Software Requirements Specification

for

Detecting Damaged Components of Rockets Using Deep Learning Model

Version 1.0 approved

Prepared by

Abhiram Krishnan RS
Akshay K
Albin Zachariah Daniel
Amal Thomas

TKM College of Engineering

16 March 2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Akshay K	28.02.2023	Initial draft prepared	1.0
Albin Zachariah Daniel	1.03.2023	Revised by adding overall description	1.0
Abhiram Krishnan RS	3.03.2023	Added External Interface Requirements	1.0
Amal Thomas	4.03.2023	Added System Features	1.0
Albin Zachariah Daniel	7.03.2023	Finalized document for initial submission	1.0

1. INTRODUCTION

This Software Requirements Specification (SRS) document describes the requirements for the development of a Deep Learning model for damage detection.

1.1 Purpose

The purpose of this document is to provide a clear and detailed description of the requirements for the development of a damage detection system for rocket components. This document will be used as a reference for the development team to ensure that all requirements are met.

This project intends to develop a Deep Learning model with the purpose of analyzing images of rocket components in order to determine the presence of any damage that may be significant during assembly.

1.2 Document Conventions

This document is based upon:

● IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications

● IEEE 830-1984 - IEEE Guide for Software Requirements Specifications

1.3 Intended Audience and Reading Suggestions

Anyone with some programming experience and familiarity with Python and its libraries related to Deep Learning like TensorFlow, can understand this document. The document is intended for developers, software architects, testers, project managers, and documentation writers.

1.4 Project Scope

Space exploration and the military depend on rockets. Rocket reliability and safety are crucial. Thus, we must improve quality assurance in component assembly.

Rocket safety depends on component damage detection. Manually, it was difficult. Early damage detection prevents catastrophic failures and saves lives. However, harsh environments and limited access make rocket component damage detection difficult.

Thus, our project will create a Deep Learning-based rocket component damage detection system. This model has several advantages:

1. Continuous Improvement of the Model: Collecting additional data, refining the model architecture, and enhancing the training process will enhance the damage detection system.
2. Standalone System: The model can be designed as a standalone system that does not rely on external data or communication links.

3. High Accuracy: The model developed must process large amounts of data quickly and accurately with minimal computational resources.
4. Increased productivity: By automating the component damage detection process, the system can reduce the time and effort required for manual inspection and testing.

1.5 References

- Data Augmentation: https://www.tensorflow.org/tutorials/images/data_augmentation
- Image Classification: https://www.tensorflow.org/tutorials/images/classification

2. OVERALL DESCRIPTION

2.1 Product Perspective

The Deep Learning model is developed to automate the detection of damages based on rocket components. The following two modules can be used to understand how it functions:

1. Standardization of Data

   The image is converted to a compatible format to be used with the trained model.

2. Prediction

   The standard input image is used to classify it as a damaged component or not using the model.

This project deals with the implementation of a Deep Learning model that classifies component images as damaged or not with high accuracy.

2.2 Product Features

1. Object detection: The system should be capable of detecting damages in rocket components using a pre-trained model, Resnet50.
2. Data pre-processing: The system should perform several data pre-processing techniques on the dataset, including image normalization, image resizing, and other data augmentation techniques.
3. Model training and evaluation: The system should be able to train and evaluate the object detection model using the dataset.
4. Security: The system should have appropriate security measures in place to protect the dataset and ensure the privacy of clients' data within the standalone model.

2.3 User Classes and Characteristics

1. Engineers and Inspectors: These are individuals responsible for the design, manufacture, and maintenance of rocket components. They have technical knowledge and expertise in the field of rocket engineering and inspection. They need our product to quickly and accurately detect damages in rocket components, allowing them to identify potential issues and take appropriate action to prevent catastrophic failures.

2. Quality Assurance Personnel: These individuals are responsible for ensuring that rocket components meet the required quality standards. They use our product to reduce inspection costs and times, leading to more efficient and effective quality control processes.

3. Developers: There is a team of developers who will refer to this document:

a. Abhiram Krishnan RS
b. Akshay K
c. Albin Zachariah Daniel
d. Amal Thomas

2.4 Operating Environment

- A computer or server with sufficient processing power and memory.
- An operating system that supports the required software and libraries.
- Sufficient storage space for storing images of rocket components.

2.5 Design and Implementation Constraints

- Processing power for running neural networks for detection of damages.
- Huge dataset so that the model can have high accuracy in predictions. Methods such as Data Augmentation can be used here in accordance with the nature of images.
- Chances of overfitting and underfitting can affect the performance.
- Annotation of dataset.

2.6 User Documentation

The soft copy of the document in PDF format and a manual will be shared with customers and developers.

2.7 Assumptions and Dependencies

Assumptions:

- The photograph should have been taken in well-lit conditions with the surface of the rocket component clearly visible.
- The system would require there to be some type of image capturing system that obtains rocket component images for input to the model for prediction.
- It is assumed that the pre-trained Deep Learning model used for training has high accuracy.

Dependencies:

- The project is dependent on the availability of the dataset.
- This project is dependent on libraries related to Deep Learning concepts like TensorFlow and Keras.

3. EXTERNAL INTERFACE REQUIREMENTS

3.1 User Interfaces

Since this is a trained Deep Learning model, the user does not directly interact with it.

Features such as an alerting system are beyond the scope of this project.

3.2 Software Interfaces

Software interfaces for Deep Learning projects may include:

1. Programming Languages:

Deep Learning models can be implemented using programming languages such as Python.

2. Development Frameworks:

Development frameworks such as TensorFlow and Keras provide pre-built tools and libraries for Deep Learning model development.

3. Object Detection Deep Learning Model

The system shall use the state-of-the-art object detection model, Resnet50, which requires fewer system requirements.

4. Integrated Development Environments (IDEs):

IDEs such as Jupyter Notebooks and Visual Studio Code provide a complete development environment for Deep Learning projects, including code editing, debugging, and project management tools.

3.3 Hardware Interfaces

Not applicable.

4. SYSTEM FEATURES

1. Data Pre-processing Techniques

The system shall perform data pre-processing techniques, including image normalization, resizing, and data augmentation. Image normalization can reduce the impact of varying lighting conditions, while resizing can help the model handle images of different sizes.

Data augmentation can generate a diverse dataset and reduce overfitting. These techniques are especially important for rocket component damage detection, as images may be captured in harsh environments with varying levels of detail and lighting.

1. Normalization: The pixel values of images are scaled to the range 0-1 using the Min-max normalization technique, which improves the convergence of gradient descent algorithms and can reduce the impact of outliers on the training process.
2. Standardization: The pixel values are scaled to a standard Gaussian with a mean of zero and a standard deviation of one. This improves model performance by ensuring that all features are on a similar scale and can prevent features with larger values from dominating the training process.
3. Centering: The mean pixel value is subtracted from each pixel value resulting in a distribution of pixel values centered on a mean of zero.
4. Flipping:
   By applying flipping techniques, the system can produce a mirrored version of the image, providing additional variations to the dataset without distorting the appearance of the damage. This technique can also help the model to generalize better to unseen data, making it more robust and accurate in real-world scenarios.
5. Rescaling:
   It is a useful technique for improving the performance of the model by reducing the computational resources required for training. Rescaling can also help to improve the accuracy of the model by reducing the impact of irrelevant image features.
6. Rotation:
   Rotation provides additional variations to the dataset which improves the model's ability to detect damages at various angles. However, it is important to ensure that the rotation does not distort or change the appearance of the damage.
7. Shear:
   Shearing can help the model learn to detect objects even when they are tilted or rotated in the image. However, it is important to note that shearing can also introduce some artifacts and distortions in the image. Therefore, it is essential to carefully evaluate the impact of this technique on the dataset and the performance of the model.
8. Cropping:
   Random cropping technique helps in reducing the dependency of the model on the specific object location in the image and improves the model's robustness. But cropping may remove important details like damaged sections from the image, leading to a decrease in the model’s accuracy.
   So, we’ve exempted using this technique in our data pre-processing.
9. Zooming:
   Zooming can help the model learn to recognize smaller details in the image that may not be visible at the original size. But it is important to be careful with the degree of zooming applied as too much zooming can distort the image or lose the damaged sections in the process and affect the model's ability to learn the underlying patterns.

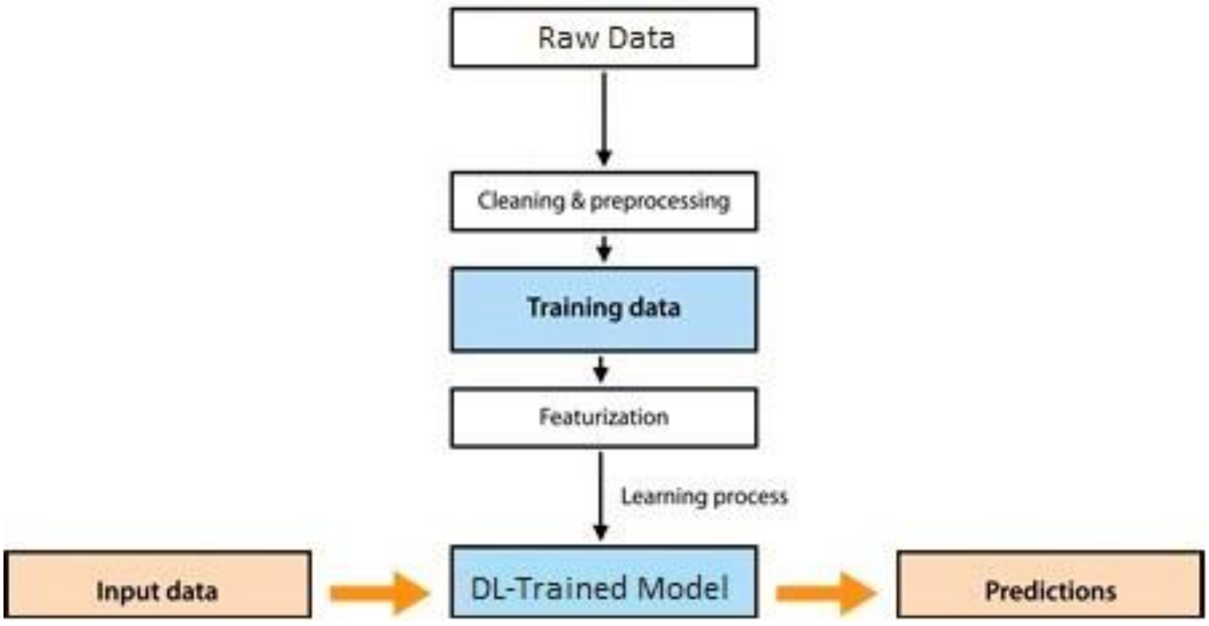
By performing these pre-processing techniques, the system can improve the accuracy and robustness of the model, allowing for earlier detection of damages and improving rocket safety and reliability.

2. Model Training

The training process is a critical component of the Deep Learning-based damaged rocket component detection system. It is important to ensure that the model is accurate and reliable.

A classification model, is developed which identifies damaged components from images. The model classifies images into either of the two classes: damaged or not damaged. These models allow the system to rapidly and correctly identify rocket component deterioration, allowing early maintenance and repair for rocket safety and dependability by alerting the users.

The process involves identifying the correct model architecture, identifying appropriate number of layers for feature extraction and correct usage of hyperparameters for better efficiency and accuracy.



3. Evaluation

During the evaluation process, several metrics are used to assess the performance of the model, such as precision, recall, and accuracy score. These metrics can help determine the effectiveness of the model in identifying damaged components.

For this solution, we take our key performance indicators to be recall and precision of the prediction while giving more importance to recall. Recall should be higher because it is more important to correctly identify all instances of damage, even if it means some non-damaged images are classified as damaged which might result in a lower precision.

It is also important to increase the value of precision which reduces the number of false alarms. False alarms need to be minimized to avoid fear, havoc and financial wastage.

The F1 score takes into account both precision and recall and provides a single number that indicates the overall performance of the model. The F1 score ranges from 0 to 1, with a score of 1 indicating perfect precision and recall.

The real-time prediction process involves capturing images of the rocket components using sensors or cameras and passing them through the damage detection model. The model then analyses the images and provides a prediction of whether or not there is damage present. This process can occur in near real time, allowing for the rapid detection of damages.

5. NON-FUNCTIONAL REQUIREMENTS

1. Performance

To ensure that the Deep Learning-based rocket component damage detection system is effective, it is important that it can process images quickly. Therefore, the system should be optimized for speed and efficiency, with a focus on reducing processing time.

2. Usability

To ensure ease of use and accessibility, the Deep Learning-based rocket component damage detection system shall have a user-friendly interface for uploading and processing images.

3. Security

To ensure the protection of the dataset and the privacy of the clients' data within the standalone model, the Deep Learning-based rocket component damage detection system shall have appropriate security measures in place. Access controls can be implemented through the use of user authentication and authorization mechanisms.

4. Portability

To ensure flexibility and accessibility, the Deep Learning-based rocket component damage detection system shall be designed to run on different operating systems. This includes Windows, Linux, and macOS.