CLOTHES DEFECT DETECTION USING OPENCV

A PROJECT REPORT SUBMITTED BY

A.T.A. SAHABANDU

(S / 20 / 507)

in partial fulfillment of the requirement for the course of

CSC 3141 Image Processing Laboratory

to the

Department of Statistics and Computer Science

of the

FACULTY OF SCIENCE

UNIVERSITY OF PERADENIYA

SRI LANKA

2025

DECLARATION

I do hereby declare that the work reported in this project report was exclusively carried out by me under the supervision of **Prof. Amalka Pinidiyaarachchi** It describes the results of my own independent work except where due reference has been made in the text. No part of this project report has been submitted earlier or concurrently for the same or any other degree.

Date: ……. ……

Signature of the Candidate

Certified by:

1. Instructor: **Mr. Kansaji Kotuwage**

   Date: ……. Signature:…….

2. Supervisor: **Prof. Amalka Pinidiyaarachchi**

   Date: ……. Signature:…….

3. Head of the Department: **Dr. Sachith Abeysundara**

   Date: ……. Signature:…….

ABSTRACT

In textile manufacturing, quality assurance plays a crucial role in reducing waste and ensuring customer satisfaction. Manual fabric defect detection, however, is time-consuming, inconsistent, and subject to human error. This project presents an automated approach to fabric defect detection using Python and OpenCV, employing classical image processing techniques. The system begins with grayscale conversion and Gaussian blurring to suppress noise, followed by adaptive thresholding using Otsu's method with dynamic adjustments based on pixel intensity distribution. Morphological operations such as opening and dilation are then applied to refine defect segmentation. Defects are detected by contour extraction, and further filtered based on statistically derived area and perimeter thresholds. The final results, including all key processing stages and defect highlighting, are visualized using Matplotlib. The proposed method effectively identifies prominent defects such as holes, stains, or misweaves under controlled lighting, demonstrating a practical, cost-effective solution for semi-automated fabric quality inspection.

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

LIST OF APPENDICES

CHAPTER 01

INTRODUCTION

1.1 INTRODUCTION

In the textile industry, ensuring fabric quality is vital for both customer satisfaction and production efficiency. Traditionally, defect detection in fabrics is done manually, which is time-consuming and inconsistent. This project aims to automate the process using Python and OpenCV. By applying classical image processing techniques such as grayscale conversion, blurring, thresholding, and morphological operations, the system can detect defects like holes, stains, and misweaves in clothing fabric. The goal is to build a cost-effective, fast, and accurate solution for real-time fabric inspection.

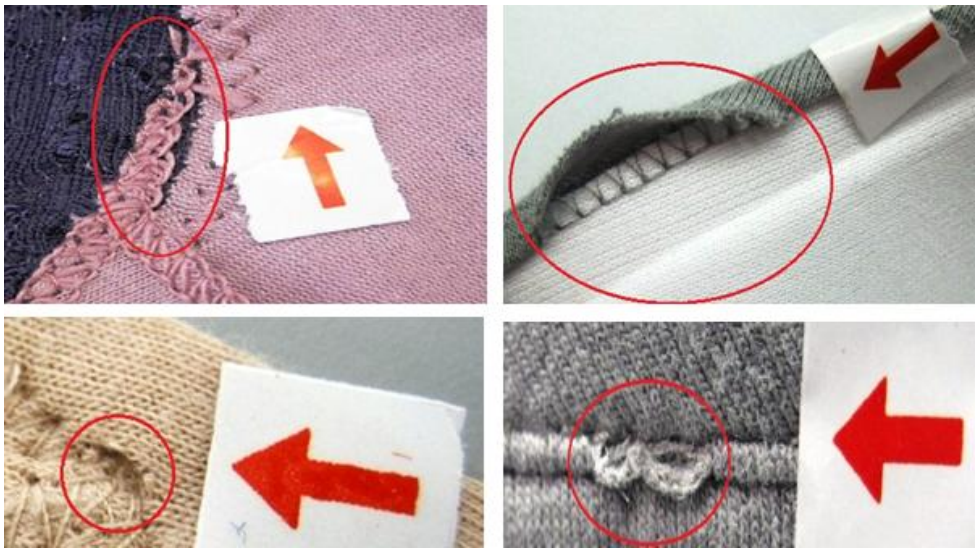

Figure 1.1 manual defect identify issue photo

1.2 PROBLEM STATEMENT

Fabric defect inspection in the textile industry is traditionally performed by human inspectors. This manual process is time-consuming, labor-intensive, and prone to inconsistencies due to human error, fatigue, and subjectivity. Additionally, it lacks the speed and precision required for modern, large-scale production environments. There is a growing need for an automated, reliable, and cost-effective method to detect common fabric defects such as holes, stains, and stitching errors. This project addresses the challenge by proposing an image processing-based solution using OpenCV to detect and highlight defects in fabric images automatically.

1.3 SOLUTION

This project proposes a low-cost, automated fabric defect detection system using Python and OpenCV. The system analyzes fabric images by applying a sequence of image processing techniques including grayscale conversion, Gaussian blurring, adaptive thresholding, and morphological operations. Defects are identified based on contour analysis and highlighted with bounding boxes. The solution requires only a standard camera and runs on basic hardware, making it scalable and easy to deploy. It eliminates the need for manual inspection and enhances detection speed, accuracy, and consistency in textile quality control.

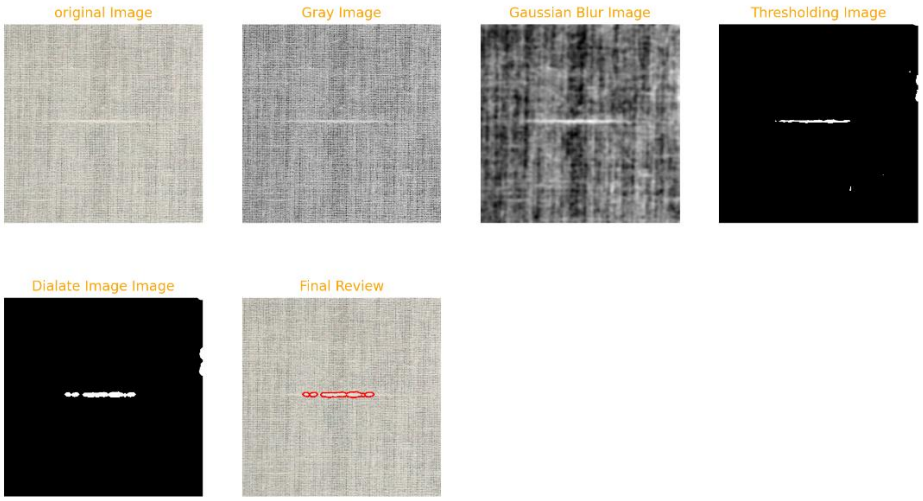

Figure 1.2 Screenshot of project output photo

CHAPTER 02

RELATED WORK

2.1 BACKGROUND

Fabric defect detection is essential in the textile industry to ensure product quality and reduce production waste. Manual inspection is commonly used but is time-consuming and prone to human error. To address this, image processing techniques can be applied to automate defect detection using simple computer vision methods.

This project uses Python and OpenCV to build a lightweight fabric defect detection system. The process begins by converting the fabric image to grayscale and applying Gaussian blur to reduce noise. Adaptive thresholding using Otsu's method is then used to segment the image, with dynamic adjustments based on pixel intensity distribution. Morphological operations such as opening and dilation are applied to clean up the segmented image and highlight the defect areas. Finally, contours are extracted and filtered based on area and perimeter, and the detected defects are marked using bounding outlines. This approach enables effective, real-time defect detection without the need for complex machine learning models or expensive hardware.

CHAPTER 03

METHODOLOGY

3.1 IMAGE PROCESSING WORKFLOW

The defect detection system uses a structured image processing workflow built with Python and OpenCV. It begins by converting the input fabric image to grayscale, followed by Gaussian blur to reduce noise. Then, Otsu's thresholding is applied to segment the image, with adjustments made dynamically based on pixel intensity distribution. If necessary, inverse thresholding is used to improve separation of defects.

Next, morphological operations are applied—opening removes small noise, and dilation expands defect regions to make them more prominent. The refined binary image is used to extract contours. To minimize false detections, contours are filtered using area and perimeter thresholds calculated from statistical metrics. The most significant contours are drawn on the original image to highlight fabric defects. Each stage of the process is visualized for clarity and verification, making the system both effective and interpretable.

3.2 USED FUNCTIONS AND THEIR APPLICATIONS

The project includes a modular collection of Python functions, each handling a specific step in the fabric defect detection process. This improves code clarity and reusability.

Table3.1 Overview of Key Functions in the Clothes Defect Detection System

Function	Purpose
cv2.cvtColor()	Converts the input image to grayscale to simplify analysis.
cv2.GaussianBlur()	Reduces noise and smooths the image using a Gaussian kernel.
cv2.threshold()	Applies Otsu's thresholding to segment the image; includes dynamic tuning.
cv2.morphologyEx()	Performs opening and dilation to refine segmented defect areas.
draw_selected_contours()	Extracts contours from the binary image to detect defect boundaries.
draw_selected_contours()	Custom function that filters and draws significant defect contours.
matplotlib.pyplot	Used to visualize each stage of the image processing workflow.

3.2 USED TOOLS, TECHNOLOGIES AND LIBRARIES

This project was built using open-source tools suitable for image-based detection tasks. The setup is designed for simplicity, requiring only basic hardware and no external training data.

Table 3.2 The key components are detailed in Table

Component	**Specification**	**Details**	**Purpose**
Hardware	CPU	Standard PC	For general image processing tasks
	Memory (RAM)	≥ 4 GB	Sufficient for single image operations
	Camera	Webcam or Fabric Image	Image acquisition or test input
Software	Operating System	Windows/Linux/macOS	Platform for running the code

	Python	3.10.8	Programming language used
	OpenCV	4.11.0.86	Core image processing library
	Matplotlib	Any stable version	Used for visualizing image outputs
	NumPy	Any stable version	Numerical processing and array operations

3.3 PROCEDURE EXPLAINED

This section illustrates the major steps of the defect detection workflow using visual outputs generated by the implemented code. Each stage transforms the fabric image closer to highlighting possible defects.

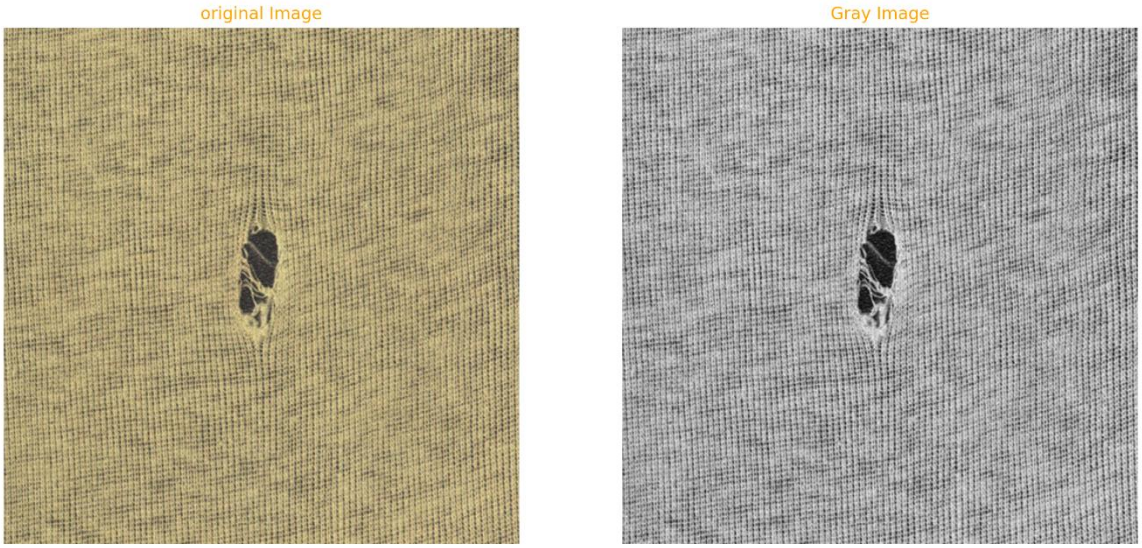

Figure 3.**1** Grayscale Conversion

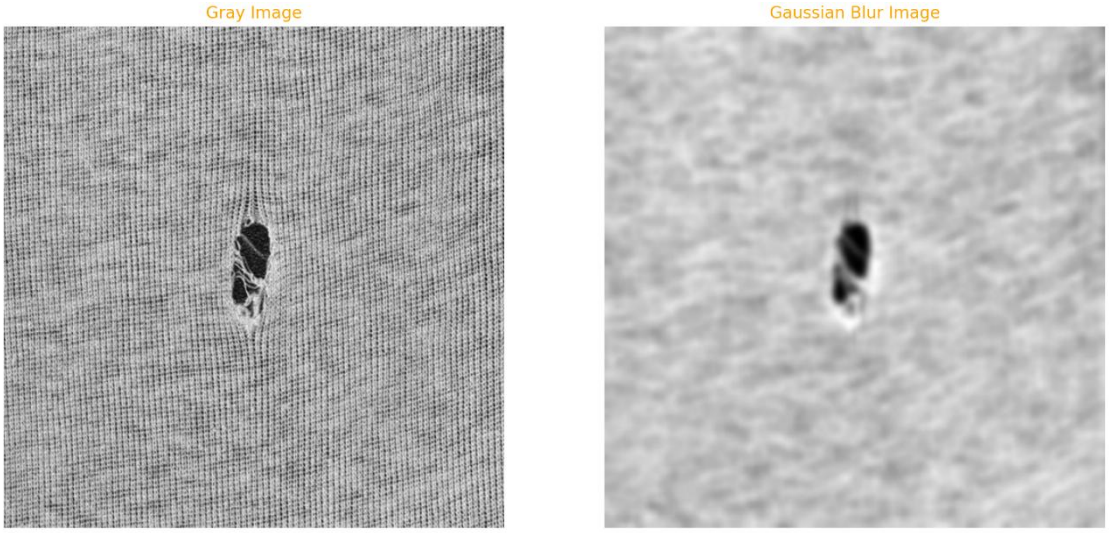

Figure 3.**2** Gaussian Blur Application

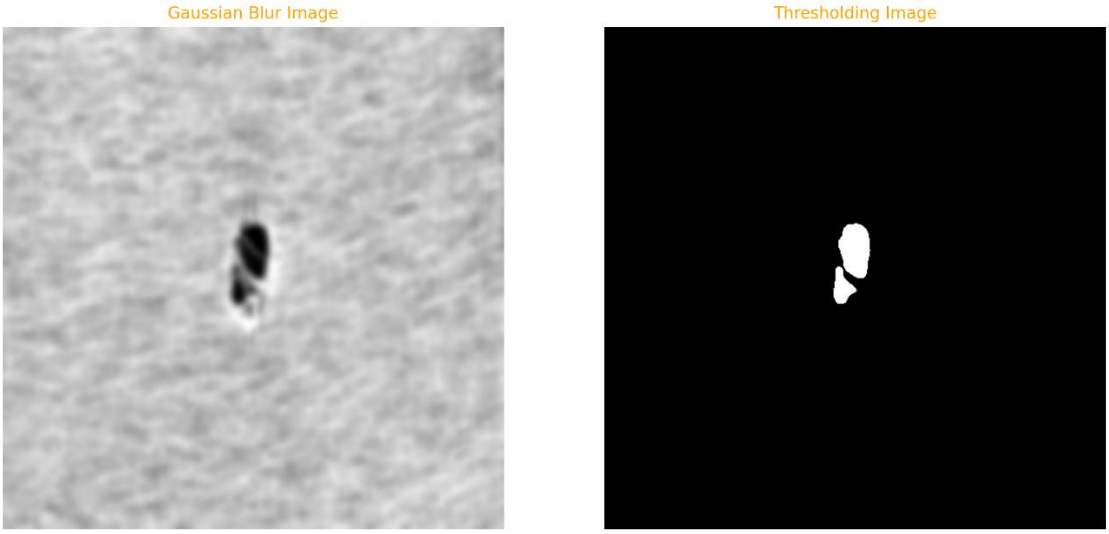

Figure 3.**3** Thresholding (Otsu's Method)

Figure 3.**4** Morphological Operations

Figure 3.**5** Contour Detection and Filtering

Figure 3.6 Final Image

This figure displays the final output of the system after completing all image processing steps. It shows a fabric image with visible damage, where the detected defect regions are clearly marked using red contours. These contours highlight the boundaries of the hole based on area and perimeter filtering from contour analysis. The image confirms that the detection workflow—including grayscale conversion, thresholding, morphological filtering, and contour selection—successfully isolates and identifies significant defects. This output serves as visual proof of the system's ability to perform accurate fabric defect detection in real-world conditions.

CHAPTER 04

RESULTS AND DISCUSSION

4.1 BACKGROUND

The results of this project are derived from testing a custom-built image processing workflow designed to identify visual defects in fabric images. The system was developed using Python and OpenCV, applying classical computer vision techniques to analyze static images of garments. The aim was to detect visible defects such as holes, tears, or irregular stitching without relying on complex machine learning algorithms. The detection pipeline included several key stages: grayscale conversion, noise reduction through Gaussian blurring, adaptive thresholding (using Otsu's method), and morphological operations for region enhancement. Contours were then extracted and filtered based on area and perimeter criteria to isolate the most significant defect regions. The output of each stage was visually analyzed using matplotlib to confirm the integrity of the process. The final results were evaluated based on visual accuracy, system responsiveness, and robustness under different image conditions. This chapter presents the main outputs of the system and evaluates its ability to correctly identify and highlight defects in clothing materials

4.2 MAJOR RESULTING STEPS

The project successfully delivered a complete and functioning fabric defect detection system using classical image processing techniques in OpenCV. The following were the key resulting outputs of the system:

Grayscale Image Generation:

The input image is first converted into grayscale, eliminating unnecessary color data to focus on structural variations in intensity.

Noise Reduction:

Gaussian blur was used to smooth the image and suppress minor pixel-level noise that might interfere with thresholding and contour extraction.

Thresholding with Dynamic Logic:

Adaptive thresholding using Otsu's method was applied. A pixel ratio–based adjustment determines whether to apply binary or inverse binary thresholding for optimal segmentation.

Morphological Enhancements:

Morphological opening removes small false areas, and dilation connects fragmented defect zones to make the shapes more detectable.

Contour-Based Defect Marking:

The final image includes only the most significant contours filtered using area and perimeter statistics, clearly outlining actual defect regions with red boundaries.

4.3 OTHER TESTED IMAGE RESULTS

In addition to the primary test image shown in the final output, the system was evaluated on several other fabric images to test its adaptability and robustness. These images included various fabric types, lighting conditions, and defect shapes such as irregular holes, misaligned stitching, and dark stains.

Plain Fabric with Hole

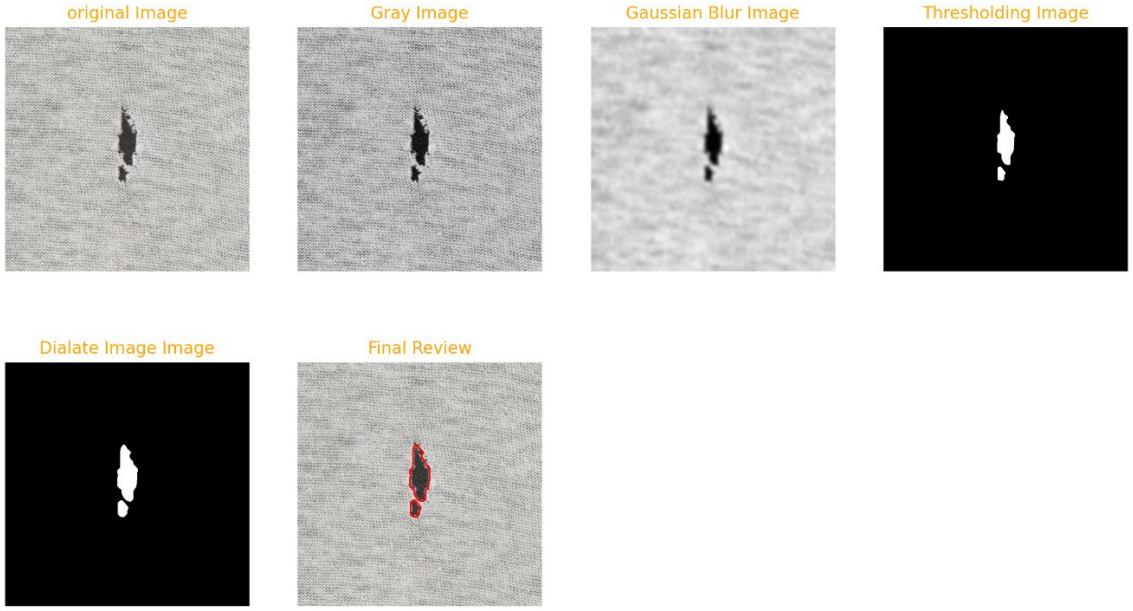

Figure 4.1 Plain Fabric with Hole image

Fabric Under Slight Shadow

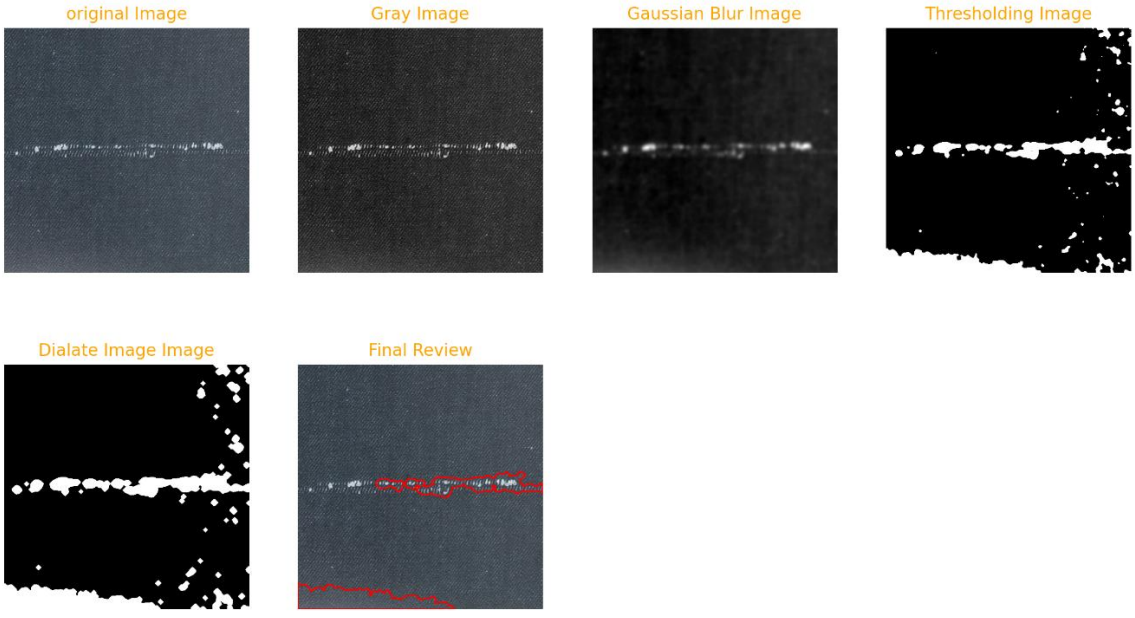

Figure 4.2 Under Slight shadow image

Dark-Colored Fabric with Faded Area

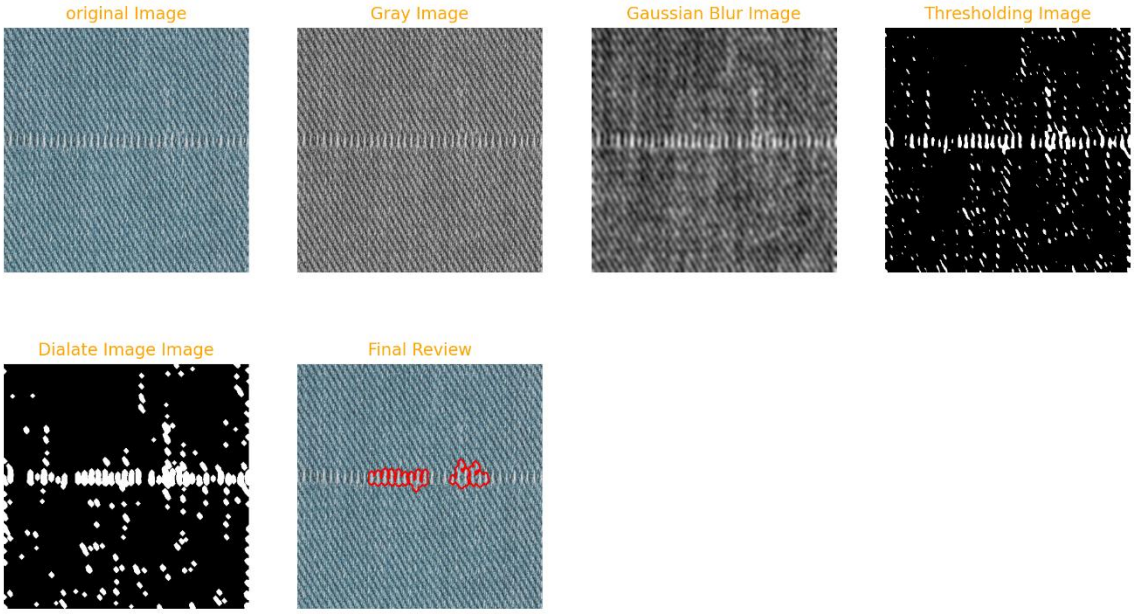

Figure 4.3 Dark-Colored Fabric with Faded Area image

4.4 DISCUSSION

The results indicate that the system performs reliably under consistent lighting and with clearly visible defects. The use of adaptive thresholding and statistical contour filtering reduces false positives and isolates meaningful defect areas. The lightweight implementation operates efficiently on low-resource devices and requires no training data, making it accessible and practical.

However, the detection quality is influenced by image conditions. Subtle defects such as minor stains or discoloration are harder to detect using only intensity-based segmentation. Likewise, irregular lighting, shadows, or wrinkles in the fabric may introduce noise or false detections. Despite these limitations, the system serves as a strong proof of concept for real-time fabric quality control.

The visual feedback at each stage supports intuitive understanding and debugging. With future improvements such as contrast enhancement, GUI integration, and support for real-time video streams, this system could evolve into a deployable tool for industrial or educational use.

LIMITATIONS DESPITE THE SUCCESSFUL RESULTS

Although the proposed fabric defect detection system performs effectively under normal conditions and meets its main objectives, several limitations remain due to the nature of classical image processing and real-world image variability:

Lighting Dependency

The system's performance is highly sensitive to lighting conditions. Strong shadows, glare, or uneven illumination can negatively affect thresholding accuracy and lead to incorrect segmentation or false positives.

Subtle Defect Detection

Minor surface defects such as light stains, faded areas, or slight texture inconsistencies may not produce enough contrast to be detected, especially in uniformly colored or textured fabrics.

Object Type Confusion

The system detects irregularities in structure but cannot classify the type of defect (e.g., hole vs. stain). Non-defective variations like fabric folds may occasionally be misidentified as faults.

Framing and Fabric Orientation

The system requires the fabric to be laid flat, evenly lit, and captured without motion blur. Wrinkles, folds, or tilted angles in the image can disrupt contour detection and create false detections.

Manual Parameter Sensitivity

While adaptive thresholding is used, the effectiveness still relies on proper image contrast and preprocessing parameters. Different fabric types or resolutions may require manual adjustment.

No Real-Time Capability

The current implementation is limited to static images. Extending the system for live camera feeds or video input would require optimization and frame-by-frame defect smoothing.

CHAPTER 05

CONCLUSIONS

5.1 KEY BENEFITS OF THE DEVELOPED SYSTEM

The implemented defect detection system offers several technical and practical benefits, making it a strong candidate for use in educational settings, small-scale textile inspections, and low-resource environments.

Cost-Effective Solution

The system is built entirely using open-source tools such as Python and OpenCV, requiring only a basic computer and a standard camera. This eliminates the need for expensive sensors or deep learning infrastructure, significantly reducing implementation costs.

Lightweight and Offline Operation

Unlike AI-based systems that require large datasets and continuous internet access, this system runs completely offline using classical image processing algorithms. It is ideal for environments where network access is limited or where data privacy is a concern.

Fully Interpretable Workflow

Each stage of the detection process—grayscale conversion, blurring, thresholding, morphology, and contour detection—is transparent and understandable. This makes the

system easy to debug, customize, and improve, especially in academic and research contexts.

Fast Processing Time

The system can process a fabric image and detect defects in approximately 1–2 seconds, enabling near real-time performance. This makes it suitable for batch image inspection or semi-automated visual quality control stations.

Modular and Extendable

The system design is modular, allowing individual components (e.g., thresholding method, contour filtering logic) to be modified or replaced. This flexibility supports adaptation to different types of fabric, lighting conditions, or application-specific requirements.

Visual Output for Verification

Each stage of the detection process is visualized using `matplotlib`, enabling easy interpretation and validation of the results. The final output image clearly highlights detected defect regions with red contours, making it intuitive for users to understand and trust the system's decisions.

5.2 SYSTEM LIMITATIONS

While the developed system achieved its primary objective of detecting fabric defects using classical image processing, several issues were encountered throughout implementation and testing:

Lighting Sensitivity

The accuracy of the detection heavily depends on uniform lighting conditions. Shadows,

glare, or inconsistent brightness across the fabric surface often led to incorrect thresholding results, causing false positives or missed defects.

Limited Detection of Subtle Defects

Defects that are faint, low in contrast, or embedded in complex textures (e.g., light stains or faded areas) were difficult to isolate. Since the detection is intensity-based, subtle variations often went unnoticed without contrast enhancement.

Background Noise and False Contours

In some cases, fabric texture, wrinkles, or stitching patterns were misinterpreted as defects, especially when their structure mimicked edges or irregular shapes. This was partially mitigated using contour filtering but not entirely eliminated.

Dependency on Centered and Flat Images

The system requires input images to be well-centered and captured from a flat angle. Tilted, folded, or curved fabrics reduced accuracy by distorting contour shapes and shadow distribution.

Lack of Real-Time Capability

The current implementation only supports single-image analysis. Real-time video input or continuous scanning is not yet supported, limiting the system's use to static inspections.

No Defect Type Classification

While the system can detect irregularities, it does not classify the type of defect (e.g., hole, stain, or loose thread), which would be useful in production environments for quality logging.

5.3 CONCLUSION

The project successfully demonstrates the feasibility of using classical image processing techniques to detect visible defects in fabric materials. By combining grayscale conversion, Gaussian blurring, adaptive thresholding, morphological operations, and contour analysis, the system was able to identify structural anomalies such as holes and misaligned stitching with a high degree of visual clarity. The use of OpenCV provided a flexible and lightweight framework that did not require any external datasets or machine learning models.

The final output images clearly marked defect regions, and intermediate visualizations confirmed the correctness of each processing stage. While the system is effective under well-lit and controlled conditions, several limitations such as lighting sensitivity and lack of defect classification were identified. Despite these challenges, the project meets its core objectives and provides a strong foundation for future improvements.

Overall, the system offers a transparent, cost-effective, and modular solution for fabric defect detection and can serve as a valuable tool in both educational and practical textile quality control applications.

REFERENCES

Youtube Tutorials:

https://www.youtube.com/watch?v=4iZSWHpqawc

https://www.youtube.com/shorts/HEvdsLgDUy8

2.OpenCV Documentation:https:

//docs.opencv.org//