OBJECT COUNTING AND DENSITY CALCULATION USING MATLAB

Submitted by

PREM KUMAR. V, BARATH. V, PRASHANTH. K

For more information contact: **premkumarbullets@gmail.com,**
prem91914@gmail.com

ABSTRACT

Counting the number of objects is an integral part of image processing. Knowing the number of objects present in the image can be useful for further analysis in a wide range of applications. In this project we propose a simple method for automatically determining the number of objects in an image . Once the number of objects are determined the objects per unit area or the density can also be estimated. Existing methods involve counting based on area of objects, color of objects, applying edge detection techniques etc. There are also applications that involve a large amount of hardware components for counting that further adds to the cost and maintenance which is a tedious work. The proposed system involves converting the input image into a format such that the number of objects can be calculated based on the connected components present in the enhanced image. This project work also aims at determining the correct value of density by clearing the objects touching the borders of the image. In this project three applications are taken into account and using Matlab with image processing toolbox the count and density values are calculated for each.

CHAPTER 1

1.INTRODUCTION

Images contain a wealth of objects and patterns which may convey information about underlying mechanism and methods. Counting involves estimating the number of objects in an image or a still video frame. Counting arises in many real-time applications such as counting cells in microscopic images, number of people, number of vehicles on road etc.

This goal is to accurately estimate the count, but however we evade the hard task of localizing or drawing boundary around the objects. Object detection is a very hard problem to be solved, especially in case of overlapping objects.

Once the count is known , the density of objects or the number of objects per unit area can be estimated.

As mentioned earlier, the existing works under counting involve a large amount of hardware which also adds to the cost. Now with a simple image processing software such as matlab many parameters can be estimated from images. In this paper we have dealt with 3 such applications namely-

- Cell counting
- People counting based on faces
- Vehicle counting

1.1 Cell counting:

COUNTING cells is often a necessary but tedious step for *in vitro* cell culture. Consistent cell concentrations ensure experimental reproducibility and accuracy. Cell counts are important for monitoring cell health and proliferation rate, assessing immortalization or transformation, seeding cells for subsequent experiments, transfect ion or infection, and
Preparing for cell-based assays.

We propose a system for counting cells in image by using matlab and image processing toolbox. It can be used for accurate calculation of object and density of image

1.2 Counting people based on the faces detected:

One of the important applications of counting includes counting people say in a hall or in a shopping mall etc,. People can be counted based on the number of faces detected. Face detection is the process in which the human faces are detected from a color image. Once the number of faces are detected the density can be calculated. There are many ways by which faces can be detected one such method is skin based detection. Rectangles can be used to map the faces in the image. Nowadays Face detection is used in many day to day application.

1.3 Vehicle counting:

Transportation research involves counting number of vehicles on road as well as finding the density of traffic in a particular area. There are many methods of detecting vehicles on road such as motion detection, installing lasers on both sides of the road, etc., which is tedious and involves large number of hardware. This method uses image processing techniques to count the number of vehicles on road and estimate the density. The number of vehicles found can be used for surveying or controlling the traffic signal.

1.4 EXISTING SYSTEMS:

1.4.1 CELL COUNT:

- Methods include counting based on the area of objects, color of objects, applying edge detection techniques etc.

1.4.2 Counting people based on the faces:

- Existing methods based on skin detection provide inaccurate count.
- Existing method involves complex algorithms and difficult to compute the faces.
- Existing method of people counting involves a lot of hardware components.

1.4.3 VEHICLE COUNT:

- In existing system a timer based traffic operation is used which does not estimate traffic parameters. This results in green light being shown to an empty road.
- Various sensors have been employed to estimate traffic parameters for updating traffic information, magnetic loop detectors have been the most used technologies in the existing system.

1.4.4 DEMERITS OF THE EXISTING SYSTEM:

- Existing methods involve a large amount of hardware components.
- Maintenance and cost are inconvenient.
- Calculating the number of objects per unit area are not accurate.
- The installation and maintenance of vehicle counting systems are very high.

1.5 PROPOSED SYSTEM:

1.5.1 CELL COUNTING:

Counting cells in Drosophila is a complex task, due to variability in image quality resulting from different cell markers. Cells are segmented according to their characteristics. But cell shape changes with cell state (i.e. arrest, mitosis, or apoptosis). For instance, during mitosis the shape is irregular and it can be difficult to determine when a dividing cell can be considered as two daughter cells. Nuclei and glia cells have a more regular shape, between elliptical and circular. Apoptotic cells have initially a very irregular shape, later on very round, and can appear subdivided into different parts depending on the timing within apoptosis. Depending on the kind of cells or cell state to be visualised, a different cell marker (i.e. antibody) is employed. As a result, different image-processing methods must be developed to quantify cells of different qualities.

Formula used:

$$\text{OBJECTS PER UNIT AREA} = \frac{\text{NO OF OBJECTS}}{\text{TOTAL SIZE OF IMAGE}}$$

1.5.2 PEOPLE COUNTING:

- In this proposed method we use skin based face detection for counting people.
- In this method multiple faces can be detected at a time or even a group of people can be detected.
- This method not only detects the faces but also count them.
- From count the density can also be derived.
- Both image and video can be given as input to the compiler.

1.5.3 VEHICLE COUNTING:

- This proposed system of counting vehicles involves comparing the images of road with and without vehicles.
- On comparing two images we can find the difference between images.
- Based on the difference obtained, the number of vehicles as well as count can be estimated.

1.6 SOFTWARE REQUIREMENTS:

- MATLAB R2010a with image processing toolbox .

CHAPTER 2

2. LITERATURE SURVEY:

2.1 CELL COUNT:

Victor Lempitsky and Andrew Zisserman-“Learning To Count Objects in Images” deals with the counting problem is the estimation of the number of objects in a still image or video frame. It arises in many real-world applications including cell counting in microscopic images, monitoring crowds in surveillance systems, and performing wildlife census or counting the number of trees in an aerial image of a forest. We take a supervised learning approach to this problem, and so require a set of training images with annotation.

Christian Wolf and Jean-Michel Jolion of Technical Report LIRIS discussed about Object count/Area Graphs for the Evaluation of Object Detection and Segmentation Algorithms Evaluation of object detection algorithms is a non-trivial task: a detection result is usually evaluated by comparing the bounding box of the detected object with the bounding box of the ground truth object.

Andr´e R.S. Mar¸cal Faculdade de Ciˆencias, Universidade do Porto DMA-‘Alternative Methods for Counting Overlapping Grains in Digital Images’ deals with Standard granulometry methods are used to count the number of disjoint grains in digital images

2.2 PEOPLE COUNT:

Bhaskar Gupta , Sushant Gupta , Arun Kumar Tiwari of *ABES Engineering College, Ghaziabad* discussed about Face Detection using Gabor Feature and Artificial Neural Network which uses a Gabor filter feature to extract facial features from the local image.

Sanjay Kr. Singh1, D. S. Chauhan2, Mayank Vatsa3, Richa Singh3 of Engineering Institute of Engineering and Technology Jaunpur, India discussed about a Robust Skin Color Based Face Detection Algorithm. They have compared the algorithms based on the color spaces RGB,YCbCr and HSI and have combined them to get a new skin color based face detection algorithm which gives higher accuracy.

Gary Chern, Paul Gurney, and Jared Starman discussed about Face detection using Color based masking. They developed an algorithm capable of locating each face in a color image of the class. They were given seven training images along with the corresponding ground truth data to develop and train algorithms on .

In 2003 Waqar Mohsin, Noman Ahmed, Chung-Tse Mar of Stanford university discussed about an method to detect faces in an color image using rejection based classification.

2.3 VEHICLE COUNT:

From the proceedings of the 26th National IT conference ,Colombo,Sri Lanka M. Pancharatnam, D.U.J. Sonnadara of Department of physics, University of Colombo, Sri Lanka discussed about **Vehicle Counting and Classification from a Traffic Scene** using adoptive background subtraction technique. After the background subtraction, using threshold and median filters, isolated image blobs are identified as individual vehicles.

Erhan ˙INCE from Department of Electrical and Electronic Engineering, Eastern Mediterranean University, Famagusta, North Cyprus, via Mersin 10 TURKEY discussed about **Measuring traffic flow and classifying vehicle types: A surveillance video based approach** based on invariant moments and shadow aware foreground masks.

Pejman Niksaz Science &Research Branch, Azad University of Yazd, Iran discussed about **Automatic Traffic Estimation Using Image Processing** based on background elimination and lane masking .

From the paper of L.G.C Wimalaratna[1] and D.U.J. Sonnadara[2], [1]Virtusa (Pvt.) Ltd., Trans Asia Commercial Complex, Colombo 2, [2]Department of Physics, University of Colombo, Colombo 3 which deals with the **Estimation of the Speeds of Moving Vehicles from Video Sequences** the concept of pre-processing video images using gray scale and median filter to extract moving vehicles from a traffic scene is surveyed. This method finds the difference between three consecutive video frame and filters through edge detection filter Blob counting.

CHAPTER 3

3. BLOCK DIAGRAM:

3.1 CELL COUNT:

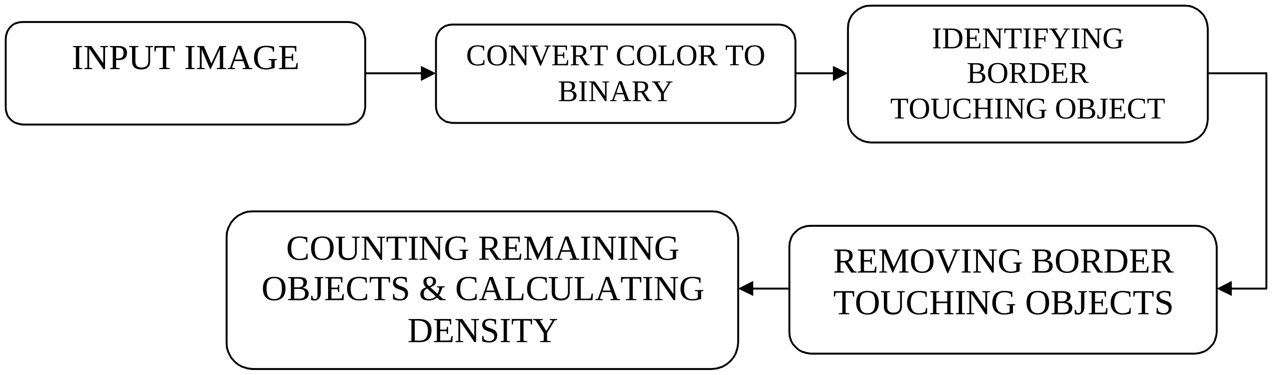

INPUT IMAGE
CONVERT COLOR TO BINARY
IDENTIFYING BORDER TOUCHING OBJECT
REMOVING BORDER TOUCHING OBJECTS
COUNTING REMAINING OBJECTS & CALCULATING DENSITY

Fig 3.1 Block diagram of the cell count

EXPLANATION:

First the input image is given to matlab, using imread() function the image is converted from color to binary. Then the objects touching the borders are identified and removed. The remaining objects are counted and the density is carried out.

3.2 PEOPLE COUNT:

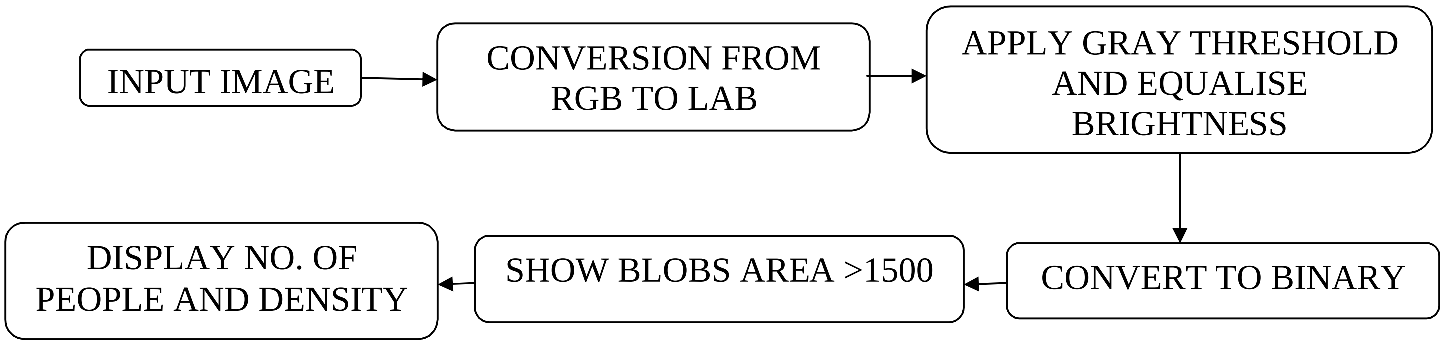

INPUT IMAGE
CONVERSION FROM RGB TO LAB
APPLY GRAY THRESHOLD AND EQUALISE BRIGHTNESS
CONVERT TO BINARY
SHOW BLOBS AREA >1500
DISPLAY NO. OF PEOPLE AND DENSITY

Fig 3.2 Block diagram of the people count

EXPLANATION:

First the input image is fed into matlab. The image is converted from rgb to lab. Then the gray threshold is applied to the converted image and brightness equalization is done. Then the equalized image is converted to binary. The blobs in the image are opened when the blobs area is greater than 1500. The number of people and density are displayed.

3.3 VEHICLE COUNTING:

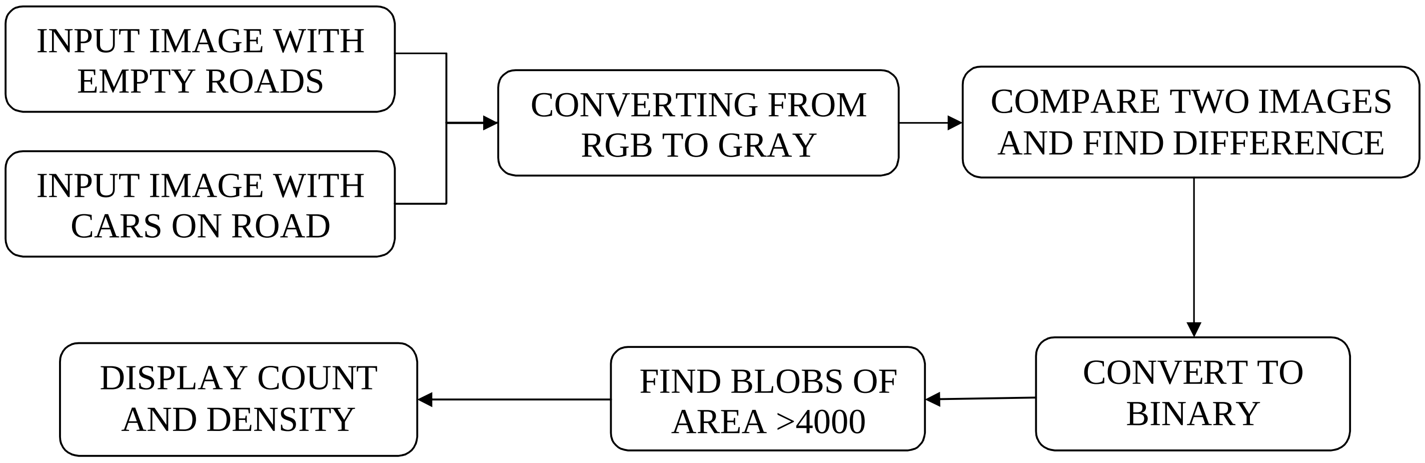

INPUT IMAGE WITH EMPTY ROADS
INPUT IMAGE WITH CARS ON ROAD
CONVERTING FROM RGB TO GRAY
COMPARE TWO IMAGES AND FIND DIFFERENCE
CONVERT TO BINARY
FIND BLOBS OF AREA >4000
DISPLAY COUNT AND DENSITY

Fig 3.3 Block diagram of the vehicle count

EXPLANATION:

In vehicle counting two input images are given, one without cars and the other with cars in it. The input image is then converted from rgb to gray. Now compare the two images and find the difference. Then the image is converted to binary. The blobs in the image are opened when the blob area is greater than 4000. Finally the count and the density are displayed.

CHAPTER 4

4. ALGORITHM:

4.1 CELL COUNT:

Step 1) Start the program

Step 2) Get the input image using imread ()

Step 3) Convert color to gray if it is a color image.

Step 4) Use clear borders command if any object is touching the border.

Step 5) Choose an appropriate threshold value using graythresh () command

Step 6) Fill blobs that occur with holes

Step 7) Apply morphological operation by using a filter of ones

Step 8) Display all blobs that have an area>100

Step 9) Count them using conncomp () command

Step 10) Calculate the density

Step 11) Stop the program

4.2 PEOPLE COUNT :

Step 1: Start the program

Step 2: Get the input image

Step 3: Change the color space from rgb to lab

Step 4: Convert to binary based on gray level value

Step 5: If patches are present fill with holes

Step 6: Set a threshold value for opening area above some value.

Step 7: Use connected components method to determine the number of objects present

Step 8: Draw bounding box

Step 9: Display output count and density

Step 10: End the program

4.3 VEHICLE COUNT :

Step1:Start the program.

Step2:Read the input background image of the empty road.

Step3:Read the new image with vehicles on road.

Step4:Convert the images to grayscale format using double precision.

Step5:Find the width and height of the image.

Step6:Set threshold value =11

Step7:Find the difference between frames based on the threshold.

Step8:If frame diff > than 11 then assign that image to a variable else "0" if no difference is found.

Step9: Increase the contrast of the output image using imadjust()

Step10:Find a graythreshhold value using graythresh() command.

Step11:Add Gaussian noise to the output difference.

Step12:Apply Weiner filter to filter the blobs

Step13:Convert to binary image

Step14:Fill holes to the blobs

Step15:Open all blobs having area greater than 5000

Step16:Count the number of cars using bwconncomp or bwlabel

Step17:Display the output image

Step18:Stop the program

CHAPTER 5

5. FLOWCHART:

5.1CELL COUNT:

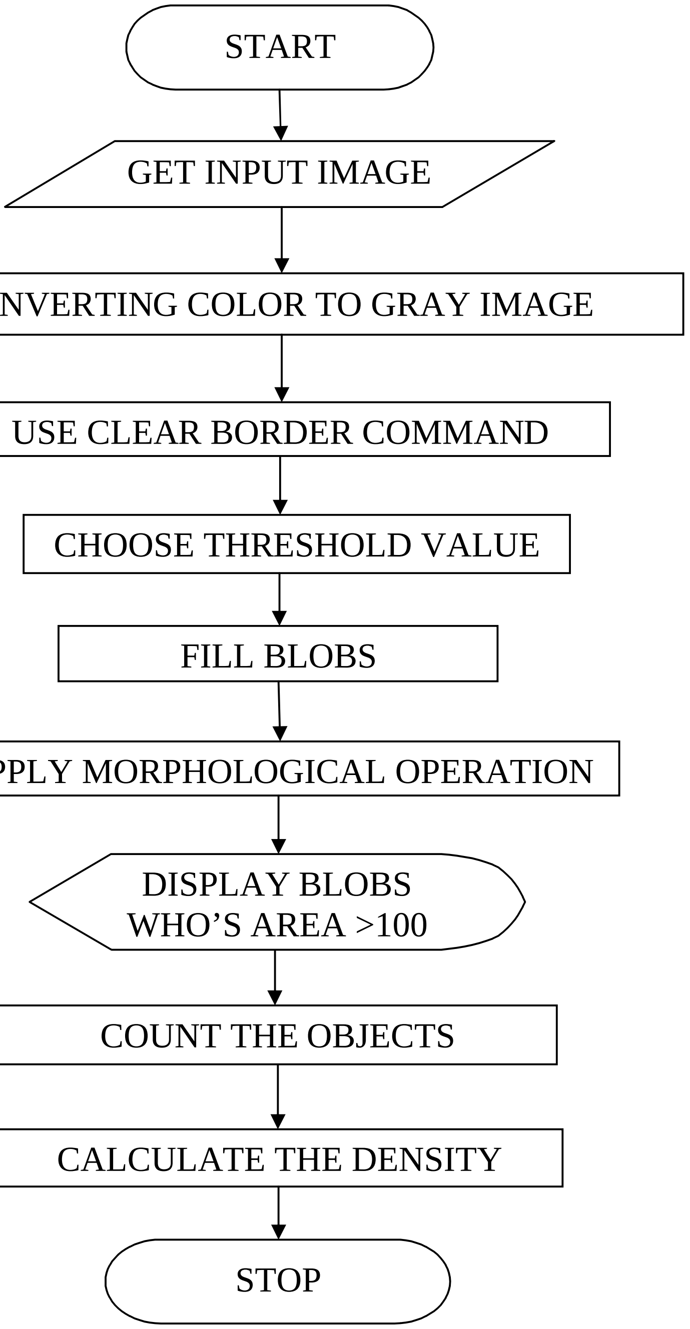


Fig 5.1 Flow chart of the cell count

5.2 PEOPLE COUNT :

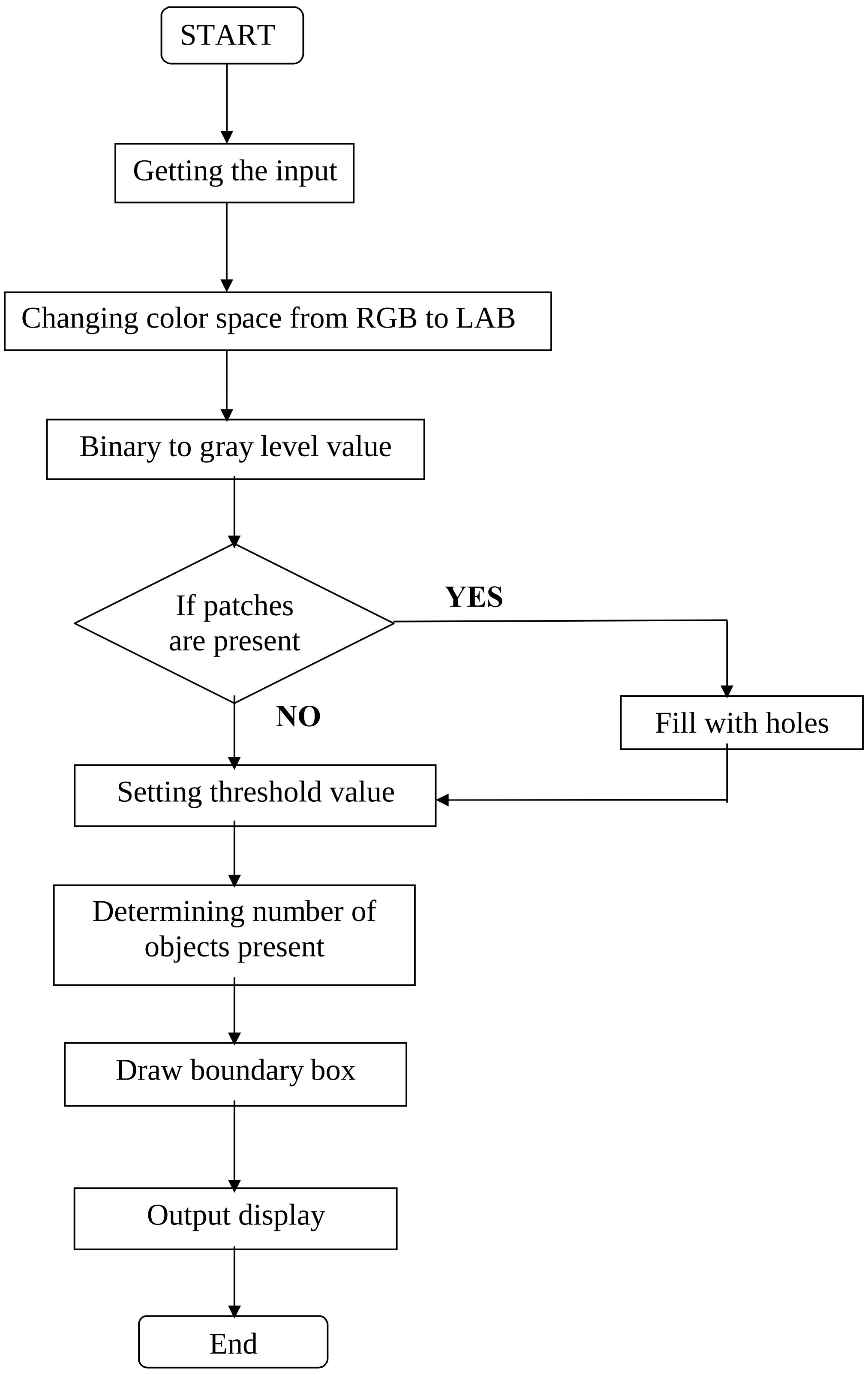
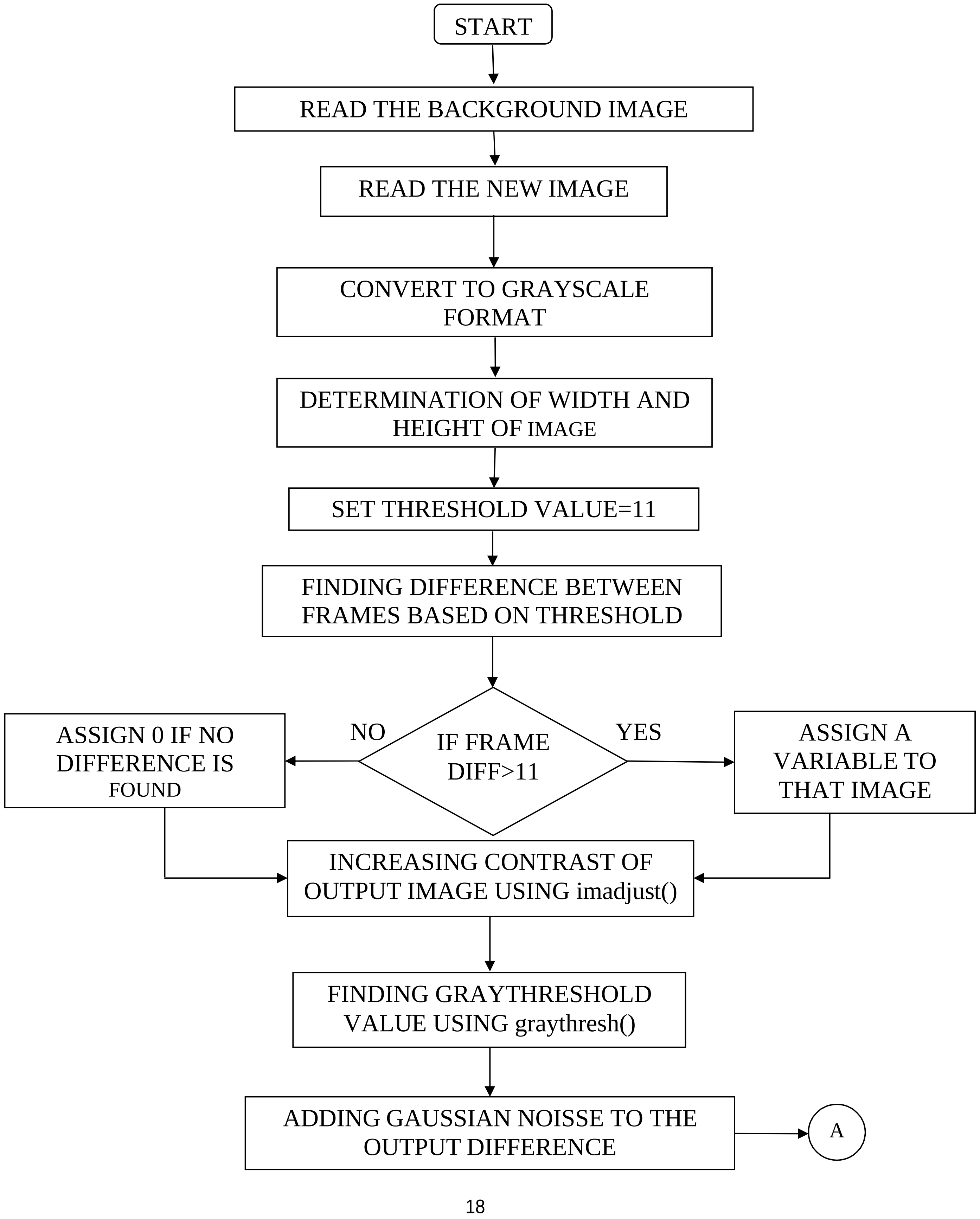


Fig 5.2 Flow chart of the people count

5.3 VEHICLE COUNT:


START
READ THE BACKGROUND IMAGE
READ THE NEW IMAGE
CONVERT TO GRAYSCALE FORMAT
DETERMINATION OF WIDTH AND HEIGHT OF IMAGE
SET THRESHOLD VALUE=11
FINDING DIFFERENCE BETWEEN FRAMES BASED ON THRESHOLD
IF FRAME DIFF>11
NO
YES
ASSIGN 0 IF NO DIFFERENCE IS FOUND
ASSIGN A VARIABLE TO THAT IMAGE
INCREASING CONTRAST OF OUTPUT IMAGE USING imadjust()
FINDING GRAYTHRESHOLD VALUE USING graythresh()
ADDING GAUSSIAN NOISSE TO THE OUTPUT DIFFERENCE
A

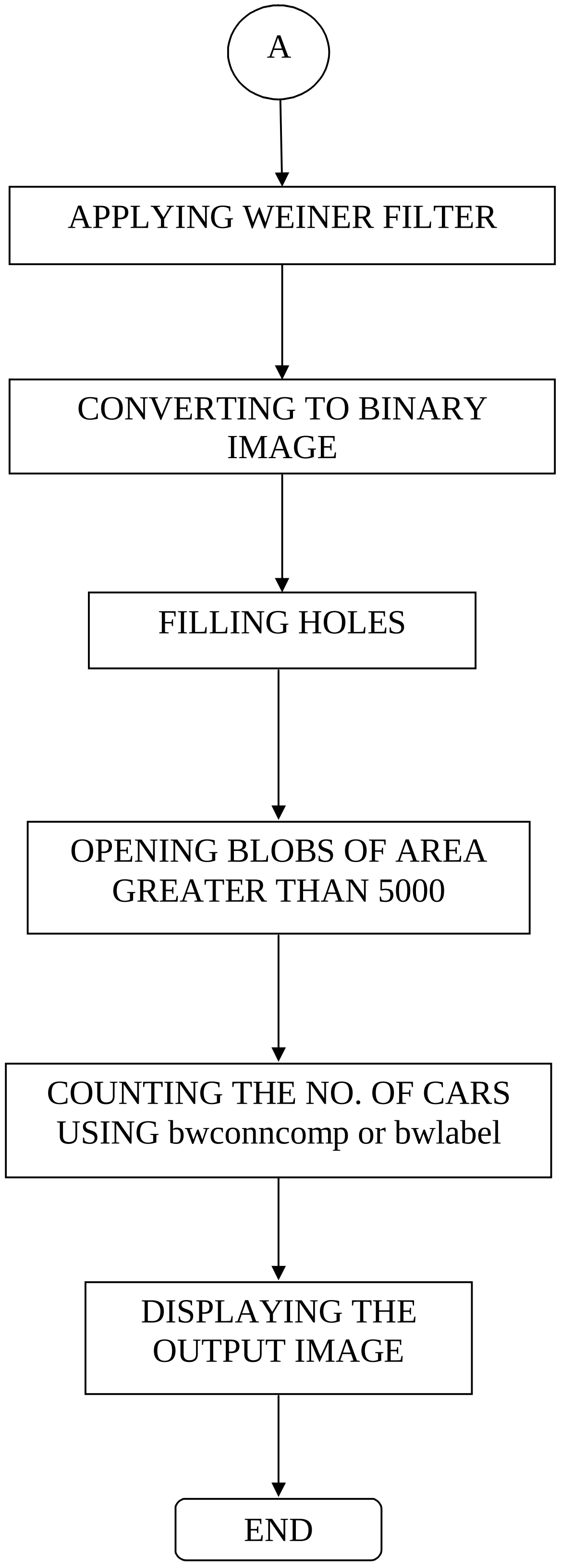

A
APPLYING WEINER FILTER
CONVERTING TO BINARY IMAGE
FILLING HOLES
OPENING BLOBS OF AREA GREATER THAN 5000
COUNTING THE NO. OF CARS USING bwconncomp or bwlabel
DISPLAYING THE OUTPUT IMAGE
END

Fig 5.3 Flow chart of the vehicle count

CHAPTER 6

6. MODULE DESCRIPTION:

6.1 CELL COUNT:

6.1.1 Image acquisition:

Imread() -Getting the input imagereads a grayscale or color image from the file specified by the string filename. If the file is not in the current folder, or in a folder on the MATLAB path, specify the full pathname.

imshow(I,[low high])- displays the grayscale image I, specifying the display range for I in [low high]. The value low (and any value less than low) displays as black; the value high (and any value greater than high) displays as white. Values in between are displayed as intermediate shades of gray, using the default number of gray levels.

6.1.2Color to gray conversion:

graythresh(J(:,:,2)) -computes a global threshold (level) that can be used to convert an intensity image to a binary image with im2bw. level is a normalized intensity value that lies in the range [0, 1].The graythresh function uses Otsu's method, which chooses the threshold to minimize the intraclass variance of the black and white pixels. Multidimensional arrays are converted automatically to 2-D arrays using reshape. The graythresh function ignores any nonzero imaginary part of I.

BW1=im2bw(J(:,:,2),L) -converts the grayscale image I to a binary image. The output image BW replaces all pixels in the input image with luminance greater than level with the value 1 (white) and replaces all other pixels with the value 0 (black). Specify level in the range [0,1]. This range is relative to the signal levels possible for the image's class. Therefore, a level value of 0.5 is midway between black and white, regardless of class.

6.1.3 Image enhancement:

imfill(BW1,'holes')-displays the binary image BW on the screen and lets you define the region to fill by selecting points interactively by using the mouse. To use this interactive syntax, BW must be a 2-D image.

bwareaopen(bw2,100)removes from a binary image all connected components (objects) that have fewer than P pixels, producing another binary image, BW2. The default connectivity is 8 for two dimensions, 26 for three dimensions.

6.1.4 Connected components:

Cc = bwconncomp()- it used for calculating objects in image.

6.1.5 Density:

Objects per_unit_area=cc.NumObjects/(size(bw4,1)*size(bw4,2))- Density is equal to connected components divided by size of image.

6.2 PEOPLE COUNT:

6.2.1 Getting the input image:

Imread() reads a grayscale or color image from the file specified by the string filename. If the file is not in the current folder, or in a folder on the MATLAB path, specify the full pathname.

6.2.2 Displaying the image:

imshow(I,[low high]) displays the grayscale image I, specifying the display range for I in [low high]. The value low (and any value less than low) displays as black; the value high (and any value greater than high) displays as white. Values in between are displayed as intermediate shades of gray, using the default number of gray levels.

6.2.3 Changing the color space:

makecform('srgb2lab') creates the color transformation structure C that defines the color space conversion specified by *type*. To perform the transformation, pass the color transformation structure as an argument to the applycform function. In this case the RGB color space is converted to LAB color space.

6.2.4 Finding gray thresh value:

graythresh(J(:,:,2)) computes a global threshold (level) that can be used to convert an intensity image to a binary image with im2bw. level is a normalized intensity value that lies in the range [0, 1]. The graythresh function uses Otsu's method, which chooses the threshold to minimize the intraclass variance of the black and white pixels.

Multidimensional arrays are converted automatically to 2-D arrays using reshape. The graythresh function ignores any nonzero imaginary part of I.

6.2.5 Converting to binary image:

BW1=im2bw(J(:,:,2),L) converts the grayscale image I to a binary image. The output image BW replaces all pixels in the input image with luminance greater than level with the value 1 (white) and replaces all other pixels with the value 0 (black). Specify level in the range [0,1]. This range is relative to the signal levels possible for the image's class. Therefore, a level value of 0.5 is midway between black and white, regardless of class.

6.2.6 Filling patches with holes:

imfill(BW1,'holes') displays the binary image BW on the screen and lets you define the region to fill by selecting points interactively by using the mouse. To use this interactive syntax, BW must be a 2-D image.

6.2.7 Opening area greater than a value:

bwareaopen(bw2,1890) removes from a binary image all connected components (objects) that have fewer than P pixels, producing another binary image, BW2. The default connectivity is 8 for two dimensions, 26 for three dimensions.

6.2.8 Finding density:

bwconncomp(bw3) returns the connected components CC found in BW3. The binary image BW3 can have any dimension. comp for finding the density of people in image.

6.2.9 Bwlabel:

Labels connected components in 2-D binary image.

6.2.10 Measuring properties of image:

Regionprops(labeledImage,'all') measures a set of properties for each connected component (object) in the binary image, BW. The image BW is a logical array; it can have any dimension.

6.2.11 Drawing bounding box:

The imagesc function scales image data to the full range of the current colormap and displays the image. imagesc(C) displays C as an image. Each element of C corresponds to a rectangular area in the image. The values of the elements of C are indices into the current colormap that determine the color of each patch.

6.3 VEHICLE COUNT:

6.3.1 Getting the input image:

Imread() reads a grayscale or color image from the file specified by the string filename. If the file is not in the current folder, or in a folder on the MATLAB path, specify the full pathname.In this module we get two input image one is the image with vehicles and the other is a background image without vehicles in it.

6.3.2 Converting color image to gray:

Rgb2gray() converts both the images to gray images. Convert both the images into gray level by using double precision.

6.3.3 Foreground detection:

Set threshold value=11, find the difference between the two images by using abs() [absolute], abs can help find the absolute between the two images. If the difference between the two images is greater than the threshold value of 11 then those will be displayed as blobs at the output.

6.3.4 Morphological operation:

imageadjust(imadjust()) used to adjust the image intensity values to the color map.

graythresh() used to set a suitable gray threshold value for the output image.

Add Gaussian noise to the output image and filter it using wiener filter.

Convert image to binary and fill holes if necessary. Open blobs of area greater than 5000, this will help detect vehicles.

6.3.5 Counting and density calculation:

Count the no of blobs by using bwconncomp, the number of blobs gives the number of vehicles present in the image. From the count the density of traffic can be estimated

CHAPTER 7

7. SIMULATION RESULTS:

7.1 RICE COUNT:

7.1.1 Input Image:

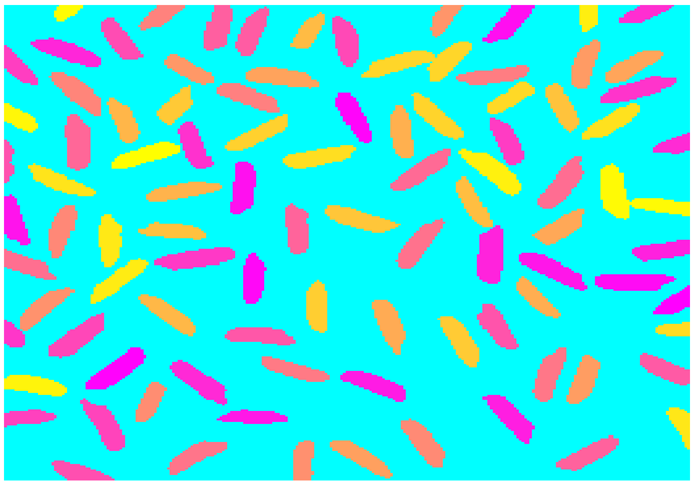


Fig 7.1.1 Input image of the rice count

7.1.2 CONVERTING INTO BINARY IMAGE:

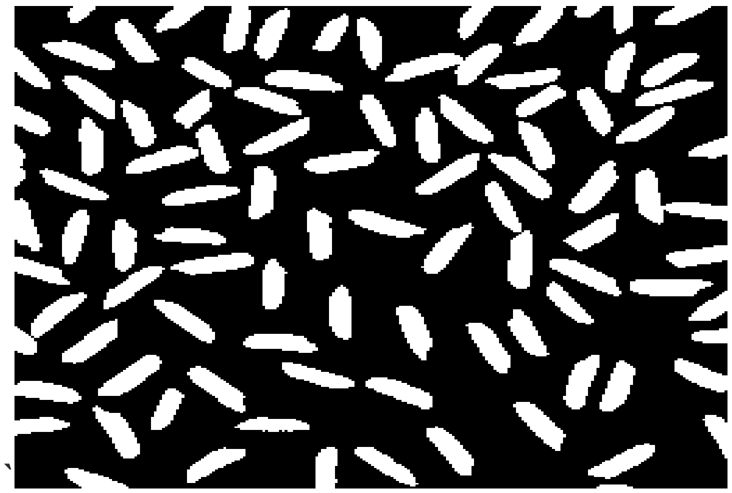


Fig 7.1.2 Binary image of the count

OUTPUT:

```
existing
Cc =
Connectivity: 8
ImageSize: [256 256]
NumObjects: 95
PixelIdxList: {1x95 cell}
Objects_per_unit_area =
0.0014
```

7.1.3 PADDING ZEROS:

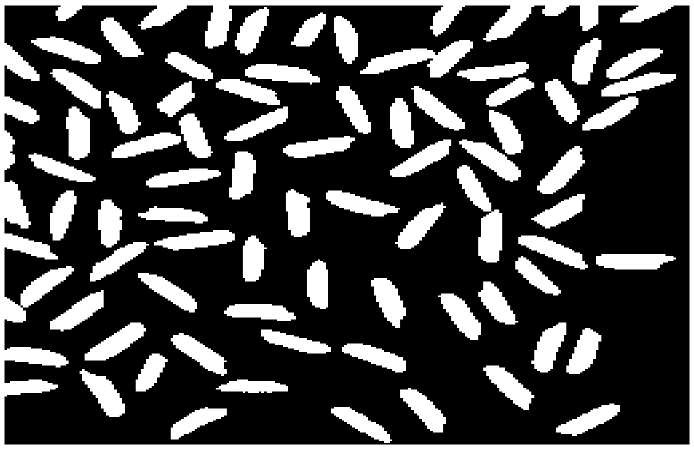


Fig 7.1.3 Objects cleared at two side borders

OUTPUT:

```
padding
cc =
   Connectivity: 8
ImageSize: [256 256]
NumObjects: 85
PixelIdxList: {1x85 cell}
objects_per_unit_area_2 =     0.0013
```

7.1.4 PROPOSED WAY OF CALCULATING DENSITY:

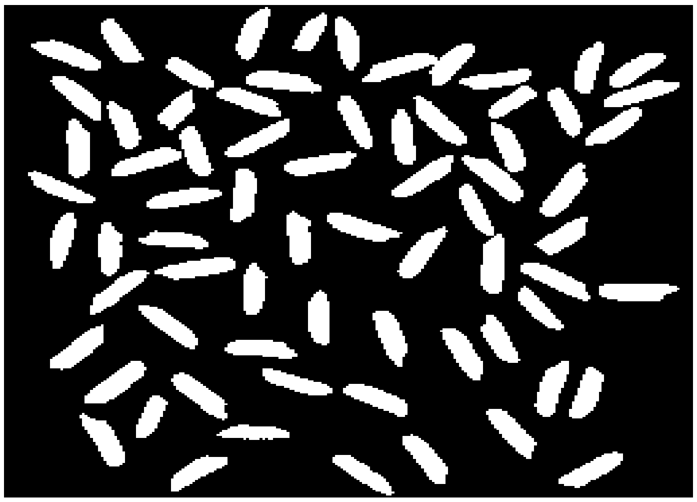


Fig 7.1.4 Objects cleared at four side borders

OUTPUT:

```
cc =
   Connectivity: 8
ImageSize: [256 256]
NumObjects: 69
PixelIdxList: {1x69 cell}
objects_per_unit_area_3 =
   0.0012
```

7.2 CELL COUNT:

7.2.1 INPUT IMAGE :

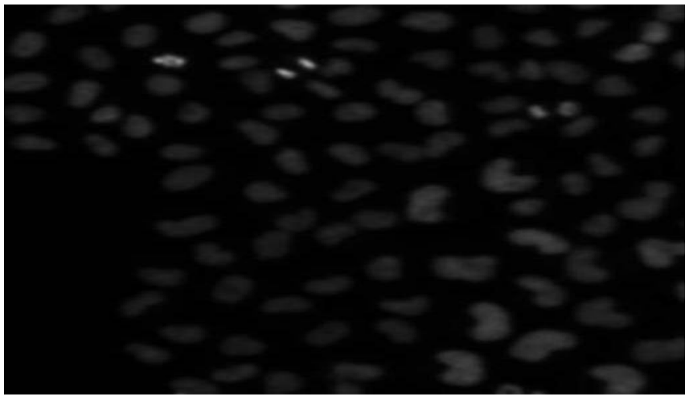


Fig 7.2.1 Input image of the cell count

7.2.2 CONVERTING INTO BINARY IMAGE:

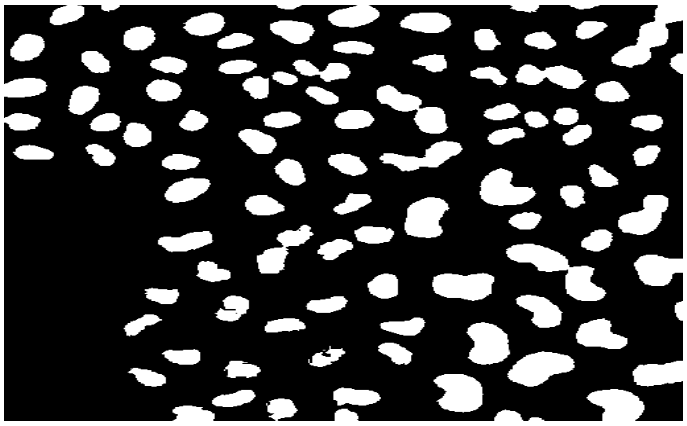


Fig 7.2.2 Binary image of cell count

Output :

```
cellseg
objects_per_unit_area =  8.0601e-005
 Connectivity: 8
      ImageSize: [530 515]
     NumObjects: 99
    PixelIdxList: {1x99 cell}
```

7.2.3 Using proposed method:

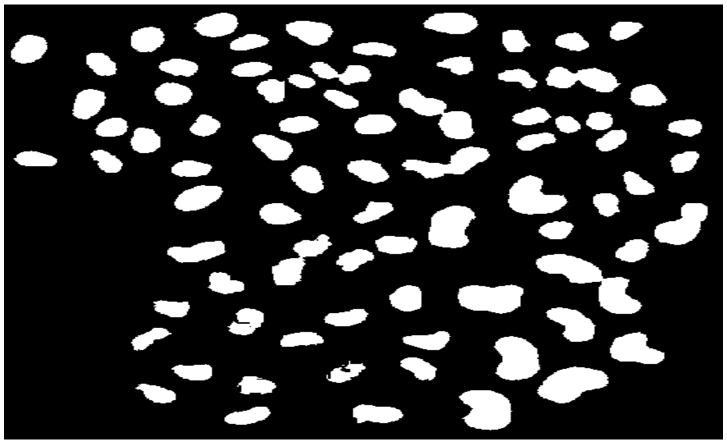


Fig 7.2.3 output image of the cell count

OUTPUT:

```
cellseg
objects_per_unit_area =
  2.0517e-004
cc =
    Connectivity: 8
       ImageSize: [530 515]
      NumObjects: 56
    PixelIdxList: {1x56 cell}
```

7.3 PEOPLE COUNTING:

7.3.1 Getting the input image:

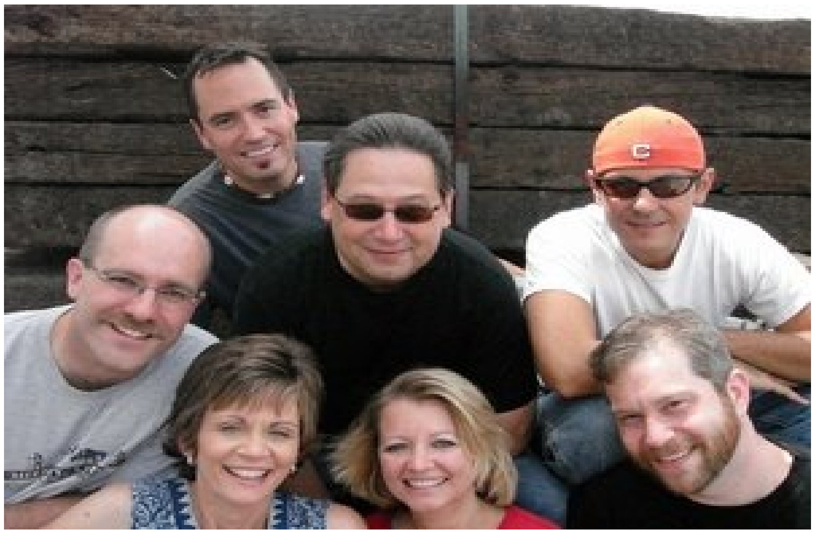


Fig 7.3.1 Input image

7.3.2 RGB color converted LAB color :

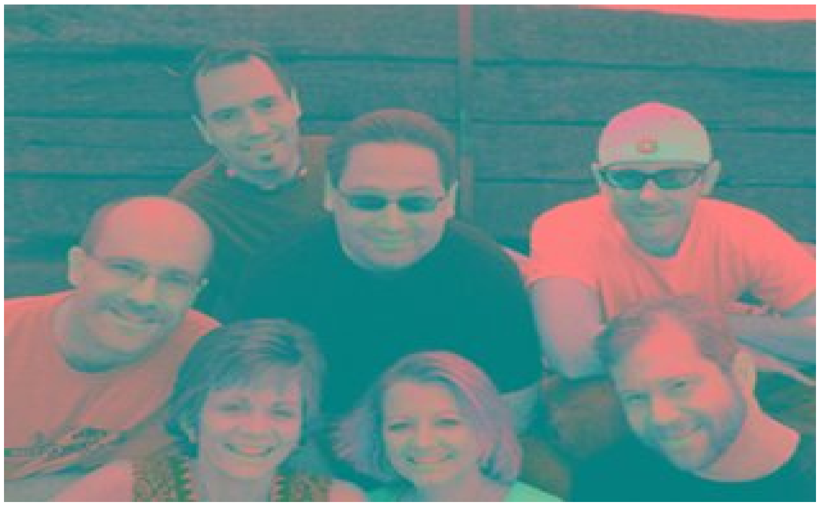
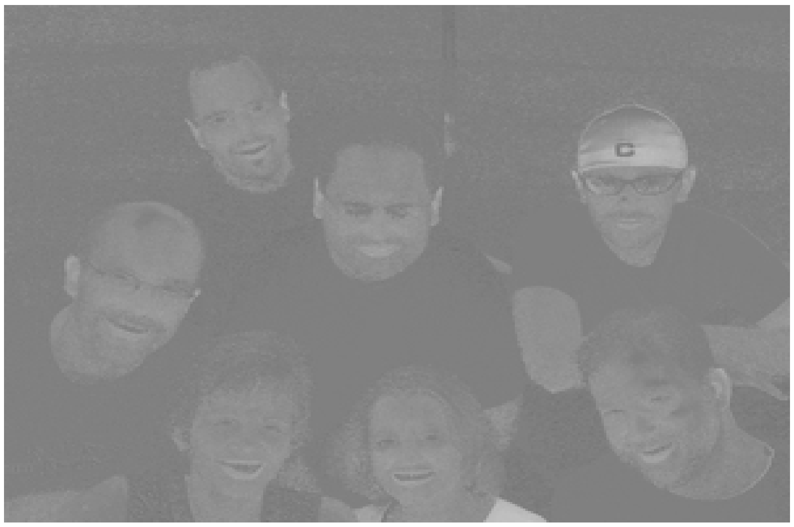
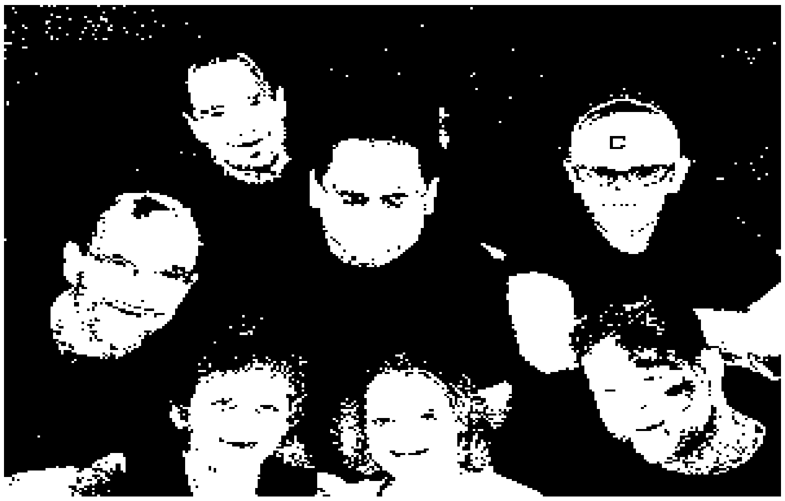


Fig 7.3.2 Converted LAB imag

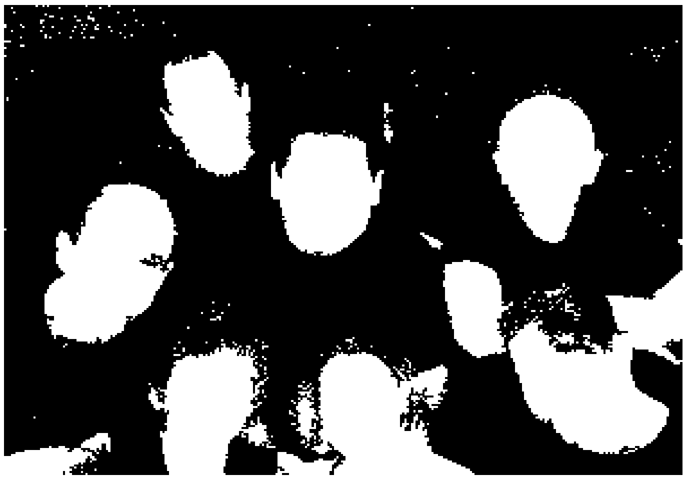
7.3.3 Equalise brightness to get skin area:



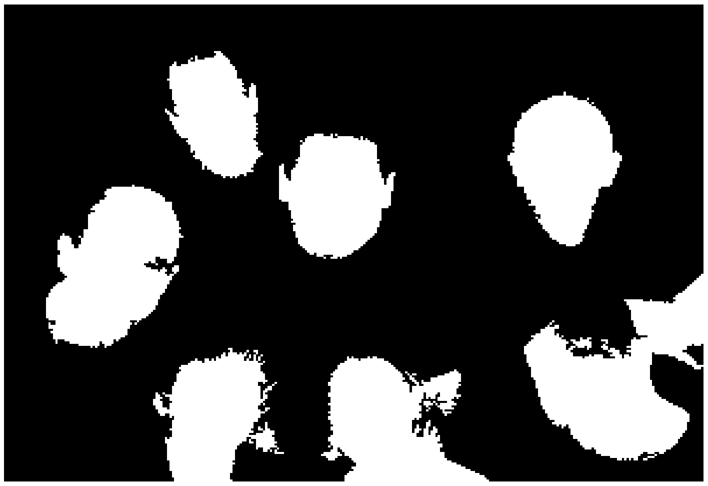
7.3.4 Converting to binary image:



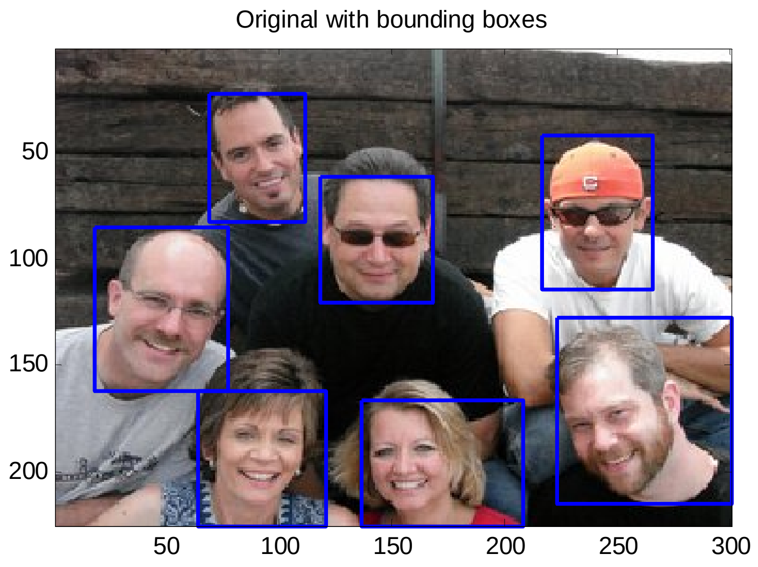
7.3.5 Filling patches with holes:



7.3.6 Opening area greater than a value:



7.3.7 Final output:

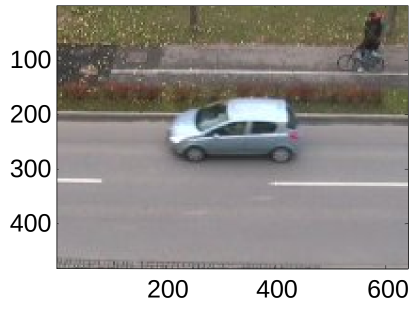


OUTPUT:

```
cc =
Connectivity: 8
       ImageSize: [225 300]
      NumObjects: 7
PixelIdxList: {1x7 cell}
density =  1.0370e-004
numberOfPeople =7
```

7.4 VEHICLE COUNT :

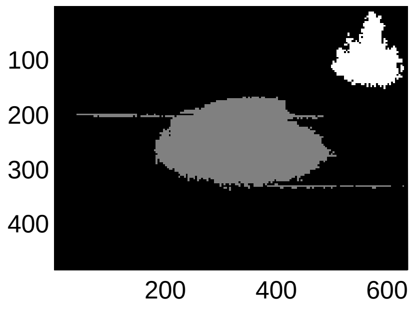
7.4.1 Orignal Frame:



7.4.2 Converted Frame:

7.4.3 BACKGND Frame:

7.4.4 Foreground:



OUTPUT:

```
cc =
 Connectivity: 8
       ImageSize: [480 640]
      NumObjects: 2
    PixelIdxList: {[39316x1 double]  [9899x1 double]}
     6.5104e-006
```

CHAPTER 8

8. CONCLUSION:

In this project work, a simple way of counting objects in the images and determining the correct value of density has been implemented. The number of objects and the density values are determined and verified for cell, people and vehicle counting applications. In cell count, the problem of counting many cells automatically as well as determining the accurate value of density has been solved. In people counting based on faces, even if too much skin is shown by a person in the image the algorithm can detect faces correctly. This provides an accurate count and density. Finally in vehicle counting, the algorithm used can detect the number of vehicles accurately including two wheelers without use of any hardware components.

In future, the same algorithm can be extended too many other applications other than the three applications covered in this project. Also, the present mat lab code used in cell count can be extended to find the correct count even if cells tend to overlap in the image. In people counting, if the two faces touch each other then inaccurate counts can be obtained. Hence methods to segment such images can be developed. In vehicle count, the same method can be extended to classify vehicles on the road such as car, truck, bike etc. Finally, the same counting applications can be shown in a simpler manner using other image processing software such as Python, ImageJ etc.

REFERENCES

Cell count:

[1] http://www.robots.ox.ac.uk/%7Evgg/research/counting/index.html.

[2] S.-Y. Cho, T. W. S. Chow, and C.-T.Leung.A neural-based crowd estimation by hybrid global learningalgorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 29(4):535–541, 1999.

[3] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. ICCV, 2009.

[4] X. Descombes, R. Minlos, and E. Zhizhina. Object extraction using a stochastic birth-and-death dynamics incontinuum. Jthisnal of Mathematical Imaging and Vision, 33(3):347–359, 2009.

[5] L. Dong, V. Parameswaran, V. Ramesh, and I. Zoghlami. Fast crowd segmentation using shape indexing.ICCV, pp. 1–8, 2007.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman.

[7] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. Machine Learning, 77(1):27–59, 2009.

[8] N. Ahuja and S. Todorovic.Extracting texels in 2.1d natural textures.ICCV, pp. 1–8, 2007.

[9] S. An, P. Peursum, W. Liu, and S. Venkatesh. Efficient algorithms for subwindow search in object detectionand localization. CVPR, pp. 264–271, 2009.

[10] D. Anoraganingrum. Cell segmentation with median filter and mathematical morphology operation.ImageAnalysis and Processing, International Conference on, 0:1043, 1999.

[11] O. Barinova, V. Lempitsky, and P. Kohli. On the detection of multiple object instances using Hough transforms.CVPR, 2010.

[12] J. L. Bentley. Programming pearls: Algorithm design techniques. Comm. ACM, 27(9):865–871, 1984.

[13] J. L. Bentley. Programming pearls: Perspective on performance. Comm. ACM, 27(11):1087–1092, 1984.

[14] L. Breiman. Random forests. Machine Learning, 45(1):5–32, 2001.

[15] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting peoplewithout people models or tracking. CVPR, 2008. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/workshop/index.html. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results.

People count:

L. H. Liang, A. H. Zhou, G. Y. Xu, B. Zhang and L. H. Zhao, "A survey of human face detection," Chinese Jthisnal of Computers, Vol. 25, No 5, pp. 450-458, 2002.

1. P. Viola and M. Jones, "Rapid object detection using a boosted cacade of simple features," Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 511-518, 2001.
2. L. M. Luo, Z. Y. Peng and G. Y. Xu, "The feature of skin color," Jthisnal of Software, Vol. 12, No 7, pp. 1032-1040, 2001.
3. R. L. Hsu, M. Abdel-Mottaleb and A. K. Jain, "Face detection in color images," IEEE Transactions on Pattern Analysis and Machine Inteligence, Vol. 24, pp. 696-706, 2002.
4. X. Tian, "Color space-based skin color subspace," Jthisnal of Xi'an University of Science and Technology, Vol. 12, No 4, pp. 369-371, 2001.
5. X. S. Gao, X. Z. Zhang and C. Y. Ji, "Face detection in color images," Techniques of Automation and Applications, Vol. 24, No. 10, pp. 54-56, 2005.
6. Y. Freund, "Boosting a weak learning algorithm by majority," Information and Computation, Vol. 121, No 2, pp. 256-285, 1995.

7. P. Viola and M. Jones, "Robust real-time face detection," International Jthisnal of Computer Vision, Vol. 57, No 2, pp. 137-154, 2001.

8. H. X. Liu and Y. C. Liu, "A face detection methord based on adaboost algorithm," Jthisnal of Shanghai Jiaotong University, Vol. 42, No 7, pp. 1119-1123, 2008.

9. M. Yang and D. J. Kriegman, "Detecting Faces in Images A Survey", IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol.24, No 1, pp.34-58, 2002.

   G. Yang and T. S. Huang, "Human Face Detection in Complex Background", Pattern Recognition, Vol.27, No 1, pp.53-63, 1994. [CrossRef]

10. C. Kotropoulos and I. Pitas, "Rule-Based Face Detection in Frontal Views", Proc. Int. Conf. Acoustics, Speech and Signal Processing, Vol.4, pp.2537-2540, 1997.

11. Sanjay K. Singh, D.S. Chauhan, MayankVatsa, Richa Singh, "A Robust Skin Color Based Face Detection Algorithm", Tamkang Jthisnal of Science and Engineering, Vol.6, No 4, pp.227-234, 2003.

12. K. Sobottka and I. Pitas, "Face Localization and Feature Extraction Based on Shape and Color Information", Proc. IEEE Int. Conf. Image Processing, Vol.3, pp.483-486, Lausanne, 1996.

13. J. Miao, B. Yin, K. Wang, L. Shen, et X. Chen, "A Hierarchical Multiscale and Multi-angle System for Human Face Detection in a Complex Background Using Gravity-Center Template", Pattern Recognition, Vol.32, No 7, pp.1237-1248, 1999. [CrossRef]

[14] H. A. Rowley, S. Baluja, and T. Kanade, "Neural Network-Based Face Detection", School of Computer Science, Carnegie Mellon University, Pittsburg Research Center, 1998.

Vehicle count:

[1] R. Chellappa “Vehicle detection and tracking using acoustic and video sensors”, 2004

[2] Sensor Line, Classax, Traffic Data Acquisition System, Germany, www.sensorline.de

[3] S. Gupte, “Detection and Classification of Vehicles”, University of Minnesota 2002.

[4] E. Atkociunas, "Image Processing in Road Traffic Analysis", Vilnius University, 2005

[5] B. Coifman "A Real-Time Computer Vision System for Vehicle Tracking and Traffic Surveillance" University of California, 1998

[6] C. Zhang "Adaptive Background Learning for Vehicle Detection and Spatio-TemporalTracking", IEEE, 2003

[7] R. Cucchiara “Vehicle Detection under Day and Night Illumination" Proc. of ISCS-IIA, Special session on vehicle traffic and surveillance, 1999

[8] Z.W. Kim "Fast Vehicle Detection with Probabilistic Feature Grouping and its Application to Vehicle Tracking", CiteSeer, 2001

[9] L. Di Stefano, ”Vehicle Detection and Tracking Using the Block Matching Algorithm", Department of Electronics, Computer Science and Systems (DEIS) University of Bologna, ITALY

APPENDIX I

MATLAB CODING

CODE FOR CALCULATING NUMBER OF CELLS AND DENSITY IN IMAGES

```
I = imread('dna.jpeg');%call image

bw0=imclearborder(I);

imshow(bw0),title('img 1');

bw1 = im2bw(bw0, graythresh(bw0));%apply threshold

figure;imshow(bw1),title('img 2');

bw2 = imfill(bw1,'holes');%fill hole in the image

figure;imshow(bw2),title('img 3');

bw3 = imopen(bw2, ones(5,5));%perform morphological ops in image

figure;imshow(bw3),title('img 4');

bw4 = bwareaopen(bw3, 100);%removes from a binary image all connected
comp of area less than 100 pixels

figure;imshow(bw4),title('img 5');

cc=bwconncomp(bw4)%count

objects_per_unit_area=cc.NumObjects/(size(bw4,1)*size(bw4,2))%Density
```

CODE FOR PEOPLE COUNT BASED ON FACES

```
%get the input image
I=imread('faces.jpg');
imshow(I),title('Image:1');
%change the color space
cform = makecform('srgb2lab');
J=applycform(I,cform);
figure;imshow(J),title('Image:2');
%equalise brightness to get skin area
K=J(:,:,2);% 2nd page of 3-d vector j
figure;imshow(K),title('Image:3');
L=graythresh(J(:,:,2));% find appropriate gray thresh value
BW1=im2bw(J(:,:,2),L);% convert to binary image based on threshold
figure;imshow(BW1),title('Image:4');
bw2=imfill(BW1,'holes');% fill patches with holes
figure;imshow(bw2)
bw3 = bwareaopen(bw2,1890); %opens area greater than 1890
cc=bwconncomp(bw3)% connected comp for finding the density of people in
image
density=cc.NumObjects / (size(bw3,1) * size(bw3,2))
figure;imshow(bw3)
labeledImage = bwlabel(bw3, 8);%same as connected components
figure;imshow(labeledImage)
blobMeasurements = regionprops(labeledImage,'all');%measure all properties of
the image
numberOfPeople = size(blobMeasurements, 1)% count the number of people
 % draw bounding boxes
imagesc(I);
```

```
hold on;
title('Original with bounding boxes');
for k = 1 : numberOfPeople % Loop through all blobs.
% Find the mean of each blob.
% directly into regionprops.
thisBlobsBox = blobMeasurements(k).BoundingBox; % Get list of pixels in
current blob.
x1 = thisBlobsBox(1);%1st side
y1 = thisBlobsBox(2);%2nd side
x2 = x1 + thisBlobsBox(3);%3rd side
y2 = y1 + thisBlobsBox(4);%4th side
 x = [x1 x2 x2 x1 x1];
y = [y1 y1 y2 y2 y1];
%subplot(3,4,2);
plot(x, y, 'LineWidth', 2);
end
```

CODE FOR VEHICLE COUNT AND DENSITY OF TRAFFIC

```
clc;
clear all;
MV = imread('cars1.png'); %To read image
MV1 = imread('backgnd.png');
A = double(rgb2gray(MV));%convert to gray
B= double(rgb2gray(MV1));%convert 2nd image to gray
[height width] = size(A); %image size?
 h1 = figure(1);
     %Foreground Detection
    thresh=11;
    fr_diff = abs(A-B);
    for j = 1:width
        for k = 1:height
        if (fr_diff(k,j)>thresh)
            fg(k,j) = A(k,j);
        else
                    fg(k,j) = 0;
        end
        end
    end
    subplot(2,2,1) , imagesc(MV), title (['Orignal Frame']);
    subplot(2,2,2) , imshow(mat2gray(A)), title ('converted Frame');
    subplot(2,2,3) , imshow(mat2gray(B)), title ('BACKGND Frame ');
    sd=imadjust(fg);% adjust the image intensity values to the color map
    level=graythresh(sd);
    m=imnoise(sd,'gaussian',0,0.025);% apply Gaussian noise
    k=wiener2(m,[5,5]);%filtering using Weiner filter
    bw=im2bw(k,level);
    bw2=imfill(bw,'holes');
    bw3 = bwareaopen(bw2,5000);
    labeled = bwlabel(bw3,8);
    cc=bwconncomp(bw3)
    Densityoftraffic = cc.NumObjects/(size(bw3,1)*size(bw3,2));
    blobMeasurements = regionprops(labeled,'all');
numberofcars = size(blobMeasurements, 1);
    subplot(2,2,4) , imagesc(labeled), title (['Foreground']);
    hold off;
    disp(numberofcars);% display number of cars
    disp(Densityoftraffic);%display number of vehicles
```