**Computer Vision for Assembly Line**

**(CVAL)**

**Software Requirement Specification**

**TABLE OF CONTENTS**

1. Introduction
   1. Purpose
   2. Scope of the project
   3. Definitions, acronyms and abbreviations
   4. References.
   5. Overview
2. System Context
   1. Product perspective
      1. System overview and Context Flow Diagram
      2. System Interface
      3. Site adaptation Requirements
   2. Product functions
   3. User characteristics.
   4. General Constraints.
   5. Assumptions and Dependencies
3. Specific requirements
   1. External interface and requirements
   2. Functional requirements
   3. Performance requirements
   4. Design constraints
   5. Software system attributes
      1. Reliability
      2. Security
      3. Maintainability
      4. Portability
      5. Installability

**1. Introduction**



The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. Aim of this document is to give a complete in-depth insight of the complete Computer vision for assembly line (CVAL) system by defining the problem statement in detail. It also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The detailed requirements of the Computer vision for assembly line system are provided in this document.

**1.1 Purpose**  
  
The aim of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to consumers. The basic purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality.

**1.2 Scope of the project**

This software system will be providing computer vision to an assembly line in the die manufacturing unit of client’s company. However, the system provided can be considered general and will be implementable in the assembly line for any other manufacturing unit. This system is designed to maximize the productivity of the client’s manufactures by providing a tool to detect defects in the manufactured goods on the assembly line which would otherwise have to be performed manually.

More specifically this system will be designed to set up a connection with a camera placed as a part of the assembly line which will be clicking pictures of the manufactured products. The pictures would then be compared with the pictures of the ideal manufactured product. If the relative error is higher, the manufactured product can be discarded else it is safe to go on. The scope of this software however is limited to scanning only the parts of the product which can be captured by the camera for errors. The parts of the product like the bottom surface kept on the conveyor cannot be scanned. An automated mechanical system can be utilized for discarding the defected product or else the process can be manual too.

**1.3. Definitions, acronyms and abbreviations**

|  |  |
| --- | --- |
| Operator | The person responsible for removing the defected objects |
| Products | The manufactured goods that come out of the assembly line |
| CVAL | Computer Vision for Assembly Line (i.e., our software product) |

**1.4. References**

* A concise Introduction to Software Engineering, Pankaj Jalote
* Wikipedia([www.wikipedia.com](http://www.wikipedia.com/))

**1.5. Overview**

This document covers the requirements received from the client’s side and development of CVAL as a software will be carried out according to this document. The document first talks about the overall functionality CVAL is going to provide the users and how the users will interact with it. The document then goes on to discuss both functional as well as non functional requirements.

**2. System Context**



**2.1. Product Perspective**CVAL is a computer vision software implemented using OpenCV libraries. It will provide the user with a simple mechanism to identify which product is defected and which product is good to go.

* CVAL provides basic functionalities like cross platform support i.e. it can be run on most of the commercial operating systems.
* Moreover, it maintains an account of the products discarded and along with the reason why. This may ease the workload of the person reviewing the products manufactured.

**2.1.1 System Overview and context flow diagram**

Our product which is responsible for detecting defects is part of assembly line.

It is dependent on the production unit for providing finished products that are ready to be tested for any manufacturing defects.

**2.1.2. System Interface**

User Interface: The user who is the person responsible for removing the defected products from the assembly will be displayed the product ids of the products which are to be removed.

Software Interface: The camera capturing the images of the product is provided an interface by CVAL as it can then receive these images and carry on the further processing.

**2.1.3. Site Adaptation Requirements**

Good quality cameras are to be installed at appropriate locations in the assembly line so as to provide a 360 degree view for proper inspection of the products.

For the camera to operate in the manufacturing unit, a DC power source of 12V is required to keep the battery of the cameras charged at all times.

A processing unit such as a PC or a laptop will be needed where the actual processing will be done. The computer preferably should be a PC which could then be a dedicated one.

**2.2. Product Functions**

The main functionality of the software is pretty simple and straightforward.

It determines whether the product being tested is defected or not, more specifically it checks defected dimensions i.e. if the product was supposed to be a circular one then the elliptical products manufactured need to be removed. In broader terms, it can be termed as a quality check.

It concentrates on finding the relative error between the existing image of an error free sample product and the one under consideration. Quite obviously if on comparison there are higher amount of differences then CVAL will identify the test item as a defected piece which will be later discarded.

**2.3. User Characteristics**

In our case the product’s target audience are the people in manufacturing units or the units themselves. The system will be designed considering there will be a person responsible for removing the defected products off the conveyor belt. The intended use of the software is a very basic level quality check of the manufactured piece i.e dimensional analysis. So the user is simply required to be able to identify how the product responds in case of a defected product and otherwise.

**2.4. General Constraints**

* Only the left,right,top and the front face with respect to the machine can be checked while the bottom surface can’t be checked.
* CVAL is designed to detect elementary/primary defects and is not reliable when it comes to reporting minute or very fine defects.
* The process captures the images of the products from various angles as they move down the assembly line. The collected pictures are then analysed for defects. Thus after taking pictures of some 'n' products the line needs to be given a short break for the software to complete its analysis.

**2.5 Assumptions and Dependencies**

Certain assumptions are taken into consideration when we take into account the functionality of CVAL one of them being that the conveyor belt on which the products are kept can pause for a short duration of time so as to provide the software enough processing time.

**3. Specific Requirements**



**3.1. External Interface Requirements**

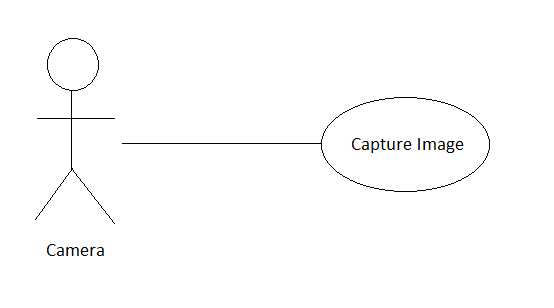
The only link to an external system is the connection to camera via which the photos of the product are fed into the software for inspection of defects.

**3.2. Functional Requirements**

This section outlines the use cases for each actors that will be present. The actors include external systems like camera and the processing unit and person like the Operator.

a) Camera Use Case

Use case: CaptureImage

Diagram:  


Brief Description:

Cameras placed at positions capture the images of the products from different angles.

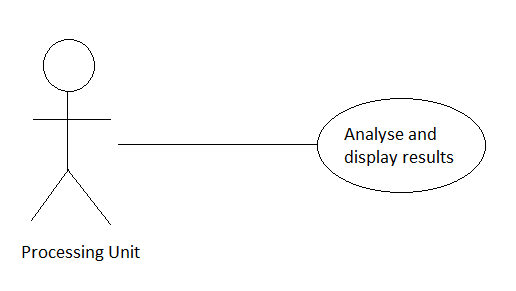
Step By Step Description:

* Product travelling on the conveyor belt comes in the proximity of the placed cameras.
* Cameras take the pictures of the product from different angles one by one.
* The clicked pictures are sent to a memory buffer in the computer from where the pictures are transferred to the product analysing module of the program.

|  |  |
| --- | --- |
| **Use Case Name** | CaptureImage |
| **Trigger** | The product comes in the proximity of the cameras. |
| **Precondition** | Camera is working. |
| **Basic Path** | The products on the conveyor belt enter the zone where cameras are placed. |
| **Alternative Paths** | No alternative path possible. |
| **Postcondition** | Clicked pictures are sent to the processing unit. |
| **Exception Paths** | None |
| **Other** | None |

b)Processing Unit Use Case

Use case: AnalyseProduct

Diagram:  


Brief Description:

The processing unit i.e. the dedicated computer analyses the products and displays the codes of the defected products on the output screen.

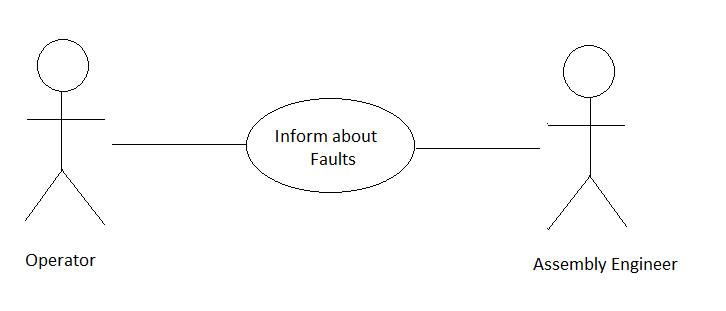
Step By Step Description:

* The images received are analysed for having any defects.
* Defected products are marked by code and displayed on the output screen.

|  |  |
| --- | --- |
| **Use Case Name** | AnalyseProduct |
| **Trigger** | The last product of the batch gets its pictures clicked. |
| **Precondition** | Dedicated PC is turned on and software is running. |
| **Basic Path** | Pictures are taken and stored in the memory buffer which is transferred. |
| **Alternative Paths** | No alternative path possible. |
| **Post condition** | Products are analysed for defects and defected product codes are displayed on screen. |
| **Exception Paths** | None |
| **Other** | None |

c) Operator Use Case

Use case: AllError

Diagram:  


Brief Description:

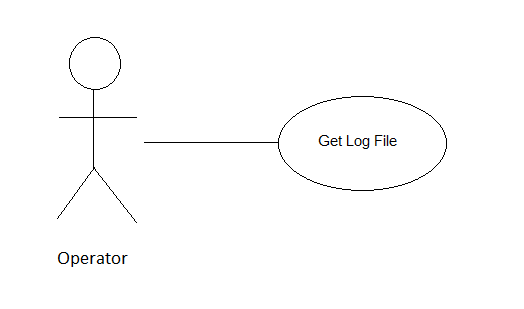
The operator informs the assembly line engineer in case the whole batch is found defected.

Step By Step Description:

* Operator observes the output screen.
* Operator informs the assembly engineer that whole batch of received products are faulty and the line needs to be checked for major failures.

|  |  |
| --- | --- |
| **Use Case Name** | AllError |
| **Trigger** | Output screen showing all received products defected. |
| **Precondition** | Whole batch of products has been analysed for defects. |
| **Basic Path** | When all the product codes are displayed on the screen. |
| **Alternative Paths** | No alternative path possible. |
| **Post condition** | Operator informs the assembly line engineer. |
| **Exception Paths** | None |
| **Other** | None |

Use case: ViewLogFile

Diagram:  


Brief Description:

The operator can authorize through a given id and password and can check a system created log file which mentions the product ids of defected products.

Step By Step Description:

* Operator types in the given id and password
* System created log file having defects listed in order of detection is shown.

|  |  |
| --- | --- |
| **Use Case Name** | ViewLogFile |
| **Trigger** | Assembly Line Engineer asks for information regarding defected products. |
| **Precondition** | Assembly line is halted and the software currently is not processing images. |
| **Basic Path** | Operator provides his id and password assigned to him. |
| **Alternative Paths** | No alternative path possible. |
| **Post condition** | A log file is shown which has all the defects listed on it in order of detection. |
| **Exception Paths** | None |
| **Other** | None |

**3.3. Performance Requirements**

CVAL is required to analyse a batch of products and report the defective product.  
When a defective product is identified the output screen displays its product code which is unique for each product in the batch.

A batch of products should be a maximum of 50 objects.

The process of capturing product images from all the different angles should be done in 2 seconds.

The processing of the whole batch shall be done in not more than 20 seconds.

**3.4. Design Constraints**

This section includes the design constraints on the software that are caused by the hardware of the dedicated PC running the software or imposed by the required standards. The software for all the computations uses the hard drive space of the PC running CVAL thereby putting no immediate restriction on the space used. However, a system with high graphic processing is required.

**3.5.** [**Software System Attributes**](#_Toc172721225)

**3.5.1 Reliability**The algorithm used in the software for matching the images of products from the ideal product yield reliable results in every test case. Hence CVAL shall provide reliability at all times.

**3.5.2. Security**

CVAL although utilizes no cryptographical techniques but no such techniques are even needed to secure the data as apart from the log files no other data is saved on the PC which could be tampered with.

**3.5.3. Maintainability**Maintainance for CVAL is not a tough job as the functionality is limited and simple. The errors that might occur will be countable because of the limited functionalities provided and hence would be very easy to sort out.

**3.5.4 Portability**CVAL is very portable as it is platform independent as well as written in a proven portable language.

**3.5.5 Installability**The software is installable at almost every platform including Windows or UNIX on account of usage of general purpose compilers available in every operating system readily.