Video Mining Software

**Software Requirements Specification**

**Version** 1.0 **Status:** Draft

**Prepared by: Abhinav Khanna(2017csb1061)**

**Prateek Mehandiratta(2017csb1098)**

**Saurav(2017csb1106)**

**Soumil Kanwal(2017csb1113)**

IIT Ropar  
29-1-2020

**Table of Contents**

**1.** **Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

**2.** **Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3.** **External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4.** **System Features (Use Cases)**

4.1 Use Case <Insert the specific use case name>

4.2 …

**5.** **Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

1. **Introduction**
   1. **Purpose**

In this project, we describe a video mining software that will process a live feed or a recorded video and will identify people in the video, and provide whatever information is available about those people. Also this software will be able to detect similar motion sequence from source in target video and trigger a predefined workflow.

* 1. **Document Conventions**

|  |  |
| --- | --- |
| Convention | Meaning |
| GUI | Graphical User Interface |
| API | Application Programming Interface |
| HTTP | Hypertext Transfer Protocol |

* 1. **Intended Audience and Reading Suggestions**

|  |  |  |
| --- | --- | --- |
| **ID** | **Stakeholder** | **Description** |
| S-1 | Customer | The person/organization that approaches the development team to build the software. The final product is delivered to this customer, who later makes it available to the users |
| S-2 | Development team | Coming up with a detailed list of requirements, both functional and non-functional, and then coming with a scalable design, and also implementing that design. |
| S-3 | Quality Assurance team | Making test-plans and test-cases. |
| S-4 | Product Manager | Estimating the resource allocation, timeline, budget, and team size and composition et al. |
| S-5 | Employees(Users) | The final end user. |

* 1. **Product Scope**

This software will be available both as a standalone software(GUI), as well as a framework, which allows it to be used as a modular function in other softwares.

Facial recognition has a large number of uses. They can be used enforcing traffic laws, and penalizing the trespassers who violate traffic lights, speed limits, overtaking policies, etc. Another very common use of this technology is phone unlocking, which has recently been introduced in various smartphones. It can also be used by law agencies for finding missing person(s), identifying suspects etc. In the medical sector, image recognition is used for diagnosing medical diseases. In the marketplace it can be used for preventing shoplifting, targeted advertising et al. Other custom uses are also possible.

* 1. **References**

1. <https://www.umlet.com/ce.htm>
2. Software engineering: A practitioner’s approach by Roger S. Pressmann
3. **Overall Description**
   1. **Product Perspective**

In today’s competitive world, every company is trying its level best to grow everyday. Vidi Mi Inc is no different. The company has decided that it will incorporate the latest advancements in Machine Learning and Deep Learning to match up to its competitors. To achieve this goal, the company wants to build a video mining software.

Facial recognition is one of the fastest growing application areas of technology. It allows faster identification of faces and motion, adding to efficiency in various business systems. It can be used on both a macro and micro levels, from identifying an individual in a crowd, to analysing crowd and traffic patterns. It is safe to say, that facial recognition is one of the best uses of the computing powers available to humans

**Product Functions**

In this project, we intend on creating a GUI, as well as a framework, both of which will serve similar functions. The user will provide a set of photos of certain objects, and information about those objects, which he/she wants identified either in other photos, or videos, which will also be provided by the user.

The GUI will be a standalone software. It will provide an interface for the user to input the photo(s)/videos to be mined. Another interface will be able to take input from the user the set of images that the user wants identified.

The framework will be used in a similar manner, except that it will be called by other software engineers/developers/testers etc, to be able to use in some other program.

In either case, the user will be authenticated before being provided access to the functionalities.

* 1. **User Classes and Characteristics**

|  |  |  |
| --- | --- | --- |
| **ID** | **User classes** | **Description** |
| U-1 | Administrator | This is the person(s) who make the first approach to get the software. They are also the people who pay for the software. Once they obtain access to the software, they can provide administrator and/or employee access(described later) to others within the organization. |
| U-2 | Employee | These are all the people who have access to use the software. |

* 1. **Operating Environment**

1. Operating System: Windows 10
2. RAM: 4GB or higher
3. Processor: Intel Pentium or higher / AMD Athlon or higher
   1. **Design and Implementation Constraints**

* *specific technologies, tools, and databases to be used:* Python, OpenCV, PyQt, sqlite
* *hardware limitations:* RAM: 4GB or higher, Processor: Intel Pentium or higher
* *interfaces to other applications:* will also be made available as a framework which can be embedded in other applications
* *standard data exchange format:* JSON
* *language requirements:* Python 3.6 and above
* *security considerations:* Only logged in user will be allowed to see the results and he/she will not be able to access other users results.
  1. **User Documentation**

While delivering the software, all the users will be provided a comprehensive manual which will contain instructions on how to use the software, features of the software, tips and tricks to solve common problems as well as customer care contact information of VidiMi Inc in case of any problems, or future business.

* 1. **Assumptions and Dependencies**

Dependencies include open source frameworks such as PyQT, OpenCV and Google login service for user authentication.

1. **External Interface Requirements**
   1. **User Interfaces**

Different types of users will be given different types of interfaces depending on the needs and requirements of the users. Two types of interfaces are available: Framework and GUI.

* **Framework:**

This is a programming interface for system developers, testers etc. Developers can utilise this API to input a set of images, and a photo/video input to be searched, and the API will return all the objects that are detected.

* **GUI:**

The GUI extends the capabilities of the RESTful API mentioned above by allowing its use not only by other developers as part of a custom software, but to make the video-mining software an independent usable entity of its own. The working of this software is described below:

1. When the user clicks on the desktop icon of the software, it opens up the start screen. This provides the user with two major options: Faces and Input Feed.
2. When the user clicks on the Faces button, it allows the user to add different photos, which he/she wants searched in the input feed. This interface interacts with the operating system, to allow the user to browse through their system and pick the desired image.

When the user is adding the image, the program asks the user to fill out certain information regarding the image, such as name, age etc.

Once the image is added, it is displayed on screen. Other options are available for the user, such as search, edit information, delete image etc.

1. When the user clicks on the Input Feed, they are given the option to add the input feed(photo or video) in which they want to search the objects. If it is a video, they are also given the option to give a time slot in which they want to search the video. Once the input is loaded, the user presses the run command. The software runs the detection algorithm and presents all the objects that have been detected in the feed.

* 1. **Software Interfaces**

Here is a tentative list of libraries and frameworks that will be used in the project:

* **Python** is an interpreted, high-level, general-purpose programming language. Created and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace
* **OpenCV** is a library of programming functions mainly aimed at real-time computer vision.The library is cross-platform and free for use under the open-source BSD license.
* **PyQt** is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in.
* **MongoDB** is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema.

1. **System Features (Use Cases)**
   1. **Use Case**
2. **Face Detection**

Successful login, addition of object images, input photo/video

|  |  |
| --- | --- |
| **Brief Description:** | This is the primary use case. In this, we describe the most natural flow in which the user logs in successfully, adds all and only those photos that he/she needs identified, and the input photo/video to be searched |
| **Preconditions:** | The user logging-in must be a pre-registered user. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Basic Flow:** When the user clicks on the software desktop icon, he/she will see the login screen. After a successful login, the see the home screen. The user presses the faces button, which opens up a new screen where he/she is able to add new object images. Once the user adds the required images, they go back to the home screen. Then they press the input button. This open up a new screen where they add the input photo/video. If the input is a video, they can also specify a time slot within the video where they want the objects searched. Once the input is loaded on successfully, they press the search button, and the results are displayed. They then logout or go back to the home screen. | | | |
| **Assumptions:** We are assuming that the user is already registered for the service. Also, this use case ignores various errors and exceptions that can occur. This will be covered in other secondary use cases. | | | |
| **Line** | **System Actor Action** | | **System Response** |
| * 1. ` | User Login | | Application asks for the login id and password. If the credentials are correct user is logged in to the system |
|  | Upload Face | | The application displays a screen where user uploads the image and fills in the details of the person(optional). All the information is added to the database. This step is repeated until all the faces are added. Data is verified |
|  | Upload target video or image | | The application displays a dialog box for searching the video/image. If the video is uploaded the user is asked to input time slot within which to search the video. |
| 4 | Start Search | | Application searches for the faces in the video or image. Results are displayed showing which person is present in which time period. |
| **Post Condition:** | | User is asked to choose between one of the following options:   1. Logout of the application 2. Add or Delete faces to search. 3. Search in a different video/image. | |

|  |  |  |
| --- | --- | --- |
| **Alternate Flow (AF1) - 1: User not registered** | | |
| If at line 1 in Basic Flow user is not registered within the application, then the user is asked to create a new account. | | |
| **Line** | **System Actor Action** | **System Response** |
| * 1. ` | User Login | Application asks for the login id and password but as user is not registered, the user will be given an option to register. |
|  | User Registration | A new window will be displayed where the user will be asked to fill their credentials. |
| Basic Use Case is continued from line 1 after successful registration of the user. | | |

|  |  |  |
| --- | --- | --- |
| **Alternate Flow (AF2) - 2: No face in photo uploaded by user** | | |
| If at line 2 in Basic Flow a photo is uploaded by the user, but there is no face in the photo. | | |
| **Line** | **System Actor Action** | **System Response** |
| 1 | User Login | Application asks for the login id and password. If the credentials are correct user is logged in to the system |
| 2 | Upload Face | The application displays a screen where user uploads the image and fills in the details of the person(optional). All the information is added to the database |
| 3 | Face not present | Application throws an error if a face is not detected in the image uploaded by the user. User is asked to input a new image. |
| Basic Use Case is continued from line 2 after successful registration of the user. | | |

|  |
| --- |
| **Activity Diagram:** |
|  |

1. **Motion Detection**

Successful login, addition of object images, input photo/video

|  |  |
| --- | --- |
| **Brief Description:** | This is the second primary use case. In this, we describe the most natural flow in which the user logs in successfully, adds a file which represents sample movement sequence and similar movement sequence is detected in target video. |
| **Preconditions:** | The user logging-in must be a pre-registered user. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Basic Flow:** When the user clicks on the software desktop icon, he/she will see the login screen. After a successful login, the see the home screen. The user presses the input button, which opens up a new screen where he/she is able to add a video showing sequence of movements. Once the user adds the required file, they go back to the home screen. Then they press the input button. This open up a new screen where they add the input video. Once the input is loaded on successfully, they press the search button, and the results are displayed. They then logout or go back to the home screen. | | | |
| **Assumptions:** We are assuming that the user is already registered for the service. Also, this use case ignores various errors and exceptions that can occur. This will be covered in other secondary use cases. | | | |
| **Line** | **System Actor Action** | | **System Response** |
| 1 | User Login | | Application asks for the login id and password. If the credentials are correct user is logged in to the system |
| 2 | Upload Movement File | | The application displays a screen where user uploads the video file. |
| 3 | Upload target video | | The application displays a dialog box for searching the video/image. If the video is uploaded the user is asked to input time slot within which to search the video. |
| 4 | Start Search | | Application searches for the same movement sequence in the video. If same sequence is found custom workflow will be triggered. |
| **Post Condition:** | | User is asked to choose between one of the following options:   1. Logout of the application 2. Change Movement File 3. Search in a different video. | |

|  |  |  |
| --- | --- | --- |
| **Alternate Flow (AF1) - 1: User not registered** | | |
| If at line 1 in Basic Flow user is not registered within the application, then the user is asked to create a new account. | | |
| **Line** | **System Actor Action** | **System Response** |
| 1 | User Login | Application asks for the login id and password but as user is not registered, the user will be given an option to register. |
| 2 | User Registration | A new window will be displayed where the user will be asked to fill their credentials. |
| Basic Use Case is continued from line 1 after successful registration of the user. | | |

|  |
| --- |
| **Activity Diagram:** |
|  |

1. **Other Nonfunctional Requirements**
   1. **Performance Requirements**

The performance of the software is measured by the number of images it is able to identify in the given input, and the time it takes to complete identification. The former depends on the quality and type of images provided by the user, and the latter, varies only in case the target is a video, where the time taken will depend on the size, quality and length of the video.

* 1. **Safety Requirements**

This software uses third-party software database services. Any damage, loss, or failure of/to the database will be the primary responsibility of the respective service providers.

* 1. **Security Requirements**

We are using Google login service for authentication purposes. All security and privacy issues regarding usage of Google login service are applicable here.

* 1. **Software Quality Attributes**
* Availability: The API should be available to all users at all times
* Correctness: The software should guarantee a certain degree of accuracy of recognition
* Flexibility: Given multiple input photos, the software will be able to give combined results on those photos
* Reusability: The GUI uses the API for the core functionality instead of implementing the algorithm again