Development of Machine Vision Algorithm for Metrology and Inspection of Mechanical Components by using Image Processing.

Pranav Salvi[1], Rahul Patil[2], Swapnil Nehe[3], Taranjyot Singh Birdi[4], Bhagyashri Badgujar[5]

[1]*Student, Mechanical Engineering, K.K. Wagh Institute of Engineering Education and Research, Maharashtra, India*
[2]*Student, Mechanical Engineering, K.K. Wagh Institute of Engineering Education and Research, Maharashtra, India*
[3]*Student, Mechanical Engineering, K.K. Wagh Institute of Engineering Education and Research, Maharashtra, India*
[4]*Student, Mechanical Engineering, Guru Gobind Singh College of Engineering and Research Centre, Maharashtra, India*
[5]*Student, Mechanical Engineering, K.K. Wagh Institute of Engineering Education and Research, Maharashtra, India*

---***---

Abstract *High efficiency, greater quality, and less labour are all requirements of Industry 4.0. Currently all types of high-tech automated machine shops are all contributing to increased production. Machine shops and automated machinery are getting more productive these days. Quality inspection machines are also being developed in order to deliver faster and more accurate results. Direct and Indirect measurement techniques are more advantageous for product inspection and measurement. Machine vision is a new type of technology that is used for automatic measurement and analysis of different quantities. It can achieve this by using various image processing techniques and hardware. The internal diameter of a hole is the most critical component of an assembly. This project shows how to measure and inspect the hole diameter using machine vision system. A robust and novel algorithm for machine vision inspection has been developed with the use of Python programming language. This system is ideal for fast moving conveyor products. Manual measurement methods and camera-equipped machine vision measurement results are combined and validated according to the newly developed process*

Keywords: **Machine Vision, Image Processing, Metrology, Quality Inspection, Python Programming language**

1. INTRODUCTION

Human Beings have become more dependent on advance technology. Automation has provided human beings readily available platform to boost productivity, quality, reduced time and satisfy customers expectations, reduced costs and improve safety factors. The fundamental goal of implementing technology is to improve the quality of human existence while reducing physical as well as mental stress. This project took an integrative way of dealing with industrial applications, merging strategic use of mechanical principles, electronics engineering, and computer science algorithms to satisfy the ever-increasing demand for automated processes in Industry 4.0.

Understanding the precise dimensions of objects remotely has been incredibly valuable in past decades, especially in the automation technology and control systems industries.

The original and most obvious approach is direct distance measurement; however it cannot be used in many situations. As a result, multiple indirect distance measurement methods been devised; the distances are generated in a variety of measuring elements, providing those easily accessible than the distance itself. Indirect distance measurements are the most flexible non-contact distance measurement technique, and many businesses are looking for ways to make challenging measurements more accessible. As a result, image processing and laser procedures are becoming increasingly used for various measurements and inspections.

1.1 Background

Overall, the operation begins with product design and manufacturing specifications and concludes with quality control and inspection. Quality Control (QC) team manually inspects the final product and which is time-consuming task and requires accuracy. The Marut Energy Pvt. Ltd., Nashik is a manufacturing firm producing fixture and jigs for CNC and VMC. Manual pneumatic pressure gauge for quality inspection of product hole diameter and depth was a major task, as it insure proper fitting of a particular component in assembly. This is a tedious and time-consuming task and requires a lot of skilled man power. Manual measurement and inspection of each component is not possible, so random sampling technique is preferred for acceptance of each batch of production.

1.2. Problem Statement

Development of a novel state of the art machine vision algorithm for measuring and inspecting the hole diameter of a mechanical component by using image processing techniques on the images captured by the digital camera. Machine Vision Algorithm should be coded by using Open Source Python Programming Language

1.3. Machine Vision

Initially, once a single frequency light source is incidental, a camera with high resolution and high definition, such as a CCD camera, collects the images. With the use of image

sensors, the illuminated analog captured image is then turned into a digital image. A digital picture is defined as the light intensity of the 2-D variable I (x, y), where x and y are linear combinations and the value of (I) at any place is proportionate to the object's intensity. After that, the digital images are divided into two pixel types. The first is defined as the colour white (or gray level), which is detected as 255, while the second is defined as the colour black, which is detected as 0. A digital picture can be formed by measuring the colour of an image at a large number of places and then assembling the image into the desired format. Pixels are arranged in rows and columns in a specific order, and they store data in an unusual method. Once data has been properly converted into digital images, indirect measurements can be performed using Edge detection. Canny Edge detection, which was based on the first derivative and defined the detection and localization criteria of a class of edges, was the most successful method.
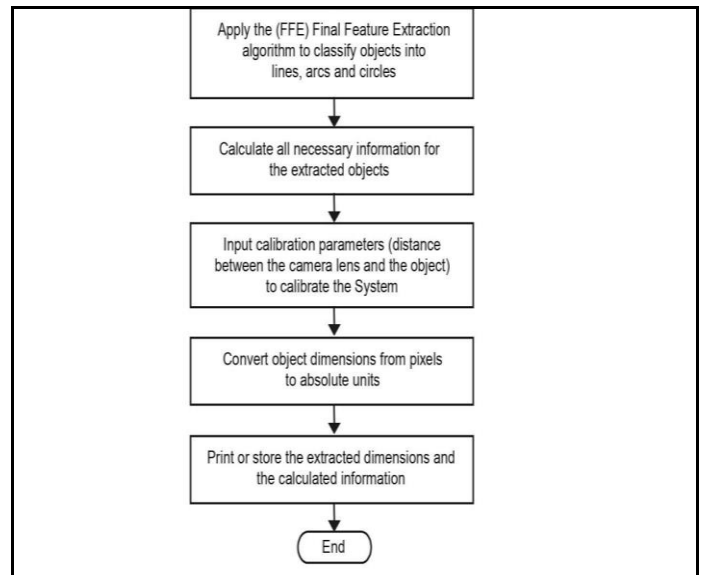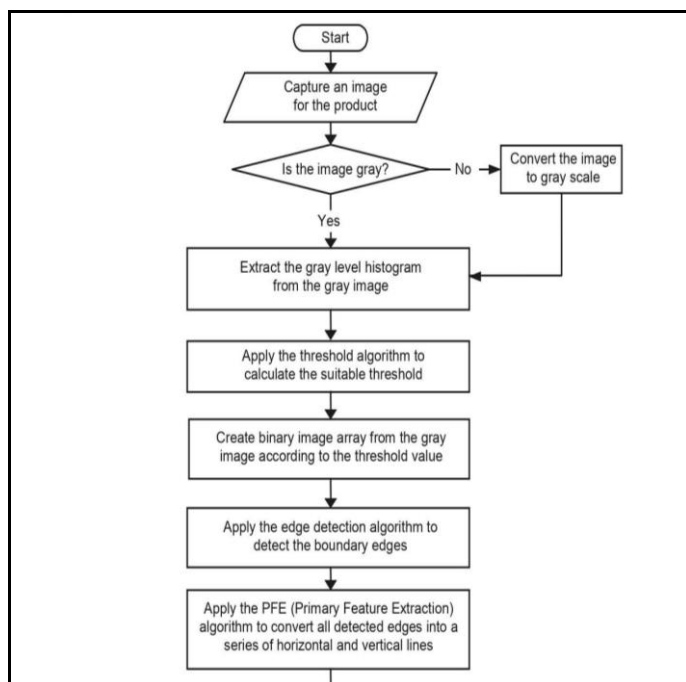

Fig.1.2. Flow Chart Image Processing Algorithm

I. Image Processing Algorithms

Segmentation, edge detection (including Canny edge detection), feature extraction, and calibration techniques are the most important image processing algorithms.

II. Segmentation Algorithms

Segmentation Algorithms are applied when the method has successfully produced the required image. The algorithm then turns the digital image into a binary image, which aids in the non-contact measurement of the needed region of the image.

III. Binary Image algorithms

This algorithm creates a two-dimensional array with the same size of the gray image to store the binary image. There are two important colors which are further converted into two fixed binary values. Black color which represents the background with negligible noise (error) is 0 and white color represented the featured edge of the object which is 255. The binary image array is filled either with 255 or 0 according to the value of the Threshold as follows:

if $g(x, y) \geq T$ then $b(x, y) = 255$ (for Edge);

if $g(x, y) < T$ then $b(x, y) = 0$ (for Background).

Where: T is the Threshold value, $g(x, y)$ is the gray-level value of a pixel at location $a(x, y)$, and $b(x, y)$ is the binary value to be stored in the binary array at the same location (x, y).

IV. Edge Detection Algorithms

On the boundary of objects or image components, edges are frequently noticed. The canny edge detection algorithms detect edges in two phases: horizontal and vertical formats. This approach consecutively actuates for each boundary edge of the item's picture. This algorithm assists in the feature extraction step by providing the coordinates of the edges discovered in the sequence. Each phase includes its

own set of colour codes that may be scanned in horizontal and vertical formats, as well as X and Y coordinates. In the case study, the greatest distance between two edges is to be determined in horizontal format, which provides the exact value via pixel measurement.

1.4. Developed Algorithm Flowchart

1. Start
- Starts the process/system/operation
- Places the products on a conveyor belt

2. Read the image
- A camera is used to scan the required aspects of the product under inquiry, and the image is saved in memory.

3. Conversion of illuminated colour image into gray scale image
- After collecting the image, a threshold scale is being used to convert the colour RGB image to a BGW image.
- For future usage, the image is turned to grayscale.

4. Resizing gray scale image as per our convenience
- After converting the image's colour, scaling is applied to allow a significant parameter to be measured within a range while avoiding undesired noise.
- Resizing is done according to the precision needed for each batch of production.

5. Detect the circular contours using the Canny Edge Detection Algorithm.
- The canny edge command [canny edge ( )] is used to process the scaled image.
- This command is used to detect all of the edges in the image that have a white indicator and a black background.
- As a result, the circular contour is identified with a white contour line and the rest of the surface is dark.

6. Reading Canny Image
- To measure the dimension of given parameter, [getpixel( )] command is used
- The getpixel( ) command scans the image in an X-Y direction from pixel to pixel using the co-ordinates provided. For e.g. Images scanned goes counting black color as (255, 255, 255) and when the white color is detected it indicates as (0, 0, 0).
- Thus the contour coordinates are measured.
- As a result, we obtain all data of pixel points obtained after scanning into required coordinates after scanning according to the required coordinates provided.

7. Assumption of pixel values of 2D image having X max and X min
- Hence, no values are obtained after scanning pixel values.
- Hence, Pixels are defined to achieve max X values and min X values,

8. Getting actual pixels of images as per X- axis and Y-axis parameters
- Logic is made in order to achieve X min and X max value

9. Print or Store X min and X max

10. Taking Difference of X max and X min
- Hence in order to achieve number of pixel points between X min and X max
- D = X max – X min
- Thus, We get the pixels between X max and X min

11. Converting Pixel to millimeters
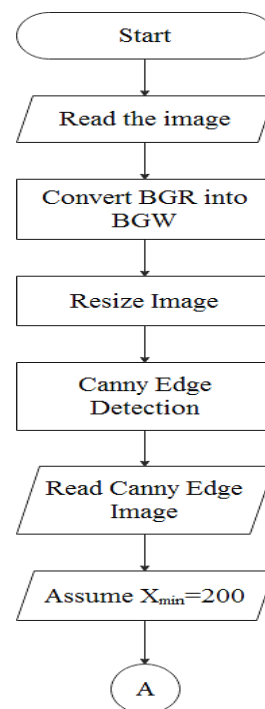- To Convert the pixels to millimeters following formula is used

$$millimeters = \frac{D \times 25.4}{dpi}, \text{ where dpi is dots per square inch}$$

, where dpi is dots per square inch
- Thus, we achieve our final dimensions

12. Print / Store extracted dimensions from images
- Save the photos' attained value, as well as the images' standard value, in a separate folder.

A

For Y Range (0,195)

For X Range (0,200)

Pixel Measurement of Canny Image

If Pixel [0]=255

If $X_{min}>X$

$X_{min}=X$

Print Zero Detected

Print (X,[])

Print one line is over

Print X_{min}

Print Y

Assume $X_{max}=0$

B

B

For Y Range (0,195)

For X Range (0,200)

Pixel Measurement of Canny Image

If Pixel[0] =255

If $X_{max}<X$

$X_{max} = X$

Print Zero Detected

Print (X,[ ])

Print One Line is Over

Print X_{max}

Print Y

C

C

Difference = X_{max} - X_{min}

Print (Difference)

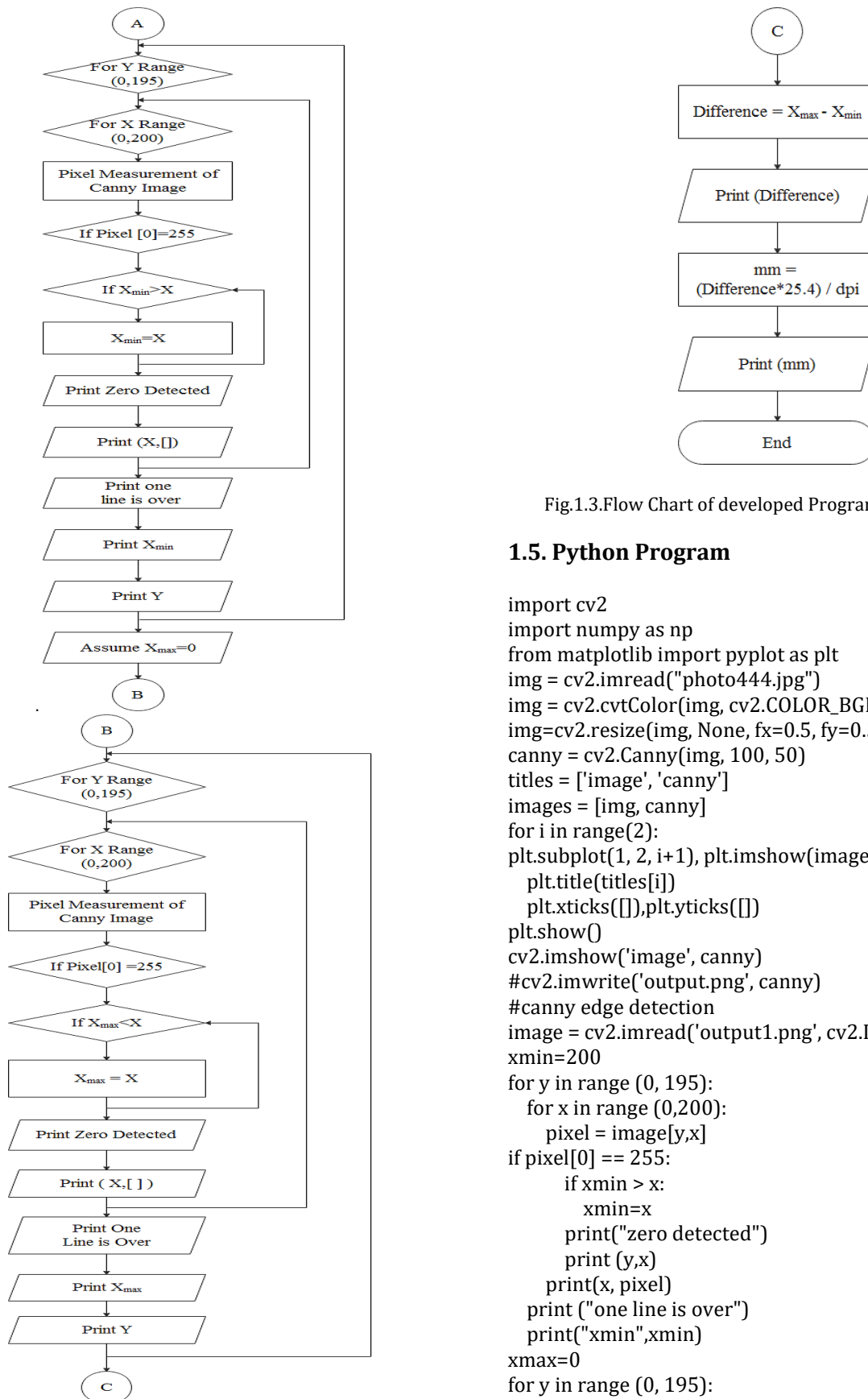mm = (Difference*25.4) / dpi

Print (mm)

End

Fig.1.3.Flow Chart of developed Programming Algorithm

1.5. Python Program

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread("photo444.jpg")
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img=cv2.resize(img, None, fx=0.5, fy=0.5)
canny = cv2.Canny(img, 100, 50)
titles = ['image', 'canny']
images = [img, canny]
for i in range(2):
plt.subplot(1, 2, i+1), plt.imshow(images[i], 'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
plt.show()
cv2.imshow('image', canny)
#cv2.imwrite('output.png', canny)
#canny edge detection
image = cv2.imread('output1.png', cv2.IMREAD_COLOR)
xmin=200
for y in range (0, 195):
   for x in range (0,200):
     pixel = image[y,x]
if pixel[0] == 255:
       if xmin > x:
         xmin=x
       print("zero detected")
       print (y,x)
     print(x, pixel)
   print ("one line is over")
   print("xmin",xmin)
xmax=0
for y in range (0, 195):
   for x in range (0,200):
     pixel = image[y,x]
```

```
if pixel[0] == 255:
        if xmax < x:
          xmax=x
      print("zero detected")
      print (y,x)
    print(x, pixel)
  print ("one line is over")
  print("xmax",xmax)
difference= xmax - xmin
print(difference)
mm = (difference * 25.4) / 300
print(mm)
Output :-
Xmax = 370
Xmax = 199
Difference = 171
Diameter = 12.477999999999998
```

2. Experiment

1. To ensure that the produced algorithm met the necessary objectives, an experimental layout was created and trials were done. The diagram depicts the experimental setup.
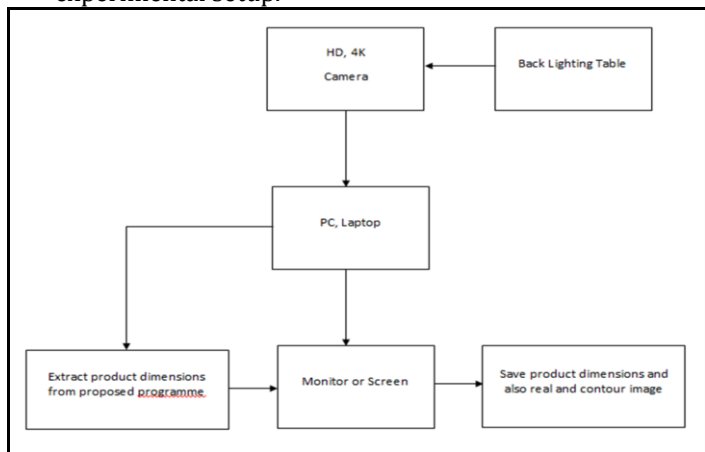

Fig.2. Experimental layout

2. For diameter measurement, the following mechanical component was considered, as shown in the diagram. The diameter of the hole is determined by the hole on the right side of the component, as shown in the diagram.

Fig.2.1. Experimentation of mechanical product

3. The mechanical component is positioned on top of the lighted light bulb, which is positioned beneath the milky screen. A photo of the relevant part of the area to be measured is taken with appropriate image capturing equipment. The image of the hole that was caught is shown in Figure. The collected image is cropped to the proper proportions to remove noise.

Fig.2.2. Captured image

4. By applying a sufficient threshold value, a colour BGR (Blue Green Red) image is converted to a BGW (Black Gray White) image. This image, as seen in Figure, can now be used to detect edges.
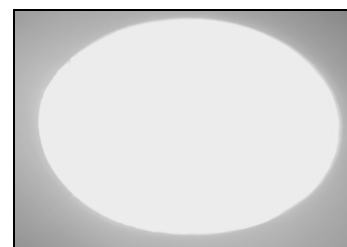

Fig.2.3. BGW version of image

5. The Canny Edge Detection and Image Processing Algorithm is used to detect the edge of the aspect to be measured in the method shown in Figure. The photos are processed using the newly developed algorithm.
    Programmed Algorithm Output:-
    X max = 370
    X max = 199
    Difference = 171 pixels
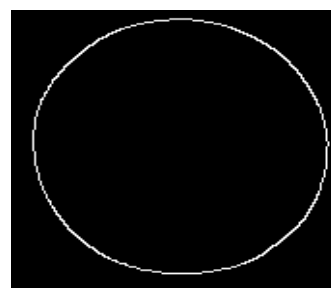   **Diameter = 12.477999999999998**

Fig.2.4. Canny Edge Detection output

2.1. Result

Object Number	True Dia (mm)	Tolerance (mm)	Measured Dia using machine vision (mm)	Measured Dia using vernier Caliper (mm)	Percent Error (%)	GO NOGO Gauge Check	Part Accepted/ Rejected
1	12	± 0.5	12.479	12.2	2.28 %	Correct	Accepted

Table.1.Result table

3. CONCLUSIONS

The existing experiment and its analysis reveal the following noteworthy points: I. This project produced a state-of-the-art method to assist contactless metrology and inspection of mechanical components.

II. The hole diameter of a mechanical component was measured using the Machine Vision Algorithm, which was programmed in Python.

III. A newly built machine vision system and traditional mechanical vernier calipers were used to gather measurements and readings. The results are in agreement, indicating that the newly created system has been validated.

IV. A machine has been presented as a means of putting the created machine vision system into practice in the industry.

V. The Machine Vision System is used to obtain precise and accurate dimensions.

VI. Component inspection is facilitated more quickly as the measurement and inspection lead times are reduced.

VII. Any dimension of a product can be calibrated with the help of pixel measurement.

REFERENCES

1. Kim J-H, Moon DK, Lee DW, Kim JS, Kang MC, Kim KH (2002) Tool wear measuring technique on the machine using CCD and exclusive jig. J Mater Process Technol 130–131:668–674

2. Ayub MA, Mohamed AB, Esa AH (2014) In-line inspection of roundness using machine vision. Procedia Technol 15:808–817

3. Bradley C, Wong YS (2001) Surface texture indicators of tool wear—a machine vision approach. Int J Adv Manuf Technol 17:435–443

4. Kumar, B. S., Vijayan, V., & Davim, J. P. (2019). Machine Vision in Measurement. In Measurement in Machining and Tribology (pp. 113-123). Springer, Cham.

5. Jawad, Huda M. & Husain, Tahseen. (2017). Measuring Object Dimensions and its Distances Based on Image Processing Technique by Analysis the Image Using Sony Camera.