# Software Requirements Specification

# For

# Object Detection using Haar Cascade

Version 1.0

Prepared by-

|  |
| --- |
| Name |
| Shubham Agarwal |
| Apoorv Khare |

Contents

CONTENTS………………………………………………………………………………………………………….II

Revisions [ii](#__RefHeading___Toc397335649)

1 Introduction [1](#__RefHeading___Toc397335650)

1.1 Document Purpose [1](#__RefHeading___Toc397335651)

1.2 Product Scope [1](#__RefHeading___Toc397335652)

1.3 References 1

2 Overall Description [2](#__RefHeading___Toc397335657)

2.1 Product Overview [2](#__RefHeading___Toc397335658)

2.2 Product Functionality 3

2.3 Design and Implementation Constraints 3

3 EXTERNAL INTERFACE REQUIREMENTS [4](#__RefHeading___Toc397335662)

3.1 Hardware Requirements [4](#__RefHeading___Toc397335663)

3.2 Software Requirements [4](#__RefHeading___Toc397335663)

1 Introduction

This project deals with making an Object detection System using the Haar Cascade algorithm in Machine Learning. This project aims to use a pre face recognition model deployment so that it can accurately predict the faces over which it is trained when the same person comes in-front of the camera.

## Document Purpose

The purpose of this SRS is to define the requirements for Object Detection using Haar Cascade. The requirements defined here will be used in assisting maintainers in further improvements and maintenance of the program, and by users who need to understand what this program does.

## Product Scope

This project introduces a car and pedestrian detection system with the objective of detecting pedestrians and other cars on the road which can be used in driverless car or to warn drivers to reduce the chances of road accidents.

Some other applications are:

1. Domain Surveillance: It can be used in public places like train stations, shopping malls and streets with security in mind.
2. E-health applications: One example is detection of falling elderly people to automatically trigger an alarm. This kind of supporting technology can allow elderly to live longer in their similar environment.

## References

* youtube.com
* kaggle.com
* opencv.org

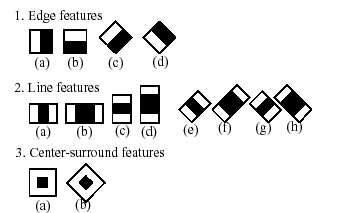
# OVERALL DESCRIPTION

## 2.1 Product Overview

First, a *cascade of boosted classifiers working with haar-like features* is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size, and negative examples - arbitrary images of the same size.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a “1” if the region is likely to show the object (i.e., face/car), and “0” otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily “resized” in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

The word “cascade” in the classifier name means that the resultant classifier consists of several simpler classifiers (*stages*) that are applied subsequently to a region of interest until at some stage the candidate is rejected or all the stages are passed. Haar-like features are the input to the basic classifiers. Haar-like features are:



In our program, we extract frames from a video and pass each frame through our classifier to detect the required object. After the classifier detects the object, it returns the dimensions of the feature that was classifies as positive which we use to draw a rectangle on it. OpenCV library is very easy to use for this purpose.

## Product Functionality

An important feature of this model is very simple to use. Only what the user has to do is to run the code by just one command. For video prediction, the user has to only run the code. For image related prediction, the user has to give the path of the image that they want to predict.

## Design and Implementation Constraints

EDUCATIONAL LEVEL: - At least, user of the system should be familiar with English language.

TECHNICAL EXPERTISE: - User should be comfortable with the general purpose applications of computer system.

SOFTWARE CONSTRAINTS: - The code will run on the system having Python3 installed and certain libraries required which will be provided along with the code.

1. External Interface Requirements

**3.1 Hardware Requirements:**

* Processor: Intel ® Core$(TM) i3 -6200U$CPU @2.30GHz 2.40GHz
* Disk Space : 1 TB
* Ram : 4 GB and above

**3.2 Software Requirements:**

* Numpy
* Opencv
* Matplotlib