Strain Measurement Using Image Processing

Abhijit Verma*, Sachin Aggarwal**, Siddhartha Srivastava*
*Under Graduate student, Department of Aerospace Engineering, Indian Institute of Technology, Kanpur
**Under Graduate Student, Department of Material Science Engineering, Indian Institute of Technology, Kanpur

Abstract

A non conventional method to measure displacements and strains in mechanical test specimens is described in this paper. A contact free strain measurement technique is discussed. This is an optical technique which uses a camera to capture the image and a computer to analyze the image to fetch the useful information out of the different images. For image processing we used OpenCV. OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision. Dev C++ was used as development environment. Dev-C++ is a full-featured Integrated Development Environment (IDE) for the C/C++ programming language. The displacements of the beam are obtained by analysis of the movements of patterns made on the specimen.

1. Introduction

Conventional strain measurement methods like one based on the "Dial Gauges" can only measure deformations at very few points. Also they are very difficult to use to find strains in small specimens. The conventional methods are also dependents on the external environmental factors like temperature.

So a new method to measure strains in mechanical test pieces is described here in which no sensors are applied to the specimens. A DSLR camera and a laptop for programming purpose is used. Image analysis techniques based on OpenCV are used to investigate relative displacements of the markers on the test specimen[1]. We used Dev C++ as our IDE. This optical technique is both full field measuring and non-contacting.

The human brain divides the vision signal into many channels that stream different kinds of information to the brain. Our brain has an attention system that identifies, in a task-dependent way, important parts of an image to examine while suppressing examination of other parts[2]. What a computer 'sees' is just a grid of numbers which can be represented in n x m matrix. Each element of the matrix contains some specific information of the image. Image processing in simple words can be explained as just some computational processing of images. Using Image Processing we help computer to find the content of the image[2]. This is achieved by developing algorithms for image processing.

This paper is organised as follows: Section 2 describes the experimental setup used. Section 3 discusses the software architecture followed by results in Section 4. Section 5 describes final conclusion.

2. Experimental Setup

Figure 1 shows the entire experimental setup. A DSLR camera was mounted on a standard tripod stand for the camera. Two light sources were used to uniformly light the specimen. This was necessary as uniform light source is required for proper analysis of the image. An aluminium beam was used as specimen. The beam was cantilevered using iron C clamps on a rigid iron base. Additionally makers were used for image analysis. Figure 2 shows the marked pattern used in the experiments. Weights were put in a pan for loading the beam from one end. The actual experimental setup is shown in figure 3.

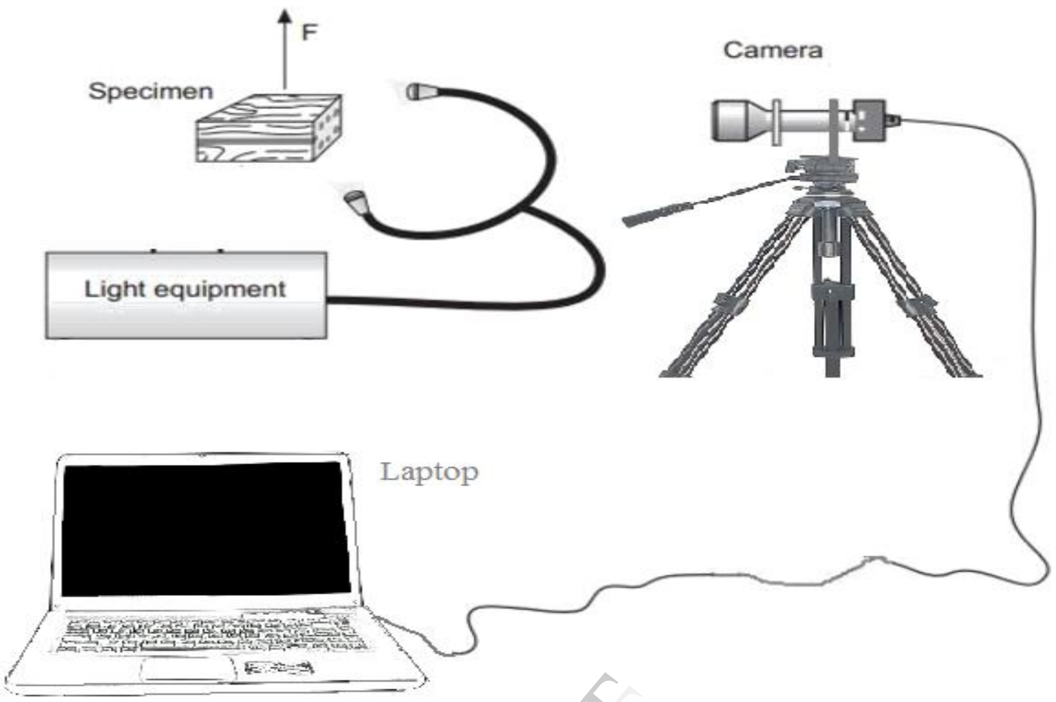

Figure: 1 Schematic of experimental Setup

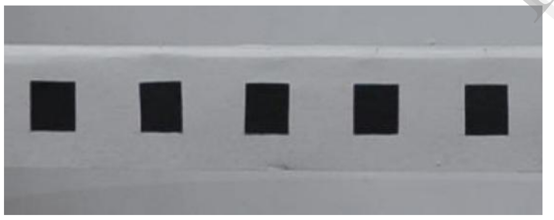

Figure: 2 Marked Pattern

We used the above pattern as markers for analysis of the image. The black squares acted as a closed area, whose centroid can be found. These marks were pated on the beam.

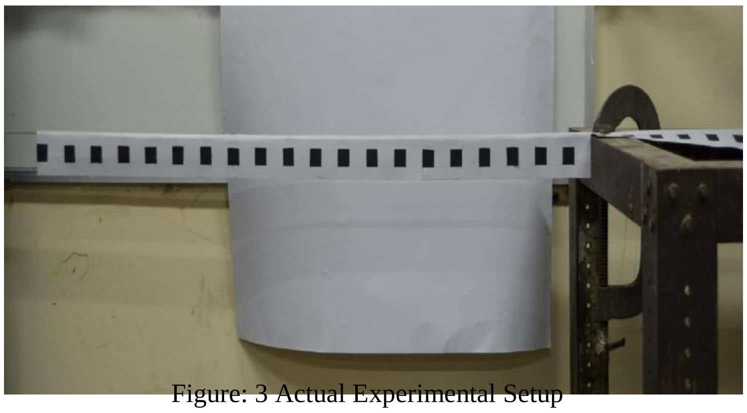

Figure: 3 Actual Experimental Setup

3. Software Architecture

Dev C++ was used to develop the codes for the processing of the image. The main algorithm used was broadly described in the flow chart shown in figure 4. The flow chart clearly explains the entire procedure of the strain measurement. After clicking the image, we convert it into the binary form. The binary image was then filtered from the background noise. After filtration we detect the centroids of each closed area. Using the relative displacement of the centroids strain was calculated.

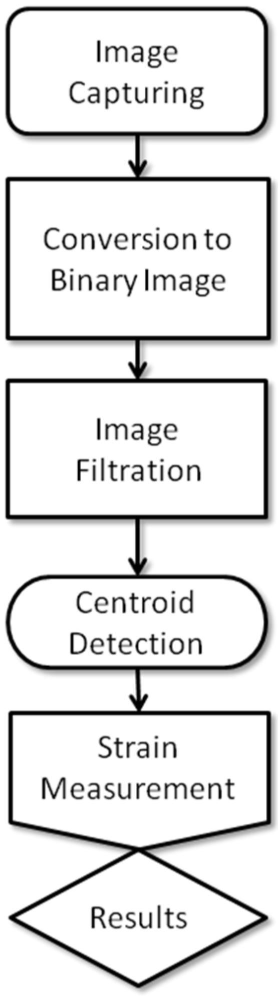

Figure: 4 Software Architecture

3.1. Binary Conversion of image

Flow chart in figure 5 shows the step wise process of conversion of colour image into a binary image. A 3-Channel RGB image of the specimen was captured from camera. Then it was converted to the single channel greyscale image. In the colored image each pixel stores three values between 0(darkest, considered as black) to 255(white) corresponding to primary colors red, blue and green. All the three colors together produce color of that pixel(so it can't be directly converted to binary). So first we converted the image into grayscale image as in grayscale image each pixel stores only one value which carries intensity information between 0 to 255[3]. Grayscale image is converted into the binary image by setting a proper threshold value depending on light intensity of the marks. Binary image can be represented as a matrix where each pixel can have value either value 0(black) or 1(white).This binary image is our Output-I as shown in figure 6.

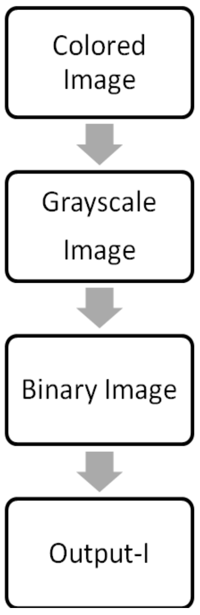

Figure: 5 Binary Conversion Of Image

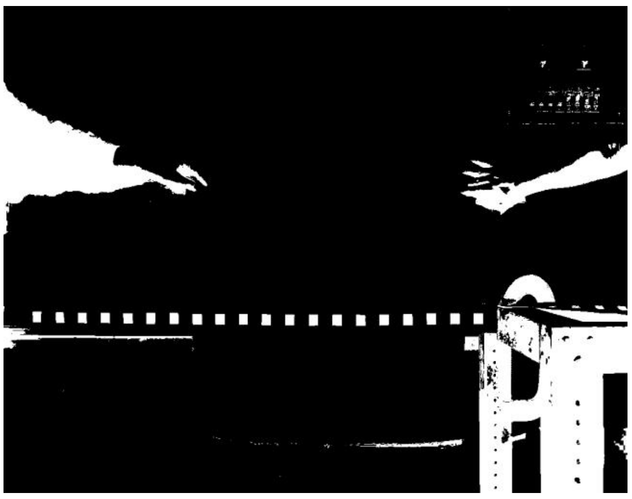

Figure: 6 Binary image

3.2 Image Filtration & Centroid Detection

Output-I is a binary image. On the binary image blob detection method was applied to find blobs. For blob detection cvBlob library in OpenCV was used. cvBlob is a library for image processing which is aimed to detect the connected regions in the binary image and those connected regions are termed as Blobs[]. Further, from this library we have access to the different properties of the connected regions like area, orientation, centroid etc[].

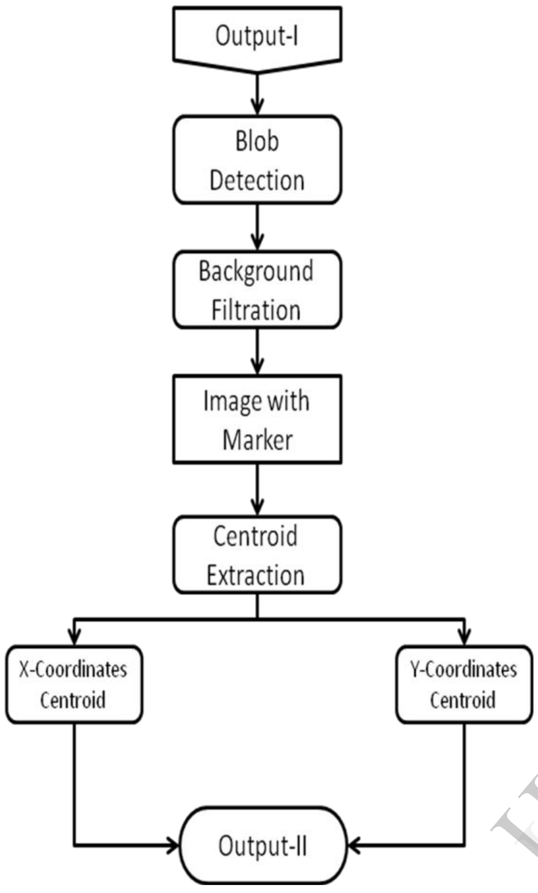

Figure: 7 Image Filtration & Centroid Detection

After detecting the connected region in binary image (output-I) area of the each blob in pixels square were extracted. Background of the image is then filtered on the basis of the extracted area (Blobs having very larger or smaller area in comparison to marker was subtracted from the background).An image (figure 8) obtained having blobs size comparable to the markers. Remaining blobs (apart from markers) are then filtered on the basis of their centroid as shown in figure. From this figure we have obtained the x-coordinates and y-coordinates of the centroid of each marker. Then we have sorted these coordinates according to x-coordinate using bubble sort. These sorted coordinates are our Output-II.

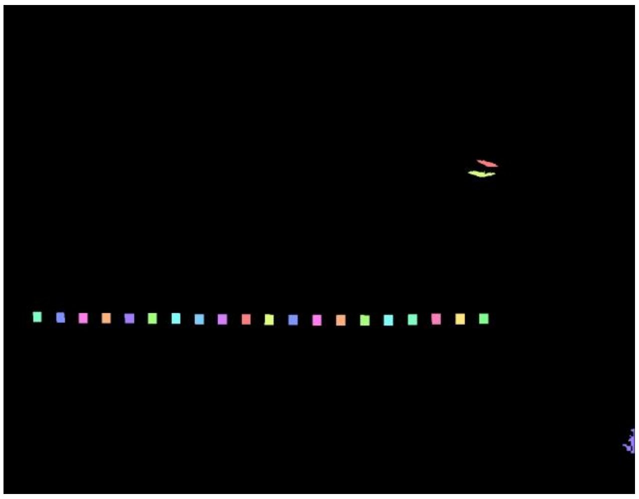

Figure: 8 Filtered Background

3.3 Strain Measurement

Two images one before deflection and one after deflection were captured to determine the strain.. As shown in figure 9 processes described in 3.1(conversion to binary) and 3.2 (image filtration and Centroid Extraction) were repeated for both the Images. From these processes we have extracted the Output-II (i.e. sorted coordinates) for both images. Now the relative displacement between the corresponding marks was obtained by comparing the output-II of both images in pixels. The pattern that was pasted on the beam had the black squares having fixed center to center distance. The fixed distance was mapped with the difference in the pixel value of two consecutive x-coordinate values in output-II. Using this mapping we found the deflection in y direction that is strain in the beam.

4. Results

Figure shows the comparison between theoretical deflection and experimental result. Experimental results were very close to the theoretical results under the consideration of error in the experiments. Figure also shows that the percentage error in the deflection at most of the points is within the of ±10%. The conventional strain measurement technique based on the Strain Gauges is more prone to errors due to physical interaction with the specimen.

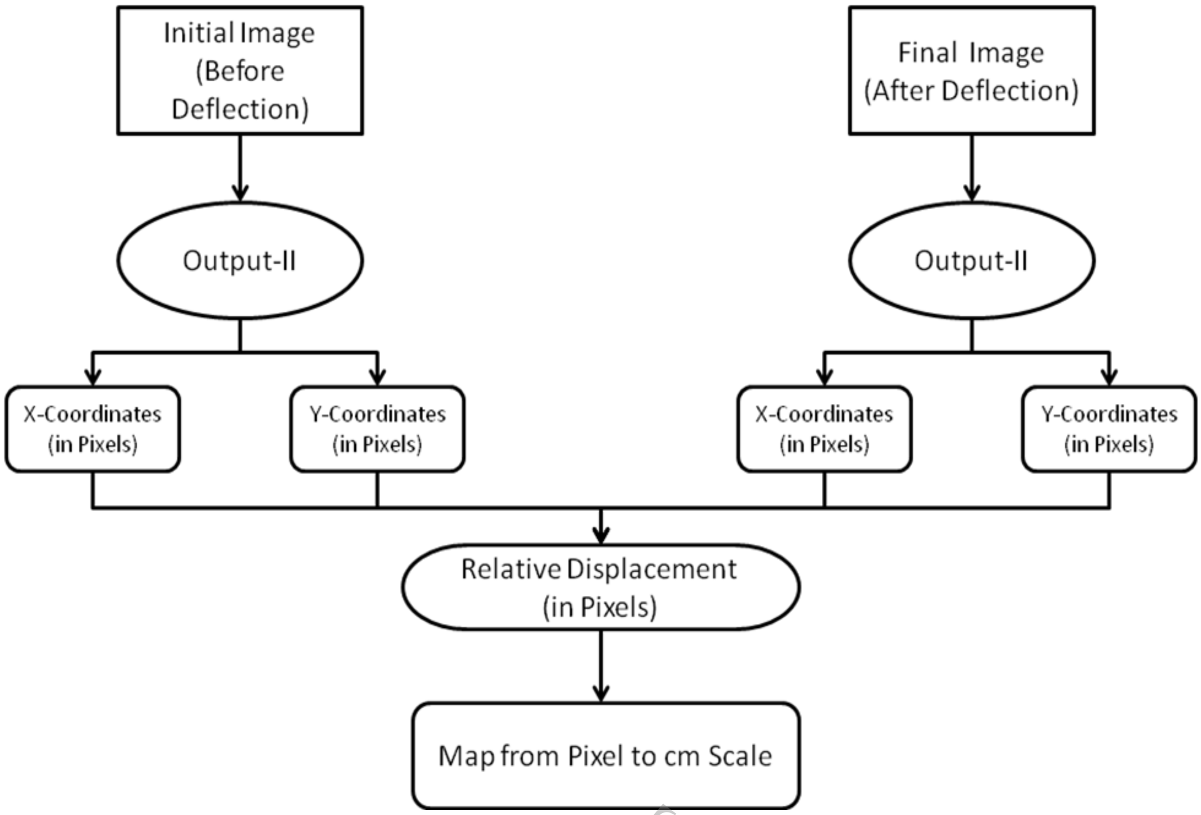
Initial Image (Before Deflection)
Final Image (After Deflection)
Output-II
Output-II
X-Coordinates (in Pixels)
Y-Coordinates (in Pixels)
X-Coordinates (in Pixels)
Y-Coordinates (in Pixels)
Relative Displacement (in Pixels)
Map from Pixel to cm Scale

Figure: 9 Strain Measurements

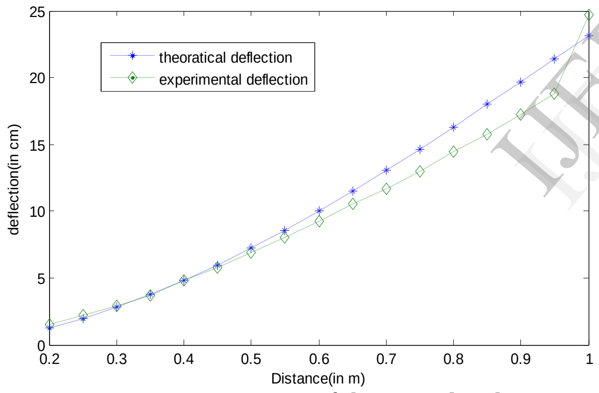

Figure: 11 Comparison of theoretical and experimental results

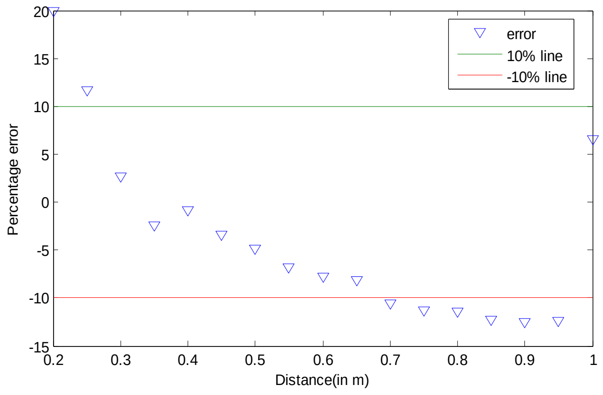

Figure: 12 Percentage Errors

5. Conclusion

A non-contacting reliable method for strain measurement has been described. The method can be used to find the strains in very small specimens. Using smaller but darker marks increases the efficiency of the algorithm. Strain rate can be calculated using a continuous sequence of images.

6. Acknowledgement

We would like to acknowledge the efforts of all the members of Structures Lab of Indian Institute of Technology. We would also like to acknowledge Robotics Club of IIT Kanpur for using their image processing facilities. Finally we would like to extend our heartily gratitude to Dr. Rajesh Kitey for his continuous inputs during the entire course of the research.

7. References

[1] “Contact Free Strain Measurement using MATLAB Image Processing Toolbox” by A.Per-Erk, E. Bertil et. Al.
[2] http://www.micc.unifi.it/lisanti/downloads/OpenCV-PPM-Histogram.pdf as on date 17th Jan,’ 2013

[3] http://en.wikipedia.org/wiki/Greyscale as on date 17th Jan,’ 2013.

Appendix A

Step wise images during binary conversion

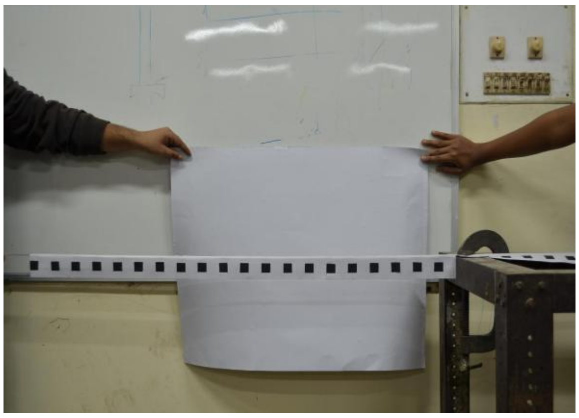
Figure13: Coloured Image

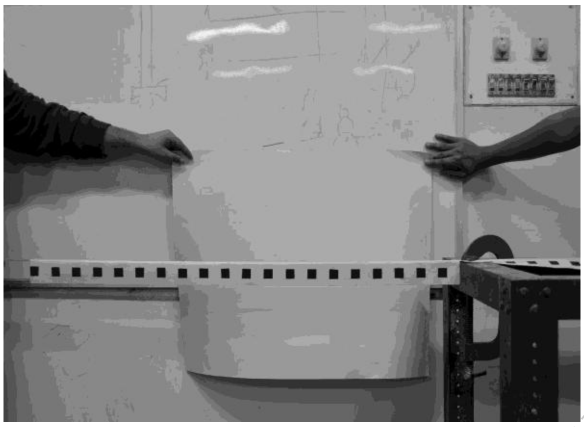
Figure14: Grey Image

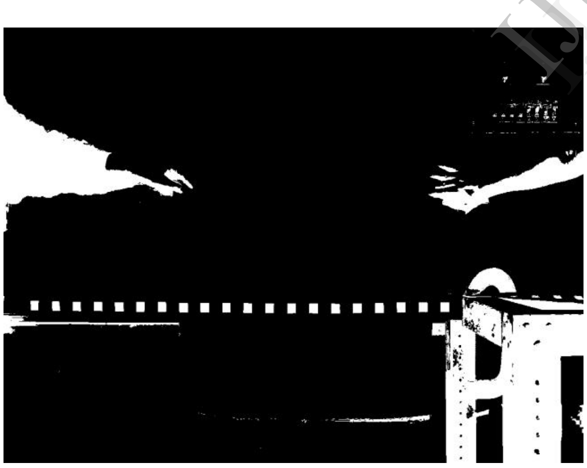
Figure15: Binary Image

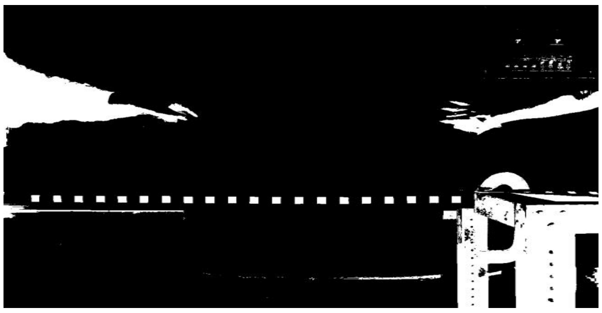
Figure16: Binary Image

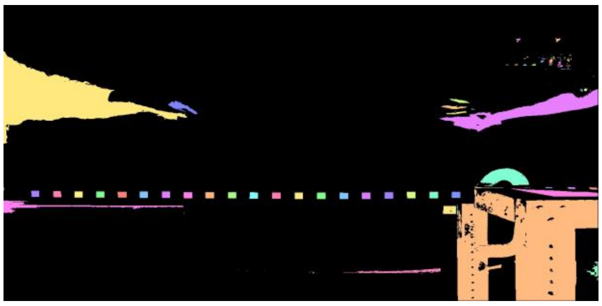
Figure16: Blobs

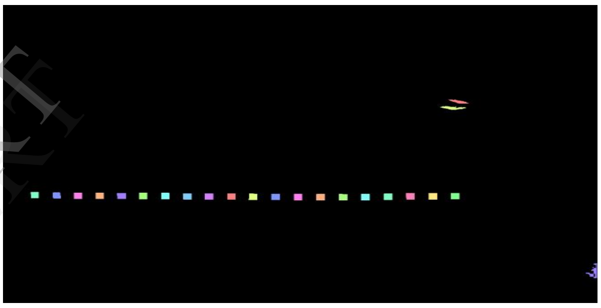
Figure16: Background Filtered

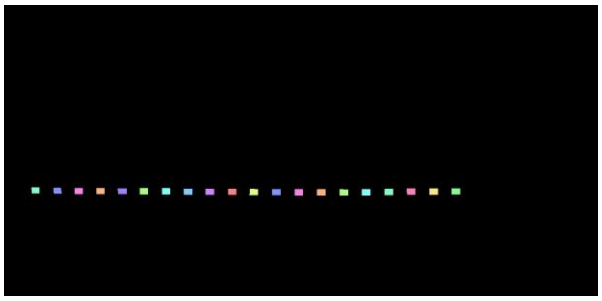
Figure17: Final Filtered Image

Step wise Images during Filtration and centroid detection