Dimensional Analysis for Quality Control in Manufacturing using Blob Detection

Aayush Rijal
Memorial University of Newfoundland
202292045

Thevananthan Thevarasa
Memorial University of Newfoundland
202094858

***Abstract*— [229 Word] The report presents a robust implementation of dimensional analysis for quality control in manufacturing and production industries using advanced blob detection techniques. The project leverages OpenCV, a powerful computer vision library, to process manufacturing images and extract essential dimensional information. The methodology encompasses image acquisition, preprocessing, morphological operations, image segmentation, and blob detection to identify and measure components in the images. The image preprocessing stage involves converting images to grayscale and applying noise reduction and contrast enhancement techniques to ensure cleaner and more reliable data. Morphological operations, including closing, erosion, and dilation, further enhance blob detection accuracy. Image segmentation techniques, like connected component analysis, efficiently identify and label individual blobs or regions of interest. Subsequently, dimensional analysis measures crucial blob characteristics, such as area, perimeter, centroid, and bounding box, to evaluate compliance with specified tolerances, ensuring product quality and consistency. While the segmentation approach encountered challenges in overlapping object scenarios and poor lighting conditions, the project effectively addresses such issues by implementing a minimum-size filtering mechanism to eliminate noise artifacts. Overall, this implementation of dimensional analysis contributes to better quality control practices in manufacturing industries by automating defect detection and dimensional analysis, reducing human errors in inspection tasks, and enhancing product consistency. The report showcases the step-by-step implementation and presents the results of the dimensional analysis, highlighting the project's effectiveness and practicality in improving quality control in manufacturing processes.**

Index Terms*— Image Acquisition, Image Processing, blob detection, Segmentation.

I. INTRODUCTION

[230 Words] In manufacturing industries, dimensional analysis plays a crucial role in ensuring product quality and conformity to specifications. However, performing dimensional analysis manually is time-consuming and error prone. Automating this process plays a critical role in mass scaling production.

This challenge aims to develop an image processing solution using blob detection techniques to automate dimensional analysis for quality control in manufacturing. The objective is to preprocess the images, apply morphological image operations for noise reduction, perform image segmentation to identify individual objects, measure the dimensions of blobs accurately, and analyze them in accordance with predefined specifications. By automating this process, manufacturers can improve efficiency, reduce errors, and maintain consistent quality standards.

The challenge lies in accurately detecting and measuring complex blob shapes, dealing with variations in lighting and background, and handling potential image noise and artifacts. Some of the Potential Applications are given below: -

1. Food Production: The solution can be used in food production for baked goods, packaged food items, and fruits/vegetables, ensuring consistent portion sizes, packaging integrity, and product quality.

2. Pharmaceutical Industry: The solution can analyze the dimensions of tablets and capsules, including their diameter, thickness, and shape consistency.

3. Consumer Goods Manufacturing: The solution can be used for quality control in the production of bottles, cans, and packaging materials. It can measure parameters such as diameter, height, and thickness of these containers, ensuring consistent sizes, shapes, and structural integrity.

II. LITERATURE REVIEW

[750 Words] Several relevant works were cited in this field. We reviewed some of the work and summarized it in detail.

1. “A Real-Time Approach for Automatic Food Quality Assessment Based on Shape Analysis” [1]:

The journal titled "A Real-Time Approach for Automatic Food Quality Assessment Based on Shape Analysis" by Luca Donati, Eleonora Iotti, and Andrea Prati addresses the importance of accurate product sorting in the agricultural industry. The authors highlight the significance of quality control measures to prevent the wastage of good products and ensure the proper disposal of rotten, broken, or deformed food items. They emphasize that existing sorting systems primarily rely on color information, which may not be sufficient to detect certain common defects. In contrast, the shape of a product can reveal important defects and is highly reliable in detecting foreign objects mixed with food. Moreover, shape analysis enables detailed measurements of a product, such as its area, length, width, and anisotropy. The paper proposes a comprehensive solution for sorting food based on shape analysis, considering real-world challenges such as accuracy, execution time, and latency. It provides an overview of a complete system implemented on advanced measurement machines, addressing the need for reliable and efficient food sorting based on shape characteristics.

2. “Smart manufacturing applications for inspection and quality assurance processes” [2]:

The conference paper titled "Smart manufacturing applications for inspection and quality assurance processes" by Maremys Galindo-Salcedo, Altagracia Pertúz-Moreno, Stefania Guzmán-Castillo, Yulineth Gómez-Charris, and Alfonso R. Romero-Conrado discusses the significant impact of smart manufacturing on inspection and quality assurance processes, specifically focusing on innovative technologies in machine learning. The paper presents a systematic review of automation applications in statistical quality control within industrial companies. The subtopics covered include artificial vision, intelligent manufacturing, inspection across various production processes, neural networks, automation using statistical process control techniques, and quality assurance. The authors analyze these technologies, highlighting their ability to improve automated manufacturing processes by enhancing efficiency, performance, and productivity. Furthermore, these technologies contribute to time optimization, cost reduction, strengthened inspection procedures, and quality assurance. The paper concludes by identifying future research opportunities for industrial applications in the field.

3. "Segmentation Techniques for Rotten Fruit detection" [3]:

The conference paper titled "Segmentation Techniques for Rotten Fruit Detection" by K. Roy, S. S. Chaudhuri, S. Bhattacharjee, S. Manna, and T. Chakraborty focuses on the development of segmentation techniques for the detection of rotten vegetables. The authors highlight the importance of automating the sorting process to distinguish between fresh and rotten vegetables, addressing potential health risks associated with consuming rotten produce. The paper presents three segmentation techniques: Marker-Based Segmentation, Color-Based Segmentation, and Edge Detection. These techniques effectively identify and isolate the rotten portions of vegetables, enabling the separation of unhealthy vegetables from the good ones. By implementing an automated system that incorporates these segmentation techniques, the sorting process for food product manufacturing units can be significantly improved in terms of time, manpower, and accuracy. The proposed techniques underwent a multi-level analysis and were evaluated using sets of images containing both healthy and rotten vegetables. The experimental results validate the efficacy of the suggested segmentation techniques for detecting and sorting rotten vegetables, thereby enhancing food safety and quality assurance processes.

4. **"**Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables" [4]:

The journal titled "Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables" by C. Sergio, A. Nuria, M. Enrique, G.-S. Juan, and B. Jose discusses the advancements in artificial vision systems for the automatic inspection and quality evaluation of fruits and vegetables. The authors highlight the various applications of these systems, including grading, quality estimation based on external parameters or internal features, monitoring fruit processes during storage, and evaluating experimental treatments. Artificial vision systems offer capabilities beyond human capacity, allowing for objective evaluation of long-term processes and detection of events outside the visible electromagnetic spectrum. By utilizing ultraviolet or near-infrared spectra, these systems can explore defects or features that are invisible to the human eye. Hyperspectral systems provide detailed information about individual components or damage, enabling the development of new computer vision systems tailored to specific objectives. In-line grading systems can inspect large quantities of fruit or vegetables individually, providing statistical data about the entire batch. Overall, artificial vision systems not only replace human inspection but also enhance its capabilities. This work presents the latest developments in applying this technology to inspecting the internal and external quality of fruits and vegetables, showcasing the potential for improved quality control and evaluation in the agricultural industry.

III. Methodology

This section of the paper focuses on detailed steps of the project which includes Image Acquisition, Image Preprocessing, Morphological Image Operations, Image Segmentation and valid Blobs Detection, and Dimensional Analysis and Filtering. The detailed explanation of each step is illustrated below: -

A. *Image Acquisition: -*

[123 word] Since this project is focused on Dimensional Analysis for Manufacturing or Production we have chosen "Food Production" to apply the developed system. Capturing high-resolution images of manufactured/production objects using appropriate imaging techniques and equipment with consistent lighting conditions are highly time consuming and hard to get access with the given context for the project under this Module. Therefore, we used *DALL·E 2* developed by OpenAI to generate the required dataset for the project with desired environment conditions.

DALL·E 2 is an AI system that can create realistic images and art from a description in natural language. URL: https://openai.com/dall-e-2

We focused on Generating two types of Datasets: Blueberries Production and Potato Chips Manufacturing. Few Samples of Datasets looks like as shown in Fig 1,2,3,4.

Figure 1 Blueberries data set image sample

Figure 2 Sample of blueberries dataset image

Figure 3 Sample image of Potato-Chips on Dataset

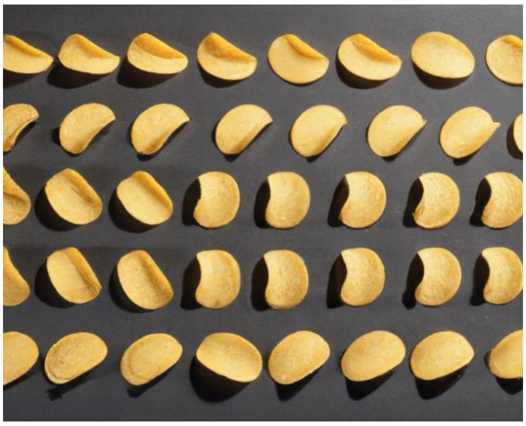

Figure 4 Another Sample of Potato Chips on Dataset

B. *Image Preprocessing.*

[365 word] After we obtain the image, we will process the image further. The purpose of image preprocessing is to enhance the image quality, remove noise, correct distortions, and extract relevant features, making it more suitable for subsequent tasks such as object detection, recognition, or image analysis. The detailed steps for image processing are given below: -

1) *Grayscale Conversion: -*

The first step of the Image Processing is the Gray Scale Conversion. In OpenCV, grayscale conversion is achieved by using the cv2.cvtColor() function with the parameter cv2.COLOR_BGR2GRAY. This function takes a color image as input and returns a single-channel grayscale image where each pixel value represents the intensity of the corresponding pixel in the original image.

2) *Noise Removal and Contrast Enhancement:-*

Secondly, the grayscale image is subjected to the Noise Removal step and goes through the Contrast Enhancement. Median blur and Gaussian blur are both image filtering techniques used for noise reduction and image smoothing in image processing[5]. Each technique applies a kernel or a window over the image to calculate the filtered pixel value based on neighboring pixels.

Contrast enhancement enhances the contrast of the images using histogram equalization to improve the visibility of the components or features. Histogram equalization, as done in OpenCV using enhanced_image = cv2.equalizeHist(blurred), is performed to enhance the contrast of the blurred image. By redistributing the pixel intensities across the entire range, histogram equalization stretches the intensity values, making the image visually more vibrant and improving the visibility of details in different regions.

The implementation of these steps in code is illustrated below: -

```
def preprocess(gray, blur, enhance):

    enhanced_image = None

    # Noise Filtering based on the Filter
    if blur == "gaussian":
        # Apply Gaussian blur to reduce noise
        enhanced_image = cv2.GaussianBlur(gray,
(11, 11), 0)
    elif blur == "median":
        # Apply Median blur to reduce noise
        enhanced_image = cv2.medianBlur(gray, 11)

    if enhance:
        # Contrast Enhancement
        enhanced_image =
cv2.equalizeHist(blurred)  # Apply histogram
equalization for contrast enhancement

    return enhanced_image
```

In summary, the "preprocess" function takes a grayscale image and applies noise filtering (Gaussian blur or Median blur) based on the "blur" parameter. Optionally, it enhances the image's contrast using histogram equalization, based on the "enhance" parameter. The resulting preprocessed and, if specified, enhanced image is then returned.

3) *Binarization/Thresholding.*

Global thresholding and adaptive thresholding are image binarization techniques tested for this application to separate the foreground objects (components of interest) from and background regions based on pixel intensities. Global thresholding is selected for our application because the images have consistent (controlled environment) foreground and background intensities across the entire image. It is computationally efficient and straightforward to implement.

The implementation of thresholding is as given below: -

```
# Performed thresholding on the blurred image to
create a binary image using cv2.threshold().
# segment the foreground objects from the
background.

# Apply Global Thresholding
def binary_threshold(image, threshold_value):

    # Apply binary thresholding
    _, binary_image = cv2.threshold(image,
threshold_value, 255, cv2.THRESH_BINARY_INV)

    return binary_image

# Apply Adaptive Thresholding
def adaptive_threshold(image, block_size,
constant):

    # Apply adaptive thresholding
    binary_image = cv2.adaptiveThreshold(image,
255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                          cv2.THRE
SH_BINARY_INV, block_size, constant)

    return binary_image
```

C. Morphological Image Operations:

[206 word] The following Morphological Processing Techniques were used to fill the holes in the blobs, separate the blobs from each other and to alter the shapes of blobs as required.

Morphological Closing: Morphological closing is an operation that combines dilation followed by erosion. It is used to fill gaps in the detected blobs and smooth the blobs. In this application, a square-shaped kernel of size (11, 11) is used for the closing operation. The cv2.morphologyEx() function is applied with cv2.MORPH_CLOSE as the operation type.

Morphological Erosion: Morphological erosion is an operation that reduces the size of bright regions (white areas in this case). It helps distinguish blobs that are close to each other. In this application, a smaller square-shaped kernel of size (3, 3) is used for the erosion operation. The cv2.erode() function is applied with iterations=1 to perform one iteration of erosion.

Morphological Dilation: Morphological dilation is an operation that increases the size of bright regions. It is used here to fill small gaps inside each blob, which might have occurred due to thresholding or erosion. In this application, a larger square-shaped kernel of size (5, 5) is used for the dilation operation. The cv2.dilate() function is applied with iterations=1 to perform one iteration of dilation.

The implementation of each step i.e., closing, erosion, dilation is given in the form of python code: -

```
def morphology(thresh, cl_ker, er_ker, di_ker):
    # Perform morphological closing to fill gaps
and smooth the blobs
    kernel = np.ones((cl_ker, cl_ker), np.uint8)
    closing = cv2.morphologyEx(thresh,
cv2.MORPH_CLOSE, kernel)

    # Perform morphological erosion to
distinguish blobs
    kernel_erosion = np.ones((er_ker, er_ker),
np.uint8)
    erosion = cv2.erode(closing, kernel_erosion,
iterations=1)

    # Perform morphological dilation to fill
small gaps inside each blob
    kernel_dilation = np.ones((di_ker, di_ker),
np.uint8)
    dilation = cv2.dilate(erosion,
kernel_dilation, iterations=1)

    # Set the Processed Image for Connected
SComponent Analysis
    processed_img = closing

    return processed_img
```

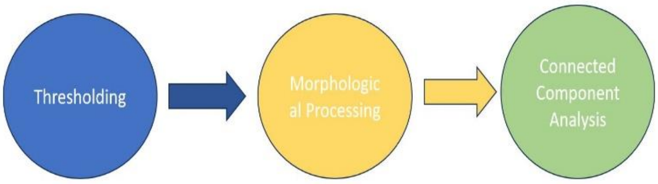

Figure 5 Steps for blob Detection

D. Image Segmentation and valid Blobs Detection.

[72 Word] The Segmentation of the foreground objects from the background is executed in the following order as shown in fig. 5.

A Valid blob is considered based on the application criteria by defining a minimum area to be eligible for a valid blob for the specific application. The Minimum Area for valid detection is configurable via command line arguments. The measurement metrics and filtering of these blobs are discussed in the next section.

E. Dimensional Analysis and Filtering.

[220 word] The End goal of the application is to perform the Dimensional Analysis and Filtering of the objects to evaluate certain statistics and information about the production or Manufacturing, report the evaluations or visualize them.

Statistical Analysis, Reporting, and Visualization: Perform statistical analysis on the collected dimensional measurements to evaluate variations, trends, and distributions. Generate reports or visual representations summarizing the dimensional analysis results, such as statistical charts, histograms, or control charts, to present the measured dimensions, deviations, and trends.

The following measurements were obtained during the analysis.
A. Area Calculation: Measured the area of each selected blob, representing the size or extent of the component.

B. Perimeter Calculation: Measured the perimeter of each blob, representing the boundary length.

C. Centroid Calculation: Determined the centroid (center of mass) of each blob, providing the spatial position information.

D. Bounding Box Calculation: Determine the bounding box dimensions of each blob, enclosing the component within a rectangular box.

For this demonstration using a batch of blueberries, the following simple filtering criteria is defined:

i. Threshold value to detect Oversized blueberries. (Configurable via command line)
ii. Threshold value to detect Undersized blueberries. (Configurable via command line)

Analyzed the measured dimension (Area) to assess their compliance with the specified tolerances. Compare the measurements against predefined criteria or dimensional specifications to determine whether the product meets quality control standards.

The code implementation for the blob detection, dimensional analysis, and filtering:

```
def detectAndMeasure(org_img, proc_img, blobMin,
filtMax, filtMin):

    # Count the objects which satisfies the constraints
    filt_stats = {
        "tot_valid_blobs": 0,
        "tot_under_Sized": 0,
        "tot_over_Sized" : 0,
        "percent_under":0,
        "percent_over":0,
    }

    # Perform connected component analysis
    num_labels, labels, stats, centroids =
cv2.connectedComponentsWithStats(proc_img,
connectivity=8)

    # Print Total Number of Blobs Detected
    print(f"Total Blobs Count: {num_labels}")

    # Define the minimum and maximum area thresholds to
filter the blobs (adjust these values as needed)
    min_area_threshold = blobMin

    # Define the Filtering Blob Size for Measurement
Analysis
    over_sized = filtMax
    under_sized = filtMin

    # Copy the original image for different markings
    org_img_copy = org_img.copy()

    # Loop through each detected blob
    for label in range(1, num_labels):  # Start from 1
to exclude the background label 0
        area = stats[label, cv2.CC_STAT_AREA]

        # Check if the area is within the specified
range
        if min_area_threshold < area:

            filt_stats["tot_valid_blobs"] += 1

            # Get the bounding box coordinates for the
blob
            x, y, w, h = stats[label,
cv2.CC_STAT_LEFT], stats[label, cv2.CC_STAT_TOP], \
                        stats[label,
cv2.CC_STAT_WIDTH], stats[label, cv2.CC_STAT_HEIGHT]

            # Draw the bounding box around the blob
based on Filtered Criteria

            # Mark All the blobs valid except
background
            cv2.rectangle(org_img_copy, (x, y), (x + w,
y + h), (0, 0, 0), 2)

            # Mark Bounding Box complying Filtering
Condition
            if  area < under_sized:
                filt_stats["tot_under_Sized"] +=1
                # Identify with BLUE bounding box
                cv2.rectangle(org_img, (x, y), (x + w,
y + h), (255, 0, 0), 2)
            elif area >= over_sized:
                filt_stats["tot_over_Sized"] +=1
                # Identify with RED bounding box
                cv2.rectangle(org_img, (x, y), (x + w,
y + h), (0, 0, 255), 2)
            else:
                # Identify with GREEN bounding box
                cv2.rectangle(org_img, (x, y), (x + w,
y + h), (0, 255, 0), 2)

            # Print the area of the blob
            #print(f"Blob {label}: Area = {area}
pixels")


    # Calculate the Percentage of Over and Under Sized
batching
    if filt_stats["tot_valid_blobs"] > 0 :
        filt_stats["percent_under"] =
round((filt_stats["tot_under_Sized"]/filt_stats["tot_va
lid_blobs"])*100, 2)
        filt_stats["percent_over"] =
round((filt_stats["tot_over_Sized"]/filt_stats["tot_val
id_blobs"])*100, 2)
    else:
        filt_stats["percent_under"] = 0
        filt_stats["percent_over"] = 0

    return org_img_copy, org_img, filt_stats
```

IV. Results. [900 Words]

A. Grayscale Conversion:

Input Image:

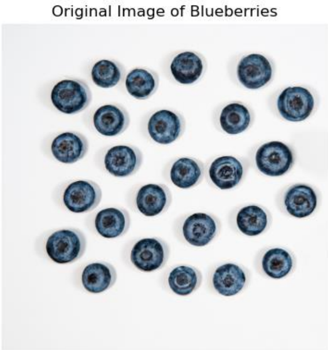

Figure 6 Original Input Image RGB

Output Image:

Figure 7 Grayscale Converted Image

Grayscale conversion involves converting a color image (commonly represented in the RGB color space) into a single-channel grayscale image. The grayscale image represents the intensity of each pixel, ranging from 0 (black) to 255 (white).

It's observed that Each pixel in the grayscale image carries only one value that corresponds to the brightness of the corresponding pixel in the color image.

B. Noise Removal and Contrast Enhancement:

Output:

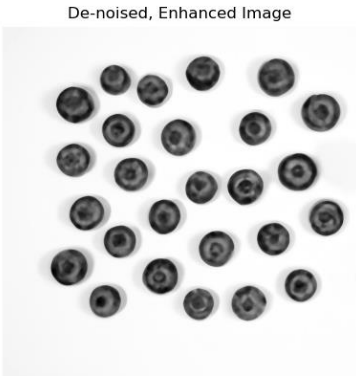

Figure 8 Noise Removed, Enhanced Image

We can see the foreground objects are smoother and more blurred after noise filtering. The application tested both types of blurring and selected Median blurring for the application because it effectively removes salt-and-pepper noise while preserving the edges and features of the objects being analyzed.

Unlike Gaussian blur, which may blur edges and boundaries, median blur replaces noisy pixels with the median value of the neighborhood, ensuring that extreme values caused by noise do not affect the overall analysis.

By incorporating noise removal and contrast enhancement in image processing applications, the resulting images are cleaner, more visually appealing, and better suited for subsequent analysis tasks, such as object detection, segmentation, and feature extraction.

C. Binarzation/Thresholding:

Output:

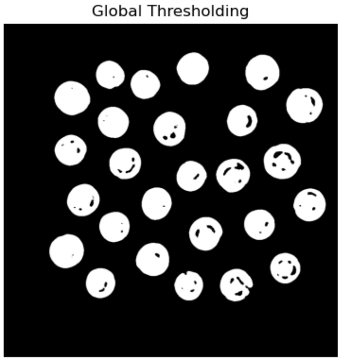

Figure 9 Thresholded Image - Global

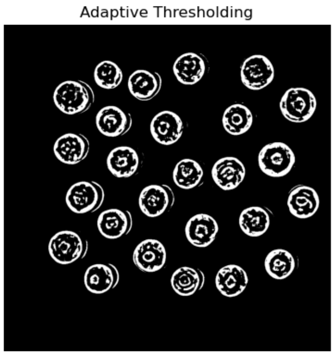

Figure 10 Thresholded Image – Adaptive

We have tested both Global thresholding and adaptive thresholding for image binarization to separate foreground objects from background regions based on pixel intensities.

Global Thresholding observations:

- Foreground objects are preserved more.
- Simple method using a single threshold value for the entire image.
- Assigns pixels below the threshold to the background and above to the foreground.
- Suitable for images with uniform lighting and consistent foreground-background separation.

Adaptive Thresholding Observations:

- Edges are preserved better than global thresholding, but the inner body is more eroded than global thresholding.

- Calculates different threshold values for different regions of the image.
- Considers local pixel intensity variations by dividing the image into smaller regions.
- Effective for images with lighting variations, shadows, or uneven backgrounds.

Therefore, Global thresholding is chosen due to the images having consistent foreground-background intensities in a controlled environment. It offers computational efficiency and easy implementation.

D. Morphological Image Operations:

Input Image:

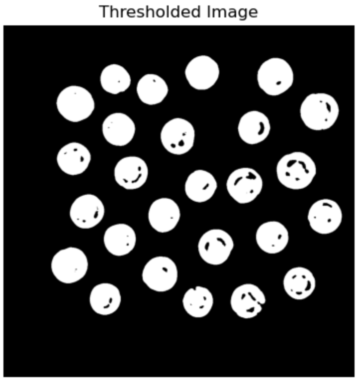

Figure 11 Input image for Morphological processing

Output Image:

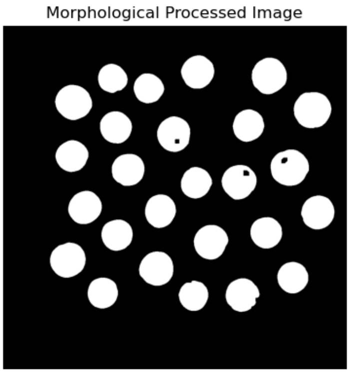

Figure 12 Output image after Morphological Processing

From the output image we can observe that the holes and non-uniformities from the thresholding are recovered from the morphological processing.

The following Morphological Processing Techniques are utilized for the application.

Morphological closing, achieved through dilation followed by erosion using an 11x11 square-shaped kernel, is employed to fill gaps in blobs and smoothen their boundaries.

Morphological erosion with a smaller 3x3 square kernel reduces the size of bright regions, aiding in distinguishing closely positioned blobs.

On the other hand, morphological dilation, employing a larger 5x5 square kernel, enlarges bright regions to fill small gaps within blobs caused by thresholding or erosion.

By employing morphological closing, erosion, and dilation in sequence, the accuracy of blob detection and the separation of adjacent blobs are significantly improved. Morphological closing fills gaps and smooths blobs, erosion aids in distinguishing closely connected blobs, and dilation ensures that small gaps within each blueberry are filled, leading to more precise and accurate detection of individual blobs in the final output.

E. Image Segmentation and valid Blobs Detection:

Figure 13 Blobs detected and added bounding boxes.

Bounding boxes of black color were calculated and masked on top of the original image to verify the blob detection and bounding box calculations.

Each blob marked with bounding boxes is segmented from the background in a stepwise manner. First, thresholding is applied to convert the preprocessed image into a binary image based on intensity, using either global or adaptive thresholding. Next, morphological processing is performed, including closing, erosion, and dilation, to enhance blob detection accuracy and separate adjacent blobs.

Connected component analysis is then conducted on the binary image to identify and label individual blobs or regions of interest. A blob is considered valid based on application criteria, with a minimum area threshold that can be configured via command-line arguments.

F. Dimensional Analysis and Filtering.

The aim of the application is to conduct dimensional analysis and filtering of objects to assess specific statistics and manufacturing information. This involves performing statistical analysis on collected dimensional measurements to identify variations, trends, and distributions. The results are then presented through reports or visual representations like statistical charts, histograms, or control charts to showcase the measured dimensions, deviations, and trends.

Statistical Analysis Output:

```
Measurement Analysis Statistics of the blobs
============================================
Total Valid Objects in the Image: 26
Total Over Sized Objects in the Image: 8
Total Under Sized Objects in the Image: 1
Percentage of LOW Sized products in the batch: 3.85%
Percentage of OVER Sized products in the batch: 30.77%
```

Figure 14 Statistical Analysis output

Visualization: Filtered Blobs

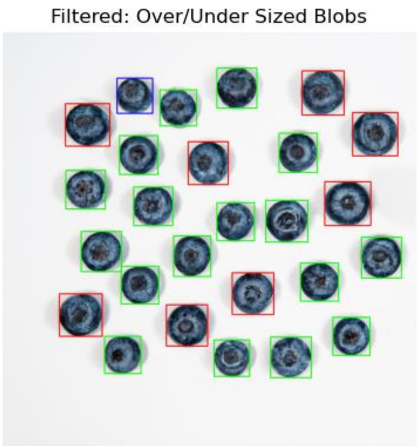

Figure 15 Visualization of Filtered Blobs

Observation:

- Undersized blobs: Blue bounding Box.
- Oversized blobs: Red bounding Box
- Acceptable range Blobs: Green bounding box.

For this demonstration using a batch of blueberries, the following simple filtering criteria is defined:

- Threshold value to detect **Oversized** blueberries. (**Configurable** via command line)
- Threshold value to detect **Undersized** blueberries. (**Configurable** via command line)

Analyzed the measured dimension (Area) to assess their compliance with the specified tolerances. Compare the measurements against predefined criteria or dimensional specifications to determine whether the product meets quality control standards.

G. Unsuccessful Result.

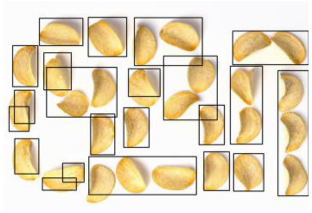

Figure 16 unsuccessful blob detection

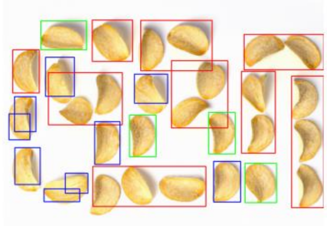

Figure 17 Unsuccessful measurement Analysis

While testing a batch of potato chips, we faced unsuccessful blob detection which results in false measurement analysis. The issue occurred when the segmentation process fails to accurately separate neighboring chips due to improper thresholding and morphological processing. This failure lead to connected components, where adjacent chips become fused together into a single blob during connected component analysis.

The issue of improper segmentation may arise in cases where the intensity or color variations between neighboring chips are subtle, making it difficult for the thresholding technique to distinguish them effectively. As a result, a single threshold value might not be sufficient to differentiate adjacent chips, causing them to be grouped together as a single object during connected component analysis.

Additionally, the application of morphological operations, such as dilation and erosion, can further exacerbate the problem if the kernel sizes are not appropriately chosen. Incorrect kernel sizes may cause neighboring chips to merge or erode, leading to distorted or fused blob shapes.

Due to the merging of neighboring chips during the segmentation process, filtering criteria based on blob size or shape might not be effectively applied. Consequently, the measurement and analysis of individual chips' dimensions become unreliable, potentially impacting the quality control assessments and batch evaluation for manufacturing.

V. Conclusion: -

[465 Words] The implemented dimensional analysis for quality control in manufacturing using blob detection, focusing on image processing techniques such as noise reduction, thresholding, morphological operations, and image segmentation, has demonstrated both effectiveness and limitations.

In scenarios where objects are well-separated and lighting conditions are favorable, the effectiveness of the project becomes evident. The blob detection algorithm accurately and reliably identifies the relevant components or features, enabling precise dimensional analysis. This success is particularly pronounced in datasets with spherical objects, such as blueberries, where the distinctive shapes and sufficient spacing between each object facilitate accurate blob detection and dimensional measurements.

The successful parameter ranges identified during the project contribute significantly to the robustness and accuracy of the dimensional analysis. The best global thresholding range of 135 to 150 has proven to be effective in segmenting objects from the background and generating binary images suitable for further processing.

The choice of parameter values for morphological operations has been critical in ensuring precise blob detection. The median filter with a kernel size of 11 effectively reduced noise and improved the quality of the preprocessed images. A closing kernel size of 11x11 and a dilation kernel size of 5 assisted in filling gaps and restoring blob shapes, respectively. Additionally, an erosion kernel size of 3 helped to smooth the blob contours.

However, it is worth noting that the segmentation approach faced challenges in cases where objects overlapped with each other or when the lighting conditions were poor. These challenging scenarios led to a degradation in the performance of the segmentation technique, resulting in inaccuracies in blob detection and potentially impacting the dimensional analysis results. Despite these limitations, the project has successfully demonstrated the practicality and relevance of employing image processing techniques in quality control for manufacturing.

One of the project's strengths lies in its ability to address noise artifacts effectively. By setting the minimum size (area) for a valid blob, such as 1000 pixels in this case, the system efficiently filtered out small noise artifacts and ensured that only significant blobs were considered during the analysis. This filtering mechanism enhanced the reliability of the dimensional analysis by focusing on meaningful components while reducing the impact of undesired artifacts. Thus, the incorporation of the minimum size criterion has proved to be valuable in improving the accuracy and validity of the measured dimensional data.

In conclusion, the dimensional analysis for quality control in manufacturing using blob detection and image processing techniques has proven effective within certain conditions and parameter ranges. While the project successfully automates dimensional analysis and defect detection, further improvements are possible by addressing challenges related to object overlapping and varying lighting conditions. This work sets the foundation for enhancing quality control practices in manufacturing industries, opening avenues for future research in developing more robust and adaptable image processing algorithms.

References

[1] D. Luca, D. Luca and A. Prati, "A Real-Time Approach for Automatic Food Quality Assessment Based on Shape Analysis," International Journal of Computational Intelligence and Applications, vol. 20, p. 03, 2021.

[2] M. Galindo-Salcedo, A. Pertúz-Moreno, S. Guzmán-Castillo and A. R. Romero-Conrado, "Smart manufacturing applications for inspection and quality assurance processes," in 12th International Conference on Emerging Ubiquitous Systems and Pervasive Networks / 11th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare, Colombia, 2022.

[3] K. Roy, S. Chaudhuri, S. Bhattacharjee and S. Manna, ""Segmentation Techniques for Rotten Fruit detection,"," in 2019 International Conference on Opto-Electronics and Applied Optics (Optronix), Kolkata, India, 2019.

[4] C. Sergio, A. Nuria, M. Enrique, G.-S. Juan and B. Jose, "Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables," Food and Bioprocess Technology, vol. 4, p. 487–504, 2021.

[5] D. Van De Ville, M. Nachtegael, D. Van der Weken, E. E. Kerre, W. Philips and I. Lemahieu, "Noise reduction by fuzzy image filtering," in IEEE Transactions on Fuzzy Systems, vol. 11, no. 4, pp. 429-436, Aug. 2003