of the system; therefore, the system should be adaptable and support users in various environments.

*Inputs:* Video feed from the camera.

*Processing:* The system, when given a live video feed, will process each frame, identifying changes and alerting the user if a fall is detected.

*Outputs:* An alert is made when a fall is detected in the live video feed.

*Error Handling:* A fall would not be detected.

**FR.3 The system must detect when an object classified as “human” falls when an object is obstructing them.**

*Source:* Customer requirement

*Priority:*

*Introduction:* It is implausible to expect that all users using the Fall Detection System will always be in an empty room by themselves. Instead, the system must be capable of detecting a fall with confidence when a “human” object falls, and an object obstructs the line of sight between the “human” object and the camera lens in use.

*Inputs:* Video feed from the camera.

*Processing:* The system, when given a live video feed, will process each frame, identifying changes and alerting the user if a fall is detected.

*Outputs:* An alert is made when a fall is detected in the live video feed.

*Error Handling:* A fall would not be detected.

**FR.4 The system must determine if an object classified as “human” is lying or sitting down.**

*Source:* Customer requirement.

*Priority:*

*Introduction:* To accurately detect a human fall, the software must be able to identify and differentiate the human form in different states. The system must not assume that a “human” object in a sitting or lying down position had previously fallen to arrive in this state.

*Inputs:* A pixel array containing a “human” object.

*Processing:* The system must label the “human” object that has been detected on screen in either a lying down or sitting down state.

*Outputs:* The “human” object will be classified into an appropriate state in which the system can handle their behavior accordingly.

*Error Handling:* N/A.

**FR.5 The system must be able to differentiate between when a person falls down and when they go to sit or lay down of their own will.**

*Source:* Customer requirement.

*Priority:*

*Introduction:* To accurately detect a human fall, the system must be able to tell the difference between when a person chooses to go lay down or sit down, and when they fall down by accident.

*Inputs:* Live video feed from camera.

*Processing:* The system must label the “human” object that has been detected on screen in either a lying down or sitting down state if the object assumes one of those positions naturally-- not through a fall.

*Outputs:* The “human” object will be classified into an appropriate state in which the system can handle their behavior accordingly.

*Error Handling:* N/A.

**FR.6 The system must access the live video feed.**

*Source:* Project description.

*Priority:*

*Introduction:* For the system to be able to record a fall, the video feed must be accessible by the machine running the software from the camera.

*Inputs:* Video feed from the camera.

*Processing:* The system must take the live video feed and display it in a window viewable by the user.

*Outputs:* Live video feed displayed in the assigned window.

*Error Handling:* The system will shut down, stop all processes, and file an error report.

**FR.7 The system must eliminate background noise of inanimate objects and non-human animated objects.**

*Source:* Project description.

*Priority:*

*Introduction:* For the system to be able to focus in on each person in the frame, the system should eliminate (blacken) all background noise that could interfere with the system's ability to track a person. This elimination should include everything that is not moving and anything that is moving that is not human.

*Inputs:* Video feed from the camera.

*Processing:* The system will blacken all background noise around each subject during each frame.

*Outputs:* Edited video feed to the screen.

*Error Handling:* The system will shut down, stop all processes, and file an error report.

**FR.8 The system must use a relational database to query templates.**

*Source:* Project description.

*Priority:*

*Introduction:* In order to compare the human form to templates, a place to store and access the templates is a necessity.

*Inputs:* A request for a .png image file.

*Processing:* The system will connect to the database and search for the image.

*Outputs:* The specified .png image file.

*Error Handling:* The system will disconnect from the database, close all running processes, and display an appropriate error message.

**3.2 Non-Functional Requirements**

**NFR.1 The system must detect when an object classified as a “human” falls within two seconds of the fall occurring.**

*Source:* FR.1, FR.2

*Priority:*

*Introduction:* For the alert, the system provides when detecting a fall successfully to positively affect the response time of emergency services, the system must detect falls fluidly as they happen.

*Inputs:* Video feed from the camera.

*Processing:* The system must be able to detect the fall, process the incident, and respond to it within two seconds.

*Outputs:* Live video feed to monitor.

*Error Handling:* A fall would not be detected.

**NFR.2 The system must detect when an object classified as a “human” falls between a distance of 5 feet and 50 feet inclusively from the viewing lens of the camera.**

*Source:* FR.1, FR.2, FR.6

*Priority:*

*Introduction:* In order to detect a human fall, the camera must have a full view of the subject and be able to recognize the human form. If a human is too close to the camera, the system will not be able to detect and classify a “human” object and, therefore, not detect a fall if one were to occur. In addition, the further away a human would appear in the view of the camera, the less likely they would be detected as their distinguishing “human” features would shrink.

*Inputs:* N/A

*Processing:* N/A

*Outputs:* N/A

*Error Handling:* N/A

**NFR.3 The system must alert the user if an object classified as a “human” falls when the source image’s RGB pixel values have a standard deviation above 10.**

*Source:* FR.1, FR.2, FR.6

*Priority:*

*Introduction:* The standard deviation between the RGB channels of the pixel values of an image gives a lot of useful information about the image. If the standard deviation of an image is 0, then all the pixel values would have the same value, in other words, the screen would be a solid color. This could be the case if the room the system will be surveying has no light source or too much light that the image is unclear. If the standard deviation is above a certain threshold; however, it is believed that the image will have enough contrast between its pixel values to detect edges, shapes, and classify a “human” object. The system must be versatile in the lighting conditions it is able to make detections in. Falls can occur in low lighting conditions, and the system must be able to detect a fall in such lighting. As long as the standard deviation is within the given threshold, it is believed the image should provide enough data to perform calculations on.

*Inputs:* A source image extracted from the live video feed of the camera.

*Processing:* The system, when given a live video feed, will process each frame and detect edges, which will be used to determine if a human is falling or not.

*Outputs:* An array of pixels, which have been manipulated to display edges formed by pixels that contrast above a given threshold from a source image, a frame from the live video feed.

*Error Handling:* If the live video feed is too dim or too bright, the contrast between pixels in a source image will be too small to detect edges. If this is the case, the system will be unable to perform as intended and will instead alert the user that the current environment is not suitable for the system to collect data.

**NFR.4 The system must detect when an object classified as “human” falls when an object obstructs the “human” object by less than 50%.**

*Source:* FR.1, FR.3

*Priority:*

*Introduction:* It is implausible to expect that all users using the Fall Detection System will always be in an empty room by themselves. Instead, the system must be capable of detecting a fall with confidence when a “human” object falls, and an unknown, alien object obstructs the line of sight between the “human” object and the camera lens in use.

*Inputs:* Video feed from the camera.

*Processing:* The system, when given a live video feed, will process each frame, identifying changes and alerting the user if a fall is detected.

*Outputs:* An alert is made when a fall is detected in the live video feed.

*Error Handling:* A fall would not be detected.

**NFR.5 The system will save and store a log of detected falls.**

*Source:* FR.2, FR.3

*Priority:*

*Introduction:* The system must be able to save and store a log of the detected falls, so that falls can be audited and reviewed to ensure accuracy.

*Inputs:* Flags thrown by detection software.

*Processing:* The system, when given a fall detection flag, will create a log of the time the fall occurred.

*Outputs:* log file entry of time fall occurred.

*Error Handling:* If the fall log has not already been created, one will be. If the fall log becomes too large obsolete data will be deleted.

**NFR.6 The system must eliminate background noise without reducing the efficiency of video processing.**

*Source:* FR.7

*Priority:*

*Introduction:* The goal of reducing background noise is to help the system to focus on accurate human detection, but this extra step of processing must not reduce the efficiency in processing.

*Inputs:* Video feed from the camera.

*Processing:* The system must be able to eliminate background noise in the video feed effectively.

*Outputs:* N/A

*Error Handling:* N/A

**NFR.7 The system must continuously track humans as they change their position in frame.**

*Source:* FR.4, FR.5

*Priority:*

*Introduction:* The system needs to fluidly update what position the subject is in be it standing, sitting, laying down, or some other position so that it can determine if the subject is changing position of their own will or if they are falling down quickly.

*Inputs:* Video feed from the camera.

*Processing:* The system must follow the subject and update what position the subject is in during real-time.

*Outputs:* N/A

*Error Handling:* N/A

**NFR.8 The system must query all templates in under 15 seconds.**

*Source:* FR.8

*Priority:*

*Introduction:* The system will have a brief phase upon initially being activated when it will query all templates stored in the database. Each template will display a human in a specified state: sitting down, lying down, standing, or falling. The system will store the templates into arrays based on the state they display. This will allow the system to quickly access the templates to make comparisons without having to directly query the database every frame.

*Inputs:* A query for all templates, grouped by human states, in the database.

*Processing:* Stores all templates into their associated arrays.

*Outputs:* Arrays of images that are grouped by their template type.

*Error Handling:* The system will disconnect from the database, close all running processes, and display an appropriate error message.

* 1. **Design Constraints**

1. The system must support the Windows operating system.
2. The system must use an RGB USB web camera to obtain the necessary data to determine falls.
3. The system will detect humans utilizing machine learning.
4. The system will detect falls utilizing templating techniques.
5. The system will use Python and its libraries to operate.
6. The system will use PostgreSQL to manage a database of templates.

**3.4 External Interface Requirements**

**3.4.1 User Interfaces**

**EIR.1 The system must display the video feed from the connected USB web camera in a window viewable to the user.**

*Source:* Project Description.

*Priority:*

*Introduction:* Displaying the video feed will be the only GUI a user will have with the system. This will allow a user to view the live feed and check falls from a live USB web camera.

*Inputs:* N/A

*Processing:* N/A

*Outputs:* N/A

*Error Handling:* The system will stop streaming the video feed from the web camera, close all processes, and output an error report.

**EIR.2 The system will output all messages through the command prompt of the machine hosting the application.**

*Source:* Project Description.

*Priority:*

*Introduction:* The system will deliver all notifications, messages, and error logs through the command prompt. This will serve as a baseline that can be built upon in future development.

*Inputs:* N/A

*Processing:* N/A

*Outputs:* N/A

*Error Handling:* The system will stop streaming the video feed from the web camera, close all processes, and output an error report.

**3.4.2 Hardware Interfaces**

**EIR.3 The hardware running the system must have at least one available USB port in working condition.**

*Source:*

*Priority:*

*Introduction:* Using an RGB USB web camera is the sole method to collect data to determine a fall and is absolutely essential for the system to run properly.

*Inputs:* N/A

*Processing:* N/A

*Outputs:* N/A

*Error Handling:* N/A

**EIR.4 The USB port must be compatible with the webcam.**

*Source:* EIR.2

*Priority:*

*Introduction:* As there are multiple types of USB’s, it is important to have compatible typings.

*Inputs:* N/A

*Processing:* N/A

*Outputs:* N/A

*Error Handling:* N/A

**3.4.3 Software Interfaces**

**EIR.5 OpenCV Libraries**

*Source:* Software Interfaces

*Priority:*

*Introduction:* OpenCV is an open-source computer vision and machine learning library.

*Inputs:* Video Feed

*Processing:* Process video feed to detect human figures in various positions.

*Outputs:* Sends signals based on values from processing.

*Error Handling:* N/A

**EIR.6 Psycopg2 Libraries**

*Source:* Software Interfaces

*Priority:*

*Introduction:* Psycopg is a popular PostgreSQL database adapter for the Python programming language.

*Inputs:* A PostgreSQL database query.

*Processing:* Processes the PostgreSQL query efficiently and securely.

*Outputs:* The resulting query based on the written PostgreSQL query.

*Error Handling:* N/A

**EIR.7 Imutils Libraries**

*Source:* Software Interfaces

*Priority:*

*Introduction:* imutils is a series of convenience functions to make basic image processing functions.

*Inputs:* Video Feed

*Processing:* To aid in video processing functionalities paired with OpenCV

*Outputs:* creates more accurate values to increase the accuracy of reports.

*Error Handling:* N/A

**EIR.8 NumPy Libraries**

*Source:* Software Interfaces

*Priority:*

*Introduction:* NumPy is a library that allows support for large, multidimensional arrays and matrices.

*Inputs:* Video Feed / OpenCV

*Processing:* To aid in video processing functionalities paired with OpenCV

*Outputs:*

*Error Handling:* N/A

**3.4.4 Communications Interfaces**

N/A

**3.5 Logical Database Requirements**

**LDR.1 Must have a database to store templates.**

*Source:* Database Restrictions

*Priority:*

*Introduction:* To allow the system to retrieve and use fall detection templates.

*Inputs:* A set of fall detection templates

*Processing:* The templates will be queried from the database

*Outputs:* Fall detection templates.

*Error Handling:*N/A

**3.6 Other Requirements**

N/A

**4. Change Management Process**

1. Discussion for changes to the SRS document can be brought up during weekly scrum meetings, or by contacting the scrum master directly to organize an on-demand meeting.
2. The scrum master will then reach out to the stakeholders, Dr. Razib Iqbal and Dr. Snigdha Chaudhari, to plan a meeting to propose the changes to the project. To abide by social distancing guidelines during the COVID-19 epidemic, all meetings will be held over Zoom.
3. During the planned meeting, the scrum master, accompanied by at least two other team members, will present the desired changes to both Dr. Razib Iqbal and Snigdha Chaudhari.
4. Either Dr. Razib Iqbal or Dr. Snigdha Chaudhari must send an official email to the scrum master’s Missouri State email account specifying that they agree or disagree with the changes presented by the team members.
5. The scrum master will be allowed to make the changes to the SRS document specified in the meeting.

**Appendices**

N/A