VISION-X: SMART NAVIGATION SYSTEM FOR VISUALLY CHALLENGED PERSONS

REPORT SUBMITTED BY

ANINDA GHOSH
Dept. of ELECTRONICS AND COMMUNICATION ENGINEERING
Registration No.: **121040110063** of **2012-13**
Roll No.: **10400312069**

ANNU SINGH
Dept. of ELECTRONICS AND COMMUNICATION ENGINEERING
Registration No.: **121040110066** of **2012-13**
Roll No.: **10400312072**

NITISH KUMAR THAKUR
Dept. of ELECTRONICS AND COMMUNICATION ENGINEERING
Registration No.: **121040110311** of **2012-13**
Roll No.: **10400312117**

RIYA SETT
Dept. of ELECTRONICS AND COMMUNICATION ENGINEERING
Registration No.: **121040110332** of **2012-13**
Roll No.: **10400312138**

UNDER THE GUIDANCE OF

Mr. TUHIN UTSAB PAUL
Dept. of ELECTRONICS AND COMMUNICATION ENGINEERING

INSTITUTE OF ENGINEERING AND MANAGEMENT
Gurukul, Y-12, Block EP, Sector – V, Salt Lake, Kolkata – 700 091

Affiliated to

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY
(formerly **WEST BENGAL UNIVERSITY OF TECHNOLOGY)**

CERTIFICATE

This is to certify that the project entitled "Smart Navigation System for Visually Challenged Persons", prepared by Aninda Ghosh (10400312069), Annu Singh (10400312072), Nitish Kumar Thakur (10400312138) and Riya Sett (10400312138) of B. Tech., Electronics and Communication Engineering, Final Year, is a bona fide work carried out by them, has been done according to the regulations of the Degree of Bachelor of Technology in Electronics and Communication Engineering, under the guidance and supervision of Prof. Tuhin Utsab Paul, Dept. of Electronics and Communication Engineering, in the 8th Semester of the academic year 2012-16. The candidates have fulfilled the requirements for the submission of the project report. The content of the report has not been submitted to any other college or university for any reward or certificate.

I am glad to inform that the work is entirely original and the performance found to be quite satisfactory.

Prof. Tuhin Utsab Paul
Asst. Professor, Dept. of ECE
Institute of Engineering and Management

Dr. Amlan Kusum Nayak
Principle
Institute of Engineering and Management

Prof. (Dr.) Malay Gangopadhyaya
H.O.D, Dept. of ECE
Institute of Engineering and Management

PREFACE

The report has been submitted towards the successful completion of the Fourth Year project of BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING of INSTITUTE OF ENGINEERING AND MANAGEMENT, MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY (formerly WEST BENGAL UNIVERSITY OF TEcHNOLOGY), KOLKATA, INDIA.

Date: June, 2016

Place: Salt Lake, Kolkata

ANINDA GHOSH ANNU SINGH

NITISH KUMAR THAKUR RIYA SETT

Declaration of Originality and Compliance of organizational ethics

We hereby declare that this report contains literature survey and original research work by the undersigned candidates are done as part of their studies in the course curriculum.

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct. We also declare that, a complete list of citation and references for the preparation of the report has been appended herewith.

Name & Roll No.: Aninda Ghosh (104003120069)
Annu Singh (104003120072)
Nitish Kumar Thakur (104003120)
Riya Sett (10400312138)

Title: Vision-X: Smart Navigation System for Visually Challenged Persons

Date: June, 2016

ANINDA GHOSH ANNU SINGH

NITISH KUMAR THAKUR RIYA SETT

ACKNOWLEDGEMENT

The sense of fulfillment that comes with the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement had been the driving force. The acknowledgement serves as a testimony for the extreme gratitude and thankfulness that I bear for all the people who had been my guiding light for this humongous task.

I would like to convey my sincere and wholehearted appreciation to my project guide Prof. Tuhin Utsab Paul, Department of Electronics and Communication Engineering, Institute of Engineering and Management, MAKAUT, for this continuous guidance and constructive advice throughout the course of the thesis work. I am indeed indebted to my guide for his doubt clearing sessions that he had organized at frequent intervals for my better understanding of the concepts and unknown facts that helped me inculcate greater interest in the area of my thesis work.

Finally, I would also like to thank my beloved friends, batch mates for their much coveted selfless support and most importantly my family for being truthfully a generous helping hand to explore the domain of technology that has enlightened and further expanded my horizon of knowledge.

There are yet a lot of people whom I would like to express my thankfulness who have contributed towards my such an extensive learning process and will surely be the same in forthcoming future. Lastly I would apologize if their names are not mentioned here just because of my bad memory or lack of space.

Date: June, 2016

Place: IEM, Salt Lake, Kolkata

ANINDA GHOSH ANNU SINGH

NITISH KUMAR THAKUR RIYA SETT

ABSTRACT

The project relates to a small electronic device that can be used to guide any visually impaired and differently abled person within any building, prior known or unknown and also in outdoor environments like fields or roads. The processor of the device uses different image processing algorithms to help the corresponding person to navigate with ease. For the visually impaired persons, the output will be a computer generated voice, instructing how to navigate and for the other persons who are able to see clearly but somehow not able to move independently, the output will be through the motor controllers, hinged to the wheels of the wheelchair.

The main purpose of this device is to help visually-impaired persons to move within any building or outdoors independently. We know that such persons use sticks to find obstacles in their way and have to depend on others to move safely. Using this device they will be much more independent in case of movement. They do not need to ask for help to cross the door or to find a clear path as this device uses information from surroundings and analyzing the information, it helps to find a clear path for the person.

Keywords: *grayscale image, segmentation, top-down approach, bottom-up approach, clustering, classifier, morphology, dilation, erosion, filtering, raspberry pi, ground detection, door detection, stair detection, human detection.*

INDEX

INTRODUCTION

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

- Importing the image via image acquisition tools;
- Analysing and manipulating the image;
- Output in which result can be altered image or report that is based on image analysis.

There are two types of methods used for image processing namely, analogue and digital image processing. Analogue image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. Digital image processing techniques help in manipulation of the digital images by using computers. The three general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement, and display, information extraction.

In this lecture we will talk about a few fundamental definitions such as image, digital image, and digital image processing. Different sources of digital images will be discussed and examples for each source will be provided. The continuum from image processing to computer vision will be covered in this lecture. Finally we will talk about image acquisition and different types of image sensors.

Digital image processing is the use of computer algorithm to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be model in the form of multidimensional systems.

Modern digital technology has made it possible to manipulate multi-dimensional signals with systems that range from simple digital circuits to advanced parallel computers. The goal of this manipulation can be divided into three categories:

- Image Processing
- Image Analysis
- Image Understanding

image in → image out
image in → measurements out
image in → high-level description out

An image defined in the “real world” is considered to be a function of two real variables, for example, a(x,y) with a as the amplitude (e.g. brightness) of the image at the real coordinate position (x,y). An image may be considered to contain sub-images sometimes referred to as...Image Processing Fundamentals regions–of–interest, ROIs, or simply regions. This concept reflects the fact that images frequently contain collections of objects each of which can be the basis for a region. In a sophisticated image processing system it should be possible to apply specific image processing operations to selected regions. Thus one part of an image (region) might be processed to suppress motion blur while another part might be processed to improve color rendition.

The amplitudes of a given image will almost always be either real numbers or integer numbers. The latter is usually a result of a quantization process that converts a continuous range (say, between 0 and 100%) to a discrete number of levels. In certain image-forming processes, however, the signal may involve photon counting which implies that the amplitude would be inherently quantized. In other image forming procedures, such as magnetic resonance imaging, the direct physical measurement yields a complex number in the form of a real magnitude and a real phase. For the remainder of this book we will consider amplitudes as reals or integers unless otherwise indicated.

1.1 Digital Image Definitions

A digital image a[m,n] described in a 2D discrete space is derived from an analog image a(x,y) in a 2D continuous space through a sampling process that is frequently referred to as digitization. For now we will look at some basic definitions associated with the digital image.

The 2D continuous image a(x,y) is divided into N rows and M columns. The intersection of a row and a column is termed a pixel. The value assigned to the integer coordinates [m,n] with {m=0,1,2,...,M–1} and {n=0,1,2,...,N–1} is a[m,n].

In fact, in most cases a(x,y)—which we might consider to be the physical signal that impinges on the face of a 2D sensor—is actually a function of many variables including depth (z), color (λ), and time (t).

1.2 Some places where image processing is needed:

- Optical imaging (cameras, microscopes).
- Medical imaging (CT, MRI, ultrasound, diffuse optical, advanced microscopes).
- Astronomical imaging (telescopes).
- Geophysical Imaging (seismics, electromagnetics).
- Radar and hyperspectral imaging (surveillance and remote sensing).
- Printing (color, dot matrix).
- Video and Imaging Compression and Transmission (JPEG, MPEG, HDTV,...).
- Computer Vision (robots, license plate reader, tracking human motion).
- Computer graphics: rendering and shading, representation.
- Commercial Software (Photoshop).
- Hardware (FPGA, DSP, cell processor implementation of compute intensive algorithms).
- Security and Digital Rights Management (Watermarking, Biometrics).

Digital Image processing is a subset of the electronic domain wherein the image is converted to an array of small integers, called pixels, representing a physical quantity such as scene radiance, stored in a digital memory, and processed by computer or other digital hardware. Digital image processing, either as enhancement for human observers or performing autonomous analysis, offers advantages in cost, speed, and flexibility, and with the rapidly falling price and rising performance of personal computers it has become the dominant method in use.

1.3 Purpose of Image processing

The purpose of image processing is divided into 5 groups. They are:

1. Visualization - Observe the objects that are not visible.
2. Image sharpening and restoration - To create a better image.
3. Image retrieval - Seek for the image of interest.

4. Measurement of pattern – Measures various objects in an image.
5. Image Recognition – Distinguish the objects in an image.

1.4 Types of Methods

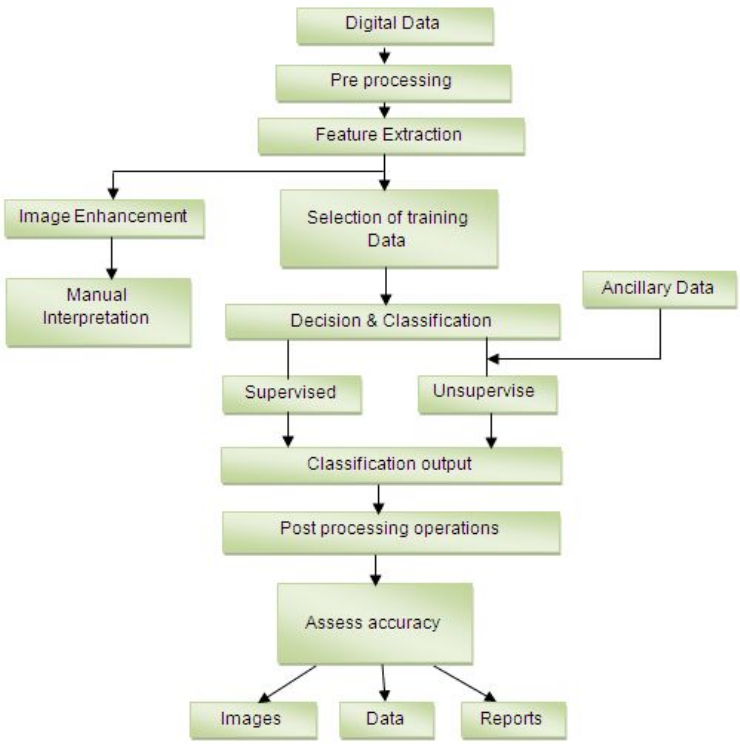
The two types of methods used for Image Processing are Analog and Digital Image Processing. Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that has to be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing.

Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from satellite platform contains deficiencies. To get over such flaws and to get originality of information, it has to undergo various phases of processing. The three general phases that all types of data have to undergo while using digital technique are Pre- processing, enhancement and display, information extraction.

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional signals while applying already set signal processing methods to them.

It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:



- Importing the image with optical scanner or by digital photography.
- Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.
- Output is the last stage in which result can be altered image or report that is based on image analysis.

1.5 Relationship between a digital image and a signal

If the image is a two dimensional array then what does it have to do with a signal? In order to understand that , We need to first understand what is a signal?

Signal

In physical world, any quantity measurable through time over space or any higher dimension can be taken as a signal. A signal is a mathematical function, and it conveys some information.

A signal can be one dimensional or two dimensional or higher dimensional signal. One dimensional signal is a signal that is measured over time. The common example is a voice signal.

The two dimensional signals are those that are measured over some other physical quantities.

The example of two dimensional signal is a digital image. We will look in more detail in the next tutorial of how a one dimensional or two dimensional single and higher signals are formed and interpreted.

Relationship

Since anything that conveys information or broadcast a message in physical world between two observers is a signal. That includes speech or (human voice) or an image as a signal. Since when we speak , our voice is converted to a sound wave/signal and transformed with respect to the time to person we are speaking to. Not only this , but the way a digital camera works, as while acquiring an image from a digital camera involves transfer of a signal from one part of the system to the other.

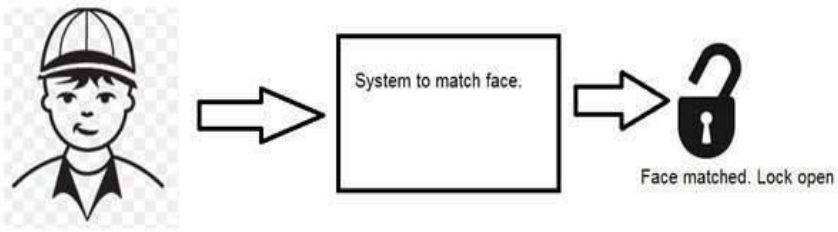
1.6 How a digital image is formed

Since capturing an image from a camera is a physical process. The sunlight is used as a source of energy. A sensor array is used for the acquisition of the image. So when the sunlight falls upon the object, then the amount of light reflected by that object is sensed by the sensors, and a continuous voltage signal is generated by the amount of sensed data. In order to create a digital image , we need to convert this data into a digital form. This involves sampling and quantization. (They are discussed later on). The result of sampling and quantization results in an two dimensional array or matrix of numbers which are nothing but a digital image.

1.7 Overlapping fields

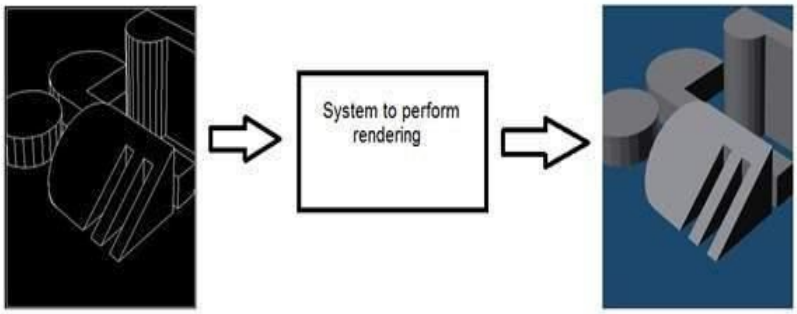
Machine/Computer vision

Machine vision or computer vision deals with developing a system in which the input is an image and the output is some information. For example: Developing a system that scans human face and opens any kind of lock. This system would look something like this.



Computer graphics

Computer graphics deals with the formation of images from object models, rather than the image is captured by some device. For example: Object rendering. Generating an image from an object model. Such a system would look something like this.



Artificial intelligence

Artificial intelligence is more or less the study of putting human intelligence into machines. Artificial intelligence has many applications in image processing. For example: developing computer aided diagnosis systems that help doctors in interpreting images of X-ray , MRI e.t.c and then highlighting conspicuous section to be examined by the doctor.

Signal processing

Signal processing is an umbrella and image processing lies under it. The amount of light reflected by an object in the physical world (3d world) is pass through the lens of the camera and it becomes a 2d signal and hence result in image formation. This image is then digitized using methods of signal processing and then this digital image is manipulated in digital image processing.

1.8 History of Photography

Origin of camera

The history of camera and photography is not exactly the same. The concepts of camera were introduced a lot before the concept of photography.

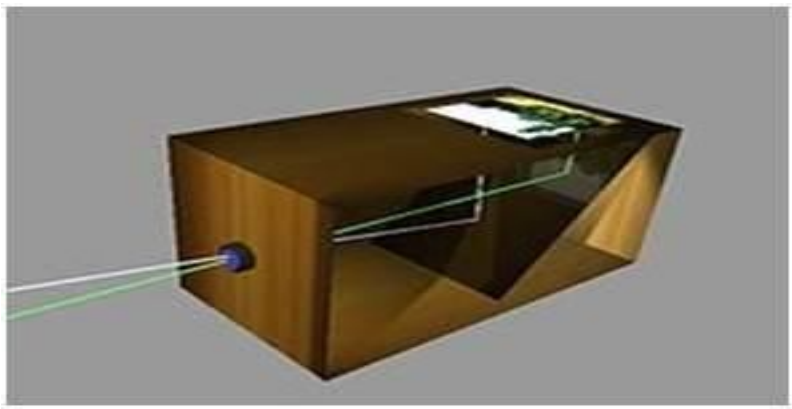
Camera Obscura

The history of the camera lies in ASIA. The principles of the camera were first introduced by a Chinese philosopher MOZI. It is known as camera obscura. The cameras evolved from this principle.

The word camera obscura is evolved from two different words. Camera and Obscura. The meaning of the word camera is a room or some kind of vault and Obscura stands for dark.

The concept which was introduced by the Chinese philosopher consist of a device, that

project an image of its surrounding on the wall. However it was not built by the Chinese.



The creation of camera obscura

The concept of Chinese was bring in reality by a Muslim scientist Abu Ali Al-Hassan Ibn al-Haitham commonly known as Ibn al-Haitham. He built the first camera obscura. His camera follows the principles of pinhole camera. He build this device in somewhere around 1000.

Portable camera

In 1685, a first portable camera was built by Johann Zahn. Before the advent of this device , the camera consist of a size of room and were not portable. Although a device was made by an Irish scientist Robert Boyle and Robert Hooke that was a transportable camera, but still that device was very huge to carry it from one place to the other.

Origin of photography

Although the camera obscura was built in 1000 by a Muslim scientist. But its first actual use was described in the 13th century by an English philosopher Roger Bacon. Roger suggested the use of camera for the observation of solar eclipses.

Da Vinci

Although much improvement has been made before the 15th century , but the improvements and the findings done by Leonardo di ser Piero da Vinci was remarkable. Da Vinci was a great artist , musician , anatomist , and a war engineer. He is credited for many inventions. His one of the most famous painting includes, the painting of Mona Lisa.

Da vinci not only built a camera obscura following the principle of a pinhole camera but

also uses it as drawing aid for his art work. In his work , which was described in Codex Atlanticus , many principles of camera obscura has been defined.

His camera follows the principle of a pinhole camera which can be described as :When images of illuminated objects penetrate through a small hole into a very dark room you will see [on the opposite wall] these objects in their proper form and color, reduced in size in a reversed position, owing to the intersection of rays.

First photograph

The first photograph was taken in 1814 by a French inventor Joseph Nicephore Niepce. He captures the first photograph of a view from the window at Le Gras, by coating the pewter plate with bitumen and after that exposing that plate to light.

First underwater photograph

The first underwater photograph was taken by an English mathematician William Thomson using a water tight box. This was done in 1856.

The origin of film

The origin of film was introduced by an American inventor and a philanthropist known as George Eastman who is considered as the pioneer of photography.

He founded the company called as Eastman Kodak , which is famous for developing films. The company starts manufacturing paper film in 1885. He first created the camera Kodak and then later Brownie. Brownie was a box camera and gain its popularity due to its feature of Snapshot.

After the advent of the film , the camera industry once again got a boom and one invention lead to another.

Leica and Argus

Leica and argus are the two analog cameras developed in 1925 and in 1939 respectively. The camera Leica was built using a 35mm cine film.

Argus was another camera analog camera that uses the 35mm format and was rather inexpensive as compared by Leica and became very popular.

Analog CCTV cameras

In 1942 a German engineer Walter Bruch developed and installed the very first system of the analog CCTV cameras. He is also credited for the invention of color television in the 1960.

Photo Pac

The first disposable camera was introduced in 1949 by Photo Pac. The camera was only a one time use camera with a roll of film already included in it. The later versions of Photo pac were water proof and even have the flash.

Digital Cameras

Mavica by Sony

Mavica (the magnetic video camera) was launched by Sony in 1981 was the first game changer in digital camera world. The images were recorded on floppy disks and images can be viewed later on any monitor screen.

It was not a pure digital camera , but an analog camera. But got its popularity due to its storing capacity of images on a floppy disks. It means that you can now store images for a long lasting period , and you can save a huge number of pictures on the floppy which are replaced by the new blank disc , when they got full. Mavica has the capacity of storing 25 images on a disk.

One more important thing that mavica introduced was its 0.3 megapixel capacity of capturing photos.

Fuji DS-1P camera by Fuji films 1988 was the first true digital camera

Nikon D1 was a 2.74 megapixel camera and the first commercial digital SLR camera developed by Nikon , and was very much affordable by the professionals.

Today digital cameras are included in the mobile phones with very high resolution and quality.

THEORETICAL DEFINITION

2.1 Conversion of RGB Image to Grayscale Image

An RGB image, sometimes referred to as a *truecolor* image, is stored as an *m*-by-*n*-by-3 data array that defines red, green, and blue color components for each individual pixel. RGB images do not use a palette. The color of each pixel is determined by the combination of the red, green, and blue intensities stored in each color plane at the pixel's location. Graphics file formats store RGB images as 24-bit images, where the red, green, and blue components are 8 bits each. This yields a potential of 16 million colors. The precision with which a real-life image can be replicated has led to the nickname "truecolor image."

An RGB array can be of class double, uint8, or uint16. In an RGB array of class double, each color component is a value between 0 and 1. A pixel whose color components are (0,0,0) is displayed as black, and a pixel whose color components are (1,1,1) is displayed as white. The three color components for each pixel are stored along the third dimension of the data array. For example, the red, green, and blue color components of the pixel (10,5) are stored in RGB(10,5,1), RGB(10,5,2), and RGB(10,5,3), respectively.

An RGB image can be converted to Grayscale Image by using the following equation [1]:

$$0.2989R + 0.5870G + 0.1140B\,.$$

For example a RGB image is shown below:

After converting the above RGB image to grayscale image, it looks like:

2.2 Segmentations

In computer vision, **image segmentation** is the process of partitioning a digital image into multiple segments (sets of pixels, also known as superpixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.[2,3] Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such ascolor, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s)[2]. When applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used to create 3D reconstructions with the help of interpolation algorithms like Marching cubes.

The segmentation process can be approached in two ways: *Top-Down* and *Bottom-Up Approach.*

2.2.1 Top-Down Approach

A top-down approach (also known as *stepwise design* and in some cases used as a synonym of *decomposition*) is essentially the breaking down of a system to gain insight into its compositional subsystems in a reverse engineering fashion. In a top-down approach an overview of the system is formulated, specifying but not detailing any first-level subsystems. Each subsystem is then refined in yet greater detail, sometimes in many additional subsystem levels, until the entire specification is reduced to base elements. A top-down model is often specified with the assistance of "black boxes", these make it easier to manipulate. However, black boxes may fail to elucidate elementary mechanisms or be detailed enough to realistically validate the model. Top down approach starts with the big picture. It breaks down from there into smaller segments [4].

Some top-down approaches for image segmentation are described as follows:

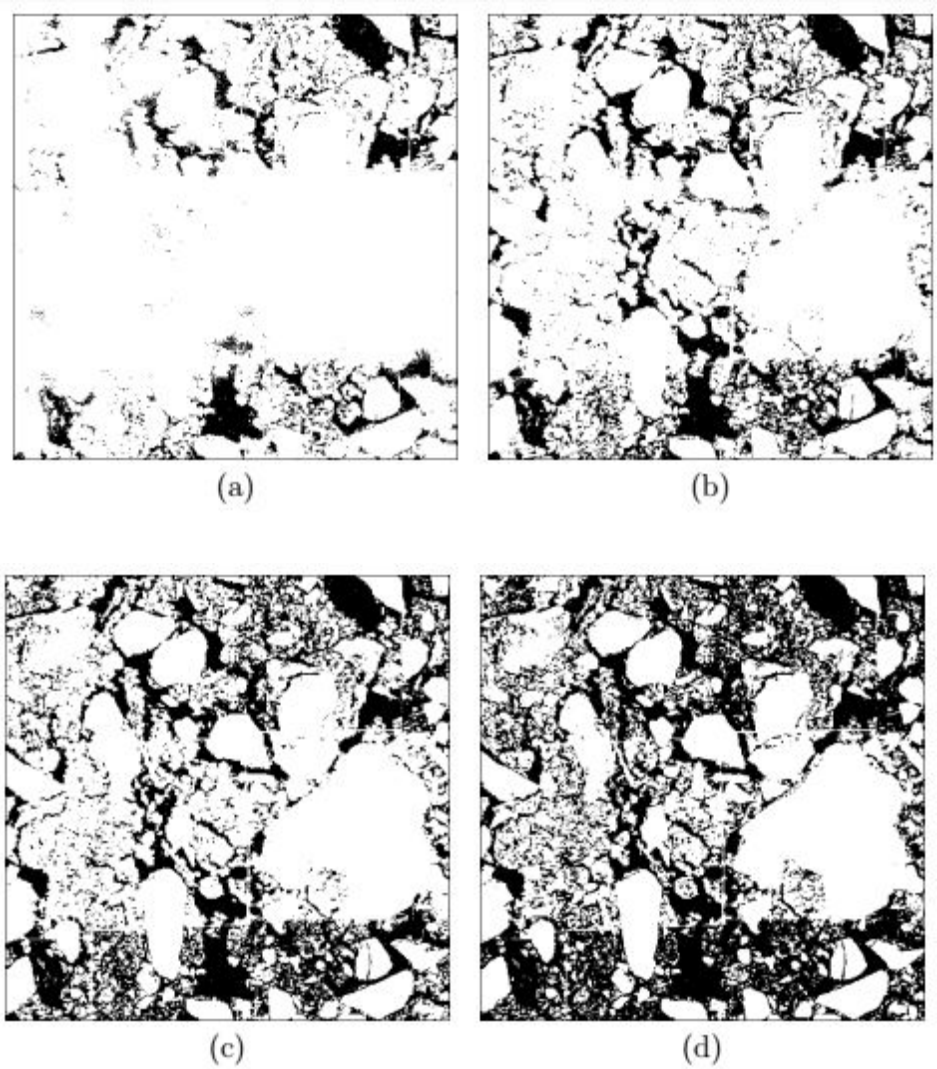
2.2.1.1. Threshold based segmentation

The simplest method of image segmentation is called the thresholding method. This method is based on a clip-level (or a threshold value) to turn a gray-scale image into a binary image.

Given a single threshold, t, the pixel located at lattice position (i, j), with grayscale value f_{ij} , is allocated to region 1 if $f_{ij} \leq t$

Otherwise, the pixel is allocated to region 2.

In many cases t is chosen manually by the scientist, by trying a range of values of t and seeing which one works best at identifying the objects of interest. Fig 1.1 shows some segmentations of the soil image. In this application, the aim was to isolate soil material from the air-filled pores which appear as the darker pixels. Thresholds of 7, 10, 13, 20, 29 and 38 were chosen in Figs 1.1(a) to (f) respectively, to identify approximately 10, 20, 30, 40, 50 and 60% of the pixels as being pores. Fig 1.1(d), with a threshold of 20, looks best because most of the connected pore network has been correctly identified, as has most of the soil matrix.



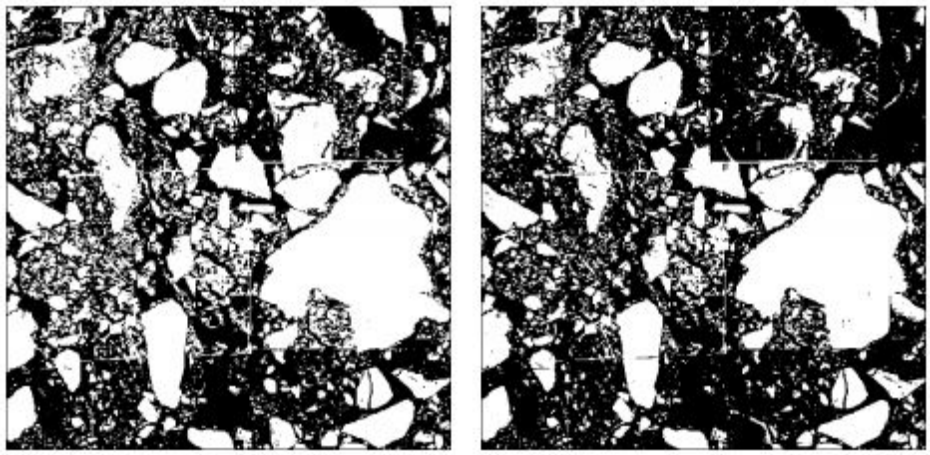


Fig. 1.1: Six segmentations of the soil image, obtained using manually-selected thresholds of (a) 7, (b) 10, (c) 13, (d) 20, (e) 29 and (f) 38. These correspond to approximately 10%, 20%,..., 60%, respectively, of the image being displayed as black.

Note that:

- Although pixels in a single thresholded category will have similar values (either in the range 0 to t, or in the range (t + 1) to 255), they will not usually constitute a single connected component. This is not a problem in the soil image because the object (air) is not necessarily connected, either in the imaging plane or in three-dimensions. In other cases, thresholding would be followed by dividing the initial categories into subcategories of connected regions.
- More than one threshold can be used, in which case more than two categories are produced.
- Thresholds can be chosen automatically

In 1.1 we will consider algorithms for choosing the threshold on the basis of the histogram of grayscale pixel values. In 1.2, manually- and automatically-selected classifiers for multivariate images will be considered. Finally, in 1.3, thresholding algorithms which make use of context (that is, values of neighbouring pixels as well as the histogram of pixel values) will be presented.

i. Histogram-based thresholding

We will denote the histogram of pixel values by $h_0, h_1, ..., h_N$, where h_k specifies the number of pixels in an image with grayscale value k and N is the maximum pixel value (typically 255). Ridler and Calvard (1978) and Trussell (1979) proposed a simple algorithm for choosing a single threshold. We shall refer to it as the inter-means algorithm. First we will describe the algorithm in words, and then mathematically.

Initially, a guess has to be made at a possible value for the threshold. From this, the mean values of pixels in the two categories produced using this threshold are calculated. The threshold is repositioned to lie exactly half way between the two means. Mean values are calculated again and a new threshold is obtained, and so on until the threshold stops changing value.

Mathematically, the algorithm can be specified as follows.

- Make an initial guess at t: for example, set it equal to the median pixel value, that is, the value for which

$$\sum_{k=0}^{t} h_k \geq \frac{n^2}{2} \geq \sum_{k=0}^{t-1} h_k ,$$

where n^2 is the number of pixels in the $n \times n$ image

- Calculate the mean pixel value in each category. For values less than or equal to t, this is given by:

$$\mu_1 = \frac{\sum_{k=0}^{t} k h_k}{\sum_{k=0}^{t} h_k}$$

Whereas, for values greater than t, it is given by:

$$\mu_2 = \frac{\sum_{k=t+1}^{N} k h_k}{\sum_{k=t+1}^{N} h_k}$$

- Re-estimate t as half-way between the two means, i.e.

$$t = [\frac{\mu_1 + \mu_2}{2}]$$

where [ ] denotes 'the integer part of' the expression between the brackets.
Repeat steps (2) and (3) until t stops changing value between consecutive evaluations.

The inter-means algorithm has a tendency to find a threshold which divides the histogram in two, so that there are approximately equal numbers of pixels in the two categories. In many applications, such as the soil image, this is not appropriate. One way to overcome this drawback is to modify the algorithm as follows.

Consider a distribution which is a mixture of two Gaussian distributions. Therefore, in the absence of sampling variability, the histogram is given by:

$$h_k = n^2 \{p_1 \phi_1(k) + p_2 \phi_2(k)\} \qquad \text{for k=1,2,...,N.}$$

Here, p_1 and p_2 are proportions (such that $p_1 + p_2 = 1$) and $\phi_1(k)$ denotes the probability density of a Gaussian distribution, that is

$$\phi_l(k) = \frac{1}{\sqrt{2\pi\sigma_l^2}} exp\{\frac{-(k-\mu)^2}{2\sigma_l^2}\} \qquad \text{for l=1,2,...}$$

where μ_1 and σ_l^2 are the mean and variance of pixel values in category l. The best classification criterion, i.e. the one which mis-classifies the least number of pixels, allocates pixels with value k to category 1 if

$$p_1 \phi_1(k) \geq p_2 \phi_2(k)$$

and otherwise classifies them as 2. After substituting for φ and taking logs, the inequality becomes

$$k^2 \{\frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2}\} - 2k\{\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\} + \{\frac{\mu_1^2}{\sigma_1^2} - \frac{\mu_2^2}{\sigma_2^2} + log\frac{\sigma_1^2 p_2^2}{\sigma_2^2 p_1^2}\} \leq 0$$

The left side of the inequality is a quadratic function in k. Let A, B and C denote the three terms in

curly brackets, respectively. Then the criterion for allocating pixels with value k to category 1 is:

$$k^2A - 2kB + C \leq 0$$

ii. Multivariate classifiers

There are many ways to extend the concept of a threshold in order to segment a multivariate image, such as:

1. Allocate pixel (i, j) to category 1 if $f_{ij,m} \leq t_m$ for m=1,2,...,M where subscript m denotes the variate number, and there are M variates.
2. Or, more generally, use a box classifier where the conditions for category 1 are: $t_{1,m} \leq f_{ij,m} \leq t_{2,m}$ for m=1,2,...,M
3. The condition could be a linear threshold: $c^T f_{ij} \leq t$

Or, the range of pixel values for category 1 could be a general set S in M-dimensional space:

$$f_{ij} \in S \subset R^M$$

Multivariate thresholding criteria are more difficult to specify than univariate ones. Therefore the approach often adopted in a segmentation which is manual (i.e. under the user's control) is to choose pixels which are known to belong to target categories (known as the training set) and then use them to classify the rest of the image. This is called supervised classification.

2.2.1.2 Edge Detection based Segmentation

Edge detection is the name for a set of mathematical methods which aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed *edges*. The same problem of finding discontinuities in 1D signals is known as step detection and the problem of finding signal discontinuities over time is known as change detection.

The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. It can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to:[5][6]

- Discontinuities in path
- Discontinuities in surface orientation
- Changes in material properties
- Variations in scene illumination

In the ideal case, the result of applying an edge detector to an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation. Thus, applying an edge detection algorithm to an image may significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image. If the edge detection step is successful, the subsequent task of interpreting the information contents in the original image may therefore be substantially simplified. However, it is not always possible to obtain such ideal edges from real life images of moderate complexity.

Edges extracted from non-trivial images are often hampered by *fragmentation*, meaning that the edge curves are not connected, missing edge segments as well as *false edges* not corresponding to interesting phenomena in the image – thus complicating the subsequent task of interpreting the image data[7].

The edges extracted from a two-dimensional image of a three-dimensional scene can be classified as either viewpoint dependent or viewpoint independent. A *viewpoint independent edge* typically reflects inherent properties of the three-dimensional objects, such as surface markings and surface shape. A *viewpoint dependent edge* may change as the viewpoint changes, and typically reflects the geometry of the scene, such as objects occluding one another.

There are many methods for edge detection, but most of them can be grouped into two categories, search-based and zero-crossing based. The search-based methods detect edges by first computing a measure of edge strength, usually a first-order derivative expression such as the gradient magnitude, and then searching for local directional maxima of the gradient magnitude using a computed estimate of the local orientation of the edge, usually the gradient direction. The zero-crossing based methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, usually the zero-crossings of the Laplacian or the zero-crossings of a nonlinear differential expression. As a pre-processing step to edge detection, a smoothing stage, typically Gaussian smoothing, is almost always applied.

Some edge detection operators are as follows:

- Robert Operator
- Prewitt Operator
- Canny Method
- Sobel Operator
- 4-Neighbour Operator
- Laplacian Edge Detector
- Laplacian of Gauss Edge Detector

2.2.1.3. Otsu Algorithm

In computer vision and image processing, **Otsu's method**, named after Nobuyuki Otsu is used to automatically perform clustering-based image thresholding[8], or, the reduction of a grayscale image to a binary image. The algorithm assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), it then calculates the optimum threshold separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal[9]. Consequently, Otsu's method is roughly a one-dimensional, discrete analog of Fisher's Discriminant Analysis.

In Otsu's method we exhaustively search for the threshold that minimizes the intra-class variance (the variance within the class), defined as a weighted sum of variances of the two classes:

$$\sigma_\omega^2(t) = \omega_0(t) \cdot \sigma_0^2(t) + \omega_1(t) \cdot \sigma_1^2(t)$$

Weights $\omega_{0,1}$ are the probabilities of the two classes separated by a threshold t and $\sigma^2_{0,1}$ are variances of these two classes.

The class probability $\omega_{0,1}(t)$ is computed from the L histograms:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i)$$
$$\omega_1(t) = \sum_{i=t}^{L} p(i)$$

Otsu shows that minimizing the intra-class variance is the same as maximizing inter-class variance:[9]

$$\sigma^2{}_b(t) = \sigma^2 - \sigma^2{}_\omega(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 = \omega_0\omega_1[\mu_0 - \mu_T]^2$$

which is expressed in terms of class probabilities ω and class means μ. While the class mean $\mu_{0,1,T}(t)$ is:

$$\mu_0(t) = \sum_{i=0}^{t-1} ip(i)/\omega_0$$

$$\mu_1(t) = \sum_{i=t}^{L-1} ip(i)/\omega_1$$

$$\mu_T(t) = \sum_{i=t}^{L-1} ip(i)$$

The following relations can be easily verified:

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T$$

$$\omega_0 + \omega_1 = 1$$

The class probabilities and class means can be computed iteratively. This idea yields an effective algorithm.

Algorithm

The following steps of Otsu Algorithm are:

1. Compute histogram and probabilities of each intensity level
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$.
3. Step through all possible thresholds $t = 1....$ maximum intensity
    a. Update ω_i and μ_i
    b. Compute $\sigma^2{}_b(t)$
4. Desired threshold corresponds to the maximum $\sigma^2{}_b(t)$

Otsu's method exhibits the relatively good performance if the histogram can be assumed to have bimodal distribution and assumed to possess a deep and sharp valley between two peaks. But if the object area is small compared with the background area, the histogram no longer exhibits

bimodality[10]. And if the variances of the object and the background intensities are large compared to the mean difference, or the image is severely corrupted by additive noise, the sharp valley of the gray level histogram is degraded. Then the possibly incorrect threshold determined by Otsu's method results in the segmentation error. Here we define the object size to be the ratio of the object area to the entire image area and the mean difference to be the difference of the average intensities of the object and the background.

2.2.2 Bottom Up Approach

A *bottom-up* approach is the piecing together of systems to give rise to more complex systems, thus making the original systems sub-systems of the emergent system. Bottom-up processing is a type of information processing based on incoming data from the environment to form a perception. From a Cognitive Psychology perspective, information enters the eyes in one direction (sensory input, or the "bottom"), and is then turned into an image by the brain that can be interpreted and recognized as a perception (output that is "built up" from processing to final cognition). In a bottom-up approach the individual base elements of the system are first specified in great detail. These elements are then linked together to form larger subsystems, which then in turn are linked, sometimes in many levels, until a complete top-level system is formed. This strategy often resembles a "seed" model, by which the beginnings are small but eventually grow in complexity and completeness. However, "organic strategies" may result in a tangle of elements and subsystems, developed in isolation and subject to local optimization as opposed to meeting a global purpose.

BU segmentation emphasizes region-based properties such as the homogeneity of color (for example, [11]), intensity, or texture (for example, [12]), the smoothness and continuity of bounding contours (for example, [13] and [14]), or a combination of these region and boundary properties (for example, [15], [16], and [17]). In this work, we use the Segmentation by Weighted Aggregation (SWA) algorithm described in Section 4 to identify a hierarchy of homogenous image regions at multiple scales.

Some bottom-up approaches of segmentation are as follows:

2.2.2.1. Clustering

Clustering is the task of assigning set of objects into groups called as clusters so that the objects in one cluster are more similar than the objects in other cluster. Clustering itself is not one specific algorithm but it is task which can be performed by various algorithms that differs from each other in their methods of computing/finding the cluster. Clustering is process of grouping similar image pixels according to some property into one cluster so that the resulting output cluster shows high intra-cluster similarities and low inter-cluster similarities. Clustering process is an unsupervised classification of data points into groups or clusters [18] [19] .

Steps of Clustering

Clustering is a task which involves number of stages. Typical clustering process used the

following steps [18] [20]:

i) Characteristics Representation: In this step the type, dimensions and features of available data are checked. For A Review of Image Data Clustering Techniques 213 that it involves processes such as feature selection or feature extraction.

ii) Similarity Measurement: In this step similarity among data points is measured. Generally various distance measurement methods viz. Euclidian Distance, Mean Square etc. are used to measure similarity between different data points.

iii) Collecting the data points: In this step the data points are grouped together into clusters, based on similarity measures obtained from the previous step.

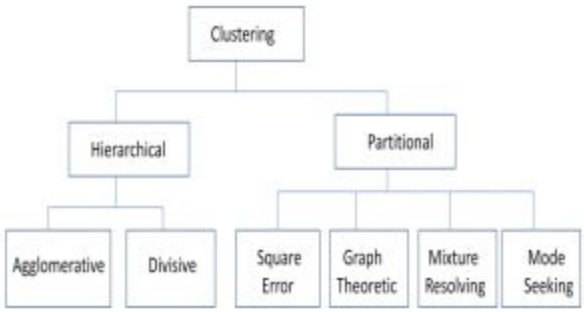
iv) Data abstraction: In this process data is represented with compact description of individual cluster and furthermore the cluster prototype i.e. the centroid of the cluster is calculated and used as final representation [18] .

v) Output Validation: This is an important stage in clustering process. In this step the resulting output clusters are observed to determine whether the resulted output is meaningful or not. It can be done by various methods; either it compares resulted output with a priori structure, or check whether the structure is intrinsically appropriate for data sets or not, or compares two derived outputs with each other and measure their exclusive merits.

Clustering is useful in number of application as it clusters the raw data and find out the hidden features / patterns in the database [21]. So it is widely used in image processing, data mining, image retrieval, pattern recognition, image segmentation and so on.

Classification of clustering

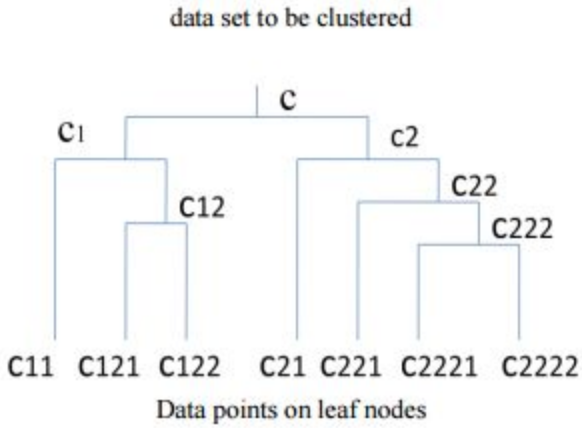
Clustering algorithms are broadly classified as i) Hierarchical clustering ii) Partitional clustering [22].The following figure shows the different techniques of clustering,



I. Hierarchical Clustering

Hierarchical clustering, clusters the given image, based on the concept of pixel being more closely related to nearby pixels than the pixels which are farther i.e. these algorithms groups the pixels into cluster based on their distances. Hierarchical clustering represents data into a tree structure form. In which the whole data set is represented by root node and the individual data points are represented by leaf node. The intermediate nodes in a tree structure represent the similarity among the pixel data points. At different stages in structure different cluster will form thus expected clustering will obtained by cutting cluster structure at respective stages [23]. The

figure below shows the hierarchical tree structure form. In this clustering technique numbers of algorithms are proposed based on the method by which distances are computed. Hierarchical clustering mainly classified as agglomerative hierarchical clustering, starts with single elements in distinct patterns and merges it until the stopping criterion reached. The cluster divisive hierarchical clustering starts with considering complete data set as single cluster and splitting it into clusters. Along with the distance function the linkage criterion is also an important factor in hierarchical clustering. Since, in this method cluster consist of multiple elements, so multiple elements are involved in computing distances. The most commonly used linkage criterions are single-linkage and complete-linkage.



II. Partitional Clustering

Partitional Clustering algorithms separate the pixels or data points into number of partitions. These partitions are referred as clusters. The partitional clustering organizes data into single partition instead of representing data into nested structure like hierarchical clustering. Partitional clustering is more useful for large data set in which it is difficult to represent data in tree structure. The partitional clustering is classified as square error clustering, Graph theoretic clustering, mixture resolving clustering and mode seeking clustering [18] .

➔ *Square Error Clustering*

In partitional clustering the data points are randomly initialize and assign it to the predefined number of clusters depend on similarity between data points and cluster center until convergence criterion has been reached. The most frequently used convergence criterion in partitional criterion is squared error algorithms. The main advantage of square error algorithms is that it works well with isolated and compact clusters [18] .

➔ *Graph Theoretic Clustering*

Graph Theoretic Clustering algorithm is divisive clustering algorithm. This algorithm first forms the minimal spanning tree (MST) for the given data. Then it obtained clusters by deleting edges of largest length from the minimal spanning tree structure [18] .

➔ *Mixture-Resolving Clustering*

Clustering can be understood by studying density distribution functions. In the mixture resolving algorithm the parametric distribution function like Gaussian distribution

are used and the vectors of component density are form. These vectors are grouped together iteratively based on maximum likelihood estimation to form the clusters [18] .

➔ *Mode-Seeking Clustering*

In non-parametric technique the algorithms are developed inspired by the Parzen window approach. It forms clusters by creating bins with large counts in multidimensional histogram of the input mixture patterns. This approach considered as mode seeking approach [18] .

Clustering Algorithms

Based on the above methods there are number of clustering algorithm available, like k-means clustering, N-cut clustering, Mean shift clustering etc.

- *K-means Clustering*

K-means is commonly used simplest algorithm which employs the square error criterion. In this algorithm the number of partitions is initially defined. The cluster centers are randomly initialized for predefined number of clusters. Each data point is then assigned to one of the nearest cluster. The cluster centers are then re-estimated which and new centroid is calculated. This process is repeated until the convergence has been reached or until no significant change occurs in cluster center [23]. k-means is easy to implement and its time complexity is less. But the output of k-means algorithm depends on selection of predefined clusters. If the initial number of clusters is not properly chosen then the output of algorithm may converge to false cluster locations and completely different clustering result [18] [24] .

The K-means algorithm may be viewed as a gradient-decent procedure,which begins with an initial set of K cluster-centers and iteratively updates it so as to decrease the error function.

This linear complexity is one of the reasons for the popularity of the K-means algorithms. Even if the number of instances is substantially large (which often is the case nowadays), this algorithm is computationally attractive. Thus, the K-means algorithm has an advantage in comparison to other clustering methods (e.g. hierarchical clustering methods), which have non-linear complexity.

Other reasons for the algorithm's popularity are its ease of interpretation, simplicity of implementation, speed of convergence and adaptability to sparse data (Dhillon and Modha, 2001).

The Achilles heel of the K-means algorithm involves the selection of the initial partition. The algorithm is very sensitive to this selection, which may make the difference between global and local minimum.

Being a typical partitioning algorithm, the K-means algorithm works well only on data sets having isotropic clusters, and is not as versatile as single link algorithms, for instance.

In addition, this algorithm is sensitive to noisy data and outliers (a single outlier can increase the squared error dramatically); it is applicable only when mean is defined (namely, for numeric attributes);and it requires the number of clusters in advance, which is not trivial when no prior knowledge is available.

The use of the K-means algorithm is often limited to numeric attributes. Haung (1998) presented the K-prototypes algorithm, which is based on the K-means algorithm but

removes numeric data limitations while preserving its efficiency. The algorithm clusters objects with numeric and categorical attributes in a way similar to the K-means algorithm. The similarity measure on numeric attributes is the square Euclidean distance; the similarity measure on the categorical attributes is the number of mismatches between objects and the cluster prototypes.

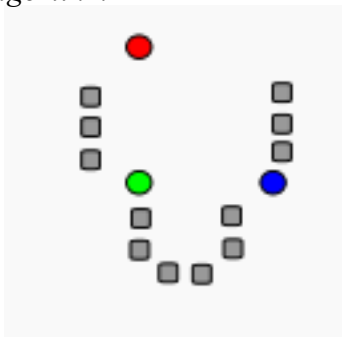
Another partitioning algorithm, which attempts to minimize the SSE is the K-medoids or PAM (partition around medoids — (Kaufmann and Rousseeuw,1987)). This algorithm is very similar to the K-means algorithm. It differs from the latter mainly in its representation of the different clusters. Each cluster is represented by the most centric object in the cluster, rather than by the implicit mean that may not belong to the cluster.

The K-medoids method is more robust than the K-means algorithm in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean. However, its processing is more costly than the K-means method. Both methods require the user to specify K, the number of clusters.

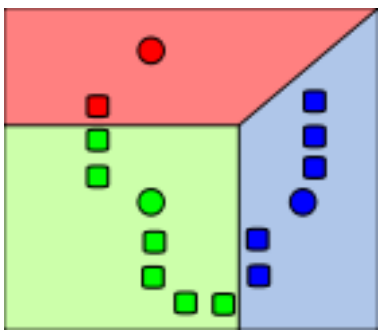
Other error criteria can be used instead of the SSE. Estivill-Castro (2000) analyzed the total absolute error criterion. Namely, instead of summing up the squared error, he suggests to summing up the absolute error. While this criterion is superior in regard to robustness, it requires more computational effort.

Algorithm

Demonstration of standard algorithm

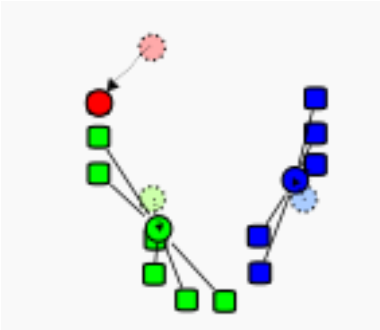


1. k initial "means" (in this case k=3) are randomly generated within the data domain (shown in color).

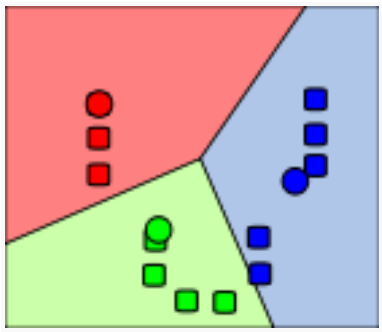


2. k clusters are created by associating every observation with the nearest mean. The partitions

here represent the Voronoi diagram generated by the means.



3. The centroid of each of the clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

As it is a heuristic algorithm, there is no guarantee that it will converge to the global optimum, and the result may depend on the initial clusters. As the algorithm is usually very fast, it is common to run it multiple times with different starting conditions. However, in the worst case, k-means can be very slow to converge: in particular it has been shown that there exist certain point sets, even in 2 dimensions, on which k-means takes exponential time, that is $2^{\Omega(n)}$, to converge.[26] These point sets do not seem to arise in practice: this is corroborated by the fact that the smoothed running time of k-means is polynomial.

The "assignment" step is also referred to as expectation step, the "update step" as maximization step, making this algorithm a variant of the generalized expectation-maximization algorithm."

The most common algorithm uses an iterative refinement technique. Due to its ubiquity it is often called the *k*-means algorithm; it is also referred to as Lloyd's algorithm, particularly in the computer science community.

Given an initial set of *k* means $m_1^{(1)},\ldots,m_k^{(1)}$ (see below), the algorithm proceeds by alternating between two steps:[27]

Assignment Step: Assign each observation to the cluster whose mean yields the least within-cluster sum of squares (WCSS). Since the sum of squares is the squared Euclidean distance, this is intuitively the "nearest" mean. Mathematically, this means partitioning the observations according to the Voronoi diagram generated by the means.

$$S_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\|^2 \leq \|x_p - m_j^{(t)}\|^2 \ \forall j, 1 \leq j \leq k\},$$

where each x_p is assigned to exactly one $S^{(t)}$, even if it could be assigned to two or more of them.

Update step: Calculate the new means to be the centroids of the observations in the new clusters.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

Since the arithmetic mean is a least-squares estimator, this also minimizes the within-cluster sum of squares (WCSS) objective.

The algorithm has converged when the assignments no longer change. Since both steps optimize the WCSS objective, and there only exists a finite number of such partitionings, the algorithm must converge to a (local) optimum. There is no guarantee that the global optimum is found using this algorithm.

The algorithm is often presented as assigning objects to the nearest cluster by distance. The standard algorithm aims at minimizing the WCSS objective, and thus assigns by "least sum of squares", which is exactly equivalent to assigning by the smallest Euclidean distance. Using a different distance function other than (squared) Euclidean distance may stop the algorithm from converging. Various modifications of k-means such as spherical k-means and k-medoids have been proposed to allow using other distance measures.

- *Distance Measures*

Since clustering is the grouping of similar instances/objects, some sort of measure that can determine whether two objects are similar or dissimilar is required. There are two main type of measures used to estimate this relation: distance measures and similarity measures.

Many clustering methods use distance measures to determine the similarity or dissimilarity between any pair of objects. It is useful to denote the distance between two instances x_i and x_j as: $d(x_i,x_j)$ be symmetric and obtains its minimum value (usually zero) in case of identical vectors. The distance measure is called a metric distance measure if it also satisfies the following properties:

1. Triangle inequality $d(x_i,x_j) = 0 \Rightarrow x_i = x_j \forall x_i$
2. Minkowski: Distance Measures for Numeric Attributes

Given two p-dimensional instances, $x_i = (x_{i1},x_{i2},..., x_{ip})$ and $x_j = (x_{j1}, x_{j2},..., x_{jp})$, the distance between the two data instances can be calculated using the Minkowski metric (Han and Kamber, 2001):

$$d(x_i,x_j) = (\sum_{k=1}^{p} |x_{ik} - y_{jk}|^p)^{1/p}$$

For $p \geq 1$, the Minkowski distance is a metric as a result of the Minkowski inequality. When $p < 1$, the distance between (0,0) and (1,1) is $2^{1/p} > 2$, but the point (0,1) is at a distance 1 from both of these points. Since this violates the triangle inequality, for $p < 1$ it is not a metric.

- *Similarity Functions*

An alternative concept to that of the distance is the similarity function s(xi, xj) that

compares the two vectors xi and xj (Duda et al., 2001). This function should be symmetrically large value when xi and xj are somehow "similar" and constitute the largest value for identical vectors. A similarity function where the target range is [0,1] is called a dichotomous similarity function. In fact, the methods described in the previous sections for calculating the "distances" in the case of binary and nominal attributes may be considered as similarity functions, rather than distances.

- *Evaluation Criteria Measures*

Evaluating if a certain clustering is good or not is a problematic and controversial issue. In fact Bonner (1964) was the first to argue that there is no universal definition for what is a good clustering. The evaluation remains mostly in the eye of the beholder. Nevertheless, several evaluation criteria have been developed in the literature. These criteria are usually divided into two categories:

➔ *Internal Quality Criteria*

Internal quality metrics usually measure the compactness of the clusters using some homogeneity, the inter-cluster separability or a combination of these two. It does not useany external information beside the data itself.

➔ *External Quality Criteria*

External measures can be useful for examining whether the structure of the clusters match to some predefined classification of the instances.
Precision-Recall Measure . The precision-recall measure from information retrieval can be used as an external measure for evaluating clusters.

The cluster is viewed as the results of a query for a specific class. Precision is the fraction of correctly retrieved instances, while recall is the fraction of correctly retrieved instances out of all matching instances. A combined F-measure can be useful for evaluating a clustering structure (Larsen and Aone,1999).

Rand Index: The Rand index (Rand, 1971) is a simple criterion used to compare an induced clustering structure (C1) with a given clustering structure (C2). Let a be the number of pairs of instances that are assigned to the same cluster in C1 and in the same cluster in C2; b be the number of pairs of instances that are in the same cluster in C1, but not in the same cluster in C2; c be the number of pairs of instances that are in the same cluster in C2, but not in the same cluster in C1; and d be the number of pairs of instances that are assigned to different clusters in C1 and C2. The quantities a and d can be interpreted as agreements, and b and c as disagreements. The Rand index is defined as:

$$RAND = \frac{a + d}{a + b + c + d}$$

The Rand index lies between 0 and 1. When the two partitions agree perfectly, the Rand index is 1.

A problem with the Rand index is that its expected value of two random clustering does not take a constant value (such as zero). Hubert and Arabie (1985) suggest an adjusted Rand index that overcomes this disadvantage.

➔ *Density-based Methods*

Density-based methods assume that the points that belong to each cluster are drawn from a specific probability distribution (Banfield and Raftery, 1993). The overall distribution of the data is assumed to be a mixture of several distributions.

The aim of these methods is to identify the clusters and their distribution parameters. These methods are designed for discovering clusters of arbitrary shape which are not necessarily convex, namely: $xi, xj \in Ck$
This does not necessarily imply that:

$$\alpha \cdot xi + (1 - \alpha) \cdot xj \in Ck$$

The idea is to continue growing the given cluster as long as the density (number of objects or data points) in the neighborhood exceeds some threshold. Namely, the neighborhood of a given radius has to contain at least a minimum number of objects. When each cluster is characterized by local mode or maxima of the density function, these methods are called mode-seeking. Much work in this field has been based on the underlying assumption that the component densities are multivariate Gaussian (in case of numeric data) or multi nominal (in case of nominal data).
An acceptable solution in this case is to use the maximum likelihood principle. According to this principle, one should choose the clustering structure and parameters such that the probability of the data being generated by such clustering structure and parameters is maximized. The expectation maximization algorithm — EM — (Dempster et al., 1977), which is a general-purpose maximum likelihood algorithm for missing-data problems, has been applied to the problem of parameter estimation. This algorithm begins with an initial estimate of the parameter vector and then alternates between two steps (Farley and Raftery, 1998): an "E-step", in which the conditional expectation of the complete data likelihood given the observed data and the current parameter estimates is computed, and an "M-step", in which parameters that maximize the expected likelihood from the E-step are determined. This algorithm was shown to converge to a local maximum of the observed data likelihood.

Assigning instances to clusters in the K-means may be considered as the E-step; computing new cluster centers may be regarded as the M-step. The DBSCAN algorithm (density-based spatial clustering of applications with noise) discovers clusters of arbitrary shapes and is efficient for large spatial databases. The algorithm searches for clusters by searching the neighborhood of each object in the database and checks if it contains more than the minimum number of objects (Ester et al., 1996).

AUTOCLASS is a widely-used algorithm that covers a broad variety of distributions, including Gaussian, Bernoulli, Poisson, and log-normal distributions (Cheeseman and Stutz, 1996). Other well-known density-based methods include: SNOB (Wallace and Dowe, 1994) and MCLUST (Farley and Raftery,1998). Density-based clustering may also employ nonparametric methods, such as searching for bins with large counts in a multidimensional histogram of the input instance space (Jain et al., 1999).

These methods attempt to optimize the fit between the given data and some mathematical models. Unlike conventional clustering, which identifies groups of objects, model-based clustering methods also find characteristic descriptions for each group, where each group represents a concept or class. The most frequently used induction methods are decision trees and neural networks.

➔ *Decision Trees*

In decision trees, the data is represented by a hierarchical tree, where each leaf refers to a concept and contains a probabilistic description of that concept. Several algorithms produce classification trees for representing the unlabelled data. The most well-known algorithms are:
COBWEB — This algorithm assumes that all attributes are independent (an often too naive assumption). Its aim is to achieve high predictability of nominal variable values, given a cluster. This algorithm is not suitable for clustering large database data (Fisher, 1987). CLASSIT, an extension of COBWEB for continuous-valued data, unfortunately has similar problems as the COBWEB algorithm.

➔ *Neural Networks*

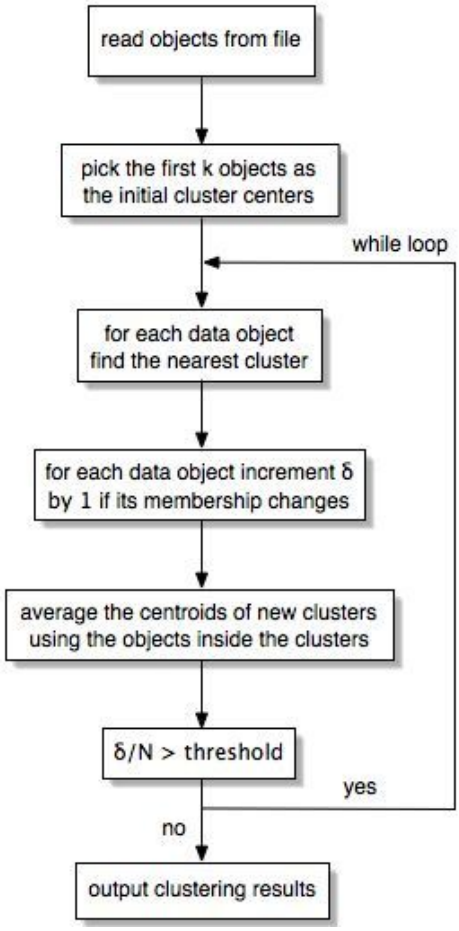
This type of algorithm represents each cluster by a neuron or "prototype". The input data is also represented by neurons, which are connected to the prototype neurons. Each such connection has a weight, which is learned adaptively during learning. A very popular neural algorithm for clustering is the self-organizing map (SOM). This algorithm constructs a single-layered network. The learning process takes place in a "winner-takes-all" fashion: The prototype neurons compete for the current instance. The winner is the neuron whose weight vector is closest to the instance currently presented. The winner and its neighbors learn by having their weights adjusted. The SOM algorithm is successfully used for vector quantization and speech recognition. It is useful for visualizing high-dimensional data in 2D or 3D space. However, it is sensitive to the initial selection of weight vector, as well as to its different parameters, such as the learning rate and neighborhood radius.

➔ *Grid-based Methods*

These methods partition the space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. The main advantage of the approach is its fast processing time (Han and Kamber, 2001).

➔ *Fuzzy Clustering*

Traditional clustering approaches generate partitions; in a partition, each instance belongs to one and only one cluster. Hence, the clusters in a hard clustering are disjointed. Fuzzy clustering extends this notion and suggests a soft clustering schema. In this case, each pattern is associated with every cluster using some sort of membership function, namely, each cluster is a fuzzy set of all the patterns. Larger membership values indicate higher confidence in the assignment of the pattern to the cluster. A hard clustering can be obtained from a fuzzy partition by using a threshold of the membership value. The most popular fuzzy clustering algorithm is the fuzzy c-means (FCM) algorithm. Even though it is better than the hard K-means algorithm at avoiding local minima, FCM can still converge to local minima of the squared error criterion. The design of membership functions is the most important problem in fuzzy clustering; different choices include those based on similarity decomposition and centroids of clusters. A generalization of the FCM algorithm has been proposed through a family of objective functions. A fuzzy c-shell algorithm and an adaptive variant for detecting circular and elliptical boundaries have been presented.



N: *number of data objects*
K: *number of clusters*

objects[N]: *array of data objects*
clusters[K]: *array of cluster centers*
membership[N]: *array of object memberships*

```
kmeans_clustering( )
1   while δ/N > threshold
2     δ ← 0
3     for i ← 0 to N-1
4       for j ← 0 to K-1
5         distance ← | objects[i] - clusters[j] |
6         if distance < dmin
7           dmin ← distance
8           n ← j
9       if membership[i] ≠ n
10        δ ← δ + 1
11        membership[i] ← n
12      new_clusters[n] ← new_clusters[n] + objects[i]
13      new_cluster_size[n] ← new_cluster_size[n] + 1
14    for j ← 0 to K-1
15      clusters[j][*] ← new_clusters[j][*] / new_cluster_size[j]
16      new_clusters[j][*] ← 0
17      new_cluster_size[j] ← 0
```

- *N-cut Clustering Algorithm*

N-cut method is a hierarchical divisive clustering process that represents the clusters in the tree structure form. This method organizes the nodes of tree into groups or cluster, such that the similarity between nodes in one cluster is high and similarity between the group is low. The output of this method results in more than two clusters, so this method is useful for the application where the data should be required to cluster into multiple clusters. In each step the subgraph with maximum number of nodes is further divided into clusters until stopping criterion has been reached [23] .

- *Mean Shift Clustering Algorithm*

The Mean Shift algorithm clusters the given data set by associating each point with a peak of the data set's probability density. For each point, Mean Shift computes its associated peak by defining a spherical window of radius 'r' at the data point and then compute the mean of the points that lie within the window. The algorithm then shifts the window to the mean and repeats until convergence, i.e., until the shift is less than the threshold. After every iteration the window will shift to a more densely populated portion of the data set until a peak is reached, where the data is equally distributed in the window [23] .

- *Evolutionary Approaches for Clustering*

Evolutionary techniques are stochastic general purpose methods for solving optimization problems. Since clustering problem can be defined as an optimization problem, evolutionary approaches may be appropriate here. The idea is to use evolutionary operators and a population of clustering structures to converge into a globally optimal clustering. Candidate clustering are encoded as chromosomes. The most commonly used evolutionary operators are: selection, recombination, and mutation. A fitness function evaluated on a chromosome determines a chromosome's likelihood of surviving into the next generation. The most frequently used evolutionary technique in clustering problems is genetic algorithms (GAs). A fitness value is associated with each cluster structure. A higher fitness value indicates a better cluster structure. A suitable fitness function is the inverse of the squared error value. Cluster structures with a small squared error will have a larger fitness value.

Input: S (instance set), K (number of clusters), n (population size)
Output: clusters

1: Randomly create a population of n structures, each corresponds to a valid K-clusters of the data.

2: repeat

3: Associate a fitness value $\forall$ structure $\in$ population.

4: Regenerate a new generation of structures.

5: until some termination condition is satisfied

The most obvious way to represent structures is to use strings of length m (where m is the number of instances in the given set). The i-th entry of the string denotes the cluster to which the i-th instance belongs. Consequently, each entry can have values from 1 to K. An improved representation scheme is proposed where an additional separator symbol is used along with the pattern labels to represent a partition. Using this representation permits them to map the clustering problem into a permutation problem such as the travelling salesman problem, which can be solved by using the permutation crossover operators. This solution also suffers from permutation redundancy.

In GAs, a selection operator propagates solutions from the current generation to the next generation based on their fitness. Selection employs a probabilistic scheme so that solutions with higher fitness have a higher probability of getting reproduced. There are a variety of recombination operators in use; crossover is the most popular. Crossover takes as input a pair of chromosomes (called parents) and outputs a new pair of chromosomes (called children or offspring). In this way the GS explores the search space. Mutation is used to make sure that the algorithm is not trapped in local optimum.

More recently investigated is the use of edge-based crossover to solve the clustering problem. Here, all patterns in a cluster are assumed to form a complete graph by connecting them with edges. Offspring are generated from the parents so that they inherit the edges from their parents. In a hybrid approach that has been proposed, the GAs is used only to find good initial cluster centers and the K-means algorithm is applied to find the final partition. This hybrid approach performed better than the Gas. A major problem with GAs is their sensitivity to the selection of various parameters such as population size, crossover and mutation probabilities, etc. Several researchers have studied this problem and suggested guidelines for selecting these control

parameters. However, these guidelines may not yield good results on specific problems like pattern clustering. It was reported that hybrid genetic algorithms incorporating problem-specific heuristics are good for clustering. A similar claim is made about the applicability of GAs to other practical problems. Another issue with GAs is the selection of an appropriate representation which is low in order and short in defining length. There are other evolutionary techniques such as evolution strategies (Ess), and evolutionary programming (EP). These techniques differ from the GAs in solution representation and the type of mutation operator used; EP does not use a recombination operator, but only selection and mutation. Each of these three approaches has been used to solve the clustering problem by viewing it as a minimization of the squared error criterion. Some of the theoretical issues, such as the convergence of these approaches, were studied. GAs perform a globalized search for solutions whereas most other clustering procedures perform a localized search. In a localized search, the solution obtained at the 'next iteration' of the procedure is in the vicinity of the current solution. In this sense, the K-means algorithm and fuzzy clustering algorithms are all localized search techniques. In the case of GAs, the crossover and mutation operators can produce new solutions that are completely different from the current ones. It is possible to search for the optimal location of the centroids rather than finding the optimal partition. This idea permits the use of ESs and EP, because centroids can be coded easily in both these approaches, as they support the direct representation of a solution as a real-valued vector. ESs were used on both hard and fuzzy clustering problems and EP has been used to evolve fuzzy min-max clusters. It has been observed that they perform better than their classical counterparts, the K-means algorithm and the fuzzy c-means algorithm. However, all of these approaches are over sensitive to their parameters. Consequently, for each specific problem, the user is required to tune the parameter values to suit the application.

- ***Simulated Annealing for Clustering.***

Another general-purpose stochastic search technique that can be used for clustering is simulated annealing (SA), which is a sequential stochastic search technique designed to avoid local optima. This is accomplished by accepting with some probability a new solution for the next iteration of lower quality (as measured by the criterion function). The probability of acceptance is governed by a critical parameter called the temperature (by analogy with annealing in metals), which is typically specified in terms of a starting (first iteration) and final temperature value. Selim and Al-Sultan (1991) studied the effects of control parameters on the performance of the algorithm. SA is statistically guaranteed to find the global optimal solution.

The SA algorithm can be slow in reaching the optimal solution, because optimal results require the temperature to be decreased very slowly from iteration to iteration. Tabu search, like SA, is a method designed to cross boundaries of feasibility or local optimality and to systematically impose and release constraints to permit exploration of otherwise forbidden regions. Al-Sultan (1995) suggests using Tabu search as an alternative to SA.

Clustering Large Data Sets

There are several applications where it is necessary to cluster a large collection of patterns. The definition of 'large' is vague. In document retrieval, millions of instances with a dimensionality of more than 100 have to be clustered to achieve data abstraction. A majority of the approaches and algorithms proposed in the literature cannot handle such large data sets. Approaches based on genetic algorithms, tabu search and simulated annealing are optimization techniques and are restricted to reasonably small data sets. Implementations of conceptual clustering optimize some criterion functions and are typically computationally expensive.
The convergent K-means algorithm and its ANN equivalent, the Kohonen net, have been used to

cluster large data sets. The reasons behind the popularity of the K-means algorithm are:

1. Its time complexity is O(mkl), where m is the number of instances; k is the number of clusters; and l is the number of iterations taken by the algorithm to converge. Typically, k and l are fixed in advance and so the algorithm has linear time complexity in the size of the data set.

2. Its space complexity is O(k+m). It requires additional space to store the data matrix. It is possible to store the data matrix in a secondary memory and access each pattern based on need. However, this scheme requires a huge access time because of the iterative nature of the algorithm. As a consequence, processing time increases enormously.

3. It is order-independent. For a given initial seed set of cluster centers, it generates the same partition of the data irrespective of the order in which the patterns are presented to the algorithm. However, the K-means algorithm is sensitive to initial seed selection and even in the best case, it can produce only hyperspherical clusters. Hierarchical algorithms are more versatile. But they have the following disadvantages:

1. The time complexity of hierarchical agglomerative algorithms is O(m2 ∗log m).

2. The space complexity of agglomerative algorithms is O(m2 because a similarity matrix of size m2 has to be stored. It is possible to compute the entries of this matrix based on need instead of storing them.

A possible solution to the problem of clustering large data sets while only marginally sacrificing the versatility of clusters is to implement more efficient variants of clustering algorithms. A hybrid approach was used, where a set of reference points is chosen as in the K-means algorithm, and each of the remaining data points is assigned to one or more reference points or clusters. Minimal spanning trees (MST) are separately obtained for each group of points. These MSTs are merged to form an approximate global MST. This approach computes only similarities between a fraction of all possible pairs of points. It was shown that the number of similarities computed for 10,000 instances using this approach is the same as the total number of pairs of points in a collection of 2,000 points. Bentley and Friedman (1978) presents an algorithm that can compute an approximate MST in O(m log m) time. A scheme to generate an approximate dendrogram incrementally in O(n log n) time was presented.

CLARANS (Clustering Large Applications based on RANdom Search) have been developed by Ng and Han (1994). This method identifies candidate cluster centroids by using repeated random samples of the original data. Because of the use of random sampling, the time complexity is O(n) for a pattern set of n elements.

The BIRCH algorithm (Balanced Iterative Reducing and Clustering) stores summary information about candidate clusters in a dynamic tree data structure. This tree hierarchically organizes the clusters represented at the leaf nodes. The tree can be rebuilt when a threshold specifying cluster size is updated manually, or when memory constraints force a change in this threshold. This algorithm has a time complexity linear in the number of instances. All algorithms presented till this point assume that the entire dataset can be accommodated in the main memory. However, there are cases in which this assumption is untrue. The following sub-sections describe three current approaches to solve this problem.

- ***Decomposition Approach:***

The dataset can be stored in a secondary memory (i.e. hard disk) and subsets of this data clustered independently, followed by a merging step to yield a clustering of the entire dataset. Initially, the data is decomposed into number of subsets. Each subset is sent to the main memory in turn where it is clustered into k clusters using a standard algorithm.
In order to join the various clustering structures obtained from each subset, a representative sample from each cluster of each structure is stored in the main memory. Then these representative instances are further clustered into k-clusters and the cluster labels of these representative instances are used to relabel the original dataset. It is possible to extend this algorithm to any number of iterations; more levels are required if the data set is very large and the main memory size is very small.

- ***Incremental Clustering***

Incremental clustering is based on the assumption that it is possible to consider instances one at a time and assign them to existing clusters. Here, a new instance is assigned to a cluster without significantly affecting the existing clusters. Only the cluster representations are stored in the main memory to alleviate the space limitations.

A high level pseudo-code of a typical incremental clustering algorithm.
Input: S (instances set), K (number of clusters), T threshold (for assigning an instance to a cluster)
Output: clusters

The major advantage with incremental clustering algorithms is that it is not necessary to store the entire dataset in the memory. Therefore, the space and time requirements of incremental algorithms are very small. There are several incremental clustering algorithms:

1. The leading clustering algorithm is the simplest in terms of time complexity which is O(mk). It has gained popularity because of its neural network implementation, the ART network, and is very easy to implement as it requires only O(k) space.

2. The shortest spanning path (SSP) algorithm, as originally proposed for data reorganization, was successfully used in automatic auditing of records. Here, the SSP algorithm was used to cluster 2000 patterns using 18 features. These clusters are used to estimate missing feature values in data items and to identify erroneous feature values.

3. The COBWEB system is an incremental conceptual clustering algorithm. It has been successfully used in engineering applications.

4. An incremental clustering algorithm for dynamic information processing was presented in (Can, 1993). The motivation behind this work is that in dynamic databases items might get added and deleted over time.

These changes should be reflected in the partition generated without significantly affecting the current clusters. This algorithm was used to cluster incrementally an INSPEC database of 12,684 documents relating to computer science and electrical engineering. Order-independence is an important property of clustering algorithms. An algorithm is order-independent if it generates the same partition for any order in which the data is presented, otherwise, it is order-dependent. Most of the incremental algorithms presented above are order-dependent. For instance the SSP algorithm and cobweb are order-dependent.

- ***Parallel Implementation***

Recent work demonstrates that a combination of algorithmic enhancements to a clustering algorithm and distribution of the computations over a network of workstations can allow a large dataset to be clustered in a few minutes. Depending on the clustering algorithm in use, parallelization of the code and replication of data for efficiency may yield large benefits. However, a global shared data structure, namely the cluster membership table, remains and must be managed centrally or replicated and synchronized periodically. The presence or absence of robust, efficient parallel clustering techniques will determine the success or failure of cluster analysis in large-scale data mining applications in the future.

- ***Determining the Number of Clusters***

As mentioned above, many clustering algorithms require that the number of clusters will be pre-set by the user. It is well-known that this parameter affects the performance of the algorithm significantly. This poses a serious question as to which K should be chosen when prior knowledge regarding the cluster quantity is unavailable. The most of the criteria that have been used to lead the construction of the clusters (such as SSE) are monotonically decreasing in K. Therefore using these criteria for determining the number of clusters results with a trivial clustering, in which each cluster contains one instance. Consequently, different criteria must be applied here. Many methods have been presented to determine which K is preferable. These methods are usually heuristics, involving the calculation of clustering criteria measures for different values of K, thus making it possible to evaluate which K was preferable.

➔ ***Methods Based on Intra-Cluster Scatter***

In general, as the number of clusters increases, the within-cluster decay first declines rapidly. From a certain K, the curve flattens. This value is considered the appropriate K according to this method. Other heuristics relate to the intra-cluster distance as the sum of squared Euclidean distances between the data instances and their cluster centers (the sum of square errors which the algorithm attempts to minimize). They range from simple methods, such as the PRE method, to more sophisticated, statistic based methods.
An example of a simple method which works well in most databases is, as mentioned above, the proportional reduction in error (PRE) method. PRE is the ratio of reduction in the sum of squares to the previous sum of squares when comparing the results of using K + 1 clusters to the results of using K clusters. Increasing the number of clusters by 1 is justified for PRE rates of about 0.4 or larger.

It is also possible to examine the SSE decay, which behaves similarly to the within cluster depression described above. The manner of determining K according to both measures is also similar.

An approximate F statistic can be used to test the significance of the reduction in the sum of squares as we increase the number of clusters (Hartigan, 1975). The method obtains this F statistic as follows:

Suppose that P(m, k) is the partition of m instances into k clusters, and P(m, k + 1) is obtained from P(m, k) by splitting one of the clusters. Also pendently over all q and i. Then the overall mean square ratio is calculated and assume that the clusters are selected without regard to xqi ~ N(μi distributed as follows:)

R = where e(P(m, k)) is the sum of squared Euclidean distances between the data instances and their

cluster centers. In fact this F distribution is inaccurate since it is based on inaccurate assumptions:

K-means is not a hierarchical clustering algorithm, but a relocation method. Therefore, the partition P(m, k + 1) is not necessarily obtained by splitting one of the clusters in P(m, k).Each xqi influences the partition. The assumptions as to the normal distribution and independence of xqi are not valid in all databases. Since the F statistic described above is imprecise, Hartigan offers a crude rule of thumb: only large values of the ratio (say, larger than 10) justify increasing the number of partitions from K to K + 1.

➔ ***Methods Based on both the Inter- and Intra-Cluster Scatter***

All the methods described so far for estimating the number of clusters are quite reasonable. However, they all suffer the same deficiency: None of these methods examines the inter-cluster distances. Thus, if the K-means algorithm partitions an existing distinct cluster in the data into sub-clusters (which is undesired), it is possible that none of the above methods would indicate this situation. In light of this observation, it may be preferable to minimize the intra-cluster scatter and at the same time maximize the inter-cluster scatter. Ray and Turi (1999), for example, strive for this goal by setting a measure that equals the ratio of intra-cluster scatter and inter-cluster scatter. Minimizing this measure is equivalent to both minimizing the intra-cluster scatter and maximizing the inter-cluster scatter.

Another method for evaluating the "optimal" K using both inter and intra cluster scatter is the validity index method (Kim et al., 2001). There are two appropriate measures:
MICD — mean intra-cluster distance; defined for the k

ICMD — inter-cluster minimum distance; defined as: dmin = min

In order to create cluster validity index, the behavior of these two measures around the real number of clusters tains large MICD. As the partition state moves towards over-partitioned (K > When the data are under-partitioned, the large MICD abruptly decreases.

The ICMD is large when the data are under-partitioned or optimally partitioned. It becomes very small when the data enters the over-partitioned state, since at least one of the compact clusters is subdivided.

Two additional measure functions may be defined in order to find the under partitioned and over-partitioned states. These functions depend, among other variables, on the vector of the clusters centers $\mu = [\mu 1, \mu 2, \ldots \mu K]$

The validity index uses the fact that both functions have small values only at
$K = K*$. The vectors of both partition functions are defined as following:
$Vu = [vu(2, \mu; X), \ldots, vu(Kmax, \mu; X)]$
$Vo = [vo(2, \mu), \ldots, vo(Kmax, \mu)]$

Before finding the validity index, each element in each vector is normalized to the range [0,1], according to its minimum and maximum values. For instance, for the Vu vector: $(K, \mu; X) = vu(K, \mu; X)$

The process of normalization is done the same way for the Vo vector. The validity index vector is calculated as the sum of the two normalized vectors:
max

K=2,...,Kmax
{vu(K, μ; X)} − min
vsv(K, μ; X) = v*(K, μ; X) + vu
Since both partition measure functions have small values only at K = K∗ smallest value of vsv is chosen as the optimal number of clusters.

➔ ***Criteria Based on Probabilistic***

When clustering is performed using a density-based method, the determination of the most suitable number of clusters K becomes a more tractable task as clear probabilistic foundation can be used. The question is whether adding new parameters results in a better way of fitting the data by the model. In Bayesian theory, the likelihood of a model is also affected by the number of parameters which are proportional to K. Suitable criteria that can used here include BIC (Bayesian Information Criterion), MML (Minimum Message Length) and MDL (Minimum Description Length). Classification, which is the task of assigning objects to one of several predefined categories, is a pervasive problem that encompasses many diverse applications. Examples include detecting spam email messages based upon the message header and content, categorizing cells as malignant or benign based upon the results of MRI scans, and classifying galaxies based upon their shapes.
(a) A spiral galaxy. (b) An elliptical galaxy.

- **Classification model**

Preliminaries

The input data for a classification task is a collection of records. Each record, also known as an instance or example, is characterized by a tuple (x, y), where x is the attribute set and y is a special attribute, designated as the class label (also known as category or target attribute). Table 4.1 shows a sample data set used for classifying vertebrates into one of the following categories: mammal, bird, fish, reptile, or amphibian. The attribute set includes properties of a vertebrate such as its body temperature, skin cover, method of reproduction, ability to fly, and ability to live in water. The class label, on the other hand, must be a discrete attribute. This is a key characteristic that distinguishes classification from regression, a predictive modeling task in which y is a continuous attribute.

Classification is the task of learning a target function f that maps each attribute set x to one of the predefined class labels y.

The target function is also known informally as a classification model. A classification model is useful for the following purposes. Descriptive Modeling A classification model can serve as an explanatory tool to distinguish between objects of different classes. For example, it would be useful—for both biologists and others—to have a descriptive model that

General Approach to Solving a Classification Problem

A classification technique (or classifier) is a systematic approach to building classification models from an input data set. Examples include decision tree classifiers, rule-based classifiers, neural networks, support vector machines, and na ïve Bayes classifiers. Each technique employs a learning algorithm to identify a model that best fits the relationship between the attribute set and class label of the input data. The model generated by a learning algorithm should both fit the input data well and correctly predict the class labels of records it has never seen before. Therefore, a key

objective of the learning algorithm is to build models with good generalization capability; i.e., models that accurately predict the class labels of previously unknown records.
Evaluation of the performance of a classification model is based on the counts of test records correctly and incorrectly predicted by the model. These counts are tabulated in a table known as a confusion matrix. The confusion matrix for a binary classification problem. Each entry fij in this table denotes the number of records from class i predicted to be of class j. For instance, f01 is the number of records from class 0 incorrectly predicted as class 1. Based on the entries in the confusion matrix, the total number of correct predictions made by the model is (f11 + f00) and the total number of incorrect predictions is (f10 + f01). Although a confusion matrix provides the information needed to determine how well a classification model performs, summarizing this information with a single number would make it more convenient to compare the performance of different models. This can be done using a performance metric such as accuracy, which is defined as follows:

Accuracy = Number of correct predictions

Equivalently, the performance of a model can be expressed in terms of its error rate, which is given by the following equation: Error rate = Number of wrong predictions Most classification algorithms seek models that attain the highest accuracy, or equivalently, the lowest error rate when applied to the test set.

Total number of predictions = f11 + f00
f11 + f10 + f01 + f00
Total number of predictions = f10 + f01
f11 + f10 + f01 + f00

Decision Tree Induction

This section introduces a decision tree classifier, which is a simple yet widely used classification technique.

How a Decision Tree Works

To illustrate how classification with a decision tree works, consider a simpler version of the vertebrate classification problem described in the previous section. Instead of classifying the vertebrates into five distinct groups of species, we assign them to two categories: mammals and non-mammals. Suppose a new species is discovered by scientists. How can we tell whether it is a mammal or a non-mammal? One approach is to pose a series of questions about the characteristics of the species. The first question we may ask is whether the species is cold- or warm-blooded. If it is cold-blooded, then it is definitely not a mammal. Otherwise, it is either a bird or a mammal. In the latter case, we need to ask a follow-up question: Do the females of the species give birth to their young? Those that do give birth are definitely mammals, while those that do not are likely to be non-mammals (with the exception of egg-laying mammals such as the platypus and spiny anteater).
The previous example illustrates how we can solve a classification problem by asking a series of carefully crafted questions about the attributes of the test record. Each time we receive an answer, a follow-up question is asked until we reach a conclusion about the class label of the record. The series of questions and their possible answers can be organized in the form of a decision tree, which is a hierarchical structure consisting of nodes and directed edges.

The tree has three types of nodes:
• A root node that has no incoming edges and zero or more outgoing edges.

• Internal nodes, each of which has exactly one incoming edge and two or more outgoing edges.
• Leaf or terminal nodes, each of which has exactly one incoming edge and no outgoing edges.
In a decision tree, each leaf node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics.

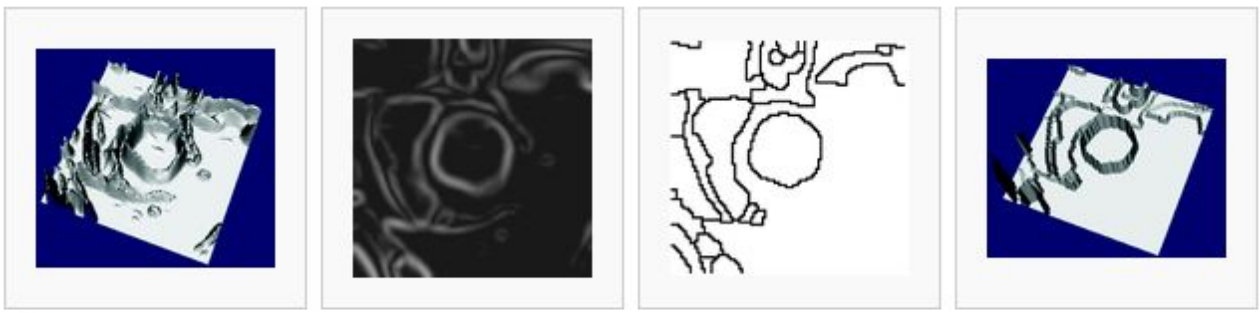
How to Build a Decision Tree

In principle, there are exponentially many decision trees that can be constructed from a given set of attributes. While some of the trees are more accurate than others, finding the optimal tree is computationally infeasible because of the exponential size of the search space. Nevertheless, efficient algorithms have been developed to induce a reasonably accurate, albeit suboptimal, decision tree in a reasonable amount of time. These algorithms usually employ a greedy strategy that grows a decision tree by making a series of locally optimum decisions about which attribute to use for partitioning the data. One such algorithm is Hunt's algorithm, which is the basis of many existing decision tree induction algorithms, including ID3, C4.5, and CART. This section presents a high-level discussion of Hunt's algorithm and illustrates some of its design issues.

2.2.2.2 Watershed Segmentation Algorithm

In the study of image processing a watershed of a grayscale image is analogous to the notion of a catchment basin of a heightmap. In short, a drop of water following the gradient of an image flows along a path to finally reach a local minimum. Intuitively, the watershed of a relief correspond to the limits of the adjacent catchment basins of the drops of water.

There are different technical definitions of a watershed. In graphs, watershed lines may be defined on the nodes, on the edges, or hybrid lines on both nodes and edges. Watersheds may also be defined in the continuous domain.[28] There are also many different algorithms to compute watersheds. Watershed algorithm is used in image processing primarily for segmentation purposes.



Relief of the gradient magnitude | Gradient magnitude image | Watershed of the gradient | Watershed of the gradient (relief)

The watershed transform can be classified as a region-based segmentation approach. The intuitive idea underlying this method comes from geography: it is that of a landscape or topographic relief which is flooded by water, watersheds being the divide lines of the domains of attraction of rain falling over the region [29]. An alternative approach is to imagine the landscape being

immersed in a lake, with holes pierced in local minima. Basins (also called 'catchment basins') will fill up with water starting at these local minima, and, at points where water coming from different basins would meet, dams are built. When the water level has reached the highest peak in the landscape, the process is stopped. As a result, the landscape is partitioned into regions or basins separated by dams, called watershed lines or simply watersheds.

When simulating this process for image segmentation, two approaches may be used: either one first finds basins, watersheds by taking a set complement; or one computes a complete partition of the image into basins, and subsequently finds the watersheds by boundary detection. To be more explicit, we will use the expression 'watershed transform' to denote a labelling of the image, such that all points of a given catchment basin have the same unique label, and a special label, distinct from all the labels of the catchment basins, is assigned to all points of the watersheds. An example of a simple image with its watershed transform is given in Fig.1.1(a-b). We note in passing that in practice one often does not apply the watershed transform to the original image, but to its (morphological) gradient [30]. This produces watersheds at the points of grey value discontinuity, as is commonly desired in image segmentation.

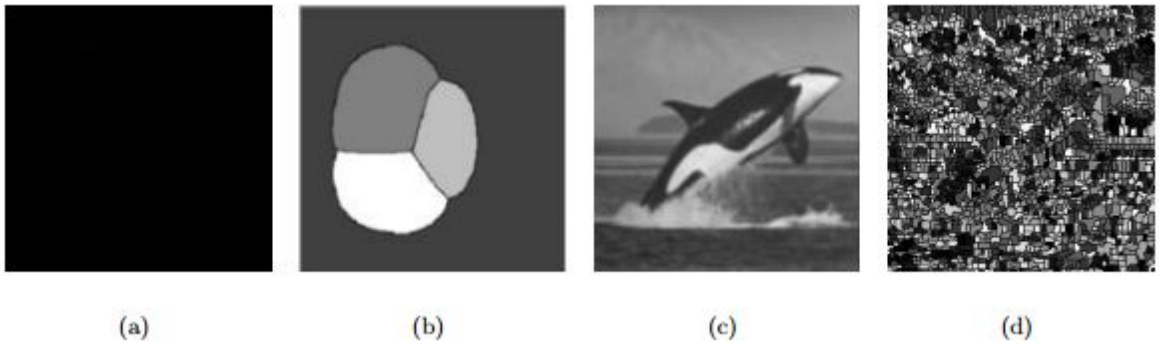


Fig.1.1: Examples of watershed segmentation by immersion. (a): synthetic image; (b): watershed transform of (a); (c): natural image; (d): watershed transform of (c). Different basins are indicated by distinct grey values.

One of the difficulties with this intuitive concept is that it leaves room for various formalizations. Different watershed definitions for continuous functions have been given later. However, our main interest here is in digital images, for which there is even more freedom to define watersheds, since in the discrete case there is no unique definition of the path a drop of water would follow. Many sequential algorithms have been developed to compute watershed transforms, see e.g. [30,31] for a survey. They can be divided into two classes, one based on the specification of a recursive algorithm by Vincent & Soille [32], and another based on distance functions by Meyer [33]. In the context of parallel implementations there exists a notable tendency for introducing other definitions of the watershed transform, enabling easier parallelization.

Definitions

A watershed is a basin-like landform defined by highpoints and ridgelines that descend into lower elevations and stream valleys.

Watershed by flooding

The idea was introduced in 1979 by S. Beucher and C. Lantuéjoul.[34] The basic idea consisted of placing a water source in each regional minimum in the relief, to flood the entire relief from sources, and build barriers when different water sources meet. The resulting set of barriers constitutes a watershed by flooding.

Watershed by topographic distance

Intuitively, a drop of water falling on a topographic relief flows towards the "nearest" minimum. The "nearest" minimum is that minimum which lies at the end of the path of steepest descent. In terms of topography, this occurs if the point lies in the catchment basin of that minimum. The previous definition does not verify this condition.

Watershed by the drop of water principle

Intuitively, the watershed is a separation of the regional minima from which a drop of water can flow down towards distinct minima. A formalization of this intuitive idea was provided in [35] for defining a watershed of an edge-weighted graph.

Inter-pixel watershed

S. Beucher and F. Meyer introduced an algorithmic inter-pixel implementation of the watershed method,[36] given the following procedure:

1. Label each minimum with a distinct label. Initialize a set S with the labeled nodes.
2. Extract from S a node x of minimal altitude F, that is to say $F(x) = \min\{F(y) | y \in S\}$. Attribute the label of x to each unlabeled node y adjacent to x, and insert y in S.
3. Repeat Step 2 until S is empty.

Topological watershed

Previous notions focus on catchment basins, but not to the produced separating line. The topological watershed was introduced by M. Couprie and G. Bertrand in 1997,[37] and beneficiate of the following fundamental property. A function W is a watershed of a function F *if and only if* W $\leq$ F and W preserves the contrast between the regional minima of F; where the contrast between two regional minima M_1 and M_2 is defined as the minimal altitude to which one must climb in order to go from M_1 to M_2.[38]

Algorithms

Different approaches may be employed to use the watershed principle for image segmentation.

- Local minima of the gradient of the image may be chosen as markers, in this case an over-segmentation is produced and a second step involves region merging.
- Marker based watershed transformation make use of specific marker positions which have been either explicitly defined by the user or determined automatically with morphological operators or other ways.

Meyer's flooding algorithm

One of the most common watershed algorithms was introduced by F. Meyer in the early 90's. The algorithm works on a gray scale image. During the successive flooding of the grey value relief, watersheds with adjacent catchment basins are constructed. This flooding process is performed on the gradient image, i.e. the basins should emerge along the edges. Normally this will lead to an over-segmentation of the image, especially for noisy image material, e.g. medical CT data. Either the image must be pre-processed or the regions must be merged on the basis of a similarity criterion afterwards.

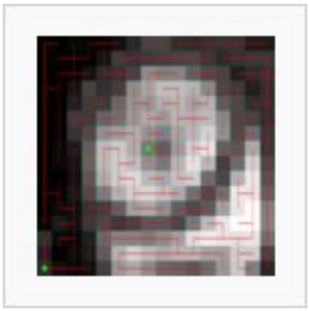
1. A set of markers, pixels where the flooding shall start, are chosen. Each is given a different label.
2. The neighboring pixels of each marked area are inserted into a priority queue with a priority level corresponding to the gradient magnitude of the pixel.
3. The pixel with the lowest priority level is extracted from the priority queue. If the neighbors of the extracted pixel that have already been labeled all have the same label, then the pixel is labeled with their label. All non-marked neighbors that are not yet in the priority queue are put into the priority queue.

Redo step 3 until the priority queue is empty.

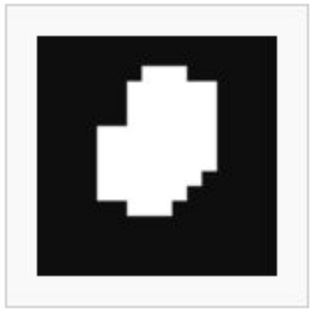
The non-labeled pixels are the watershed lines.

Optimal spanning forest algorithms (watershed cuts)

Watersheds as optimal spanning forest have been introduced by Jean Cousty et al[39]. They establish the consistency of these watersheds: they can be equivalently defined by their “catchment basins” (through a steepest descent property) or by the “dividing lines” separating these catchment basins (through the drop of water principle). Then they prove, through an equivalence theorem, their optimality in terms of minimum spanning forests. Afterward, they introduce a linear-time algorithm to compute them. It is worthwhile to note that similar properties are not verified in other frameworks and the proposed algorithm is the most efficient existing algorithm.



An image with two markers (green), and a Minimum Spanning Forest computed on the gradient of the image.



Result of the segmentation by Minimum Spanning Forest

2.3. Morphological Operation

Morphological image processing is a tool for extracting or modifying information on the shape and structure of objects within an image. Morphological operators, such as dilation, erosion and skeletonization, are particularly useful for the analysis of binary images, although they can be extended for use with grayscale images. Morphological operators are nonlinear, and common usages include filtering, edge detection, feature detection, counting objects in an image, image segmentation, noise reduction, and finding the midline of an object.

2.3.1 Mathematical Morphology

The field of mathematical morphology contributes a wide range of operators to image processing, all based around a few simple mathematical concepts from set theory and, in the case of binary images, (Boolean) logic operations such as “AND”, “OR”, “XOR” (exclusive OR) and “NOT”. The “union” operation, AB, for example, is equivalent to the “OR” operation for binary images; and the “intersection” operator, AB, is equivalent to the “AND” operation for binary images.

2.3.1.a Connectivity

In binary images an object is defined as a connected set of pixels. With two-dimensional images connectivity can be either 4-connectivity or 8-connectivity (Fig. 1). In 4-connectivity, each pixel (P) has four connected neighbors (N) – top, bottom, right and left. The diagonally touching pixels are not considered to be connected. In 8-connectivity, each pixel (P) has eight connected neighbors (N) – including the diagonally touching pixels. For three-dimensional images neighborhoods can be 6-connected, 18-connected or 26-connected.

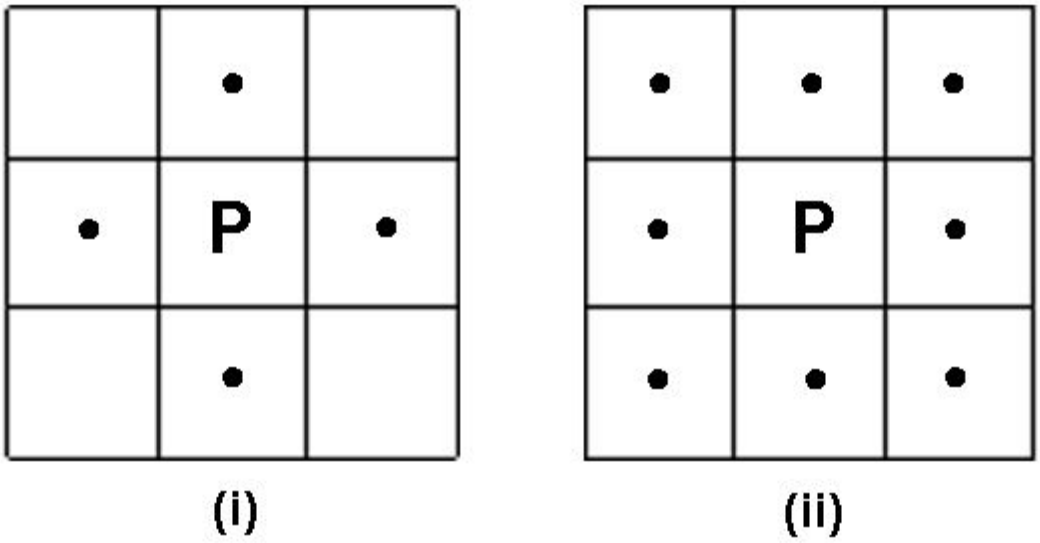


Figure 1 Connectivity in two-dimensional images. (i) 4-connectivity - each pixel (P) has four connected neighbors (●) (ii) 8-connectivity - each pixel (P) has eight connected neighbors (●).

This leads to different ideas of distance. In a 4-connected neighborhood, N4, the distance is known as the city-block, taxicab or Manhattan distance by analogy with a city based on an orthogonal grid of roads. It is the distance a taxicab would drive in Manhattan (if there were no one way streets!). The distance in a 4-connected neighborhood is given by

$$d_4(x,y) = |x_1 - y_1| + |x_2 - y_2| \qquad (1)$$

A diagonal step has a distance of two since it requires a horizontal and a vertical step. Equal distances from a certain position would form diamonds centered on it. In an 8-connected neighborhood, N8, the distance is known as the Chebyshev or chessboard distance, by analogy with the moves available to a king in chess. The distance in an 8-connected neighborhood is given by

$$d_8(x,y) = max\{|x_1 - y_1|, |x_2 - y_2|\} \qquad (2)$$

A diagonal step has a distance of one, the same as a horizontal or vertical step. Equal distances from a certain position would form squares centered on it. Neither is the same as Euclidean distance, which is given by

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \qquad (3)$$

A diagonal step is given by a distance of $1/\sqrt{2}$, and equal distances from a certain position

form circles centered on it. In physical space the Euclidean distance is the most natural distance, because the length of a rigid body does not change with rotation. Alternating the two metrics (N4-N8 or N8-N4) is an approximation to Euclidean distance.

2.3.2 Morphological Operators

There are a number of morphological operators, but two most fundamental operations are dilation and erosion; all other morphological operations are built from a combination of these two.

2.3.2.a Dilation and Erosion

In binary images dilation is an operation that increases the size of foreground objects, generally taken as white pixels although in some implementations this convention is reversed. It can be defined in terms of set theory, although we will use a more intuitive algorithm. The connectivity needs to be established prior to operation, or a structuring element defined (Fig. 2).

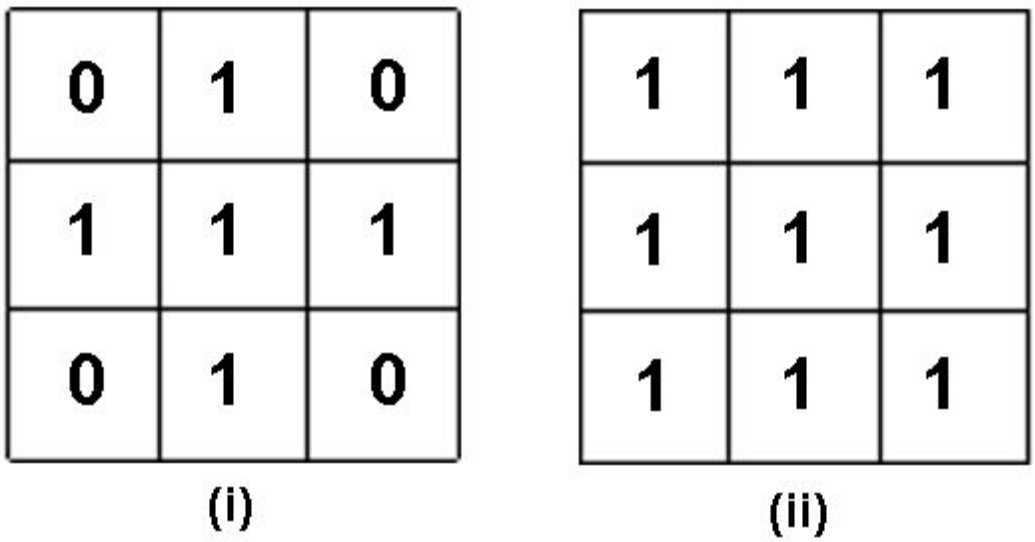


Fig. 2: Structuring elements corresponding to (i) 4-connectivity (ii) 8-connectivity.

The algorithm is as follows: superimpose the structuring element on top of each pixel of the input image so that the center of the structuring element coincides with the input pixel position. If at least one pixel in the structuring element coincides with a foreground pixel in the image underneath, including the pixel being tested, then set the output pixel in a new image to the foreground value. Thus, some of the background pixels in the input image become foreground pixels in the output image; those that were foreground pixels in the input image remain foreground pixels in the output image. In the case of 8-connectivity, if a background pixel has at least one foreground (white) neighbor then it becomes white; otherwise, it remains unchanged. The pixels which change from background to foreground are pixels which lie at the edges of foreground regions in the input image, so the consequence is that foreground regions grow in size, and foreground features tend to connect or merge (Fig. 3). Background features or holes inside a foreground region shrink due to

the growth of the foreground, and sharp corners are smoothed (Fig. 4). Repeated dilation results in further growth of the foreground regions (Fig. 5). The pattern of growth depends on the structuring element used, as can be seen in activity 1.

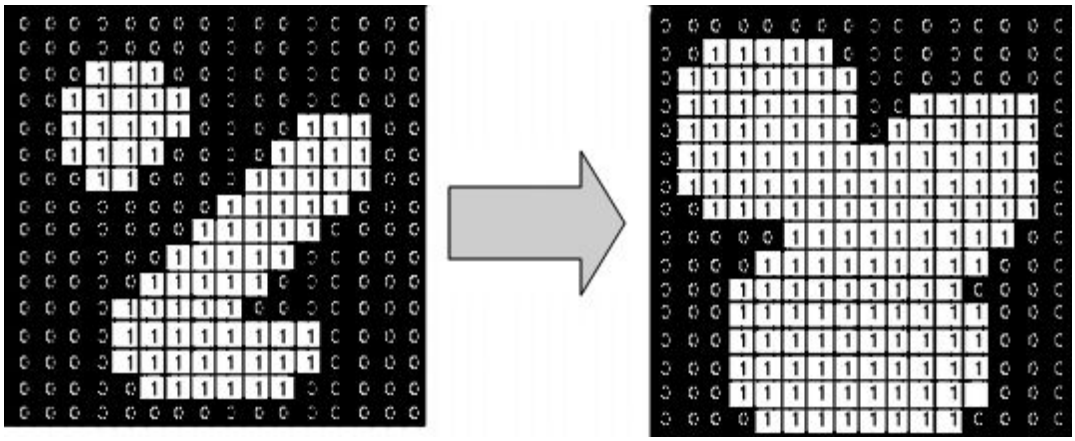


Fig. 3: The effect of dilation in connecting foreground features, using a structuring element corresponding to 8-connectivity

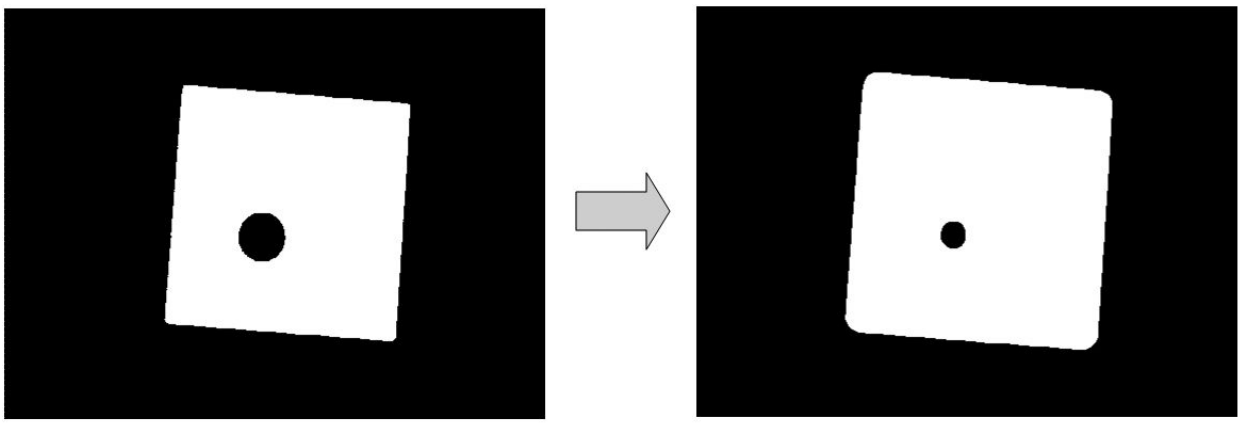
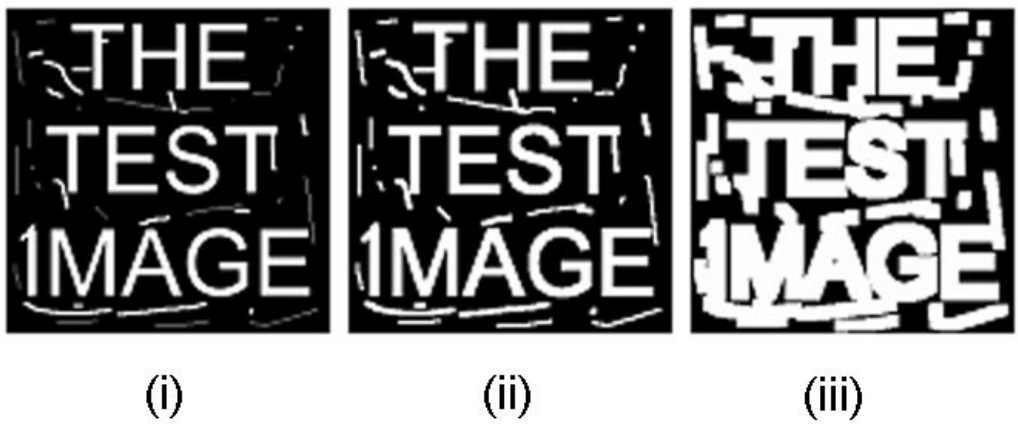


Fig. 4: The effect of repeated dilation in shrinking background features and smoothing sharp corners, using a structuring element corresponding to 8-connectivity



.

Fig. 5: (i) Original image (ii) after a single dilation (iii) after several dilations.

The structuring element can be considered analogous to a convolution mask, and the dilation process analogous to convolution although dilation is based on set operations whereas convolution is based on arithmetic operations. After being reflected about its own origin, it slides over an image pushing out the boundaries of the image where it overlaps with the image by at least one element. This growing effect is similar to the smearing or blurring effect of an averaging mask. One of the basic applications of dilation is to bridge gaps and connect objects. Dilation with a 3x3 structuring element is able to bridge gaps of up to two pixels in length.

Dilation can be used to create the outline of features in an image (Fig. 6). If a binarized image is dilated once, and the original image subtracted pixel-by-pixel from the dilated image, the result is a one-pixel wide outline of the features in the original image. This operation tends to be more robust than most edge enhancement operations in the presence of image noise. The outline can be used in subsequent feature extraction operations to measure size, shape and orientation, for example, and these derived measurements can be used in feature classification .

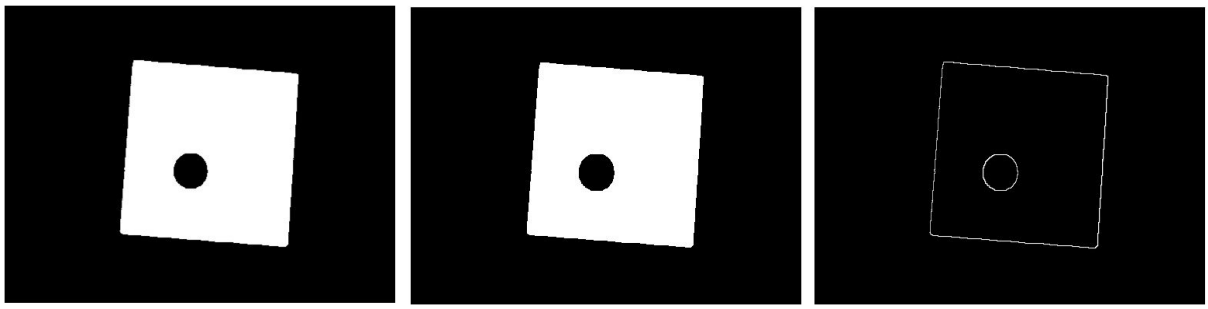


Fig. 6: Outlining features in an image. (i) Original image, (ii) image dilated (once), (ii) result of subtracting image (i) from image (ii).

In contradistinction erosion is an operation that increases the size of background objects (and shrinks the foreground objects) in binary images. In this case the structuring element is superimposed on each pixel of the input image, and if at least one pixel in the structuring element coincides with a background pixel in the image underneath, then the output pixel is set to the background value. Thus, some of the foreground pixels in the input image become background pixels in the output image; those that were background pixels in the input image remain background pixels in the output image. In the case of 8-connectivity, if a foreground pixel has at least one background (black) neighbor then it becomes black; otherwise, it remains unchanged. The pixels which change from foreground to background are pixels at the edges of background regions in the input image, so the consequence is that background regions grow in size, and foreground features tend to disconnect or further separate (Fig. 7). Background features or holes inside a foreground region grow, and corners are sharpened (Fig. 8). Further erosion results in further growth of the background, or shrinking of the foreground (Fig. 9).

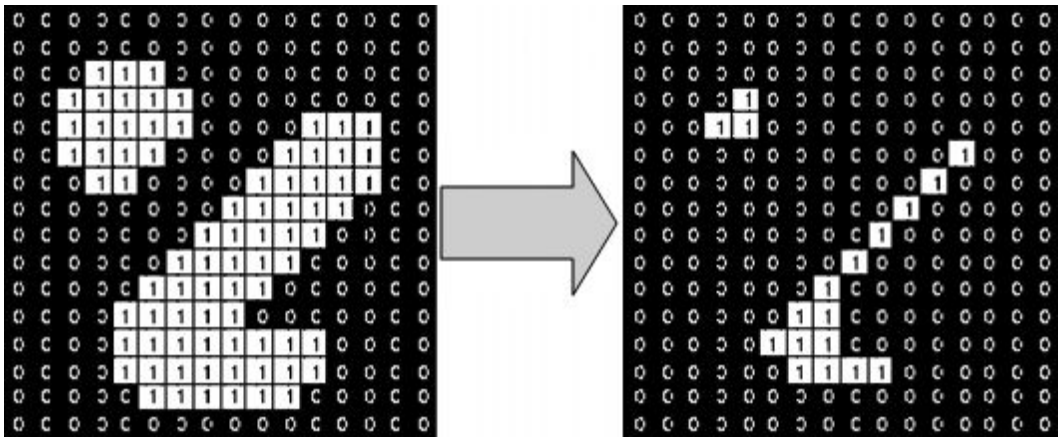


Fig. 7: The effect of erosion in further separating foreground features, using a structuring element corresponding to 8-connectivity.

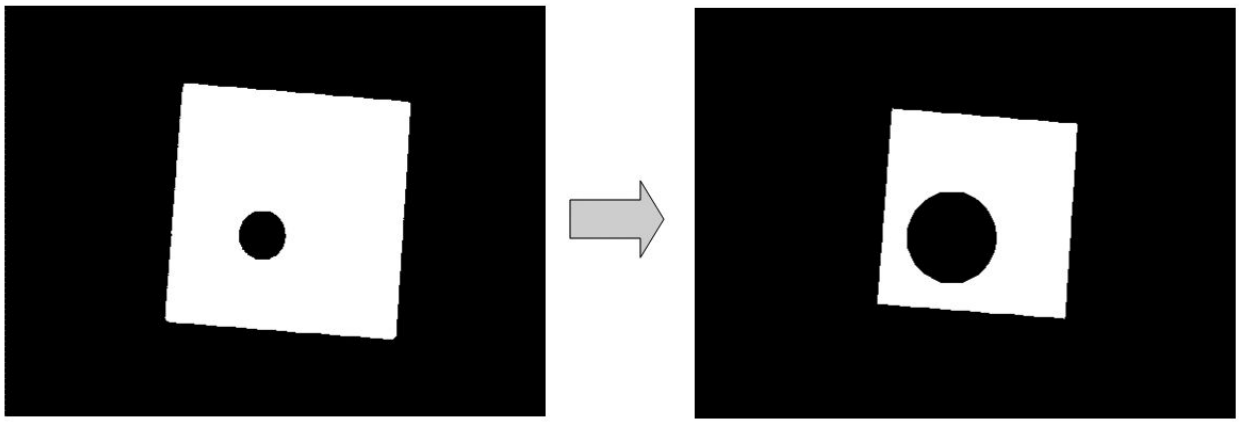


Fig. 8: The effect of erosion in growing background features and sharpening corners, using a structuring element corresponding to 8-connectivity.

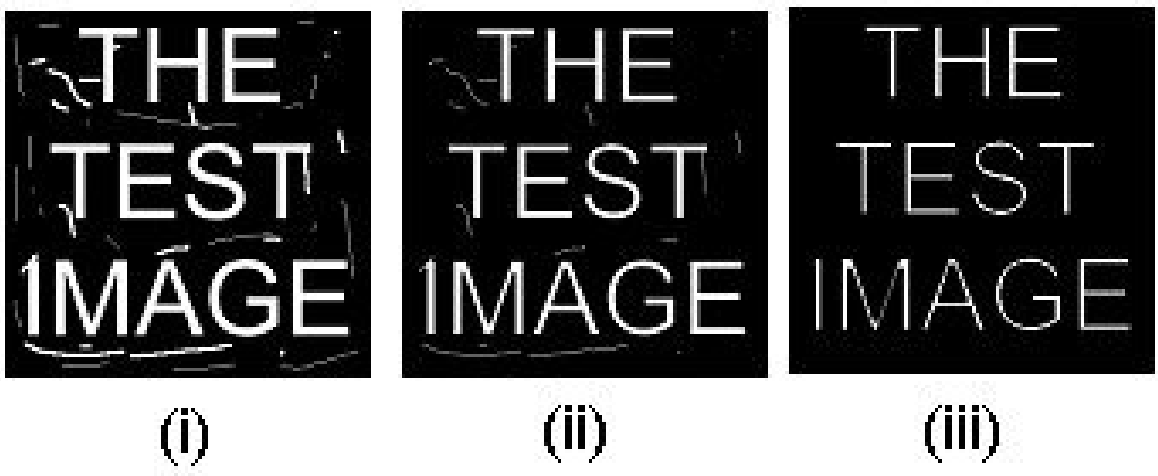


Fig. 9: (i) Original image (ii) after a single erosion (iii) after two erosions.

Again erosion can be considered analogous to convolution. As the structuring element moves inside the image, the boundaries of the image are moved inwards because image foreground pixels in the image are changed to background pixels wherever the structuring element overlaps the background region by at least one element. One of the basic applications of erosion is to eliminate irrelevant detail, below a certain size, in an image. A structuring element eliminates detail smaller than about its own size. Erosion can be also used to create a one-pixel wide outline of the features in an image by subtracting the eroded image from the original image.

Erosion is the dual of dilation, i.e. eroding foreground pixels is equivalent to dilating background pixels. However, erosion of an image followed by dilation of the result, or vice versa, does not produce the original image; isolated foreground pixels removed during erosion, for example, are not re-instated during dilation.

Erosion can help in the counting of features which touch or overlap in an image (Fig. 10). The first stage in counting the features is to segment the image, i.e. simplify it by reducing it to black and white. If the features still touch each other, they can be separated by erosion (activity 2).

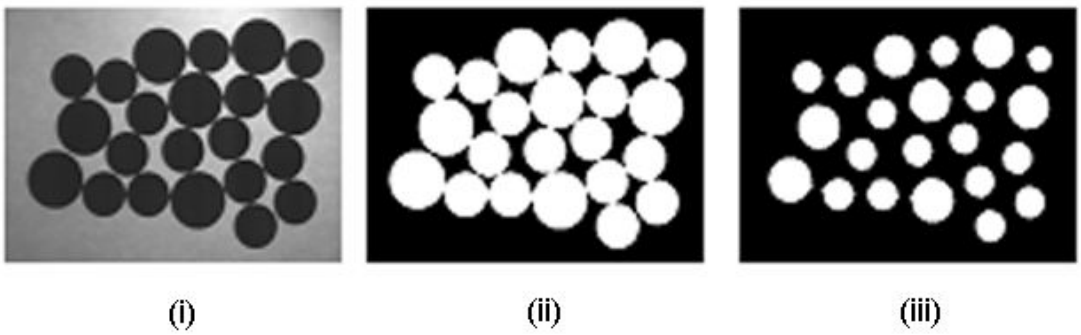


Fig. 10: (i) Grayscale image with features that touch each other (ii) the image after segmentation (iii) erosion of image (ii).

It is possible to do a constrained or conditional dilation. An image, known as the seed image, is dilated but not allowed to dilate outside of a supplied mask image, i.e. the resulting features are never larger than the features in the mask image. This is illustrated in activity 3. This can be a useful function in feature extraction and recognition. An image (Fig. 11(i)) could be thresholded to give the mask image (Fig. 11(ii)), and then further processed to isolate parts of a certain subset of features (say, only those larger than a certain size) in a seed image (Fig. 11(iii). The features can then be grown back to their original shape using the mask image to constrain the dilation (Fig. 11(iv)).

In a binary image, each feature is considered to be a connected set of pixels (either 4-connected or 8-connected). Before we can measure the properties of these features (for example, their areas), we need to label them. Labeling involves finding a foreground pixel in the image, giving it a label, and recursively giving the same label to all pixels that are connected to it or to its neighbors. This process is repeated until all the foreground pixels have been assigned to a feature

and have a label; the label can be used to colorize the features (Fig. 11 (v)). Labeling can be done in a two-pass process. The image is examined in raster order. When a foreground (ON) pixel is found, neighboring pixels are examined. (In 4-connectedness, it is sufficient to examine the pixel to the left and the pixel above it; in 8-connected ness the pixel on the top left diagonal should also be examined). Four situations can occur. If none of these neighbors is ON, the current pixel is given a new label; if one of the neighbors is ON, the current pixel is given the same label; if more than one pixel is ON, and they are labeled similarly, the current pixel is given that same label; and if more than one pixel is ON, but they are labeled differently, the current pixel is given one of those labels and these labels are merged to a single label since they are connected and belong to the same feature. In the second pass the labels are re assigned sequentially. The properties of each individual feature can now be measured. For example the area of a feature is the number of foreground pixels that have that particular label; when all the features are measured, their size distribution can be displayed.

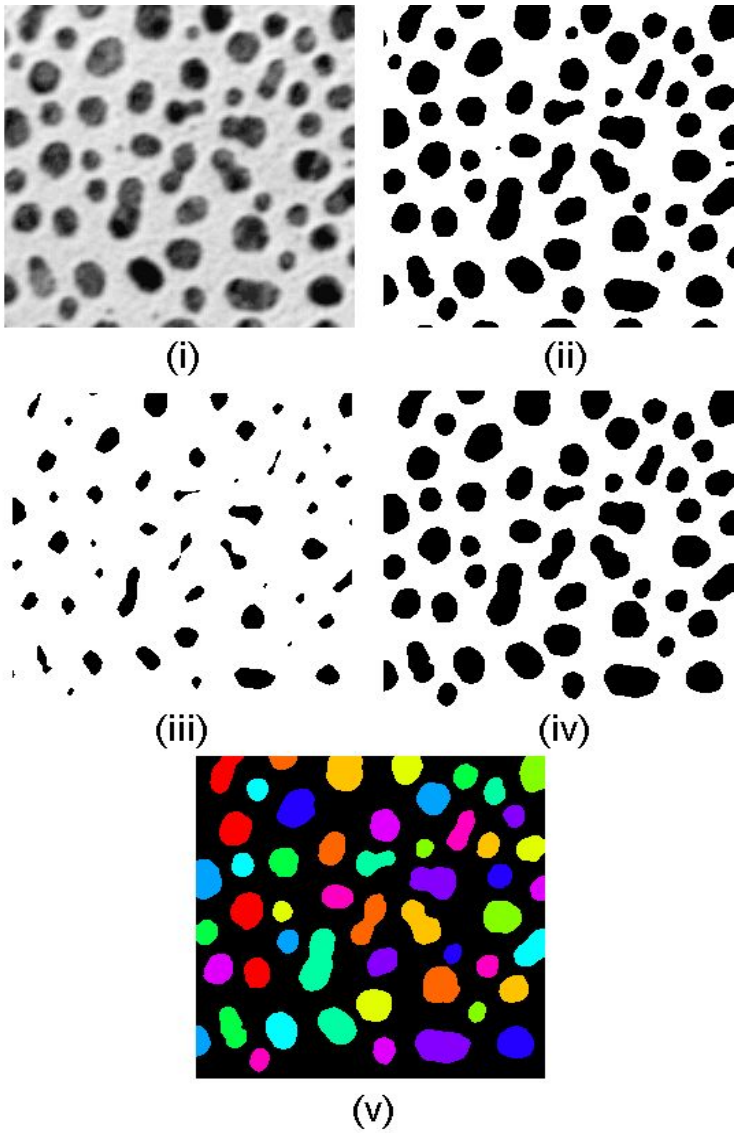


Fig. 11: (i) Original image, (ii) after thresholding, (iii) after 4 erosions, (iv) after 12 conditional dilations (the small objects have been removed) (v) after labeling and displaying each object in a different color.

2.3.3 Opening and Closing

All the other mathematical morphology operators can be defined in terms of combinations of erosion and dilation along with set operators such as intersection and union. Some of the more important of these other operators are opening, closing and skeletonization.

Opening is defined as erosion followed by dilation using the same structuring element for both operations. The erosion part of it removes some foreground (bright) pixels from the edges of regions of foreground pixels, while the dilation part adds foreground pixels. The foreground features remain about the same size, but their contours are smoother. As with erosion itself, narrow isthmuses are broken and thin protrusions eliminated.

The effect of opening on a binary image depends on the shape of the structuring element. Opening preserves foreground regions that have a similar shape to the structuring element, or that can completely contain the structuring element, while it tends to eliminates foreground regions of dissimilar shape. Thus binary opening can be used as a powerful shape detector to preserve certain shapes and eliminate others. The image in figure 12 (i) comprises a mixture of lines and circles, with the diameter of the circles being greater than the width of the lines. If a circular structuring element with a diameter just smaller than the diameter of the smallest circles is used to open this image, the resulting image (Fig 12(ii)) contains just the circles and the lines have been eliminated. Activity 4 contains examples of opening used as a shape detector.

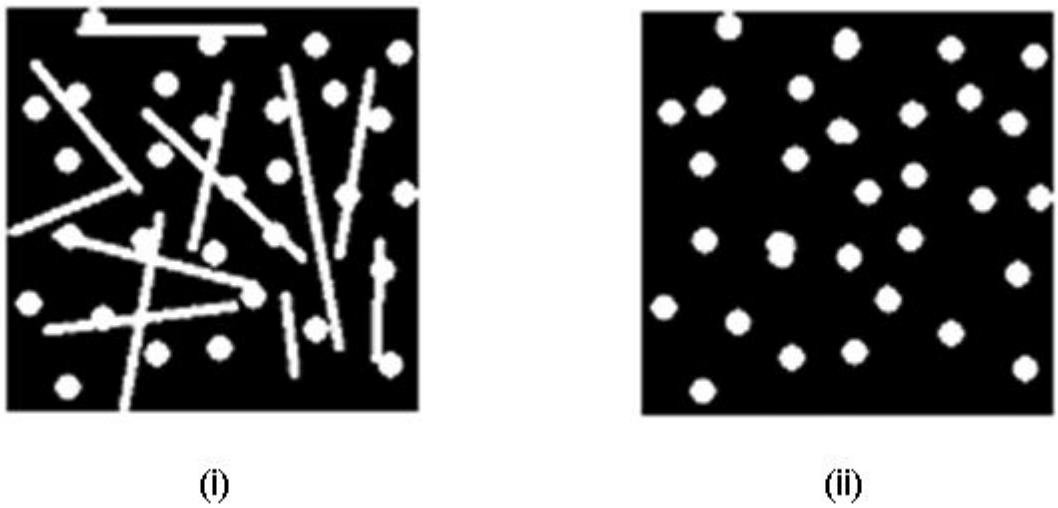


Fig. 12: Binary opening used as a shape detector. (i) An image comprising both lines and circles, (ii) the result after opening (i) with a circular structuring element.

This is how to think of opening. Take the structuring element and slide it around inside each foreground object. Pixels which can be covered by the structuring element with it remaining entirely within the object are preserved. Foreground pixels which can not be reached by the structuring element without it protruding outside the object are eroded away. When the structuring element is a circle, or a sphere in three dimensions, this operation is known as a rolling ball, and is useful for subtracting an uneven background from grayscale images.

Closing is defined as dilation followed by erosion using the same structuring element for both operations. Closing smoothes the contours of foreground objects but, in contradistinction to opening, it merges narrow breaks or gaps and eliminates small holes. Figure 13 illustrates how closing can be used to eliminate the smaller holes in the image. A circular structural element of size mid-way between the diameter of the two sets of holes was used to close the image in Fig. 13(i); the resulting image (Fig. 13(ii) contains only the larger holes, since only they allow the structuring element to move freely inside them without protruding outside.

Opening and closing are frequently used to clean up artifacts in a segmented image prior to further analysis (Fig 14). The choice of whether to use opening or closing, or a sequence of erosions and dilations, depends on the image and the objective. For example, opening is used when the image has foreground noise or when we want to eliminate long, thin features. It is not used when there is a chance that the initial erosion operation might disconnect regions. Closing is used when a region has become disconnected and we want to restore connectivity. It is not used when different regions are located closely such that the first iteration might connect them. Usually a compromise is determined between noise reduction and feature retention by testing representative images.

You can practice using these operations in activity 5.

(i) (ii)

Fig. 13 (i) An image containing holes of two different sizes, and (ii) the result of closing (i) with a circular structuring element mid way in size between the two sets of holes.

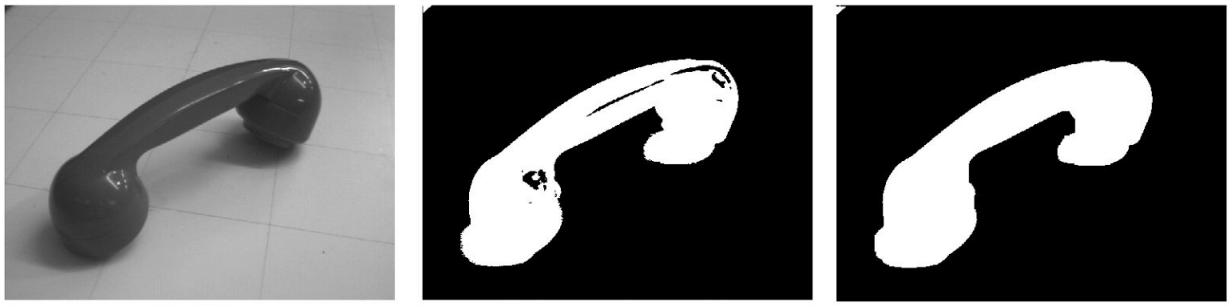


Fig. 14: (i) Original grayscale image, (ii) segmented image showing various artifacts, (ii) the result of closing (ii) with a circular structuring element.

As in the case of erosion and dilation, opening and closing are the duals of each other, i.e. opening the foreground pixels with a particular structuring element is equivalent to closing the background pixels with the same element. Opening and closing are also idempotent operations, i.e. repeated application of either of them has no further effect on an image.

2.3.4 Hit-or-Miss Transform

The hit-or-miss transform is a basic tool for shape detection or pattern recognition. Indeed almost all the other morphological operations can be derived from it.

The structuring element is an extension of those we have used before which contained only 1's and 0's. In this case it contains a pattern of 1's (foreground pixels), 0's (background pixels) and x's ("don't cares"). An example, used for finding a bottom left right-angle corner point, is shown in Fig. 15.

x	1	x
0	1	1
0	0	x

Fig. 15: Example of the extended type of structuring element used in hit-or-miss operations.

The hit-or-miss operation is performed by translating the center of the structuring element to

all points in the image, and then comparing the pixels of the structuring element with the underlying image pixels. If the pixels in the structuring element exactly match the pixels in the image, then the image pixel underneath the center of the structuring element is set to the foreground color, indicating a "hit". If the pixels do not match, then that pixel is set to the background color, indicating a "miss". The x's or "don't care" elements in the structuring element match with either 0's or 1's. When the structuring element overlaps the edge of an image, this would also generally be considered as a "miss". Look at the white pixel at the bottom left hand corner of the feature in figure 16, and imagine the structuring element of figure 15 placed on it. This is recognized as a bottom left corner of the object because of the pattern of three 1's in the foreground, and the pattern of three 0's describing the background, which are matched in the structuring element. The other three neighboring pixels can be either 0's or 1's and this central pixel remains a corner point; hence they are designated x's (don't cares) in the structuring element.

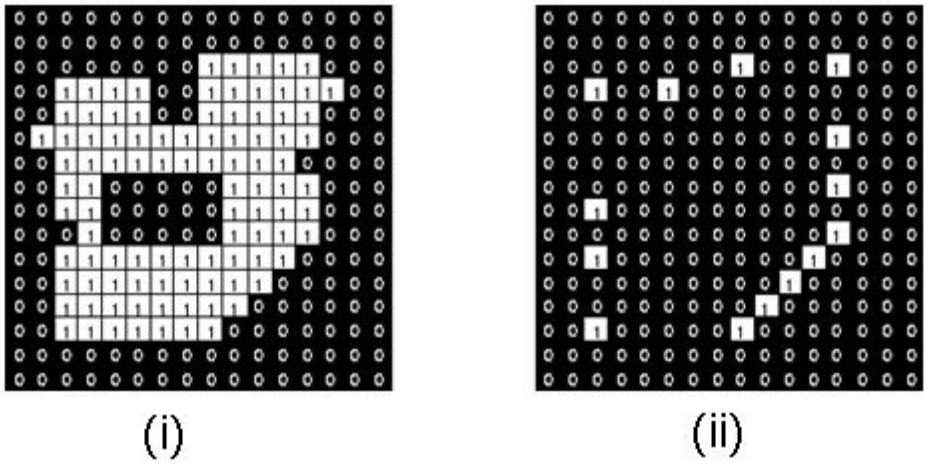
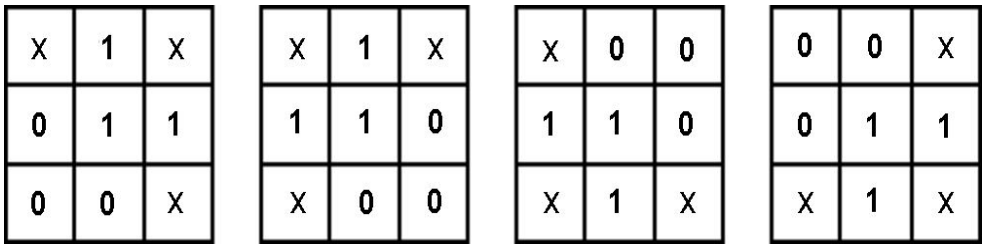


Fig. 16: (i) Image of a white feature (ii) the final result, locating all the right angle corners of the feature by combining the results of using the hit-or-miss transform with the four structuring elements of Fig. 15.

In order to find all the corners in a binary image we need to run the hit-or-miss transform four times with four different structuring elements representing the four kinds of right angle corners found in binary images (Fig. 17), and then combine the four results, using a logical "OR", to get the final result showing the locations of all right angle corners in any orientation. Figure 16 shows the final result of locating all the right angle corners of a feature. Activity 6 illustrates other practical examples of using the hit-or-miss transform.

Different structuring elements can be used for locating other features within a binary image, for example isolated points in an image, or end-points and junction points in a binary skeleton.



(i) (ii) (iii) (iv)

Fig. 17: The four structuring elements used for finding corners in a binary image using the hit-or-miss transform. The leftmost one detects bottom left corners (as we saw in Fig. 15), and the others are derived from it with various rotations to detect the bottom right, top right and top left corners respectively.

2.3.4 Thinning and Skeletonization

Thinning is a morphological operation that successively erodes away foreground pixels from the boundary of binary images while preserving the endpoints of line segments. Thickening is the dual of thinning, i.e. thickening the foreground is equivalent to thinning the background.

The thinning operation is related to the hit-and-miss transform and can be expressed quite simply in terms of it. The thinning of an image I by a structuring element J is:

thin (I,J) = I – Hit-or-miss (I,J) (4)

where the subtraction is a logical subtraction defined by X – Y = X Not Y.

For example the structuring element of Fig. 18(i), and the three rotations of it by 900, are essentially line detectors. If a hit-or-miss transform is applied to the rectangle of Fig. 18(ii), using this structuring element, a pixel-wide line from the top surface of the rectangle is produced, which is one pixel short at both right and left ends. If the line is subtracted from the original image, the original rectangle is thinned slightly. Repeated thinning produces the image shown in Fig. 18(iii). If this is continued, together with thinning by the other three rotations of the structuring element, the skeleton shown in Fig. 18(iv) is produced.

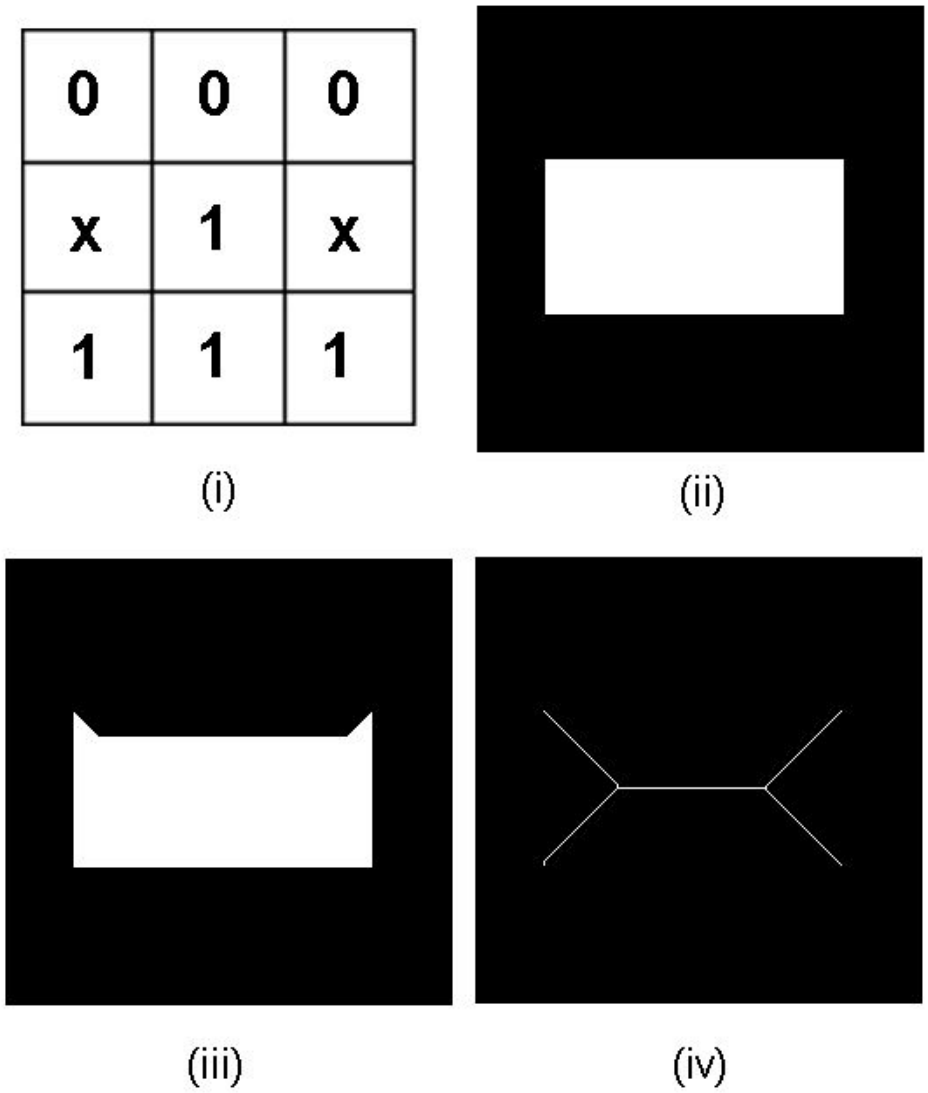


Fig. 18: (i) Structural element for line detection, (ii) image of rectangle, (iii) image of rectangle after 12 iterations of thinning with the structural element of (i), (iv) thinning to convergence using the structural element of (i) and its three 900 rotations.

Repeated thinning can be used to obtain a single-pixel wide skeleton or center line of an object. One of the most common uses of skeletonization is to reduce the thresholded output of an edge detector such as the Sobel operator to lines of a single pixel thickness. Skeletonization needs to be implemented as a two-step process that does not break the objects. The first step is normal thinning, but it is conditional; that is, pixels are marked as candidates for removal, but are not actually eliminated. In the second pass, those candidates which can be removed without destroying connectivity are eliminated, while those that cannot are retained. The process is then repeated several times until no further change occurs, i.e. until convergence, and the skeleton is obtained. Skeletonization preserves the topology, i.e. the extent and connectivity, of an object. The skeleton should be minimally eight connected, i.e. the resulting line segments should always contain the minimal number of pixels that maintain eight-connectness, and the approximate end-line locations

should be maintained. Various implementations have been proposed; the algorithm of Zhang and Suen [Zhang and Suen, 1984] is probably the most widely used realization.

The skeleton is useful because it provides a simple and compact representation of the shape of an object. Thus, for instance, we can get a rough idea of the length of an object by finding the maximally separated pair of end points on the skeleton. Similarly, we can distinguish many qualitatively different shapes from one another on the basis of how many junction points there are, i.e. points where at least three branches of the skeleton meet. Although skeletonization can be applied to binary images containing regions of any shape, it is most suitable for elongated (Fig. 19), as opposed to convex or blob-like, shapes. For example, it is useful for visualizing the center line of blood vessels in an angiogram and in automated recognition of hand-written characters.

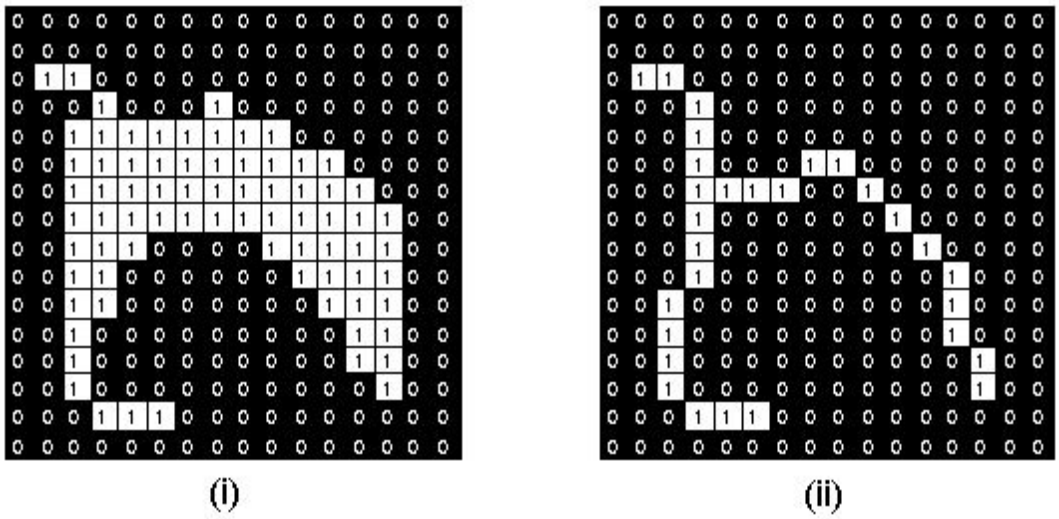


Fig. 19: Skeletonization by morphological thinning. (i) Binary image showing an elongated white foreground object (ii) its skeleton.

Skeletons produced by this method tend to leave parasitic components or spurs as a result of small irregularities in the boundary of the original object. These spurs can be removed by a process called pruning, which is in fact just another form of thinning. The structuring element for this pruning operation is shown in figure 20. Pruning is normally carried out for only a limited number of iterations to remove short spurs, since pruning until convergence actually removes all pixels except those that form closed loops (activity 7).

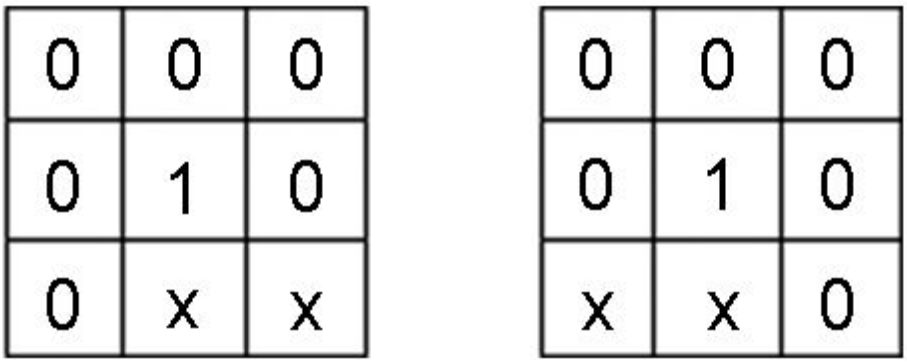


Fig. 20: Structural elements used for pruning. At each iteration, each element must be used in each of its four 90° rotations.

Skeletonization can be understood in terms of the prairie fire analogy. Imagine that the foreground region in a binary image is made of some uniform slow-burning material such as dry grass on a bare dirt background. If fires were to be started simultaneously at all points along the boundary of the region, the fire would proceed to burn inwards towards the center of the region until all the grass was consumed. At points where the fire traveling from two different boundaries meets itself, the fire extinguishes itself and the points at which this happens form the so-called quench line. This line is the skeleton. Another way to think about the skeleton is as the loci of centers of bi-tangent circles that fit entirely within the foreground region being considered. Figure 21 illustrates this for a rectangular shape.

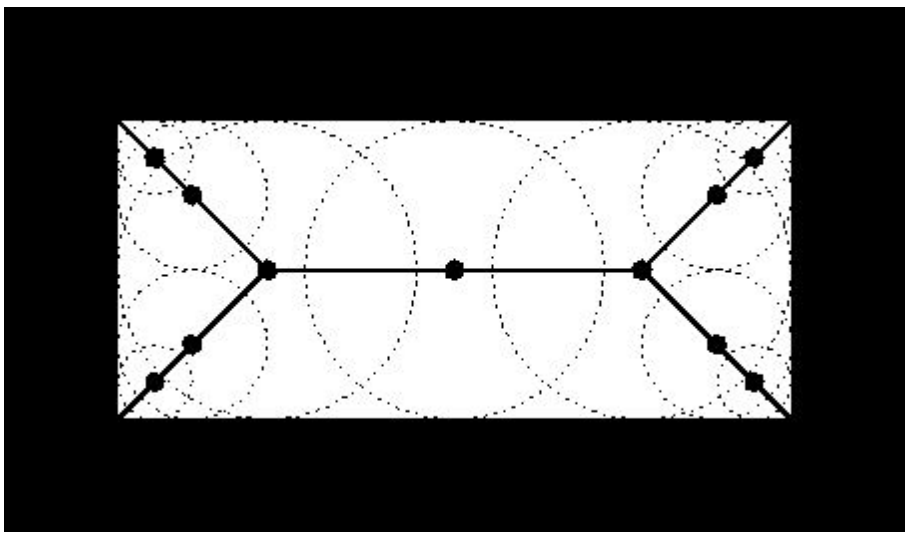


Fig. 21: The skeleton of a rectangle defined in terms of bi-tangent circles.

2.3.5 The Medial Axis Transform

The terms medial axis transform (MAT) and skeletonization are often used interchangeably but they are different. Skeletonization produces a binary image showing the simple skeleton. The medial axis transform on the other hand produces a grayscale image where each point on the

skeleton has an intensity which represents its distance to a boundary in the original object. Thus the medial axis transform (but not the skeleton) can be used to exactly reconstruct the original shape, which makes it useful for lossless image compression, by constructing circles of radius equal to the pixel value around each pixel. The skeleton is the medial axis transform, thresholded such that only the center pixels, one pixel in width, are above the threshold.

The medial axis transform is closely linked to the distance transform, which is the result of performing multiple successive erosions with a structuring element that depends on which distance metric has been chosen, until all foreground regions of the image have been eroded away, and labeling each pixel with the number of erosions that had to be performed before it disappeared (Fig. 22). The distance transform can also be used to derive various other symmetries from binary shapes. Although there are many different implementations the medial axis transform is essentially the locus of slope discontinuities (i.e., the ridges) in the distance transform; if the distance transform is displayed as a three-dimensional surface plot with the third dimension representing the gray value, the medial axis transform can be imagined as the ridges on the three-dimensional surface.

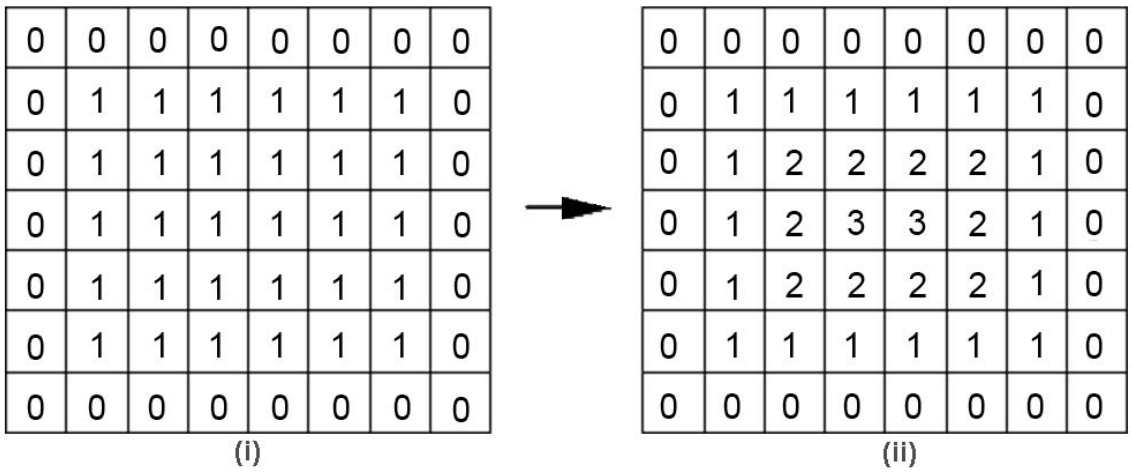


Fig. 22: Schematics of (i) a binary image of a rectangle and (ii) its distance transform image (note: using the N8 distance metric).

The skeletons and the medial axial transforms, obtained from the distance transforms, of a number of images are compared in Fig. 23.

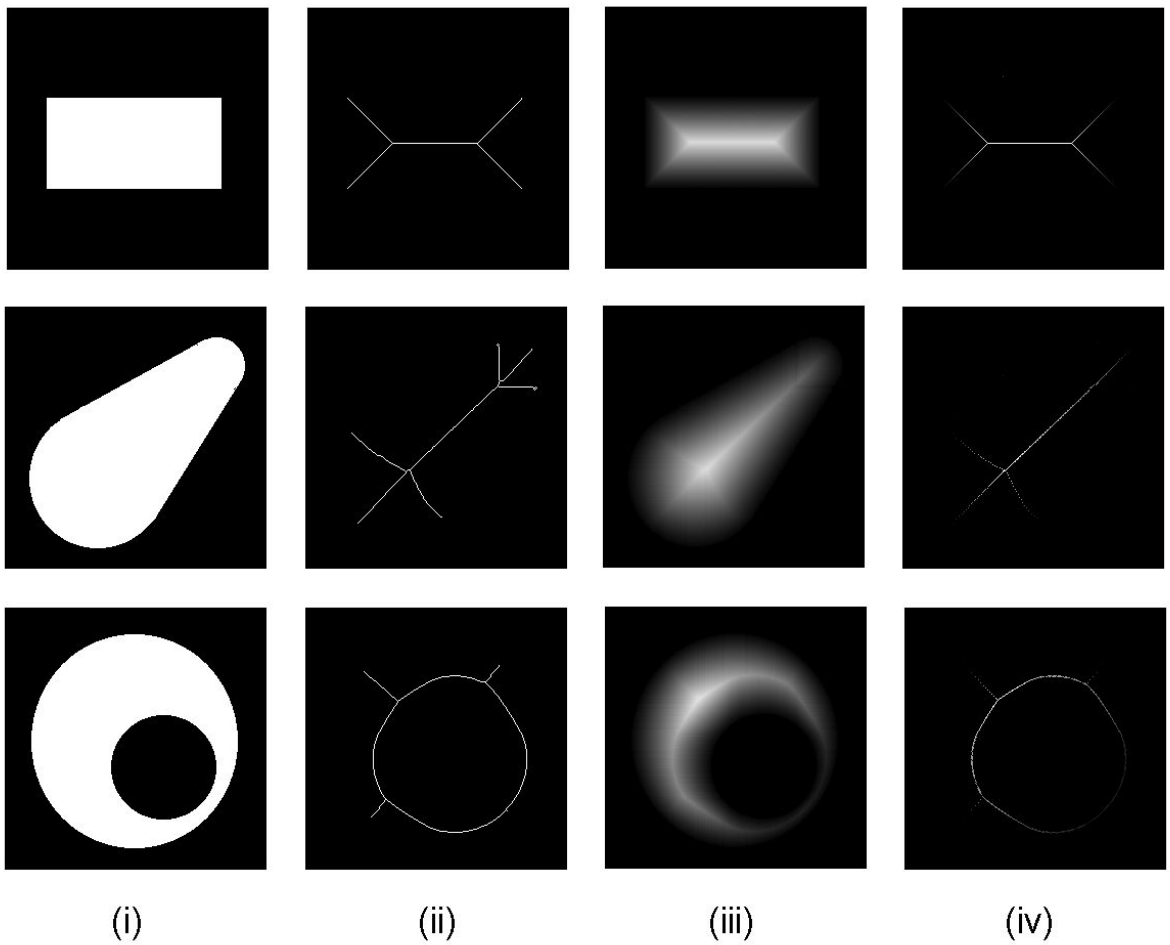


Fig. 23: (i) Original images, (ii) their skeletons, (iii) their Euclidean distance transforms (after contrast enhancement) and (iii) their medial axis transforms.

Both the skeleton and the medial axis transform are sensitive to small changes in the boundary of the object, which can produce artifactually more complex skeletons (Fig. 24; compare to Fig. 23(ii), top). The skeletonized image in figure 25 shows the very complex skeleton produced by skeletonizing a thresholded image of a telephone receiver and the less complex skeleton, more representative of the true shape of the telephone receiver, produced when the thresholded image is closed prior to skeletonization.

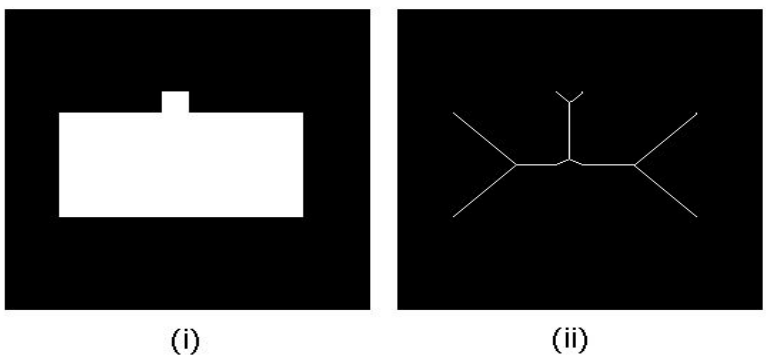


Fig. 24: (i) An image of a rectangle with a small change in its boundary; (ii) the result of skeletonizing image (i).

The skeleton can be further improved by pruning insignificant spur features. These examples indicate that additional processing may often be required prior to skeletonization.

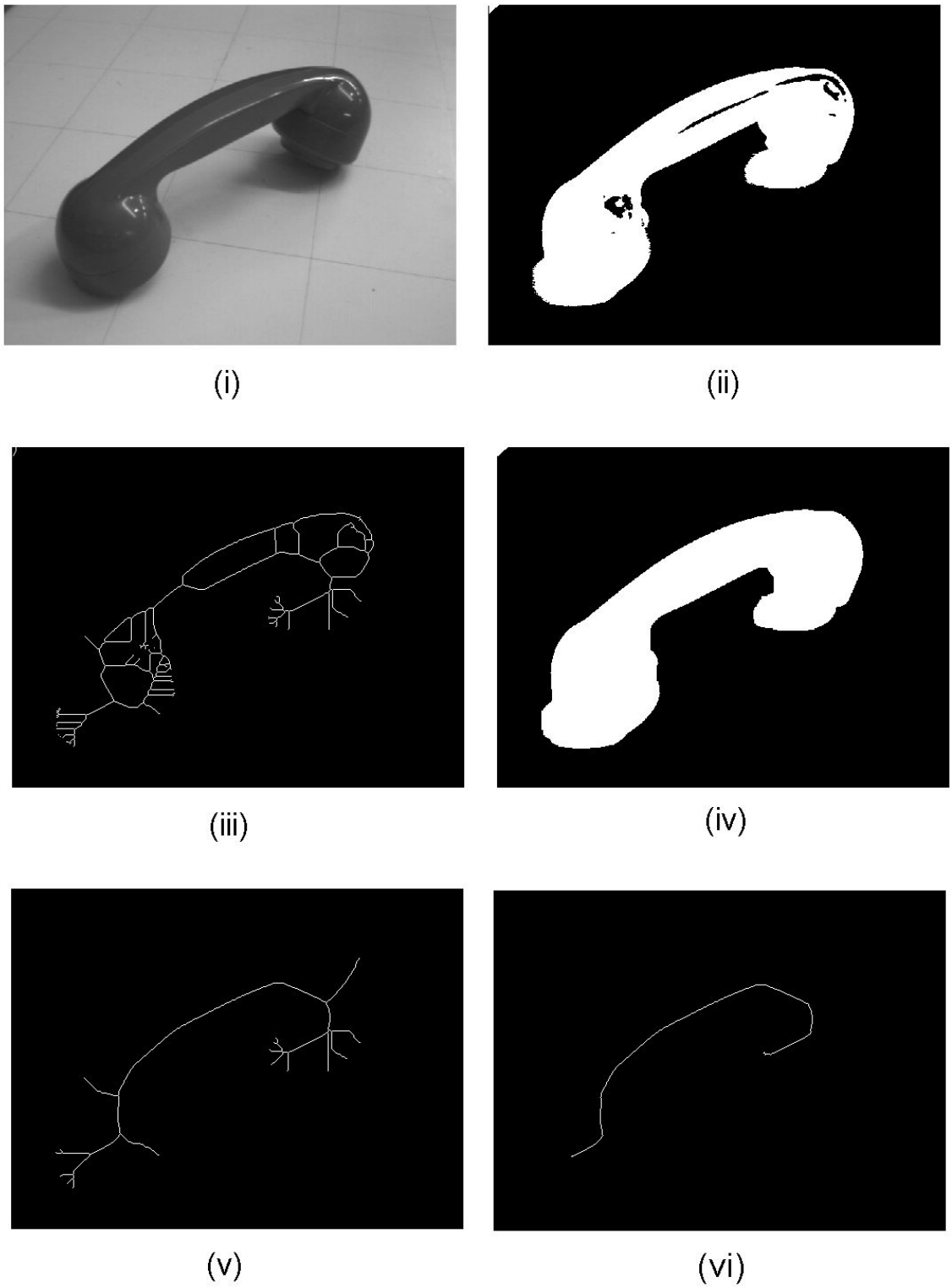


Fig. 25: (i) Grayscale image of a telephone receiver; (ii) after thresholding image (i); (iii) after skeletonizing image (ii); (iv) after closing image (ii) with a circular structural element; (v) after skeletonizing image (iv); (vi) after pruning (v).

Both skeletonization and the medial axis transform are also very sensitive to noise. If some "pepper noise" is added to the image of a white rectangle (Fig. 26(i)), the resulting skeleton (Fig. 26(ii)) connects each noise point to the skeleton obtained from the noise free image (see Fig. 23(ii), top).

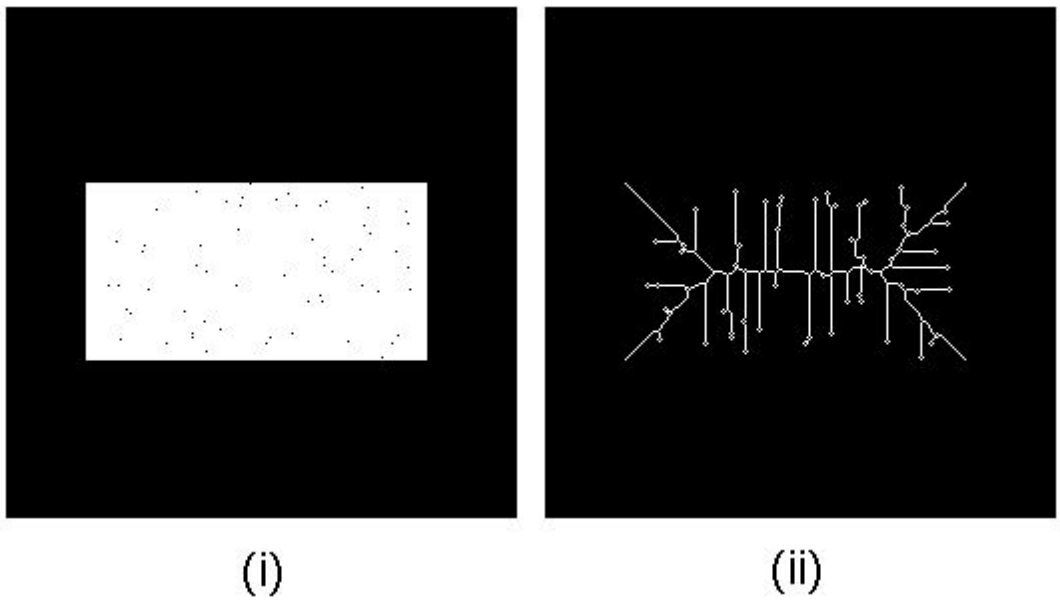


Fig. 26: (i) Image containing "pepper" noise and (ii) the resulting skeleton.

Just as the skeleton of objects or features in an image can be determined, it is also possible to skeletonize the background. This gives the so-called "skiz" (skeleton of influence zone) image (Fig. 27 and 28). This effectively divides the image into regions or zones of influence around each feature. Discontinuous lines can be easily removed; starting at each end point (points with a single neighbor) connected pixels are eliminated until a node (a point with more than two neighbors) is reached. The skiz is actually the generalized Voronoi diagram.

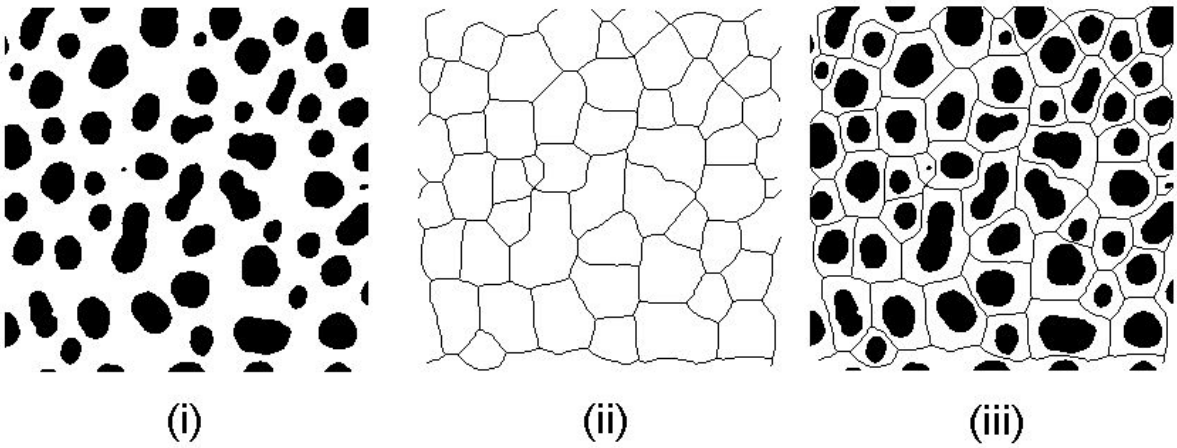


Fig. 27: (i) Image containing features (ii) the skeleton of the background (or skiz) (iii) skiz superimposed on original image to show zones of influence.

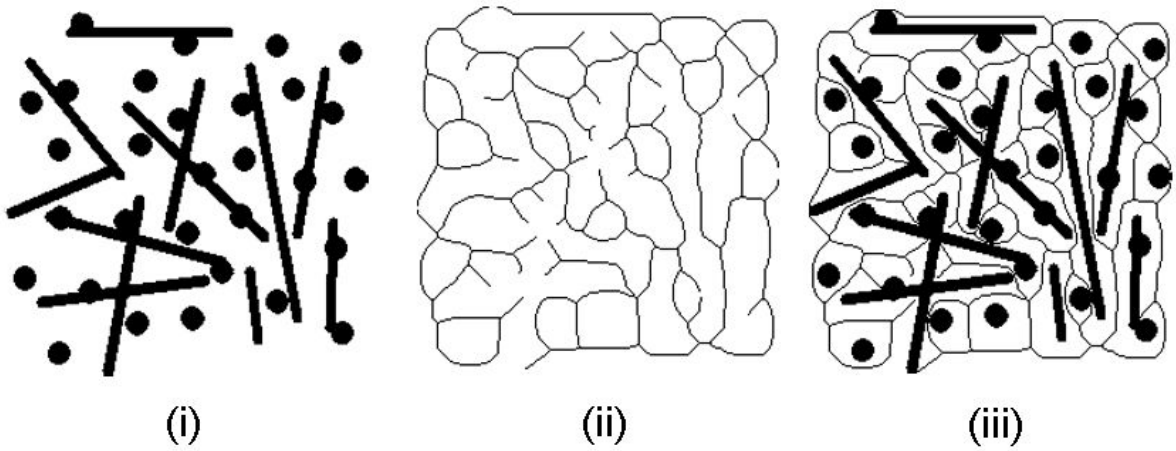


Fig. 28: (i) Image containing features (ii) the skeleton of the background (or skiz); note that there are some discontinuous lines which should be eliminated) (iii) skiz superimposed on original image to show zones of influence.

2.3.6 The Convex Hull

The convex hull of a binary feature can be visualized quite easily by imagining stretching an elastic band around the feature. The elastic band follows the convex contours of the feature, but `bridges' the concave contours. The resulting shape has no concavities and contains the original feature (Fig. 29). Where an image contains multiple disconnected features, the convex hull algorithm determines the convex hull of each of them, but does not connect disconnected features, unless their convex hulls happen to overlap.

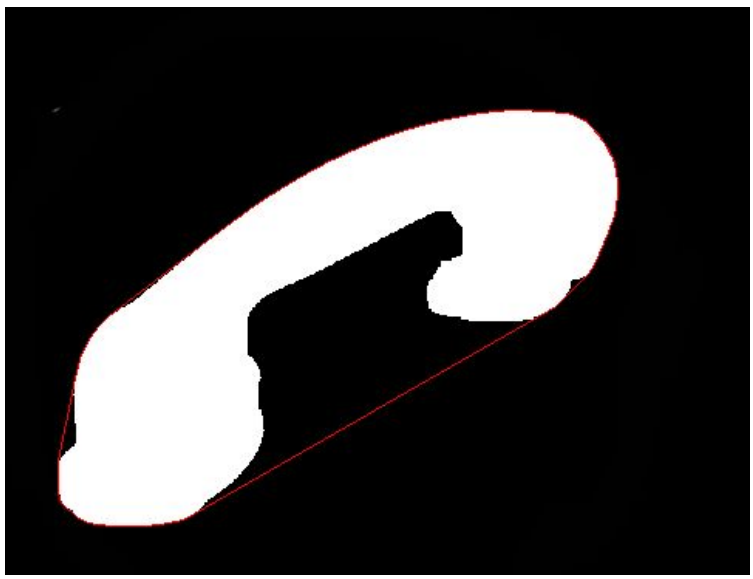


Fig. 29: A feature enclosed by its convex hull (shown in red).

The convex hull is the smallest convex polygon that contains the object in an image. Its simple shape often suffices to perform matching or recognition, and it delineates the area of influence of an object or region; if another region or its convex hull overlaps this convex hull, then it is said to encroach on the first region's area of influence.

An approximate convex hull can be computed using thickening with the structuring elements shown in figure 30. The convex hull computed using this method is actually a '45° convex hull' approximation, in which the boundaries of the convex hull must have orientations that are multiples of 45°. Note that this computation can be very slow.

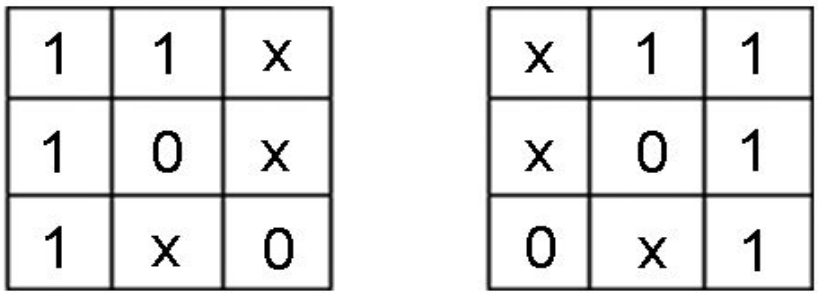


Fig. 30: Structuring elements for determining the approximate convex hull using thickening. During each iteration, each structuring element should be used in turn, and then in each of their 90° rotations, giving 8 effective structuring elements in total. The thickening is continued until no further changes occur.

Fig. 31 (i) shows an image containing a number of cross-shaped binary objects. The 45° convex hull algorithm described above results in convex hulls which depend on the orientation of the individual cross-shaped objects in the original image (Fig. 31(ii)). The process took a considerable amount of time - over one hundred thickening passes with each of the eight structuring elements!

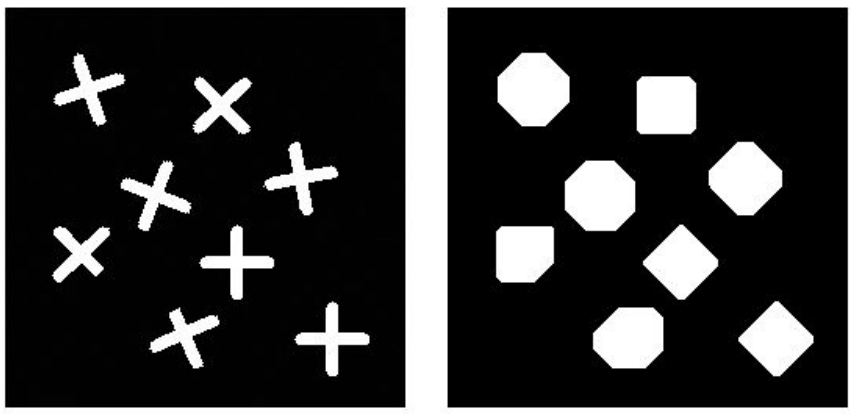


Fig. 31: (i) An image and (ii) its approximate (45°) convex hull.

Other more exact implementations of the complex hull exist, for example using angular sorting of the corners of an object (Graham scan), but they are beyond the scope of this text.

For a set of points the Voronoi diagram and its dual, the Delaunay triangulation are mathematically related to the convex hull. The Voronoi diagram is obtained by drawing bisectors of the lines between points and connecting them to form convex polygons. These polygons are then the polygons of influence around the points (Fig. 32); they are not as general as the skiz where the zones of influence are not constrained to be polygons. The Delaunay triangulation is a set of triangles with the points as vertices, such that no point is inside the circumcircle of any of the triangle (Fig. 33). (Delaunay triangulations are often used to build meshes for the finite element method). The outer boundary of the Voronoi diagram is the convex hull of all the points. Voronoi cells can also be defined by measuring distances to features that are not points. The Voronoi diagram with these cells is the medial axis.

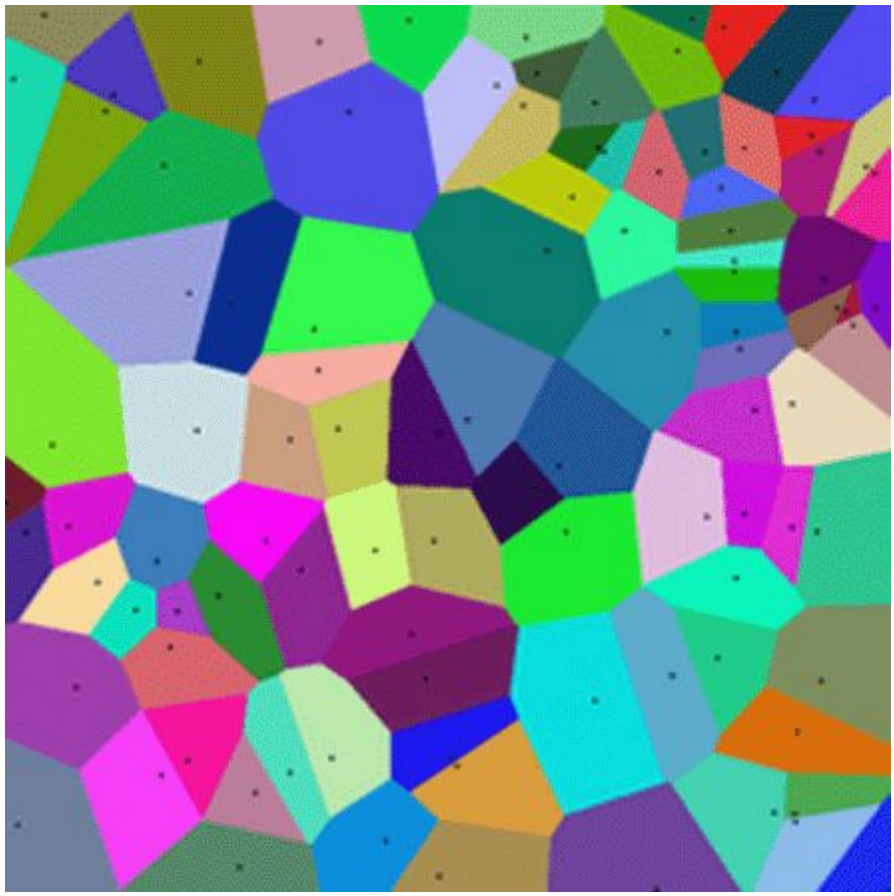


Fig. 32: The Voronoi diagram of a set of points, showing the polygons of influence.

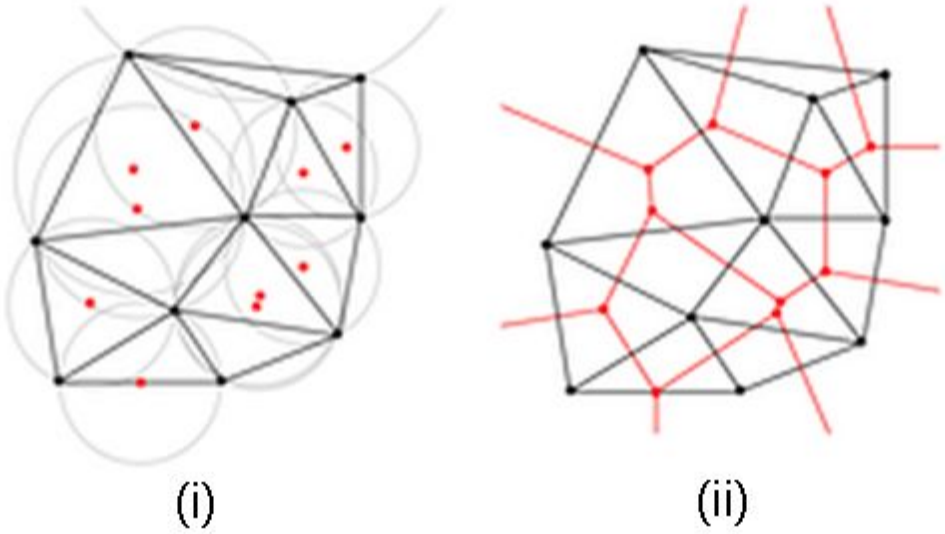


Fig. 33: (i) A set of points (in red) with their Delaunay triangulation and circumscribed circles (ii) connecting the centers of the circumscribed points produces the Voronoi diagram (in red).

2.3.8 Extension to Grayscale Images

The basic binary morphological operations can be extended to use with grayscale images; the results of such operations are grayscale images.

In grayscale dilation, for example, the structuring element is defined by a pattern of 1's and is superimposed on top of each pixel of the input image in turn. Only those pixels with a 1 on top of them are considered and the output pixel, which replaces the central image pixel, is the maximum of the pixel values under consideration. For grayscale erosion, the image pixel is replaced by the minimum of the pixels considered by the structuring element. Thus dilation brightens and expands brighter areas of an image, and darkens and shrinks darker areas. Erosion is the dual, and has the opposite effect.

Grayscale dilation and erosion are thus seen to be identical to convolution with the maximum and minimum rank masks, which operate like the median mask. The neighborhood around each pixel and the pixels are ordered by rank. If the center pixel is replaced by the maximum value in the neighborhood, grayscale dilation occurs. If the center pixel is replaced by the minimum value in the neighborhood, grayscale erosion occurs. And if the center pixel is replaced by the median value in the neighborhood, median filtering occurs (Fig. 34).

Grayscale opening and closing have the same form as their binary counterparts, i.e. grayscale opening is grayscale erosion followed by grayscale dilation, and grayscale closing is grayscale dilation followed by grayscale erosion.

Open = Max (Min (Image)) (5a)

Close = Min (Max (Image)) (5b)

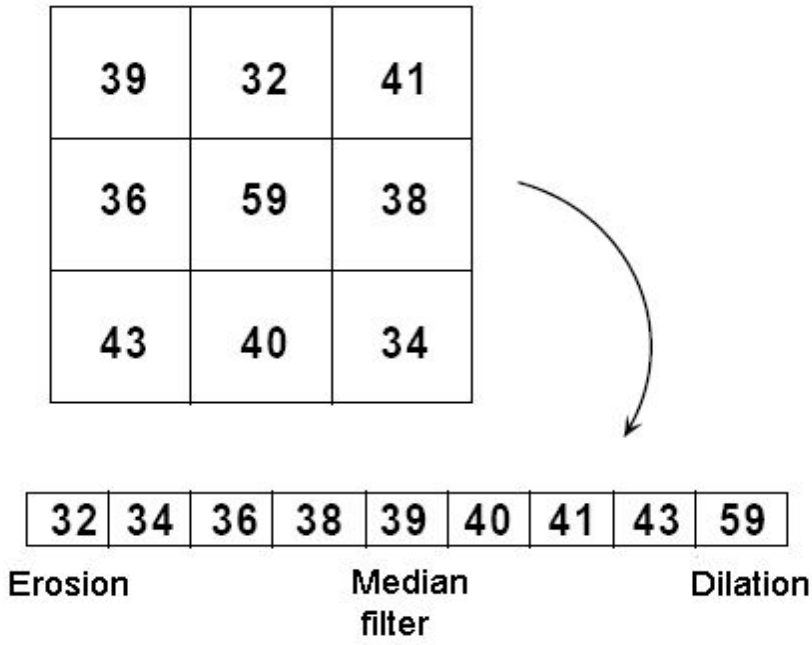


Fig. 34: Schematic showing pixels in a 3 x 3 neighborhood being ranked, as a prelude to replacing the center pixel by the maximum, median or minimum value. Each option corresponds to grayscale dilation, median filtering and grayscale erosion respectively.

Opening a grayscale image with a circular structuring element can be viewed as having the structuring roll under the profile of the image pushing up on the underside.

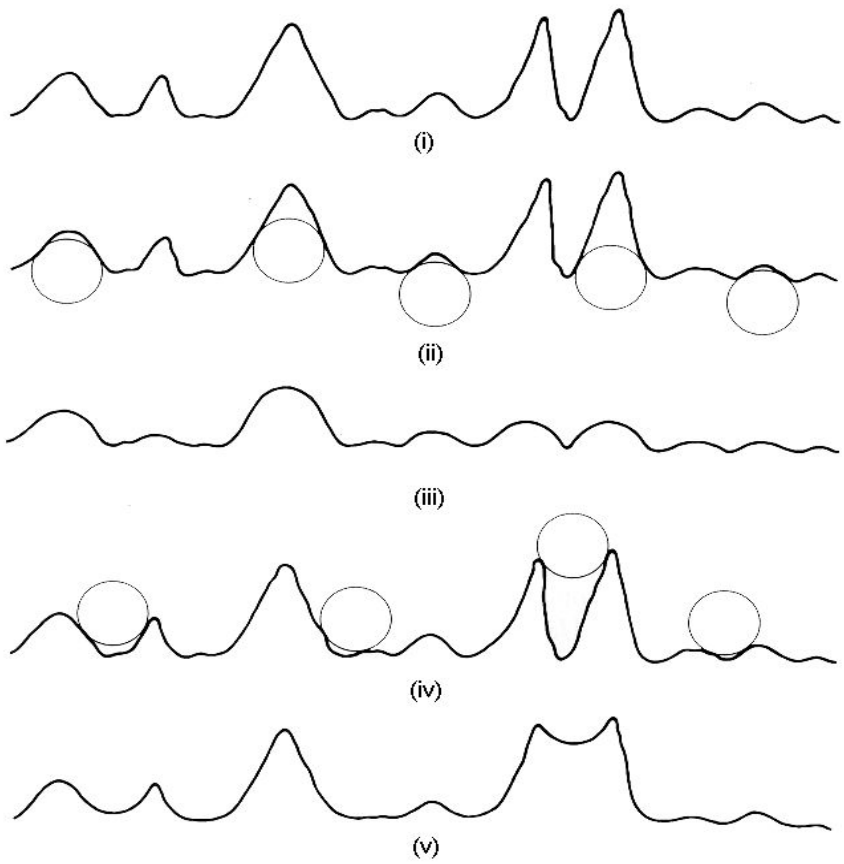


Figure .35 (i) a grayscale image profile (ii) positions of rolling ball for opening (iii) result of opening (iv) positions of rolling ball for closing (v) result of closing. (After Gonzalez and Woods, 2002).

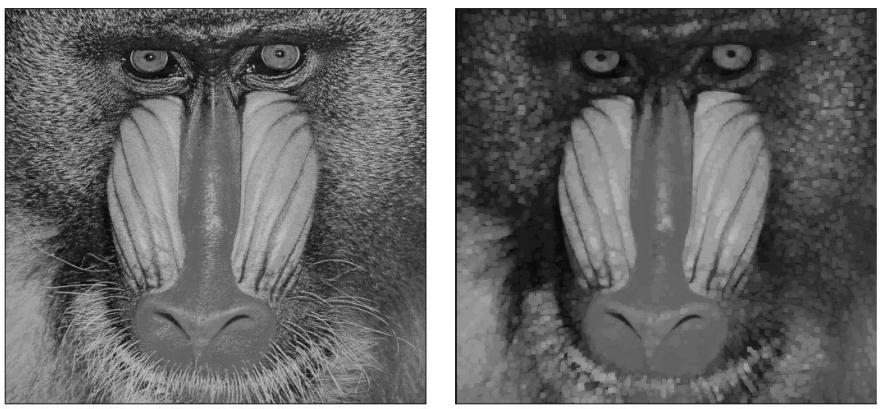
The result of the opening is the surface of the highest points reached by any part of this rolling ball (Fig. .35). Conversely, grayscale erosion can be viewed as the rolling ball traversing the image profile and pressing down on it, with the result being the surface of the lowest points reached by any part of the rolling ball.

Properties such as duality and idempotence also apply to the grayscale operators.

Applications of Grayscale Morphological Processing

Non-linear processing is often used to remove noise without blurring the edges in the image. Recall how the median mask out-performed the linear averaging mask in removing salt-and-pepper noise. Morphological processing is often used because of its ability to distinguish objects based on their size, shape or contrast, viz. whether they are lighter or darker than the background. It can remove certain objects and leave others intact, making it more sophisticated at image interpretation than most other image processing tools.

Grayscale opening smoothes an image from above the brightness surface, while grayscale closing smoothes it from below. They remove small local maxima or minima without affecting the gray values of larger objects. Grayscale opening can be used to select and preserve particular intensity patterns while attenuating others. Figure .36 illustrates the effect of grayscale opening with a flat 5×5 square structuring element. Bright features smaller than the structuring element are greatly reduced in intensity, while larger features remain more or less unchanged in intensity. Thus the fine grained hair and whiskers in the original image are much reduced in intensity, while the nose region is still at much the same intensity as in the original. Note that the opened image does have a more matt appearance than before since the opening has eliminated small fluctuations in texture.



(i) (ii)

Fig.36: (i) Original image (ii) after grayscale opening with a flat 5 × 5 square structuring element

Similarly, opening can be used to remove ‘salt noise’ in grayscale images. Figure .37 shows an image containing salt noise, and the result of opening with a 3 × 3 square structuring element. The noise has been entirely removed with relatively little degradation of the underlying image.

Fig. 37: (i) Original image with ‘salt’ noise (ii) after grayscale opening with a flat 3 x 3 square structuring element

A sequential combination of these two operations (open-close or close-open) is referred to as morphological smoothing can be used to remove ‘salt-and-pepper’ noise (see activity .8).

In images with a variable background it is often difficult to separate features from the background. Adaptive processing is a possible solution. An alternative solution is so-called morphological thresholding, in which a morphologically smoothed image is used to produce an image of the variable background which can then be subtracted from the original image. The process is illustrated in fig. 38. Activity 9 contains several practice images.

Morphological sharpening can be implemented by the morphological gradient, MG, operation

$$MG = \tfrac{1}{2} (\text{Max (Image)} - \text{Min (Image)}) \qquad (6)$$

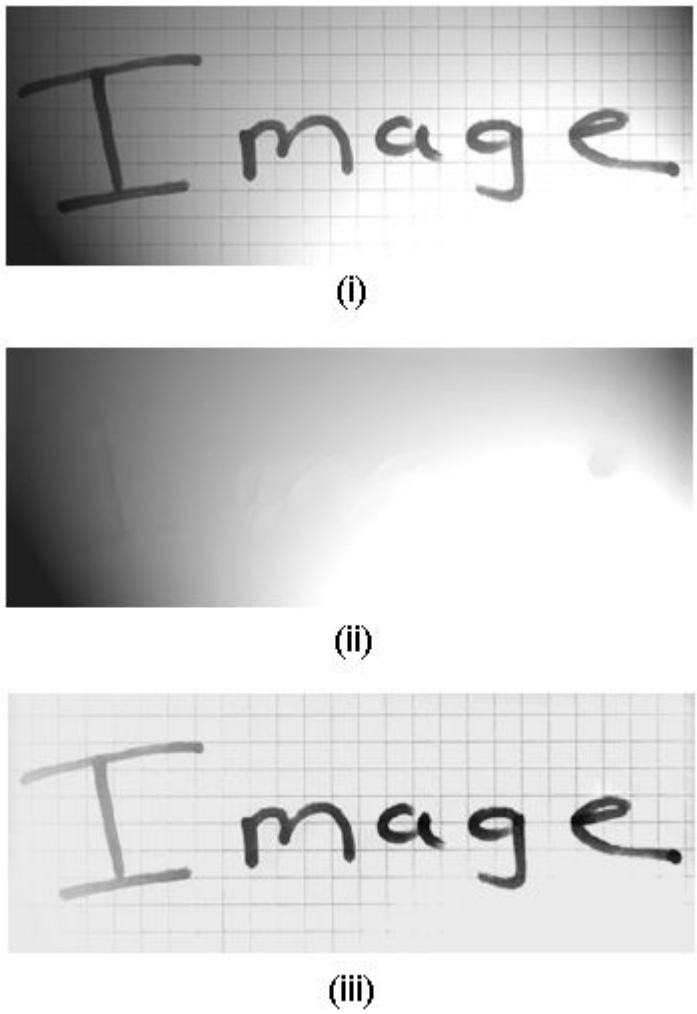


Fig. 38: (i) Image of text on variable background; (ii) morphological thresholding produces variable background; (iii) subtraction of (ii) from (i) to separate text.

The effect of the morphological gradient on a one-dimensional gray-level profile is shown in figure 39. The edges of the original image are replaced by peaks.

If a symmetrical structuring element is used, such sharpening is less dependent on edge directionality than using sharpening masks such as the Sobel masks.

The morphological top hat transformation, TH, is defined by

TH = Image – Open (Image) (.7)

It is the analog of unsharp masking, and is useful for enhancing detail in the presence of shading.

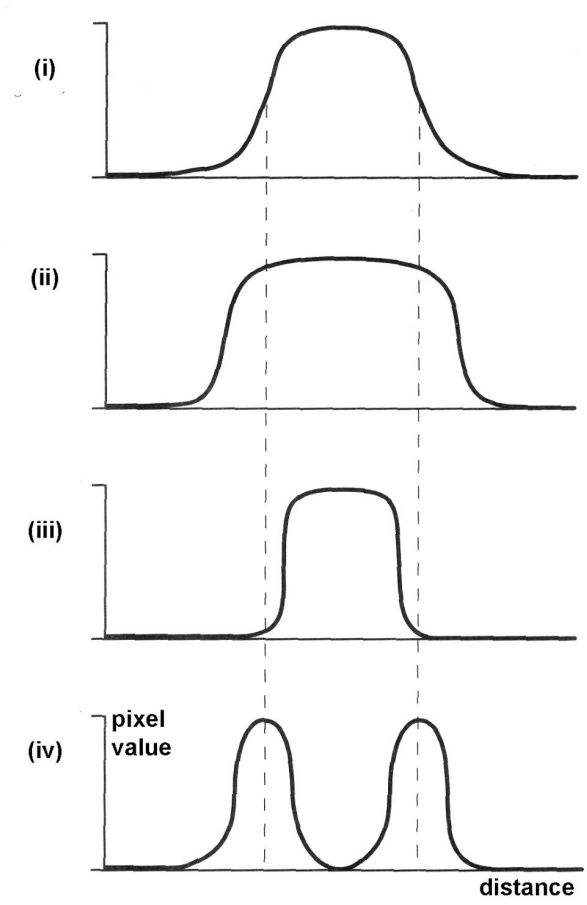


Fig. 39: (i) Profile of gray-level image (ii) profile of dilated image (iii) profile of eroded image (iv) profile of morphological gradient. (After Baxes, 1994).

In local contrast stretching the amount of stretching that is applied in a neighborhood is controlled by the original contrast in that neighborhood. It is implemented by:

$$G = (A - \text{Min}(A)) * \text{scale} / (\text{Max}(A) - \text{Min}(A)) \quad (8)$$

The Max (dilate) and Min (erode) operations are taken over the structuring element; "scale" is a small number. This operation is an extended version of the point operation for contrast stretching.

Granulometry is the name given to the determination of the size distribution of features within an image, particularly when they are predominantly circular in shape. Opening operations with structuring elements of increasing size can be used to construct a histogram of feature size, even when they overlap and are too cluttered to enable detection of individual features. The difference between the original image and its opening is computed after each pass. At the end of the process, these differences are normalized and used to construct a histogram of feature-size distribution. The resulting histogram is often referred to as the pattern spectrum of the image (Fig.

.40 and activity .10).

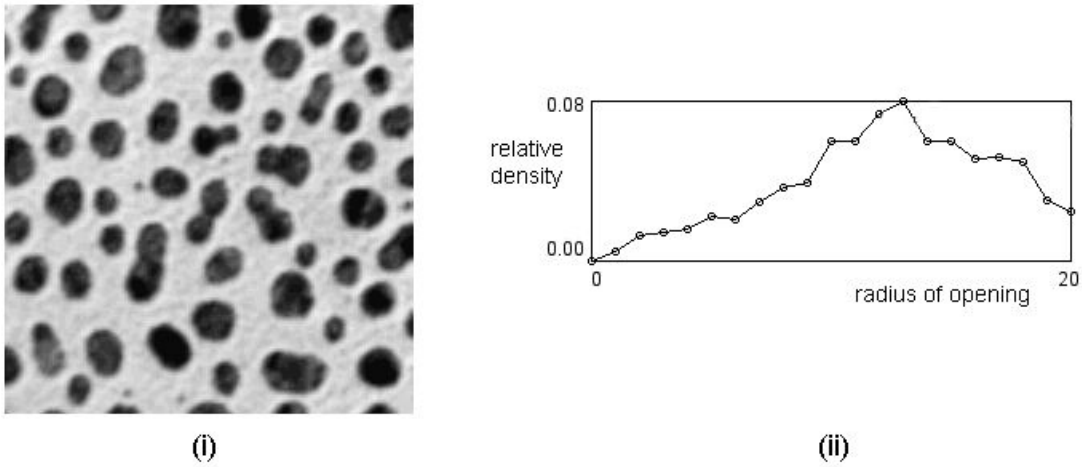


Fig. 40: (i) Image containing features of different size (ii) the histogram of feature-size distribution.

Computer-Based Activities

Activity .1 Neighborhood shapes

Open the image deltaim, which has a single white ('ON') pixel in the center of a 2562 image, in ImageJ. Using Plugins Plugins/Morpho plugin, set the operation to Max (equivalent to dilation), the number of iterations to 1 and the connectivity to 8. Repeat the dilation on the result, and then dilate again, and again. Observe how the point is dilated to show the neighborhood, using chessboard distances. Repeat with connectivity set to 4. Observe how the 4-connected neighborhood is built up from city-block distances. The neighborhood shapes represent PSFs of different widths. Starting from deltaim dilate using 4-connectivity, then 8-connectivity, then 4-connectivity, and so on. Note the shape of the neighborhood. What shape would result from Euclidean distances if it was possible to implement such a scheme? Which of the three schemes tested is closest to Euclidean?

Activity .2 Applications

Open box in ImageJ. Dilate the image using Process/Binary/Dilate, and subtract the original image (using Process/Image Calculator) from the result to obtain a one-pixel wide outline. You can also get edges by subtracting an eroded image from the original. The edges in both these images are one pixel shifted from each other. Repeat the outlining process using mri.

Open objects and threshold the image (Image/Adjust/Threshold …). Note the gray-level histogram. A threshold between ~50 and ~80 separates the darker objects from the lighter background, but the objects remain connected in the thresholded image. Note that reducing the

threshold value does not disconnect the objects. Instead some of the objects begin to disappear. Why is this? It is often useful to smooth an object with a small (e.g. 3x3) median mask, which avoids blurring the edges, prior to thresholding. After choosing the threshold, apply it to the image (Process/Binary/Make Binary) to obtain a binary image (check its histogram using Analyze/Histogram). The objects in the resulting binary image can be disconnected using erosion (Plugins/Binary/Erode). The objects themselves are reduced in size, and you might then consider increasing them to their former size approximately by dilating them by the same number of iterations which it took to separate them.

Try to separate the individual cells in the image cells. Threshold and binarize as before. The cells are black on a white foreground. Invert the image (Edit/Invert) to change to the more conventional situation of white (foreground) objects on a black background. Use erosions to disconnect the cells, but note that they become smaller and that the internal holes grow. You need to try and balance the erosions with dilations in an order that optimally disconnects the cells but preserves as much of their shape as possible. Determine whether 4-connectivity or 8-connectivity is better for this particular image.

Now try to separate the individual spots in the image spots.

Activity .3 Constrained/conditional dilation

Open cermet in ImageJ, smooth it slightly (Process/Filters/Mean … and use a radius=1), threshold it (Image/Adjust/Threshold) and save the result as mask. Make a duplicate (Image/Duplicate …) of this image, erode it four times (Process/Binary/Options, set to 4 iterations, then Process/Binary/Erode) and save the new result as seed. Make a duplicate of seed and dilate it ten times (Process/Binary/Options, set to 10 iterations, then Process/Binary/Dilate). Note how the objects in the resulting image have grown much larger than those in the original thresholded image, mask.

Conditionally dilate (Plugins/Ch.9 Plugins/BinaryConditionalDilate) the seed image ten times with seed as the seed image and the thresholded image, mask, as the mask image, and the number of iterations set to 10) Compare the result with the earlier result using (unconditional) dilation. The objects in the (conditionally) dilated image are not allowed to grow bigger than their size in the mask image.

Delete all the open images and record all the previous steps in a macro. (Use Plugins/Macros/Record … , name the macro ConDilate, and proceed to work through the processing steps as before. Observe how they are recorded. When you have finished, click “Create”). The macro can be re-run at any time by Plugins/Macros/Run … and choosing CondDilate.txt.

Activity .4 Opening as a shape detector

Open shapes in ImageJ, and use the cursor to find the diameter of the circles. Open Plugins/Ch.9 Plugins/Teacher Open and choose “Disk” as the type of structuring element, and a size just smaller than the diameter of the smallest circles. (”Square” uses an N8 neighborhood, “Cross” uses N4 and “Disk” uses a structuring element as close to a circle as you can get using a square matrix). Observe that the resulting image contains just the circles, i.e. you have used binary opening as a shape detector.

Open mixed cells. The image contains two kinds of cell; small, black ones and larger, gray ones. Threshold the image (Image/Adjust/Threshold …) to separate the cells from the background. Use a circular structuring element (Plugins/Ch.9 Plugins /Teacher Open) to remove the small cells and retain the larger cells. You should be able to do this fairly successfully by choosing an appropriate size of structuring element. (Note that it is not possible to isolate the small cells directly using this method).

Activity .5 Applications of closing and opening

Open holes in ImageJ. Find the diameter of both the small holes and the large holes in the image using the cursor. Use Plugins/Ch.9 Plugins/Teacher Close with an appropriate structural element to eliminate the smaller holes.

Open telephone and threshold it (Image/Adjust/Threshold …). The gray-level histogram is bimodal, showing two regions. The region with the lower pixel values roughly corresponds to the (dark) telephone receiver, and the other region corresponds to the background. Set the left-hand threshold to zero, and the right-hand threshold to mid-way between the two regions. Note that the resulting thresholded image includes the dark shadows under the telephone, and that some pixels within the telephone appear white because they reflected more of the illuminating light. You can vary the right-hand threshold to minimize these effects. If you increase the threshold value the white area shrinks but the shadow area expands, and vice versa if you reduce the threshold value. Choose a value just below the mid way value to minimize the shadow effect somewhat, and click “Apply”. Since we consider the dark telephone as the object of interest, we do an opening to try to clean it up; remember that opening and closing are duals of each other. Use Plugins/Ch.9 Plugins/Teacher Open with a circular structuring element of size 9 pixels. Observe the result. Try to improve on it by using other sizes, or by using the “cross” structuring element (shaped like a “plus” sign) and various sizes.

Activity .6 Hit-or-miss transform

Open rectangle in ImageJ and use the hit-or-miss transform (Plugins Plugins/BinaryHitorMiss) with a suitable structuring element (e.g.

212

011

002, where 0 indicates black, 1 is white and 2 is used to denote “don’t care”) to detect corners in the image. Choose 900 rotations and check white foreground. The result should be the four vertices of the rectangle.

Open binaorta, and use the hit-or-miss transform with a structuring element of

000

111

111 and no rotations. Observe the result, which shows the positions at which this pattern was matched.

We would like to be able to detect junction points, either bifurcations (splittings) or vessel crossings, at all orientations in this image. Experiment with different structuring elements to try to achieve this, making use of the 450 rotation feature in the plugin. A limitation is that the plugin only uses 3 x 3 structuring elements.

Activity .7 Thinning and skeletonization

Open rectangle in ImageJ. The built-in skeletonization within ImageJ is not very reliable. Invert the original image (Edit/Invert) and skeletonize it (Process/Binary/Skeletonize).

Instead use a macro that makes use of an alternative binary thinning plugin. Open rectangle and run (Plugins/Macros/Run) the macro skeleton1.txt. The result is the 8-connected skeleton. Repeat with the macro skeleton3.txt to obtain the 4-connected skeleton. Repeat using shape1, shape2, and box2.

Open telephone and smooth it (Process/Smooth) to reduce noise which can confound successful thresholding. Threshold it (Image/Adjust/Threshold), minimizing the shadows; binarize the result (Process/Binary/Make Binary) and invert it (Edit/Invert). Open it (Process/Binary/Open) to remove the black holes in the white object, and then run (Plugins/Macros/Run) the macro skeleton1.txt. Prune the spurs in the resulting skeleton by stages using (Plugins/Macros/Run) and the macro Prune1.txt as often as required, or run PruneAll.txt to prune until convergence.

Skeletonization can be used to find the center line of blood vessels. Open angio1, an angiogram, and angio2, the image after thresholding. Binarize angio2 (Process/Binary/Make Binary) and invert it (Edit/Invert). Run the 8-connectivity skeletonization macro (skeleton1.txt), followed by the stage-by-stage pruning (Prune1.txt). Compare your result with angio3.

Activity .8 Morphological smoothing

Open salt and pepper in ImageJ, and perform a grayscale opening (Plugins/Ch.9

Plugins/Teacher Open) with a disc-shaped (circular) structural element of size 3 pixels. Notice how the 'salt' is removed from the image. Close this resulting image (Plugins/Ch.9 Plugins/Teacher Close) with a circular structural element to remove the 'pepper'. Compare the final result with mri, the original image before salt-and-pepper noise was added.

Activity .9 Morphological thresholding

Open image in ImageJ. The generally darker text appears on a lighter, but variable background. However simple thresholding (Image/Adjust/Threshold …) is unable to separate the text from the background. Try it!.

Adaptive processing is a possible solution. Use Plugins/Ch.9 Plugins/Adaptive Threshold and try different parameters. (A neighborhood/mask size of 11, constant of 3, and mean thresholding gives a reasonable result). The text is separated from the variable background but the grid lines remain. They can be removed by opening; Process/Binary/Make Binary then Process/Binary/Open.

Morphological thresholding is an alternative solution. Use a sufficiently large structuring element and a close-open (Plugins/Ch.9 Plugins/Teacher Close followed by Plugins/Ch.9 Plugins/Teacher Open and a circular disk of size 15, say) to smooth out both the dark and light objects in the image, and produce an image of the variable background which can then be subtracted (Process/Image Calculator… using "Subtract' and "32-bit Result") from the original image. This is similar to the effect of Process/Subtract Background; try different parameter values.

Use these different techniques on yeast, uneven, sonnet and the two mammographic images lcc (a left cranio-caudal view) and rcc (a right cranio-caudal view).

Activity .10 Granulometry

Open cermet in ImageJ, and start Plugins/Ch.9 Plugins/Granulometry. Choose the minimum and maximum radii as 0 and 20 pixels respectively, and the step size as 1 pixel; check 'yes' to view intermediate images. The density distribution or pattern spectrum takes a few iterations before it appears. What is the radius of the predominant particles within this image?

2.4. Filtering Techniques:

Filtering is an image processing operation where the value of a pixel depends on the values of its neighboring pixels. Each of the pixels are processed separately with a predefined *window* (or *template,* or *mask*). Weighted sum of the pixels inside the window is calculated using the weights given by a mask, see Fig. 41. The result of the sum replaces the original value in the processed image:

$$f(x) = \sum_{i=1}^{9} w_i \cdot x_i$$

In the case of border pixels, the part of the mask lying outside of the image is assumed to have the same pixel values as that of the border pixels, see Fig. 42. Note that the filtering is a parallel operation, i.e. the neighboring values used in the calculations are always taken from the original image, not from the processed image.

w1	w2	w3
w4	w5	w6
w7	w8	w9

Fig. 41: General mask for filtering with a 3´3 window.

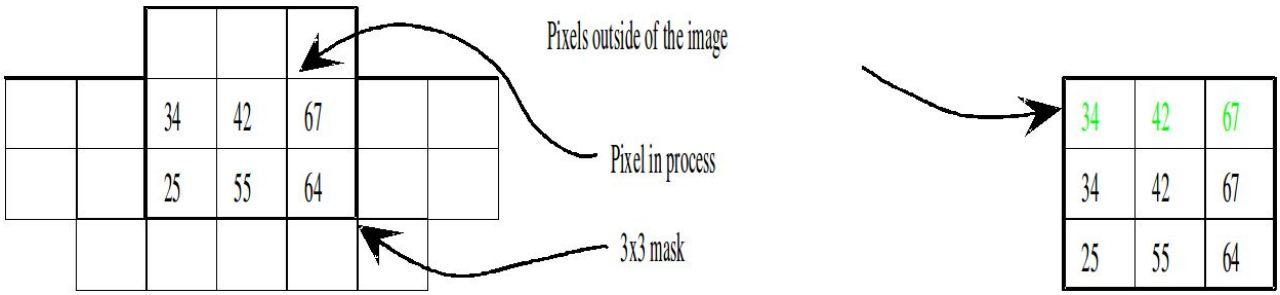


Fig. 42: Example of filtering operation in the case of border pixel.

2.4.1 Low-pass filtering

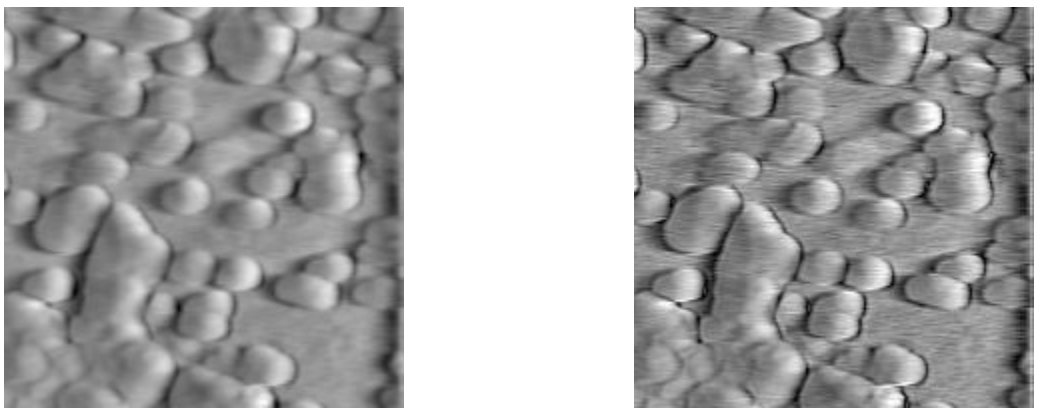
Low-pass filtering (or *averaging filtering*, or *smoothing*) reduces the *high frequency components* (or *noise*) in the image by averaging the pixel values over a small region (block). see Figure 43. This reduces noise and makes the image generally smoother, especially near the edges. The level of smoothing can be changed by increasing the size of the window.

(a) (b)

Fig. 43: (a) Original image "airplane" (512´512´8); (b) Smoothed by 3´3 averaging filter

2.4.2 High-pass filtering

High-pass filtering is the opposite operation to low-pass filtering. The *low frequency components* are eliminated and only the high frequency components in the image are retained. The operation can be applied in image enhancement by adding the result of the filtering to the original image. This is known as *sharpening*. It enhances the pixels near edges and makes it easier to observe details in the image, see Figure 44. The use of negative weights in the mask may result in negative values, thus the pixel values must be scaled back to [0, 255].



(a) (b)

Fig. 44: **(a)**Original image "sample-1" (200´200´8) ; (b) Sharpened by 3´3 window

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Mask for low pass filter

-1	-1	-1
-1	8	-1
-1	-1	-1

Fig. 45: Mask for high-pass filter.

2.4.3 Median filtering

Low-pass and high-pass filter are in the class of *linear filters*; they can always be described by a weighting mask. *Median filtering*, on the other hand, belong to a class of *rank filters*. Here the pixels within the window are *ranked* (or *sorted*) and the result of the filtering is chosen according to the ordering of the pixel values. In median filtering the new value of a pixel is the median of the pixel values in the window. The parameter of the filtering is the size and the shape of the filtering window (mask).

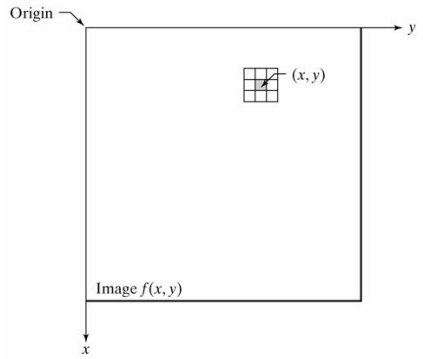
The median filter is used for removing noise. It can remove isolated impulsive noise and at the same time it preserves the edges and other structures in the image. Contrary to average filtering it does not smooth the edge. In the left image each pixel has been affected by the noise with a probability of 5 %, and in the right example with a probability of 20 %. A pixel that is affected by the noise is replaced either with a black or a white pixel.

2.4.4 Spatial Filtering

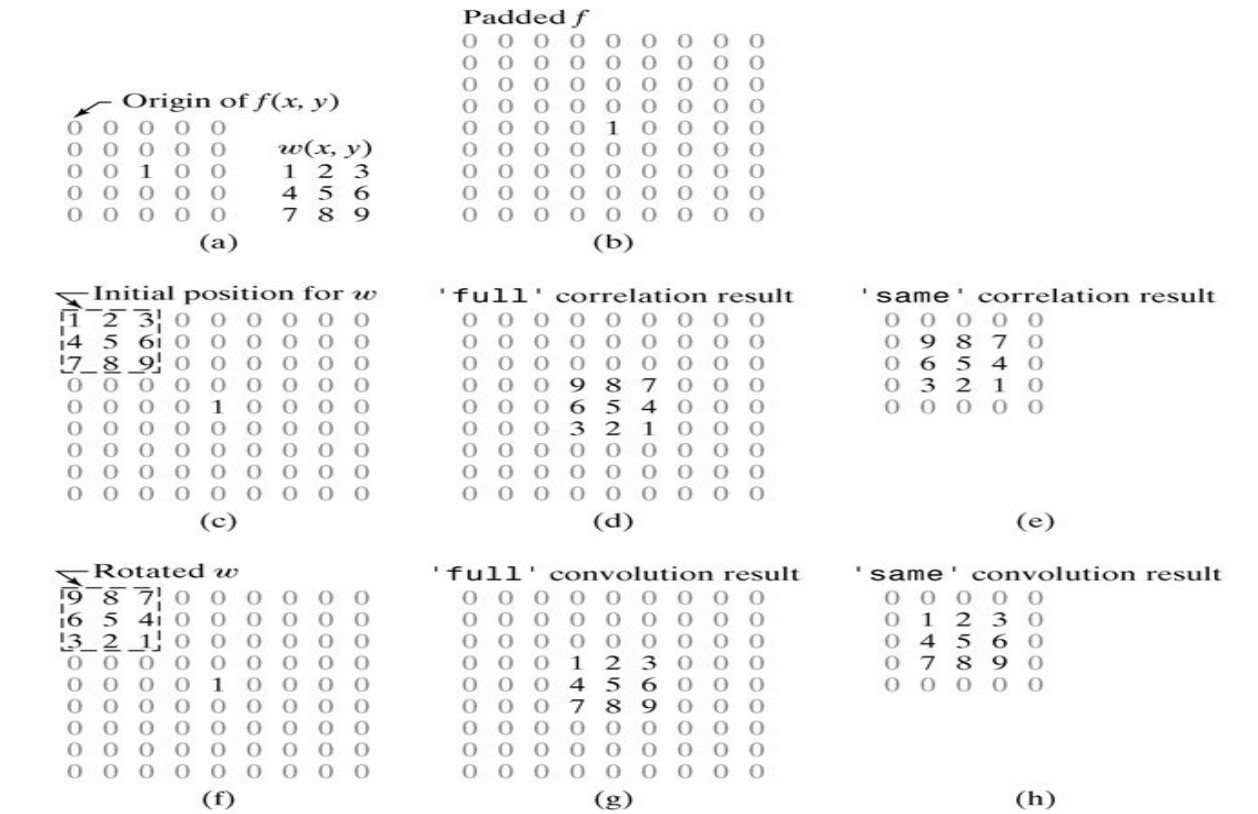
A large variety of image processing tasks can be accomplished by a technique called spatial filtering. Spatial filtering involves a mask, consists of an array of values (*a*-*i*) and has a center (gray).

The mask is placed on the image of interest:

MASK



and translated across all possible pixel positions on the image. A new (filtered) image is produced by replacing the intensity value at the center by a linear combination of the intensity values of the center pixel and all neighboring pixels covered by the mask. The coefficients (*a*-*i*) in the mask array give the weights of each pixel in the linear combination. Here is an example:



Filters can perform many different functions. The function of the filter is essentially determined by the value of the mask coefficients (*a*-*i*). In this tutorial, we consider primarily two classes of filters: 1) Smoothing filters, which reduce noise in an image, and 2) edge detection (or derivative) filters, which can detect object borders.

RECENT WORKS

In the very beginning of the research in the arena of launching artificial intelligence to establish techniques of Computer Vision for supporting blind and vision impaired people, disclosed the paradigms for serving people who are unable to go alone. There are a lot of interesting publications, projects and mobile apps available in this field as this is a burning topic nowadays. Here is an overview on them with indicating significant points.

Volodymyr Ivanchenko in 2008 described Clear Path Guidance Guiding blind and visually impaired wheelchair users along a clear path that uses computer vision to sense the presence of obstacles or other terrain features and warn the user accordingly. Since multiple terrain features can be distributed anywhere on the ground, and their locations relative to a moving wheelchair are continually changing, it is challenging to communicate this wealth of spatial information in a way that is rapidly comprehensible to the user. To develop a novel user interface that allows the user to interrogate the environment by sweeping a standard (unmodified) white cane back and forth: the system continuously tracks the cane location and sounds an alert if a terrain feature is detected in the direction the cane is pointing. Experiments are described demonstrating the feasibility of the approach.

Hend S. Al-Khalifa in 2008 proposed Utilizing QR Code This paper proposed a barcode-based system to help the visually impaired and blind people identifying objects in the introduced environment. The system is based on the idea of utilizing QR codes (two dimensional barcode) affixed to an object and scanned using a camera phone equipped with QR reader software. The reader decodes the barcode to a URL and directs the phone's browser to fetch an audio file from the web that contains a verbal description of the object. Their proposed system is expected to be useful in real-time interaction with different environments and to further illustrate the potential of new idea.

Jerey P. Bighamy in 2010 introduced VizWiz::LocateIt. This method enables blind people to take a picture and ask for assistance in finding a specific object. The request is first forwarded to remote workers who outline the object, enabling efficient and accurate automatic computer vision to guide users interactively from their existing cell phones. A two-stage algorithm is presented that uses this information to conduct users to the appropriate object interactively from their phone. They produced an app in ios and app is live in itunes, currently iPhone users can use and now they are trying to make the app for android mobiles.

Jing Su in September 2010 introduced Timbremap. A sonification interface enabling visually impaired users to explore complex indoor layouts using off-the shelf touch-screen mobile devices. This is achieved using audio feedback to guide the user’s finger on the device’s touch interface to convey geometry. The user study evaluation shows Timbremap is effective in

conveying non-trivial geometry and enabling visually impaired users to explore indoor layouts.

Roberto Manduchi in 2010 explored Blind Guidance using mobile computer vision Here they focused on the usability of a way-finding and localization system for persons with visual impairment. This system uses special color markers, placed at key locations in the environment, which can be detected by a regular camera phone. Three blind participants tested the system in various indoor locations and under different system settings. Quantitative performance results are reported for the different system settings, the reduced field of view setting is the most challenging one. The role of frame rate and of marker size did not result fully clear in the experiments.

Barcodes Ender Tekin and James M. Coughlan in 2010 presented Find and Read Product Barcodes They described a mobile phone application that guides visually impaired user to the barcode on a package in real-time using the phone's built-in video camera. Once the barcode is located by the system, the user is prompted with audio signals to bring the camera closer to the barcode until it can be resolved by the camera, which is then decoded and the corresponding product information read aloud using text-to-speech. Experiments with a blind volunteer demonstrate proof of concept of the system, which allowed the volunteer to locate barcodes which were then translated to product information that was announced to the user.

Y. Tian in 2010 explored Door Detection- Indoor Way finding. Here they presented a robust image-based door detection algorithm based on doors' general and stable features (edges and corners) instead of appearance features (color, texture, etc). A generic geometric door model is built to detect doors by combining edges and corners. Furthermore, additional geometric information is employed to distinguish doors from other objects with similar size and shape (e.g. bookshelf, cabinet, etc). The robustness and generalizability of the proposed detection algorithm are evaluated against a challenging database of doors collected from a variety of environments over a wide range of colors, textures, occlusions, illuminations, scale, and views.

F. Bellotti in May 2011 presented LodeStar A location-aware multimedia mobile guide, called LodeStar, was designed to enhance visually impaired people's fruition and enjoyment of cultural and natural heritage. It aimed at supporting visually impaired persons in the construction of location awareness by providing them with location-related added value information about object-to-self and object-to-object spatial relations. It presented the underlying psychological and theoretical basis, the description of the CHI design of the mobile guide and user test results coming from trials conducted with real users in two contexts of authentic use: the Galata Sea Museum and the Villa Serra naturalistic park in Genoa. Results mainly revealed that the guide can contribute to people's ability to construct overall awareness of an unfamiliar area including geographical and cultural aspects. The users still appreciate the guide descriptions of point of interests in terms of content and local/global spatial indications. Interviews highlighted for improving the visit experience of visually impaired people.

João José in May 2011 invented a system Local Navigation Aid Here the proposed SmartVision prototype is a small, cheap and easily wearable navigation aid for blind and visually

impaired persons. Its functionality addresses global navigation for guiding the user to some destiny, and local navigation for negotiating paths, sidewalks and corridors, with avoidance of static as well as moving obstacles. Local navigation applies to both inand outdoor situations. In this article they focused on local navigation: the detection of path borders and obstacles in front of the user and just beyond the reach of the white cane, such that the user can be assisted in centering on the path and alerted to looming hazards. Using a stereo camera worn at chest height, a portable computer in a shoulder-strapped pouch or pocket and only one earphone or small speaker, the system is inconspicuous, it is no hindrance while walking with the cane, and it does not block normal surround sounds. The vision algorithms are optimized such that the system can work at a few frames per second.

Brilhault, A. in 2011 introduced Pedestrian Positioning They designed an assistive device for the Blind based on adapted GIS, and fusion of GPS and vision based positioning. The proposed assistive device may improve user positioning, even in urban environment where GPS signals are degraded. The estimated position would then be compatible with assisted navigation for the Blind. Interestingly the vision module may also answer Blind needs by providing them with situational awareness (localizing objects of interest) along the path. Note that the solution
proposed for positioning could also enhance autonomous robots or vehicles localization.

Bharat Bhargava in 2011 explored Pedestrian Crossing A mobile-cloud collaborative approach provided here for context-aware outdoor navigation, where the computational power of resources made available by cloud computing providers for real-time image processing. The system architecture also has the advantages of being extensible and having minimal infrastructural reliance, thus allowing for wide usability. They have developed an outdoor navigation application with integrated support for pedestrian crossing guidance and report experiment results, which suggest that the proposed approach is promising for real-time crossing guidance for blind.

Boris Schauertey in 2012 proposed Find lost things Here they proposed a computer vision system that helps blind people to find lost objects. To this end, they combine color- and SIFT-based object detection with signification to guide the hand of the user towards potential target object locations. This way, it is able to guide the user's attention and effectively reduce the space in the environment that needs to be explored. They verified the suitability of the proposed system in a user study. They experimentally demonstrated that the system makes it easier for visually impaired users to find misplaced items, especially if the target object is located at an unexpected location.

Alexander Dreyer Johnsen in 2012 proposed Touch-based mobile for the visually impaired They revealed the idea to use touch-based phones by the visually impaired. Two possible technologies are screen-readers and haptics (tactile feedback). In this paper they suggested a solution based on a combination of voice and haptics. Design research, was chosen as the methodology for the project, highlights the importance of developing a solution over the course of several iterations, and to perform product evaluation using external participants. The research contribution is an Android based prototype that demonstrates the new user interface, allowing the visually impaired to seamlessly interact with a smartphone. Operation relies on voice and haptic feedback, where the user receives information when tapping or dragging the finger across the screen. The proposed solution is unique in several ways, it keeps gestures to a minimum, it does

not rely on physical keys, and it takes the technologies of screen readers and haptics one step further.

K. Matusiak in June 2013 invented Object recognition in mobile phone In this work, they described main features of software modules developed for Android smart phones that are dedicated for the blind users. The main module can recognize and match scanned objects to a database of objects. e.g. food or medicine containers. The two other modules are capable of detecting major colors and locate direction of the maximum brightness regions in the captured scenes. The paper was concluded with a short summary of the tests of the software aiding activities of daily living of a blind user.

OUR WORK

4.1 Design

Our design relates to a small electronic device that can be used to guide any visually impaired and differently-abled person within a building as well as outdoor environment. Here, two simple cameras are used as a part of the computer vision. The stereo-image is obtained from both the cameras together. Both cameras are used for depth analysis and one of them will be used for scene analysis. Here the only used sensor is camera and thus we have reduced the number of individual elements so as to make it more user-friendly and economically feasible.

Here, we have used Raspberry pi-2 microcomputer where all the processings will be done. Along with the microcomputer, the device is mounted with two USB/Bluetooth cameras, Bluetooth earphone and a portable power source are included in the hardware. And in the software part we have applied opencv language as the platform of video processing.

- **Raspberry Pi-2 :**

The Raspberry Pi 2 Model B is the second generation Raspberry Pi. It replaced the original Raspi Model B+erry Pi 1 in February 2015. Compared to the Raspberry Pi 1 it has:

1. A 900MHz quad-core ARM Cortex-A7 CPU
2. 1GB RAM

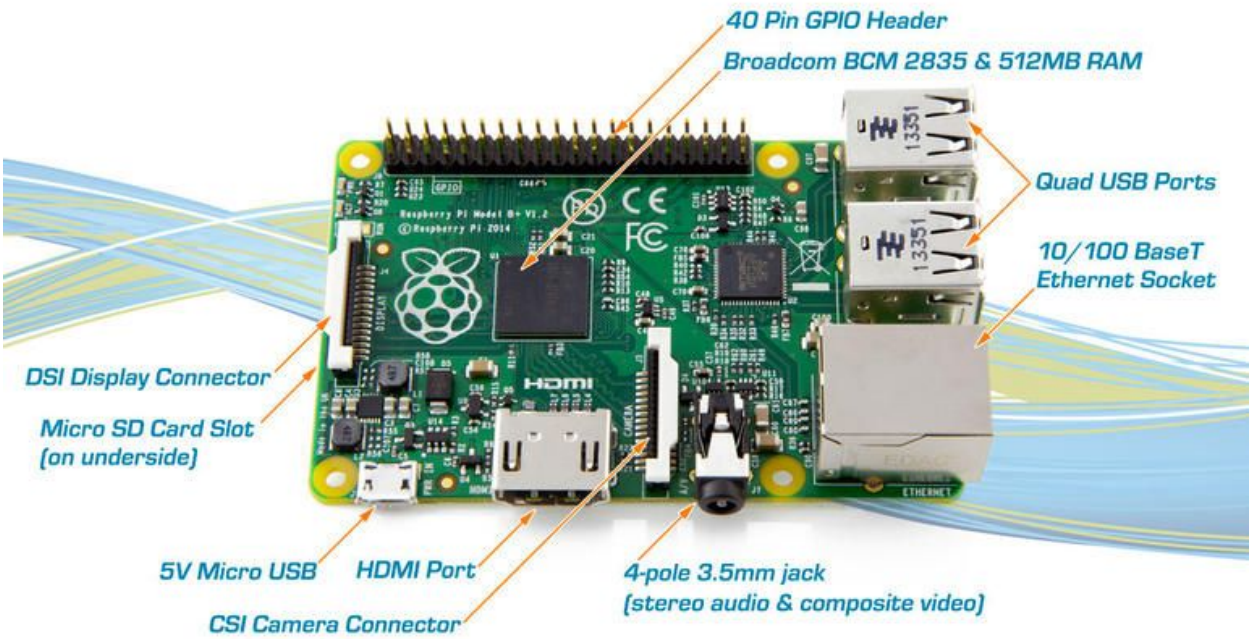
Like the (Pi 1) Model B+, it also has:

1. 4 USB ports
2. 40 GPIO pins
3. Full HDMI port
4. Ethernet port
5. Combined 3.5mm audio jack and composite video
6. Camera interface (CSI)
7. Display interface (DSI)
8. Micro SD card slot
9. VideoCore IV 3D graphics core

Because it has an ARMv7 processor, it can run the full range of ARM GNU/Linux distributions, including Snappy Ubuntu Core, as well as Microsoft Windows 10. The Raspberry Pi 2 has an identical form factor to the previous (Pi 1) Model B+ and has complete compatibility with Raspberry Pi 1. When idle it consumes 1.15W, and when all cores are being utilized power consumption goes up to 4W. Using a 5000mAH battery will last for about 5 hours

.

(Raspberry Pi 2)



(Different Parts of Raspberry Pi-2)

- **USB Cameras :**

USB Cameras are imaging cameras that use USB 2.0 or USB 3.0 technology to transfer image data. USB Cameras are designed to easily interface with dedicated computer systems by using the same USB technology that is found on most computers. The accessibility of USB technology in

computer systems as well as the 480 Mb/s transfer rate of USB 2.0 makes USB Cameras ideal for many imaging applications. An increasing selection of USB 3.0 Cameras is also available with data transfer rates of up to 5 Gb/s.

Here we have used two USB 2.0 2M camera modules with UXGA resolution (1600X1200). Model No.: AH5020B23-S1-2Z1. AH5020B23-S1-2Z1 is an USB Video Class (UVC) compliant camera module with video feature, designed for portable notebook PC image applications. It is made up of the following components, CMOS sensor, lens, holder, backend, PCB, image processing circuit and connector, to come out a digital video device. It shall be a reliable device which is embedded in notebook PC to transfer video data through USB interface to notebook PC.

AH5020B23-S1-2Z1 not only offers up to UXGA resolution (1600X1200) for image applications to take still image, but also offers video stream for end user to preview/record motion image through USB 2.0 interface. It can support VGA (640x480) resolution up to 30 fps at YUY2 mode.

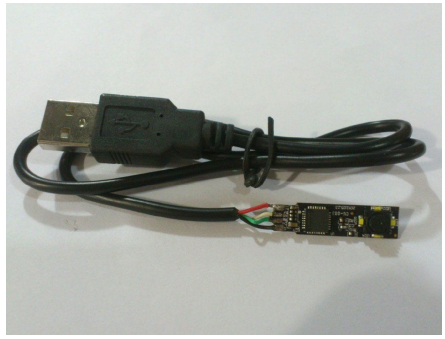
AH5020B23-S1-2Z1 builds in AE, AWB and AGC for automatic image control supported by CMOS sensor. For image quality control, it also offers UVC standardized User Interface (UI) to let end user well tune image by property page.

The features are as follows:

- ➔ Compliant to USB2.0 and USB Video Class
- ➔ Support still image capture and Video Streaming
- ➔ Convert Bayer RGB to YUY2 color space
- ➔ Video Resolution: 2.0M pixel
- ➔ Black Clamping-Gamma Correction
- ➔ Gain and offset adjustment in RGB space
- ➔ Window image statistics collection for AE and AWB
- ➔ Gain and offset adjustment in YUY2 space
- ➔ Input Voltage: standard 3.3V ~ 5V

Its applications are as follows:

- ➔ Notebook PC
- ➔ LCD PC
- ➔ LCD Monitor
- ➔ Industrial PC



(USB Camera Module) (Camera Module with USB cable)

4. Key Specification

Module Specification		
Size(LWH/mm)		60.0 x 8.0 x 4.3±0.2 mm (include PCB Thickness)
PCB Thickeness		0.6 mm
Output Interface		USB 2.0
Image/Video Format		YUY2
Output size	VGA	640x480 (default)
	QQVGA	160x120
	CIF	176x144
	QVGA	320x240
	QCIF	352x288
	SXGA	1280x1024
	UXGA	1600x1200
Video Class Compliant		YES
Operating Temperature		0℃ to +55℃

Power Consumption			
	Min	Type	Max
Input Supply Voltage		5 DC	
Un-configured Current	–	30 mA	–
Operating Current	110 mA	125 mA±5%	140 mA

Power Consumption			
	Min	Type	Max
Input Supply Voltage		3.3 DC	
Un-configured Current	–	20 mA	–
Operating Current	100 mA	115 mA±5%	130 mA

Max Frame Rates (fps)							
	160×120	176×144	320×240	352×288	640×480	1280×1024	1600×1200
YUY2	25 ~ 30	25 ~ 30	25 ~ 30	25 ~ 30	25 ~ 30	5~7	3~4

We can also use bluetooth cameras in place of USB cameras.

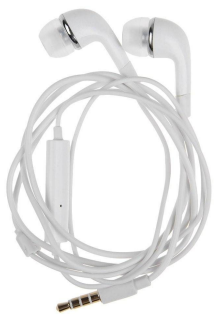
- **Earphones :**

Headphones are a pair of small listening devices that are designed to be worn on or around the head over a user's ears. They are electroacoustic transducers, which convert an electrical signal to a corresponding sound in the user's ear. Headphones are designed to allow a single user to listen to an audio source privately, in contrast to a loudspeaker, which emits sound into the open air, for anyone nearby to hear. Headphones are also known as earspeakers, earphones or, colloquially, cans [40]. Circumaural and supra-aural headphones use a band over the top of the head to hold the speakers in place. The other type, known as earbuds or earphones consist of individual units that plug into the user's ear canal. In the context of telecommunication, a headset is a combination of headphone and microphone. Headphones either connect directly to a signal source such as an audio amplifier, radio, CD player, portable media player, mobile phone, video game consoles, electronic musical instrument, or use wireless technology such as bluetooth or FM radio. Early headphones were first used by radio pioneers (crystal sets) and also by radio telephone and telegraph operators allowing a better audio reception without disturbing others around. Initially the audio quality was mediocre and a step forward was the invention of high fidelity headphones [41].

Headphones are made in a range of different audio reproduction quality capabilities. Headsets designed for telephone use typically cannot reproduce sound with the high fidelity of expensive units designed for music listening by audiophiles. Headphones that use cables typically have either a 1/4 inch (6.35mm) or 1/8 inch (3.5mm) phone jack for plugging the headphones into the audio source. As of 2015, most headphones are amplified by a headphone amplifier, either an integrated amplifier (e.g., in an iPod) or a standalone unit. In the 2010s, headphones are used by people in everyday life to listen to audio material such as recorded music, podcasts, or radio shows. Headphones are also used by people in various professional contexts, such as audio engineers mixing sound for live concerts or sound recordings and DJs, who use headphones to cue up the next song they will play without the audience hearing, aircraft pilots and call center employees. The latter two types of employees use headphones with an integrated micropath.

Here we have used Samsung wired headset HS1303. Its specifications: 3 buttons remote controller (with MIC), Impedance: 32 Ohm, Frequency Response: 20Hz ~ 20kHz, Cable Length: 1.2m, Weight: 11.8g.

We can also use bluetooth earphones in place of wired headsets.



(Earphone)

- **Portable Power Source :**

Powerbanks are mostly popular for charging smartphones and mobile tablet devices. A power bank is a portable device that can supply USB power using stored energy in its built-in batteries. Power banks usually recharge with USB power supply. Basically, powerbanks comprises the rechargeable batteries, consisting of either Lithium-ion or Lithium-Polymer cells and comes under protective casing, guided by sophisticated PCB ensuring various protective and safety measures.

The specifications of a power bank are as follows:

- Capacity in mAh:[42] mAh stands for milliampere hour and measures the amount of power flow that can be supplied by a certain power bank. Amount of mA × time at 5V ideally. Many manufacturers measure this at the voltage of battery inside, hence they show more than actual.
- Simultaneous charging and discharging: need to specify if the power bank can be used while it is charging.
- Number of output USB ports: This specifies the number of devices that can be charged simultaneously.
- Output current rating: This specifies the current rating that it can charge maximum. The higher the number, the better the power bank. This can vary from output port to output port.
- Input Current Rating: Input current rating is the amount of current the power bank is able to draw at its maximum level while getting charged.
- Safety Protections: Over Voltage Protection, Over Charge Protections, Over Current Protections, Overheat Protections, Short-Circuit Protections and Over Discharge Protections are the common safety measures observed with standard power banks.
- LED Indications: The Led glows as per indicating the amount of charging ability left with the power bank.

We have used Intex IT-PB10KW Power bank for our project. Specifications for Intex IT-PB10KW are:

1. Battery Capacity: 10000mAh
2. Battery Type: Lithium-ion
3. Power Source: AC Adapter
4. Connectors: Mini

(Portable Power Source or Power Bank)

- **Software :**

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 7 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

Officially launched in 1999, the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of OpenCV, the goals of the project were described [43] as:

- Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.
- Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.
- Advance vision-based commercial applications by making portable, performance-optimized code available for free—with a license that did not require to be open or free themselves.

OpenCV's application areas include:

➔ 2D and 3D feature toolkits

➔ Egomotion estimation

➔ Facial recognition system

➔ Gesture recognition

➔ Human–computer interaction (HCI)

➔ Mobile robotics

➔ Motion understanding

➔ Object identification

➔ Segmentation and recognition

➔ Stereopsis stereo vision: depth perception from 2 cameras

➔ Structure from motion (SFM)

➔ Motion tracking

➔ Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

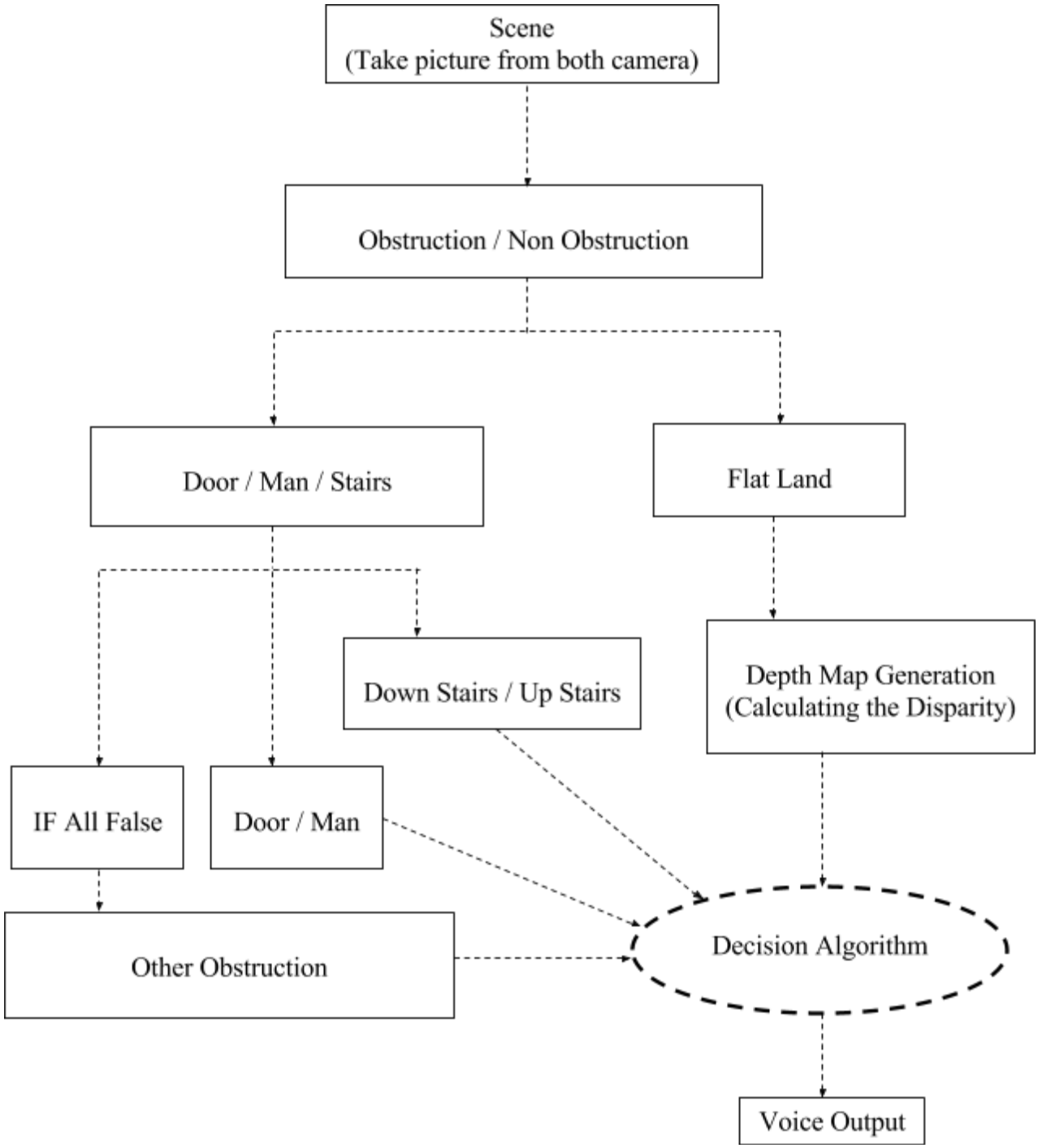
- Boosting
- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)

Programing Language

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation.[44] Wrappers in other languages such as C#, Perl,[45] and Ruby have been developed to encourage adoption by a wider audience.

All of the new developments and algorithms in OpenCV are now developed in the C++ interface.

4.2. Flow Chart



4.3. Algorithms

To start utilizing the benefits of the device one will have to power on the device and let the OS boot up for the first time, Application starts after the boot up process is complete, for the first time software needs to be calibrated according to height of the user so as to identify possible interested regions within a captured frame e.g. identifying amount of ground area to be present for subsequent movement. Once calibrated the hardware works independently and starts taking snaps of the environment continuously and processes the snaps and get to a conclusion whether the path in front is safe or not, if not then finding the direction where a most probable safe path can exist depending on the clearance in front of user. The device doesn't classify all the objects in front but does classify some special cases which seems to play important role in navigation of a visually impaired person e.g. **Doors**, **Staircases, Human** and **Ground**. The feedback provided to the user is based on speech signals. According to previous medical research, for any visually impaired person the ears are the next powerful sense-organs and it is quite reliable to use voice feedback. But keeping ears busy all the time with the speech output will not be a great advantage in hand rather it will interfere with the natural navigation system of human body. Whenever a major change in environment is observed, this information is given to user in the form of speech signals. Major change in environment includes the appearing of a staircase or door, which doesn't allow a natural flow of walk, thus enabling the user to know about the obstacles beforehand.

The flow of the task that is performed is as follows:

1. Acquire Image from the Camera.
2. Send the digital image to the device via USB Cable or Bluetooth.
3. Followed by Object Segmentation.
4. After the classes of objects are segregated out, the next task is to extract the features out of them based on some predefined criteria.
5. The features so extracted form the base for classification of the particular object into Door, Staircase, Human or Open Ground.
6. Once the classification is done a judgment algorithm is run so as to decide whether a major change has been observed in the environment or not. If Observed, user is made aware of the change in the form of a suitable phrase in a language he has selected while installation of the system.
7. The previous steps are followed repeatedly until the device is switched of or the battery has been drained out.

4.3.1. Ground and Obstacle Detection :

The first step of a sequence of algorithms is to detect obstacles from one captured image. All images are processed as grayscale images in this study, because the information quantity can be reduced to one third of color images, and the color information is unlikely to be crucial in planetary environment. Since natural environment is difficult to be recognized by only one criterion, the obstacle detection method described in this section combines two fundamental image processing: edge extraction and thresholding.

The edge of the image represents the outline of obstacles, and the surface and the shadow of obstacles have brighter or darker pixels. After deriving the edge and the surface extraction results, they are integrated in order to compensate errors of one result with the other [46].

The edge of obstacles is extracted by calculating the variance in a small square window which has width of w[pixel]. The variance value V (p, q) at (p, q) on image is namely calculated as follows.

$$V(p,q) = \frac{1}{w^2} \sum_{i=p-\frac{w}{2}}^{p+\frac{w}{2}} \sum_{j=q-\frac{w}{2}}^{q+\frac{w}{2}} (A(p,q) - x(i,j))^2 \quad (1)$$

Where A(p, q) is the average of pixel values in the window, x(i, j) is the pixel value at (i, j) [46].

The variance is calculated on each pixel and then binaries by the threshold Th, which is one of the unique points in the proposed obstacle detection method, given as follows,

$$T_h = \overline{V(p,q)} + t\sigma \quad (2)$$

Where σ is a standard deviation of V (p, q), t is a experimentally decided constant. By this expression, the threshold, which is the key parameter in binarizing process, is decided according to the captured image [46].

This thresholding process expects rover to properly detect obstacle areas even in natural and unknown environment such as planetary surface because the algorithm is robust to the change of light condition. The surface of obstacles is detected by extracting bright and dark pixels. Thresholds are derived in the same way as Eq. (2) in order to secure the robustness. Finally, these results are mixed inclusively, and areas which do not have both edge and surface information simultaneously are rejected, since such areas tend to represent no obstacles but tiny roughness of the surface.

4.3.2. Modified K-means Algorithm :

The primary idea of this algorithm is to detect the peaks in histogram and to employ the corresponding gray-levels as initial cluster centroids for K-means clustering. For the first centroid, the gray-level of the highest peak in the histogram is selected. Then the local maximum with the largest weighted distance to all other known centroids is chosen as a new centroid [47].

For the distance metric, the product of the gray-level difference $d_{n,i}$ between centroid C_n and local maximum i, and the height h_i of the local maximum is used. This process is iteratively repeated until the desired number of centroids is found. The product (instead of the sum) of all weighted distances is used as criterion, since iff. all ($h_i.d_{n,i}$) were large, the product would be maximized [47]. The flowchart is as follows [47]:

1. Calculate the image histogram.
2. Detect all local maxima in the histogram. We assume a total number of I local maxima are detected.
3. Choose the global maximum of the histogram as the first centroid C1.
4. Identify the remaining centroids as follows:

   (a) Calculate

$$P_{N+1}(i) = \prod_{n=1}^{N} (h_i * d_{n,i})$$

   for each local maximum (i = 1, 2, ..., I). N is the number of already found centroids. The computational complexity can be reduced by using a recursive implementation:

$$P_2(i) = h_i * d_{1,i}$$

$$P_{N+1}(i) = P_N(i) * (h_i * d_{N,i})$$

   which employs similar idea of Viterbi algorithm.

   (b) Then the new centroid $C_{N+1} = arg\ max(P_{N+1}(i))$ N=N+1.

5. Repeat 4 until N equals the user-specified number of classes K.

4.3.3. Human Detection :

There are three main contributions of our object detection framework. We will introduce each of these ideas briefly below and then describe them in detail in subsequent sections. The first contribution of this paper is a new image representation called an integral image that allows for very fast feature evaluation. Motivated in part by the work of Papageorgiou et al. our detection

system does not work directly with image intensities [49]. Like these authors we use a set of features which are reminiscent of Haar Basis functions (though we will also use related filters which are more complex than Haar filters). In order to compute these features very rapidly at many scales we introduce the integral image representation for images. The integral image can be computed from an image using a few operations per pixel. Once computed, any one of these Haar-like features can be computed at any scale or location in constant time. The second contribution of this paper is a method for constructing a classifier by selecting a small number of important features using AdaBoost [50]. Within any image subwindow the total number of Haar-like features is very large, far larger than the number of pixels. In order to ensure fast classification, the learning process must exclude a large majority of the available features, and focus on a small set of critical features. Motivated by the work of Tieu and Viola, feature selection is achieved through a simple modification of the AdaBoost procedure: the weak learner is constrained so that each weak classifier returned can depend on only a single feature . As a result each stage of the boosting process, which selects a new weak classifier, can be viewed as a feature selection process. AdaBoost provides an effective learning algorithm and strong bounds on generalization performance [[51], [52], [53]].

The third major contribution of this paper is a method for combining successively more complex classifiers in a cascade structure which dramatically increases the speed of the detector by focusing attention on promising regions of the image. The notion behind focus of attention approaches is that it is often possible to rapidly determine where in an image an object might occur [[54], [55], [56]]. More complex processing is reserved only for these promising regions. The key measure of such an approach is the "false negative" rate of the attentional process. It must be the case that all, or almost all, object instances are selected by the attentional filter.

We will describe a process for training an extremely simple and efficient classifier which can be used as a "supervised" focus of attention operator. The term supervised refers to the fact that the attentional operator is trained to detect examples of a particular class. In the domain of face detection it is possible to achieve fewer than 1% false negatives and 40% false positives using a classifier constructed from two Harr-like features. The effect of this filter is to reduce by over one half the number of locations where the final detector must be evaluated. Those sub-windows which are not rejected by the initial classifier are processed by a sequence of classifiers, each slightly more complex than the last. If any classifier rejects the sub-window, no further processing is performed. The structure of the cascaded detection process is essentially that of a degenerate decision tree, and as such is related to the work of Geman and colleagues [[56], [57]].

This object detection procedure classifies images based on the value of simple features. The most common reason of using features rather than the pixels directly is that features can act to encode ad-hoc domain knowledge that is difficult to learn using a finite quantity of training data.

The simple features used are reminiscent of Haar basis functions which have been used by Papageorgiou et al. [53]. More specifically, we use three kinds of features. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent (see Figure 1).

A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles.

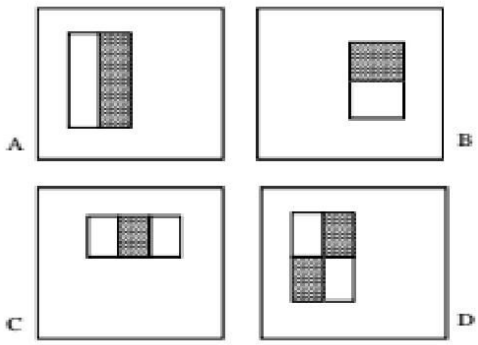


Figure 1: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

Rectangle features can be computed very rapidly using an intermediate representation for the image which we call the integral image. The integral image at location (x,y) contains the sum of the pixels above and to the left of (x,y) inclusive:

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y')$$

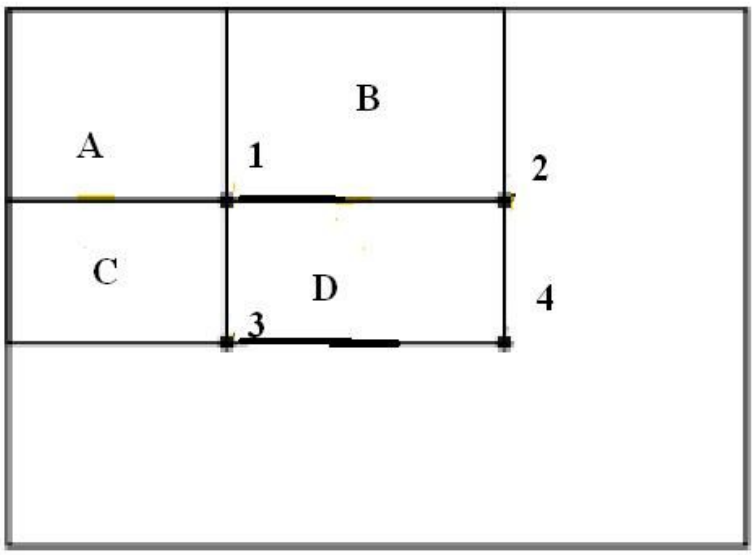


Figure 2: The sum of the pixels within rectangle computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value of location 2 is A+B, at location 3 is A+C, and at location 4 is A+B+C+D. The sum within D can be computed as 4+1-(2+3).

where ii(x,y) is the integral image and i(x,y) is the original image. Using the following pair of recurrences:

$$s(x,y) = s(x,y-1) + i(x,y)$$
$$ii(x,y) = ii(x-1,y) + s(x,y)$$

Where $s(x,y)$ is the cumulative row sum, $s(x,-1) = 0$ and $ii(-1,y) = 0$. The integral image can be computed in one pass over the original image.

Using the integral image any rectangular sum can be computed in four array references (see Figure x). Clearly the difference between two rectangular sums can be computed in eight references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references, eight in the case of the three-rectangle features, and nine for four-rectangle features.

Rectangle features are somewhat primitive when compared with alternatives such as steerable filters [[58], [59]]. Steerable filters, and their relatives, are excellent for the detailed analysis of boundaries, image compression, and texture analysis. In contrast rectangle features, while sensitive to the presence of edges, bars, and other simple image structure, are quite coarse. Unlike steerable filters the only orientations available are vertical, horizontal, and diagonal. The set of rectangle features do however provide a rich image representation which supports effective learning. In conjunction with the integral image , the efficiency of the rectangle feature set provides ample compensation for their limited flexibility.

Given a feature set and a training set of positive and negative images, any number of machine learning approaches could be used to learn a classification function. In our system a variant of AdaBoost is used both to select a small set of features and train the classifier [50]. In its original form, the AdaBoost learning algorithm is used to boost the classification performance of a simple (sometimes called weak) learning algorithm. There are a number of formal guarantees provided by the AdaBoost learning procedure. Freund and Schapire proved that the training error of the strong classifier approaches zero exponentially in the number of rounds. More importantly a number of results were later proved about generalization performance [60]. The key insight is that generalization performance is related to the margin of the examples, and that AdaBoost achieves large margins rapidly. The AdaBoost algorithm for classifier learning is as follows:

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For t = 1, … , T

1. Normalize the weights,

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

So that w_t is a probability distribution.

2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\varepsilon_j = \sum_i w_i |h_j(x_i) - y_i|$

3. Choose the classifier, h_t , with the lowest error ε_t.

4. Update the weights:

$$w_{t+1,i} = w_{t,i}\beta_t^{1-\varepsilon_i}$$

Where $\varepsilon_i = 0$ if example x_i is classified correctly, $\varepsilon_i = 1$ otherwise, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$.

- The final strong classifier is :

$$h(x) = 1 \quad \sum_{t=1}^{T} \alpha_t h_t(x) \geq \sum_{t=1}^{T} \alpha_t$$

$$0 \quad \text{otherwise}$$

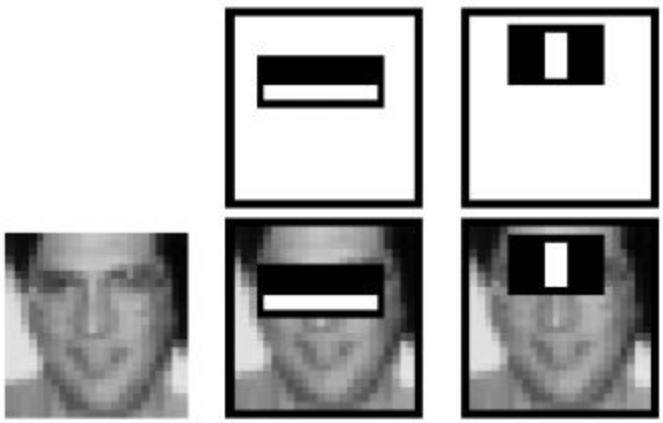


Figure: The first and second features selected by AdaBoost. The two features are shown in the top row and then overlaid on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

The overall form of the detection process is that of a degenerate decision tree, what we call a “cascade”. A positive result from the first classifier triggers the evaluation of a second classifier which has also been adjusted to achieve very high detection rates. A positive result from the second classifier triggers a third classifier, and so on. A negative outcome at any point leads to the immediate rejection of the sub-window.

Stages in the cascade are constructed by training classifiers using AdaBoost and then adjusting the threshold to minimize false negatives. Note that the default AdaBoost threshold is designed to yield a low error rate on the training data. In general a lower threshold yields higher detection rates and higher false positive rates.

Computation of the two feature classifier amounts to about 60 microprocessor instructions. It seems hard to imagine that any simpler filter could achieve higher rejection rates. By comparison, scanning a simple image template, or a single layer perceptron, would require at least 20 times as many operations per sub-window.

The structure of the cascade reflects the fact that within any single image an overwhelming majority of subwindows are negative. As such, the cascade attempts to reject as many negatives as possible at the earliest stage possible. While a positive instance will trigger the evaluation of every classifier in the cascade, this is an exceedingly rare event.

Much like a decision tree, subsequent classifiers are trained using those examples which pass through all the previous stages. As a result, the second classifier faces a more difficult task than the first. The examples which make it through the first stage are “harder” than typical examples. The more difficult examples faced by deeper classi- fiers push the entire receiver operating characteristic (ROC) curve downward. At a given detection rate, deeper classi- fiers have correspondingly higher false positive rates.

The cascade training process involves two types of tradeoffs. In most cases classifiers with more features will achieve higher detection rates and lower false positive rates. At the same time classifiers with more features require more time to compute. In principle one could define an optimization framework in which:

i) the number of classifier stages,

ii) the number of features in each stage, and

iii) the threshold of each stage,

are traded off in order to minimize the expected number of evaluated features. Unfortunately finding this optimum is a tremendously difficult problem.

In practice a very simple framework is used to produce an effective classifier which is highly efficient. Each stage in the cascade reduces the false positive rate and decreases the detection rate. A target is selected for the minimum reduction in false positives and the maximum decrease in detection. Each stage is trained by adding features until the target detection and false positives rates are met (these rates are determined by testing the detector on a validation set). Stages are added until the overall target for false positive and detection rate is met.

The complete face detection cascade has 38 stages with over 6000 features. Nevertheless the cascade structure results in fast average detection times. On a difficult dataset, containing 507 faces and 75 million sub-windows, faces are detected using an average of 10 feature evaluations per

subwindow. In comparison, this system is about 15 times faster than an implementation of the detection system constructed by Rowley et al. [61]

4.3.4. Door Detection :

The method is based on Canny edge detection algorithm and the Hough transform, and uses fuzzy logic to determine the probability of existence of doors which fit a set of predefined rules. The implementation of the algorithm is composed of a few core steps: region of interest (ROI) preprocessing, door line extraction, and door classification.

4.3.4.1 ROI Preprocessor

The preprocessing steps locate regions of interest (ROI) that are likely to contain doors. Door corners are likely among the strongest corners in the image; however, regular corner detection not only detects the corners of the door, but also any other strong corners in the image. To differentiate the cases, the preprocessing algorithm takes into account that the corners for doors are near the intersection of strong horizontal and vertical lines. This sacrifices some degree of rotational invariance, but significantly reduces the number of undesirable corners.

The image is converted to grayscale and cropped slightly to remove border effects. Canny edge detection is then performed, and the resulting image is dilated with a 3x3 cross structuring element to improve connections at the corners. Dilation with horizontal and vertical line structuring element is performed before corners are computed. Region counting is then performed and each line segment is assigned a weighting based on the region size.

Next the corners are determined by multiplying the squares of the horizontal and vertical segment images similar to as would be performed with a Harris corner detector with a sensitivity parameter equal to zero. It can be noticed that the corners of the door are discernibly the strongest, especially near the top where the horizontal segment is clearer in the image.

After corner detection is performed, multiple sets of three corners with an implicit fourth corner are used to generate quadrilateral regions where a door would likely be. During the initial search, the interior angles of the polygon must be within a tolerance comparable to those of a door viewed with a slightly rotated camera and have a height to width ratio which is acceptable. Next, the quadrilateral regions are each evaluated based on overall area and independently on a shape metric. Particular emphasis is given to quadrilaterals which could be described as rectangles and/or parallelograms and those which match the appropriate 2.2:1 standard for U.S. doors.

The quadrilaterals selected based on area are merged into one or more quadrilaterals of maximum extent. The quadrilaterals selected based on shape are evaluated for overall connection in the edge image. Each remaining quadrilateral region is converted to a rectangular region, and each region is cropped from the original image, padded, and sent to the line and feature extraction algorithm. If the selected region is small, the image is upsampled before it is sent.

The ROI preprocessor increases the robustness of the door detection algorithm, and, depending on the image and parameters, can speed up detection. The detection rate increase is notable especially for images with small doors.

4.3.4.2. Door Line and Feature Extraction

With the ROI preprocessor removing most of the context around the door, the remaining image is grayscale and processed to extract dominant line features which are put into a line database.

The database extraction is a multistep process. First, the input image is converted into an edge map using Canny edge detection, with a very low fixed threshold.

Morphological erosion with 3x1 pixel line structuring elements is used to extract horizontal and vertical lines from the edges.

Region counting is applied to remove small vertical and horizontal lines from the respective images. Only regions of sufficient size are retained. A Hough transform is applied to each horizontal and vertical line. The theta and rho values are stored, along with the line segment locations, in a database. The final line database consists of horizontal and vertical lines that meet a length criterion and an angle criterion.

8-level Otsu thresholding is applied to the original image passed by the ROI preprocessor. Note that the Otsu thresholding function was leveraged from [63]. Each Otsu level is turned into a binary image with all levels below the current level set to 0 and all levels equal to or larger than the current level set to 1. Each binary image is region counted looking for features that are the correct size for hinges or door knobs.

The hinge locations are stored in a database, while the door knobs are stored in a separate database. The result is then passed onto an algorithm utilizing fuzzy logic to calculate confidence.

4.3.4.3. Fuzzy Logic to Extract Doors

A heuristic method is used to extract parallel lines from the line database. The parallel lines are used to find the four corners of the door.

The heuristic method employs three metrics to determine the likelihood of line segments being part of a door. One metric is assigned to vertical line pairs, the second metric is assigned to the top horizontal line, and the third metric is assigned to the bottom horizontal line. High metric values indicate more confidence that the line combination constitutes a door. The steps used to calculate vertical line metrics are as follows:

Calculate the mean length of each pair of vertical line segments. If the distance between the pair of lines is greater than 1/3rd and less than 2/3rd of the mean length, the pair is considered further.

The difference between the two vertical lines is subtracted from the metric. If the distance between the vertical lines multiplied by 2.2 is close to the length of the vertical line pair, the metric is increased. Vertical line pairs with a metric less than 50% of the highest vertical line metric are removed from consideration.

Once the vertical line pairs are scored, the fuzzy logic scores the horizontal lines with respect to each pair of vertical lines still under consideration. The average top and bottom Y coordinates of vertical lines are calculated. Each horizontal line has a metric calculated based on how close the average Y coordinate of the horizontal line matches the top or bottom average Y values of the vertical line pair. The lines with the highest metric are stored as the top and bottom that match with the vertical line pair. Horizontal lines that do not span the entire distance between the two vertical lines score a lower horizontal metric.

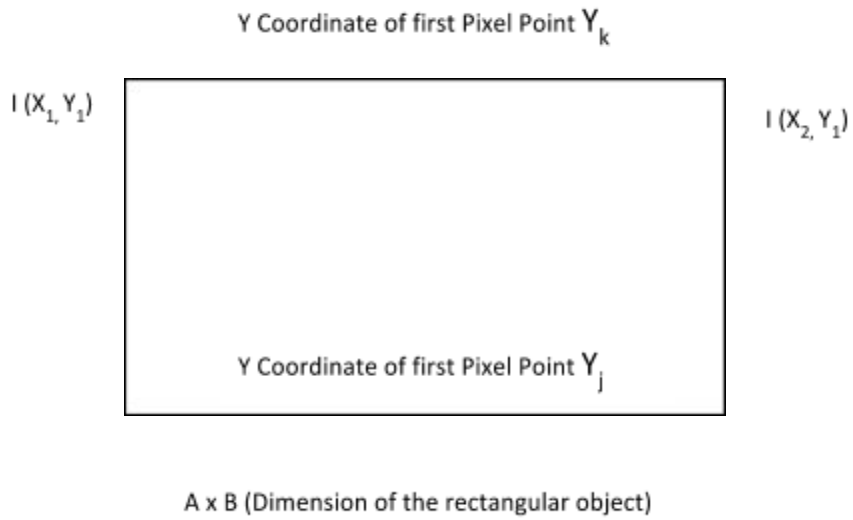
Once all three base metrics are calculated, the four lines constituting the door are passed to the door feature detection section of the fuzzy logic. The door feature detection section uses the location of the door as described by the four lines to search for features which improve the confidence score. The door feature code works as follows. A door handle near either vertical line in the second quadrant from the bottom of the door adds half the difference between the vertical line score and the top and bottom horizontal line scores independently. Hinges near either vertical line add a sixth to the difference between the vertical line score and top and bottom horizontal line scores independently. The door feature detection helps significantly with close door detection, which is defined as doors closer than 6 ft. This is because the top or bottom of the door is commonly missing from these images. However, door handles and hinges become more easily identifiable with thresholding as these features are larger in close door pictures than other picture types.

The feature metric values are summed with the original vertical and horizontal door metrics to create the final door confidence metric. These final values are sorted from highest to lowest. The last section of door confidence is determining if the metrics constitute a door, and how many. The highest ranked combined metric, is indicative of the existence of a door, if its horizontal metrics are at least 1/4th of the vertical line metric. In addition, all lower confidence doors are thrown away if their area overlaps 50% or more of the higher scored rectangle. If the top few candidates have close metrics and their areas do not overlap, it is indicative of multiple doors in the image.

The overall algorithm of sensing a door is as follows:

1. Obtain the High Pass Filtered image such that the edges are obtained.
2. Count the Number of White colored pixels and Black Colored Pixels.
3. Now for the Object of interest is generated based on minimum variance so obtained.

4. Apply Vertical Hough Transform to the image.
5. Set the visited array as 0
6. Move along the image towards right and check if the transition occurs from Black pixel to White pixel, If occurs mark the first visited node. Again check for the transition White pixel to black, If occurs mark the second visited node.
7. If both Visited node is marked then sum up the difference between the visited node x coordinates
8. Repeat the step 5 and 6 until the condition in 6 satisfies or the image frame ends.
9. Generate the mean.
10. Repeat Step 4, 5, 6 to obtain the minimum square error for the object.
11. (Mean Square Error / Mean), White Pixel Count forms the feature set for door detection.



$$\text{Mean(M)} = \frac{\sum_{i=1,B}(X_{2i}-X_{1i})}{|Y_k-Y_j|}$$

$$\text{Mean Square Error(MSE)} = \frac{\sum(M-(X_{2i}-X_{1i}))^2}{|Y_k-Y_j|}$$

4.3.5. Stair Detection:

The following algorithm of sensing the presence of stair is

1. Obtain the High pass Filter Image.

   A high pass filter is the basis for most sharpening methods. An image is sharpened when contrast is enhanced between adjoining areas with little variation in brightness or darkness .

   These filters emphasize fine details in the image – exactly the opposite of the low-pass filter. High-pass filtering works in exactly the same way as low-pass filtering; it just uses a different convolution kernel.

2. Apply Horizontal Hough Transform to the image.
3. Scan the image through the middle of the frame with a pre-defined kernel of dimension 4x3.
4. Generate two variable $Temp_{Var_1}$ & $Temp_{Var_2}$. First one gives the Correlation with the kernel middle values, second one gives correlation with the entire kernel at each step.
5. If $Temp_{Var_1} = 6$ & $5 < Temp_{Var_2} < 7$ then increment the $Stair_{Count}$.
6. If $Stair_{Count} > 1$ then scan again through the middle of the frame and obtain the coordinates of the point of interest for kernel.
7. Get the difference between consecutive coordinate values and store them as the step height.
8. Get the Mean Step Height by using ($\frac{Step_{Height}}{Stair_{Count}}$) and Mode Step Height ($Step_{Height}$ mostly occurring).
9. If Mean Step Height deviates largely from Mode Step Height then it's not a Stair otherwise it is Stair.

Kernel Used:

0	0	0	0
1	1	1	1
1	1	1	1
0	0	0	0

4.3.6. Hough Transform :

The Hough Transform is an algorithm presented by Paul Hough in 1962 for the detection of features of a particular shape like lines or circles in digitalized images. In its classical form it was restricted to features that can be specified in a parametric form. However, a generalized version of the algorithm exists which can also be applied to features with no simple analytic form, but it is very complex in terms of computation time.

Although Hough Transform is a standard algorithm for line or circle detection it has weak points, especially its computational complexity. However a faster version called Fast Hough Transform has been developed.

Classical Hough Transform

The classical Hough Transform is a standard algorithm for line and circle detection. It can be applied to many computer vision problems as most images contain feature boundaries which can be described by regular curves. The main advantage of the Hough transform technique is that it is tolerant of gaps in feature boundary descriptions and is relatively unaffected by image noise, unlike edge detectors.

Line Detection

The simplest case of Hough transform is the linear transform for detecting straight lines. In the image space, the straight line can be described as $y = mx + b$ and can be graphically plotted for each pair of image points (x, y). In the Hough transform, a main idea is to consider the characteristics of the straight line not as image points x or y, but in terms of its parameters, here the slope parameter m and the intercept parameter b.

We incorporate the results in an Accumulator Array. At the beginning, the Accumulator Array is initialized to zero for all cells. After that, for every value of a with a corresponding value of b, the value of that cell is incremented by one.

However, this method has its drawbacks. If the line is horizontal, then a is 0, and if the line is vertical, then a is infinite. So, a more general representation of a line will be $\rho = x \cdot cos\ \theta + y \cdot sin\ \theta$.

Circle and Ellipse Detection

As stated before the Hough transform can also be used for detection of circles and other parameterizable geometric figures. In theory any kind of curve can be detected if you can express it as a function of the form

$$f(a_1, a_2, ..., a_n, x, y) = 0$$

For example, a circle can be represented as

$$(x - a)^2 + (y - b)^2 - r^2 = 0$$

Then we have a n dimensional parameter space (three dimensional space for a circle). This model has three parameters: two parameters for the centre of the circle and one parameter for the radius of the circle. For ellipses and other parametric objects the algorithm is quite similar, but the computation complexity (dimension of the Hough space) increases with the number of the variables.

Advantages of Hough Transform

One important difference between the Hough Transform and other approaches is resistance of the former to noise in the image and its tolerance towards holes in the boundary line. Figures 3.1 and 3.2 compare the Hough transform of a plain straight line with a dotted one. As can be seen there is almost no differences in the results.

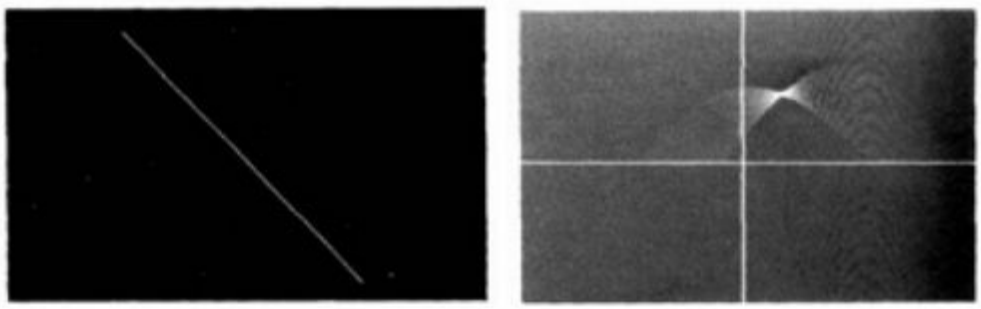


Figure 3.1: *Straight line and its Hough transform.*

Disadvantages of Hough Transform

Since the Hough Transform is a kind of Brute-Force method it is very complex in computation. It has two main drawbacks: large memory requirement and slowness. In order to find the plane parameters accurately, parameter space must be divided finely in all three directions, and an accumulator assigned to each block. Also it takes a long time to fill the accumulators when there are so many.

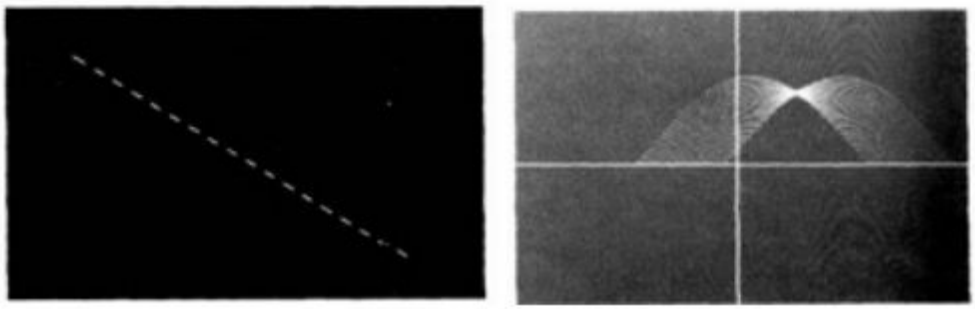
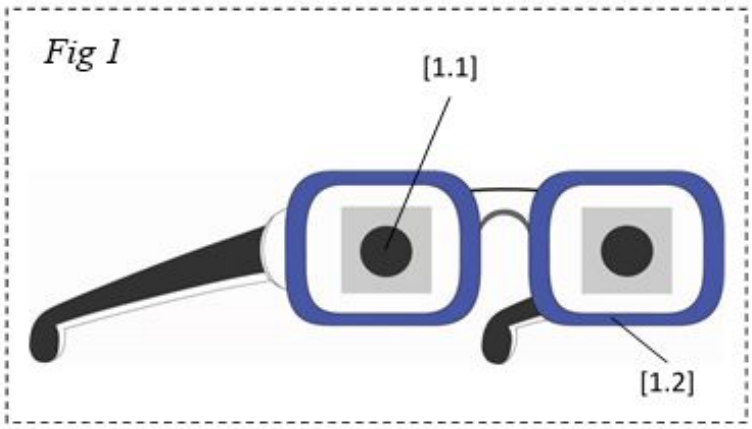


Figure 3.2: *Straight dashed line and its Hough transform.*

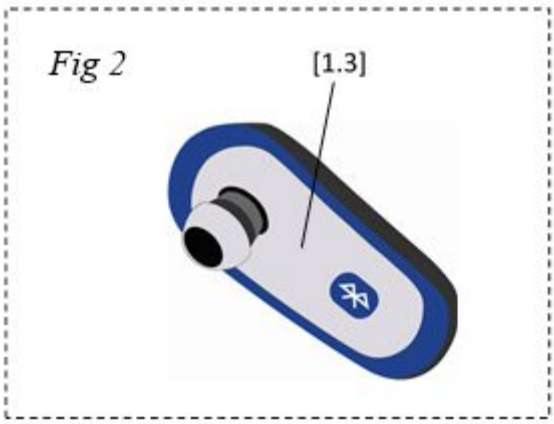
The Fast Hough Transform gives considerable speed up and reduces memory requirement. Instead of dividing parameter space uniformly into blocks, the FHT homes in on the solution, ignoring areas in parameter space relatively devoid of votes. The relative speed advantage of the FHT increases for higher dimensional parameter spaces.

Another weakness of the Hough Transform is that it often recognises many similar lines instead of the one correct one. The algorithm only returns a line without a starting and ending point. For that different postprocessing algorithms have to be used.

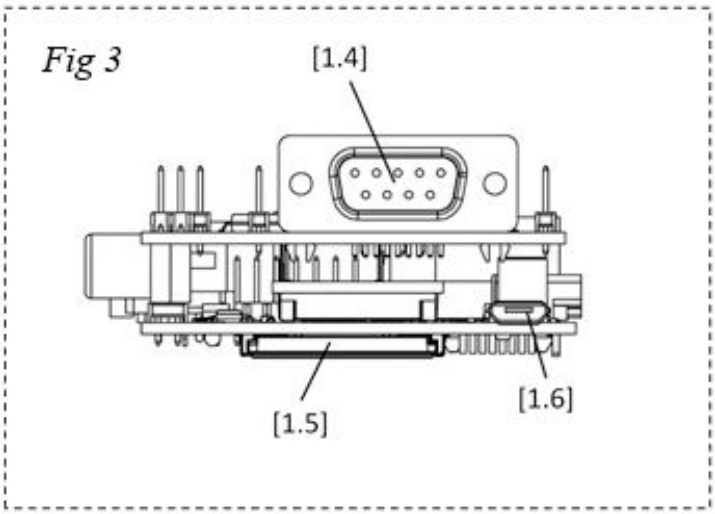
4.4 Hardware Diagrams:



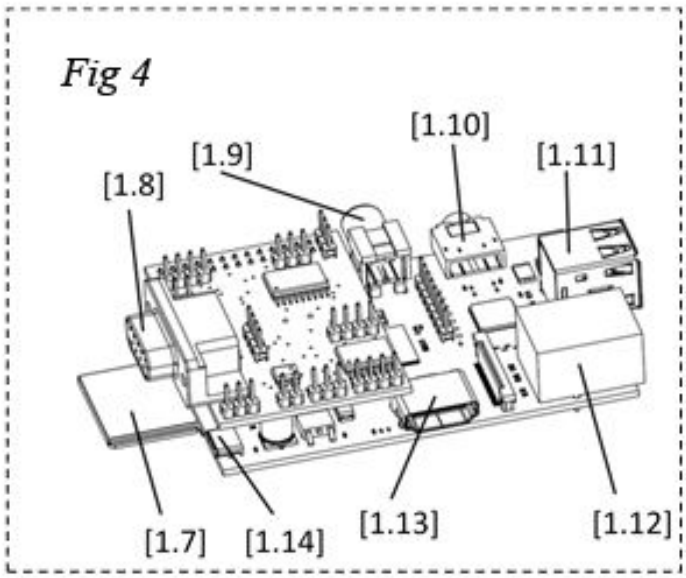
USB Camera for getting input image frames



Bluetooth audio device for audio feedback

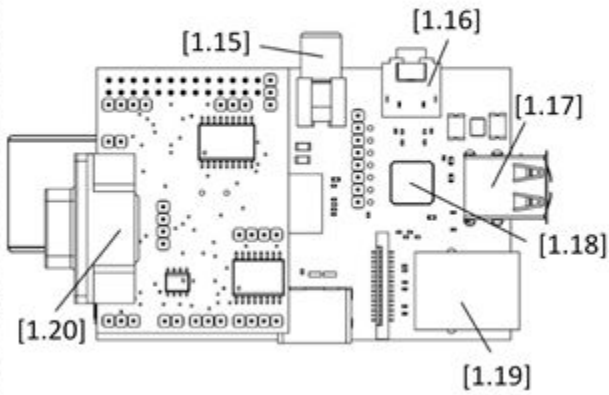


Lateral view showing VGA port and SD-Card slot



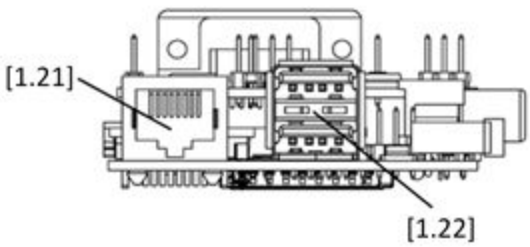
Lateral view showing major peripherals

Fig 5



Top View showing the peripherals

Fig 6

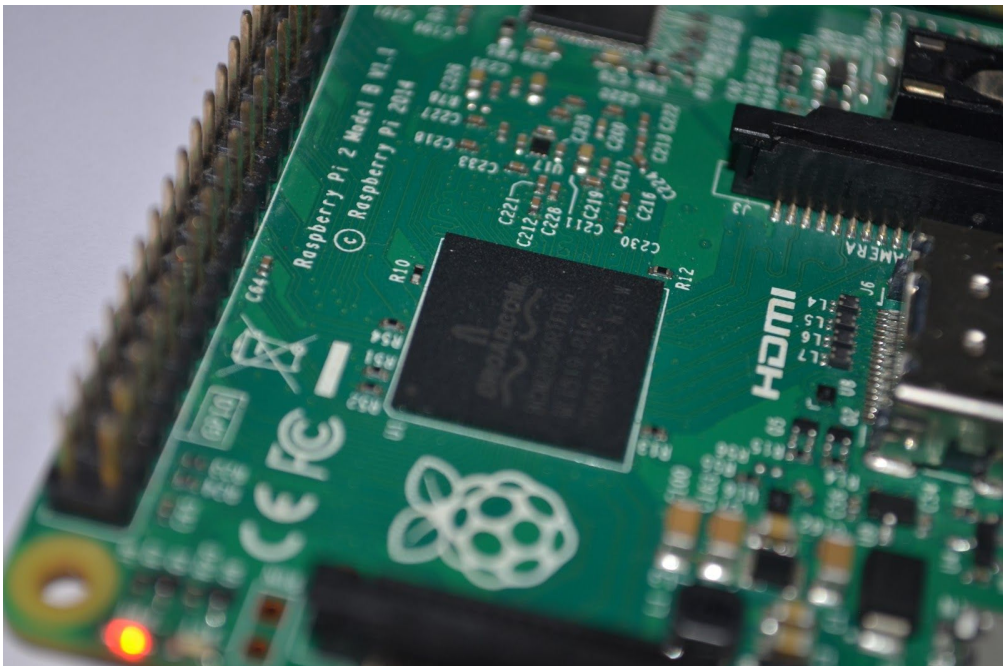


Lateral view showing Ethernet and USB ports

4.5 Actual Hardware Drawings:

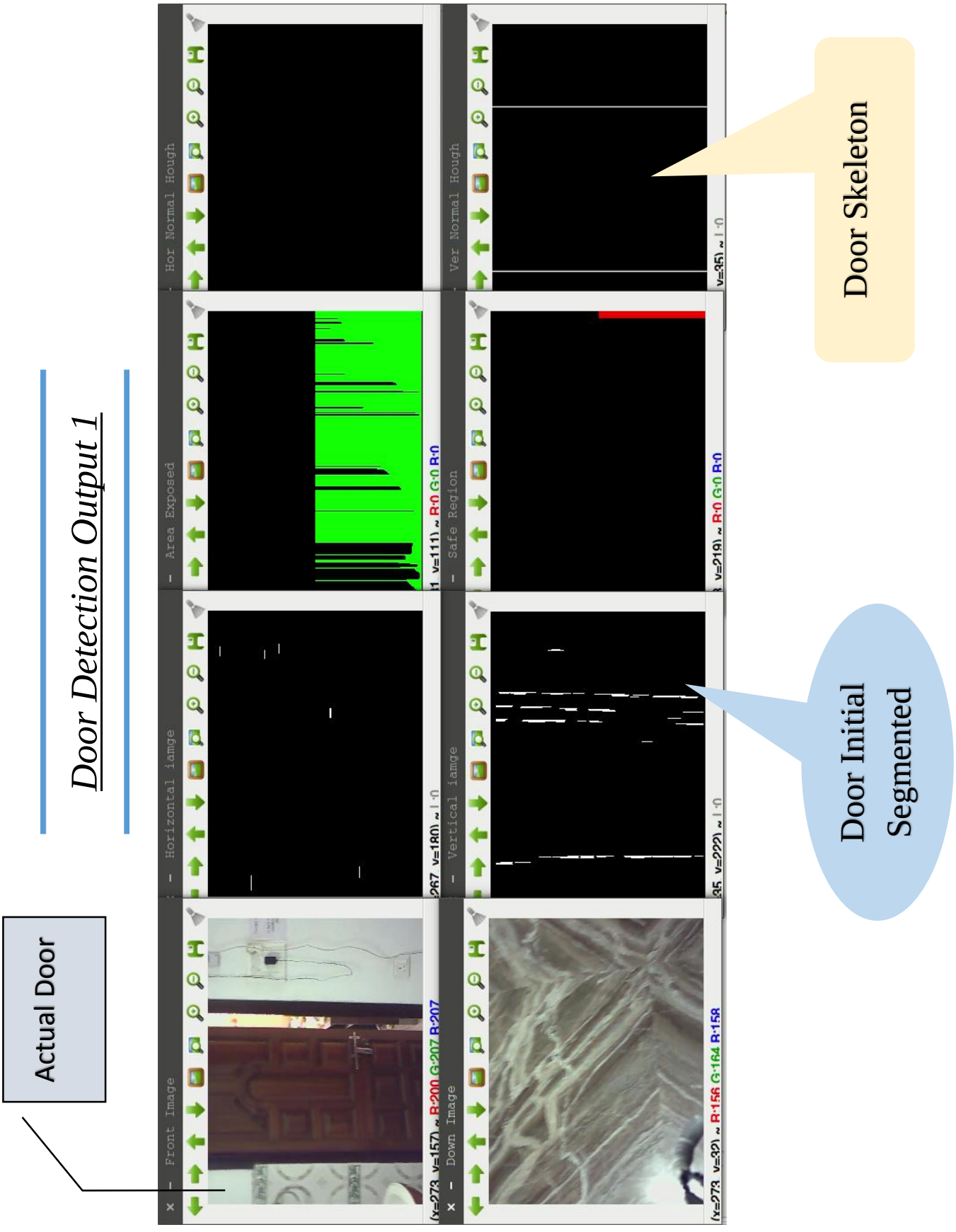
Actual head mount system

Head Mount Device with the wire output

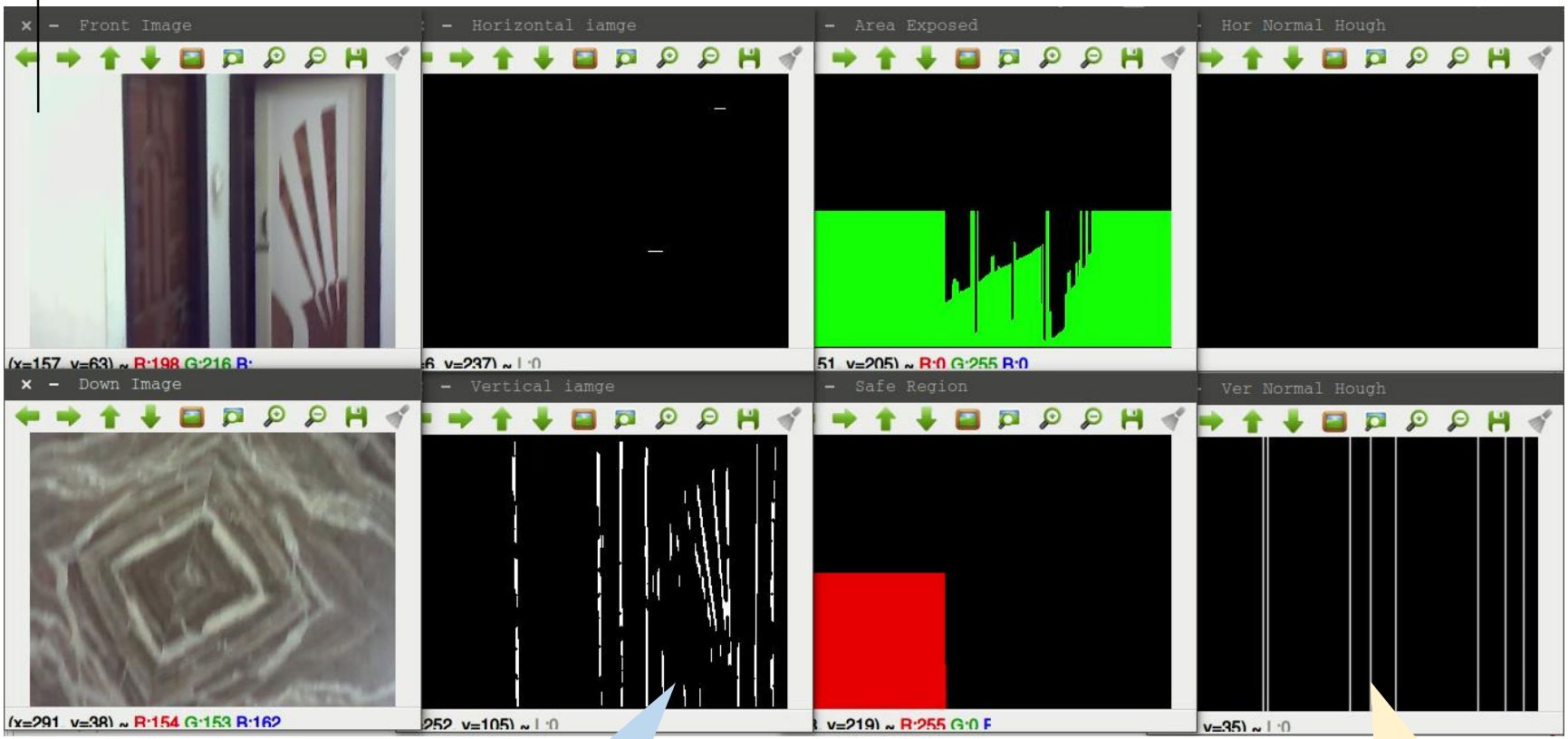


Raspberry pi 2 Development board

4.6 Output:

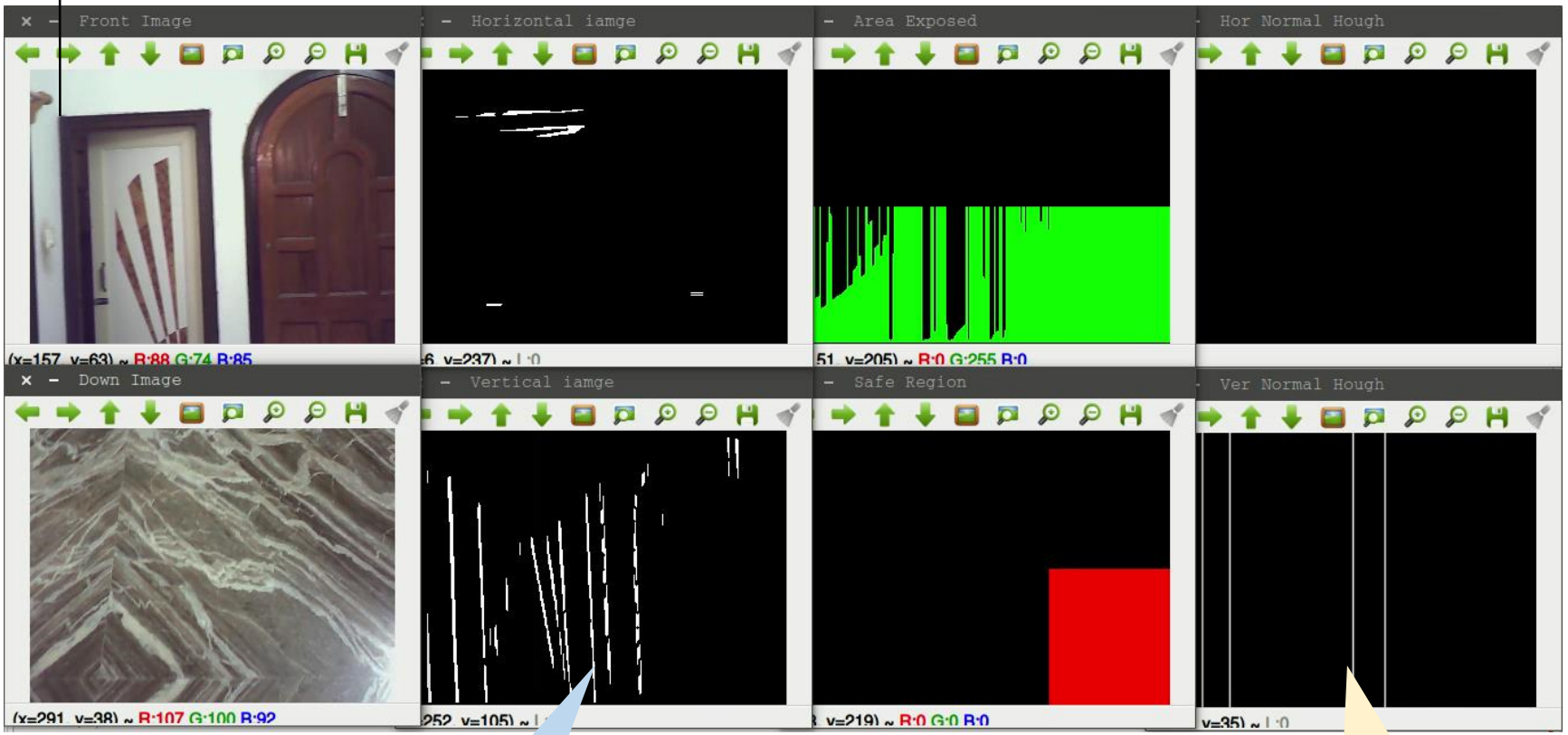

Door Detection Output 1
Actual Door
Door Initial Segmented
Door Skeleton
Front Image
Horizontal iamge
Area Exposed
Hor Normal Hough
Down Image
Vertical iamge
Safe Region
Ver Normal Hough

Door Detection Output 2



Actual Door

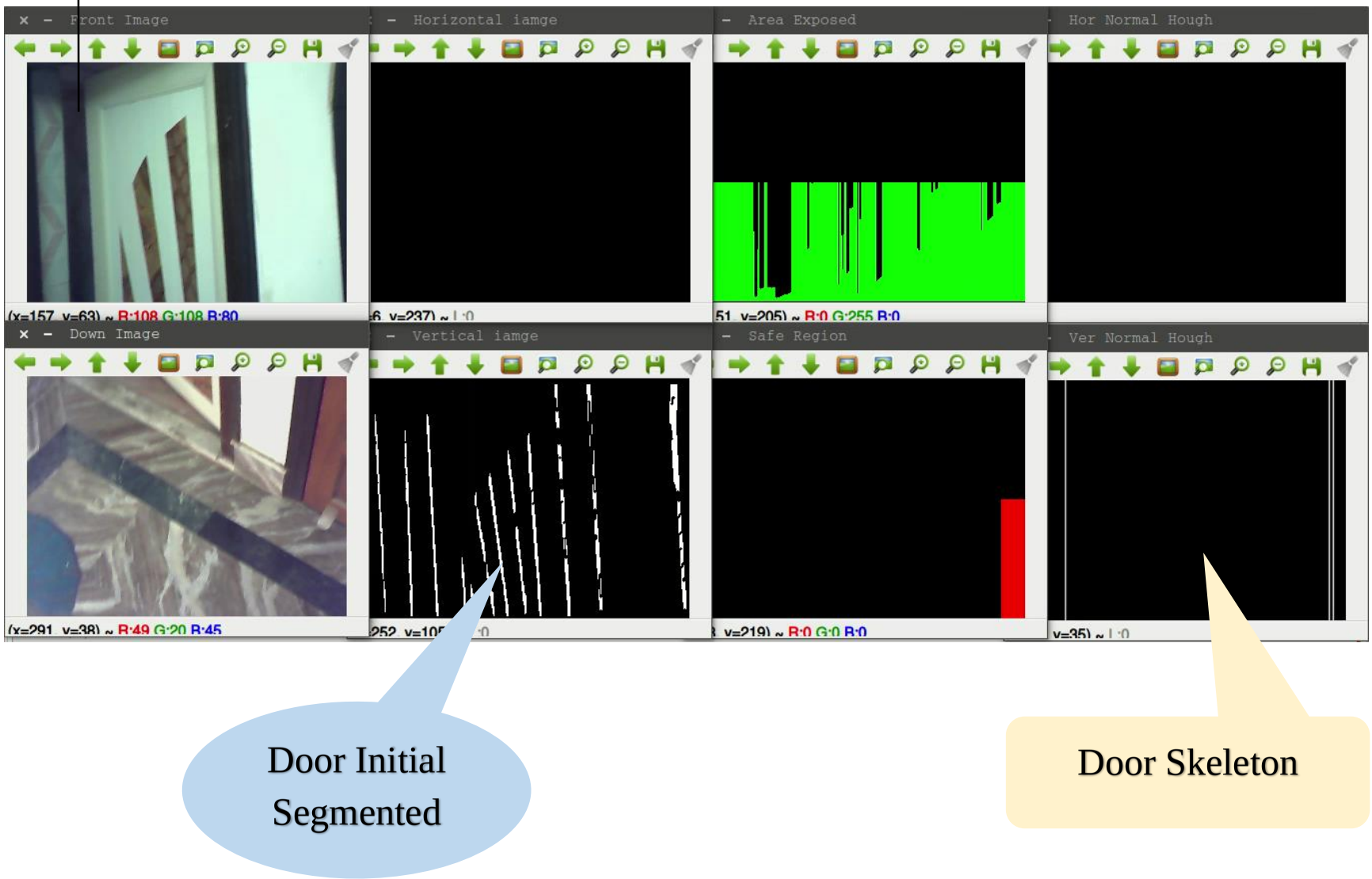
Door Detection Output 3



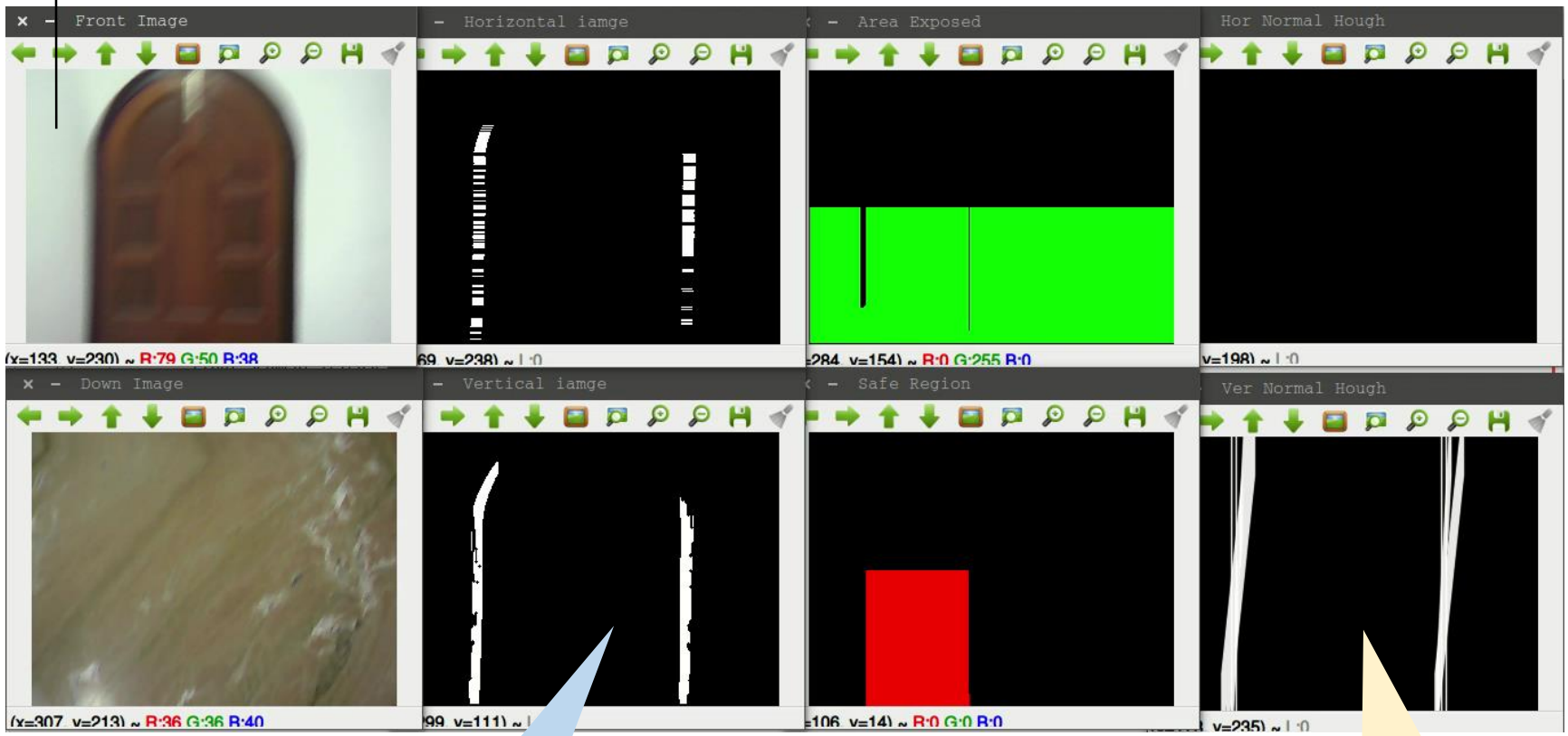
Door Initial Segmented

Door Skeleton

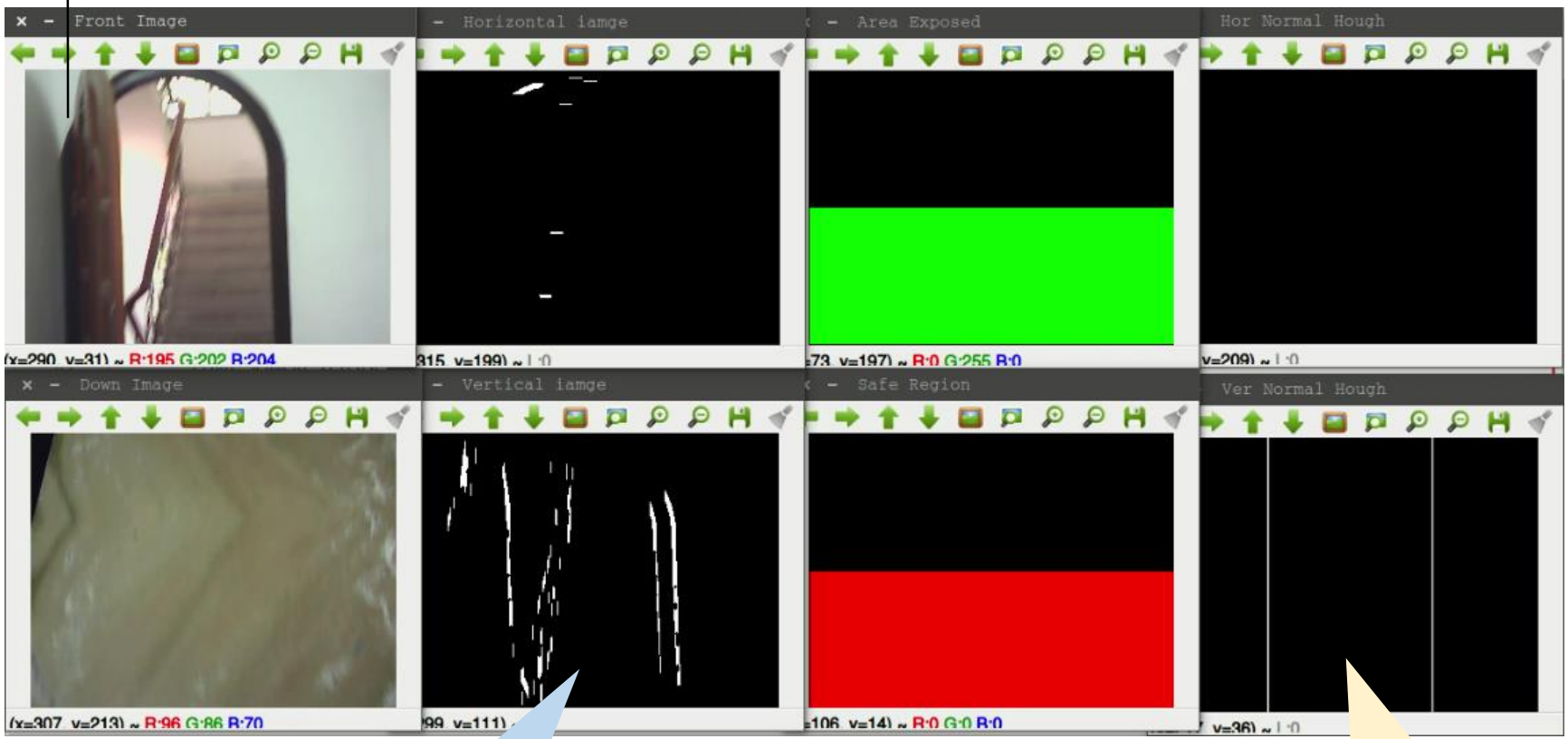
Door Detection Output 4



Door Detection Output 5

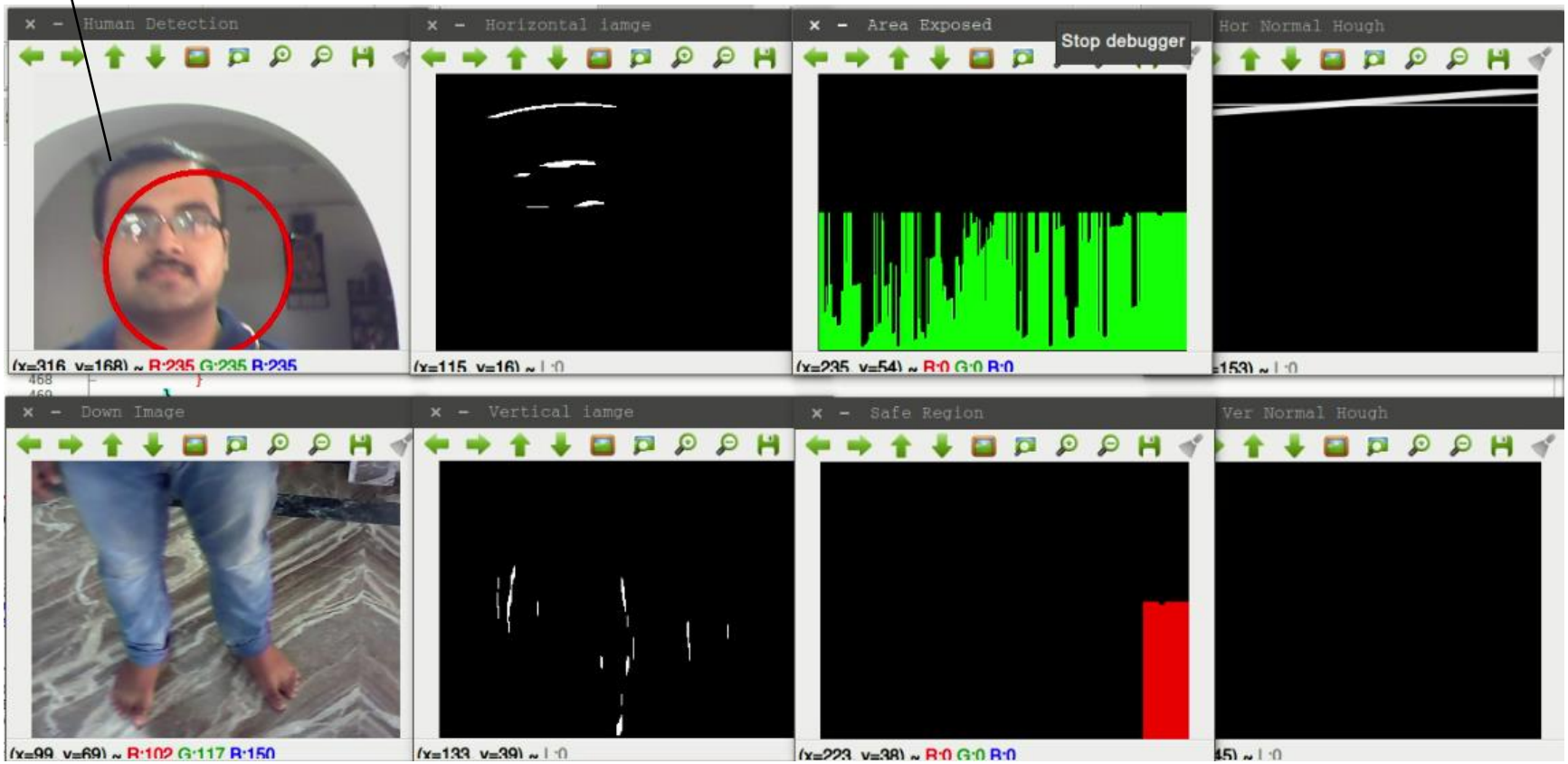


Door Detection Output 6

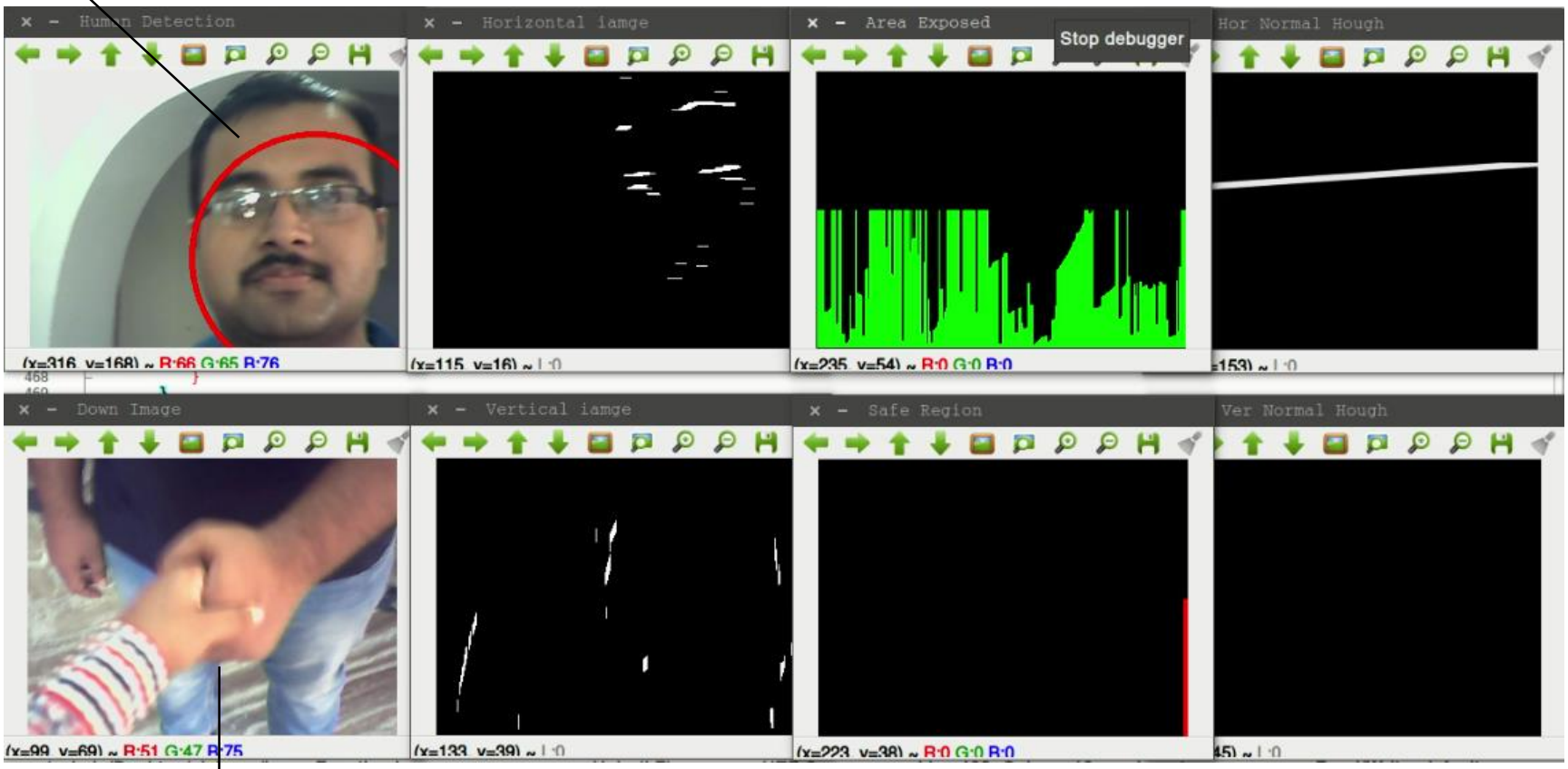


Human Detected

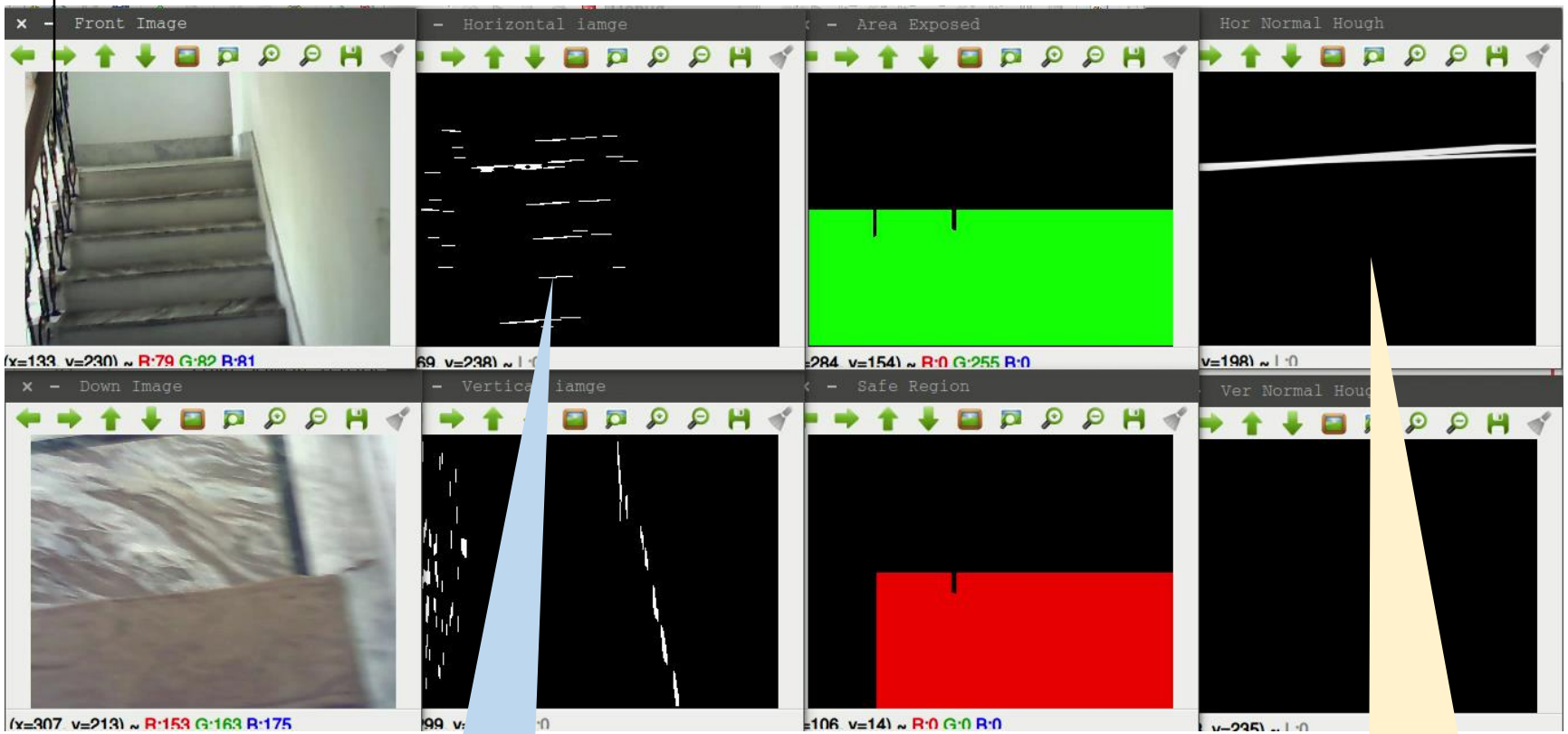
Human Detection Output 1



Human Detection Output 2

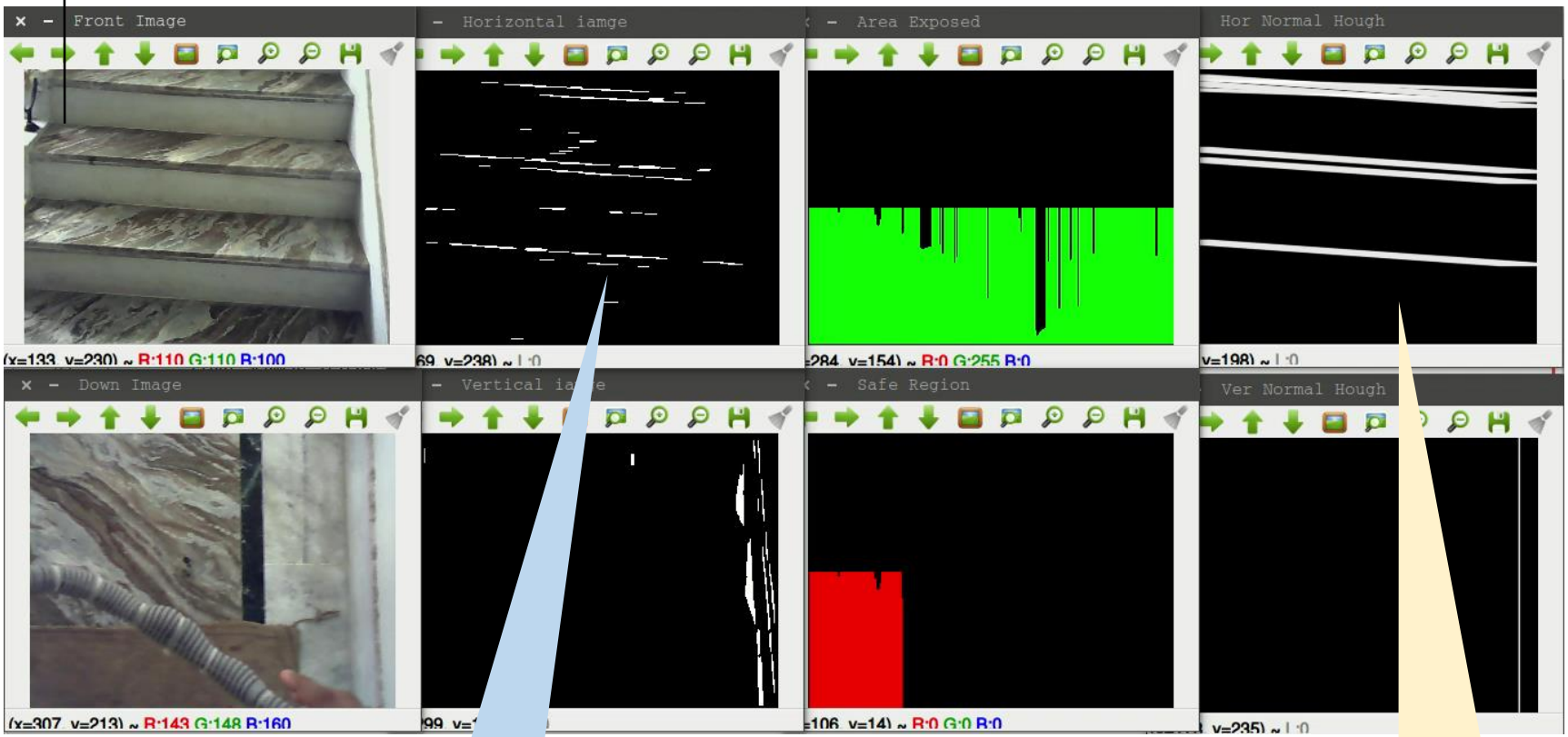


Stair Detection Output 1



Actual Stair

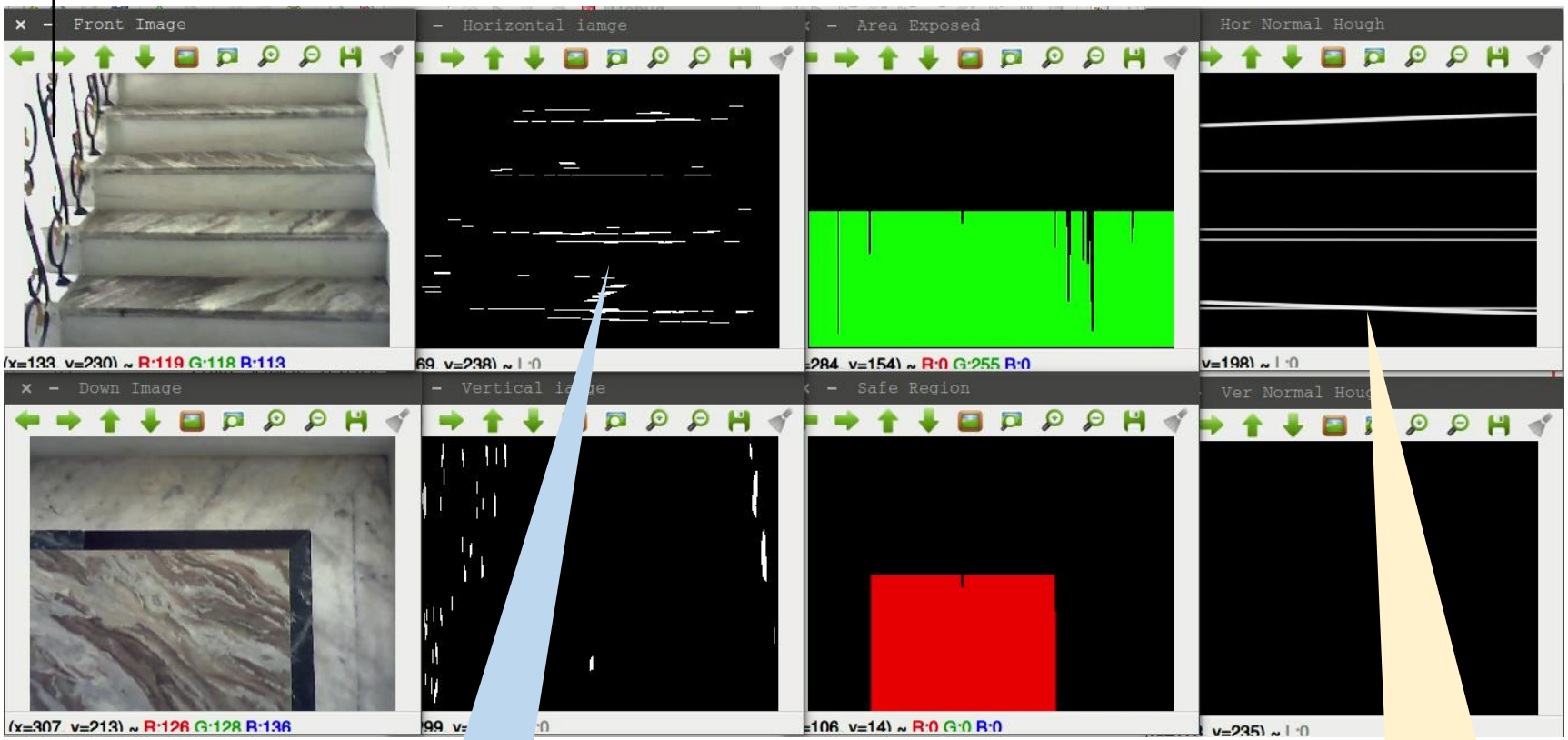
Stair Detection Output 2



Stair Initial Segmented

Stair Skeleton

Stair Detection Output 3



CONCLUSION

The main advantage of this project is that it is a standalone package and platform independent. The software is compatible with any Debian version OS.

Further modifications for future implementation are Android Based product for more accessibility and portability than the current version, as well as implementation in a wheelchair for usage in hospitals where at least one nurse has to be present with each patient to move him in wards. So our invention will be helpful in a way that the source and destination can be fed by the doctor personnel and the wheelchair will certainly take the patient to correct destination. Our claims in this project are:

- This is a navigation system having two webcams mounted on spectacles, an ARM processor development board, a simple wireless earphone and a power source.
- It can be implemented in android platform using the processor of the smartphone as it is also an ARM processor, so we do not need to carry any extra processor with the spectacles.
- It is independent of GPS (Global Positioning System) and Internet. The invention is free from network interference.
- Only two simple Cameras are used for capturing image to be processed further later.
- The Invention is economically feasible.
- No other Sensor is used other than an imaging device, i.e. camera.
- The invention can detect any kind of obstacles in its way.
- It can easily detect doors and staircases, both upwards and downwards and human beings.
- The program Generates speech signals as output which are fed out to the user via earphones.
- The program is a Guided navigation system.
- The invention can be used for robotics navigation.
- It is power efficient. When idle it consumes 1.15W, and when all cores are being utilized power consumption goes up to 4W. Using a 5000mAH battery will last for about 5 hours.
- It is pollution free.
- The process of communication of the output to the user is Painless and user friendly.
- The device works efficiently in almost all light conditions, i.e. low of day light.

REFERENCES

[1] Mathworks rgb2gray (http://in.mathworks.com/help/matlab/ref/rgb2gray.html).

[2] Linda G. Shapiro and George C. Stockman (2001): "Computer Vision", pp 279-325, New Jersey, Prentice-Hall.

[3] Barghout, Lauren, and Lawrence W. Lee. "Perceptual information processing system." Paravue Inc. U.S. Patent Application 10/618,543, filed July 11, 2003.

[4] "Top-Down Design (Introduction to Statistical Computing)". bactra.org. September 24, 2012. Retrieved September 9, 2015.

[5] H.G.Barrow and J.M. Tenenbaum(1981) "Interpreting line drawings as three-dimensional surfaces", Artificial Intelligence, vol 17, issues 1-3, pages 75-116.

[6] Lindeberg, Tony (2001), "Edge detection", in Hazewinkel, Michiel, *Encyclopedia of Mathematics*, Springer, ISBN 978-1-55608-010-4.

[7] T. Lindeberg (1998) "Edge detection and ridge detection with automatic scale selection", International Journal of Computer Vision, 30, 2, pages 117--154.

[8] M. Sezgin and B. Sankur (2004). "Survey over image thresholding techniques and quantitative performance evaluation". *Journal of Electronic Imaging* **13** (1): 146–165.doi:10.1117/1.1631315.

[9] Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". *IEEE Trans. Sys., Man., Cyber.* **9** (1): 62–66. doi:10.1109/TSMC.1979.4310076.

[10] Kittler, Josef and Illingworth, John (1985). "On threshold selection using clustering criteria".*Systems, Man and Cybernetics, IEEE Transactions on*. SMC-15 (5): 652–655.

[11] A. Weeks and G. Hague, "Color Segmentation in the HSI Color Space Using the k-Means Algorithm," Proc. SPIE, vol. 3026, pp. 143-154, Feb. 1997.

[12] J.D. Buf, M. Kardan, and M. Spann, "Texture Feature Performance for Image Segmentation," Pattern Recognition, vol. 23, 1990.

[13] U. Montanari, "On the Optimal Detection of Curves in Noisy Pictures," Comm. ACM, vol. 14, 1971.

[14] A. Shashua and S. Ullman, "Structural Saliency: The Detection of Globally Salient Structures Using a Locally Connected Network," Proc. Int'l Conf. Computer Vision, 1988.

[15] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 6, Nov. 1984.

[16] D. Mumford and J. Shah, "Boundary Detection by Minimizing Functionals," Proc. Computer Vision and Pattern Recognition (CVPR '85), 1985.

[17] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," Proc. Computer Vision and Pattern Recognition (CVPR), 1997.

[18] S. Anitha Elavarasi, Dr. J. Akilandeswari, Dr. B. Sathiyabhama," A survey on partition clustering algorithms", International Journal of Enterprise Computing and Business SystemInternational Systems, vol. 1, pp. 1-13, 2011.

[19] Harikrishna Narasimhan, Purushothaman Ramraj" Contribution-Based Clustering Algorithm for Content-Based Image Retrieval", 2010 5th International Conference on Industrial and Information Systems, ICIIS 2010, pp. 442-447.

[20] Wenbing Tao, Hai Jin and Yimin Zhang," Color Image Segmentation Based on Mean Shift and Normalized Cuts", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND

CYBERNETICS—PART B: CYBERNETICS, VOL. 37, NO. 5, OCTOBER 2007, pp. 1382-1389.
[21] Shi Na, Liu Xumin, Guan yong," Research on k-means Clustering Algorithm An Improved k-means Clustering Algorithm", Third International Symposium on Intelligent Information Technology and Security Informatics, pp. 63-67, 2010.
[22] Monika Jain, Dr. S.K.Singh," A Survey On: Content Based Image Retrieval Systems Using Clustering Techniques For Large Data sets", International Journal of Managing Information Technology (IJMIT) Vol.3, No.4, November 2011, pp. 23-39.
[23] Juntao Wang, Xiaolong Su," An improved K-Means clustering algorithm", IEEE proceeding, pp. 44-46, 2011.
[24] Periklis Andritsos," Data Clustering Techniques", March 11, 2002.
[25] Siddheswar Ray, Rose H. Turi," Determination of Number of Clusters in K-Means Clustering and Application in Colour Image Segmentation", IEEE proceeding.
[26] Vattani., A. (2011). "k-means requires exponentially many iterations even in the plane"(PDF). *Discrete and Computational Geometry* **45** (4): 596–616. doi:10.1007/s00454-011-9340-1
[27] MacKay, David (2003). "Chapter 20. An Example Inference Task: Clustering" (PDF).*Information Theory, Inference and Learning Algorithms*. Cambridge University Press. pp. 284–292. ISBN 0-521-64298-1. MR 2012999.
[28] L. Najman and M. Schmitt. Watershed of a continuous function. In Signal Processing (Special issue on Mathematical Morphology.), Vol. 38 (1994), pages 99–112
[29] Serra, J. Image Analysis and Mathematical Morphology. Academic Press, New York, 1982.
[30] Meyer, F., and Beucher, S. Morphological segmentation. J. Visual Commun. and Image Repres. 1, 1 (1990), 21–45.
[31] Vincent, L. Algorithmes Morphologiques a Base de Files d'Attente et de Lacets. Extension aux Graphes. PhD thesis, Ecole Nationale Sup´erieure des Mines de Paris, Fontainebleau, 1990.
[32] Vincent, L., and Soille, P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Patt. Anal. Mach. Intell. 13, 6 (1991), 583–598.
[33] Meyer, F. Topographic distance and watershed lines. Signal Processing 38 (1994), 113–125.
[34] Serge Beucher and Christian Lantuéj workshop on image processing, real-time edge and motion detection *(1979).*
[35] J. Cousty, G. Bertrand, L. Najman and M. Couprie. Watershed Cuts: Minimum Spanning Forests and the Drop of Water Principle, Pattern Analysis and Machine Intelligence, IEEE Transactions on 31(8) pp. 1362-1374, 2009.
[36] Serge Beucher and Fernand Meyer. The morphological approach to segmentation: the watershed transformation. In *Mathematical Morphology in Image Processing* (Ed. E. R. Dougherty), pages 433–481 (1993).
[37] M. Couprie, G. Bertrand. Topological gray-scale watershed transform. In Proc. of SPIE Vision Geometry V, volume 3168, pages 136–146 (1997).
[38] G. Bertrand. On topological watersheds. Journal of Mathematical Imaging and Vision, 22(2–3), pages 217–230 (2005).
[39] Jean Cousty, Gilles Bertrand, Laurent Najman, and Michel Couprie. Watershed Cuts: Minimum Spanning Forests and the Drop of Water Principle. IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (8). August 2009. pp. 1362–1374.
[40] Stanley R. Alten *Audio Basics* Cengage 2011 ISBN 0-495-91356-1 page 63.

[41] "Headphones : The Ultimate buying guide - Hi-fidelity headphones". *StereoCompare*. Retrieved 2016-03-03.
[42] "Buying the perfect power bank for your smartphone: 12 things to consider - Powerbanky". *Powerbanky*. Retrieved 2015-12-06.
[43] Bradski, Gary; Kaehler, Adrian (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc. p. 6.
[44] OpenCV C interface: http://docs.opencv.org
[45] CPAN: http://search.cpan.org/~yuta/Cv-0.29] Ch,[Ch OpenCV :http://www.softintegration.com/products/thirdparty/opencv/
[46] Mariko Sakuta, Shogo Takanashi and Takashi Kubota "An image based path planning scheme for rovers" in 2011 IEEE International Conference on Robotics and Biomimetics December 7-11, 2011, Phuket, Thailand.
[47] An automatic histogram-based initializing algorithm for K-means clustering in CT
[48] Rapid object detection using a boosted cascade of simple features- viola jones
[49] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In International Conference on Computer Vision, 1998.
[50] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In Computational Learning Theory: Eurocolt '95, pages 23–37. Springer-Verlag, 1995.
[51] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. Ann. Stat., 26(5):1651–1686, 1998.
[52] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1997.
[53] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In International Conference on Computer Vision, 1998.
[54] J.K. Tsotsos, S.M. Culhane, W.Y.K. Wai, Y.H. Lai, N. Davis, and F. Nuflo. Modeling visual-attention via selective tuning. Artificial Intelligence Journal, 78(1-2):507–545, October 1995.
[55] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. IEEE Patt. Anal. Mach. Intell., 20(11):1254–1259, November 1998.
[56] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers, 1997.
[57] F. Fleuret and D. Geman. Coarse-to-fine face detection. Int. J. Computer Vision, 2001.
[58] William T. Freeman and Edward H. Adelson. The design and use of steerable filters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(9):891–906, 1991.
[59] H. Greenspan, S. Belongie, R. Gooodman, P. Perona, S. Rakshit, and C. Anderson. Overcomplete steerable pyramid filters and rotation invariance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1994.
[60] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In Proceedings of the Fourteenth International Conference on Machine Learning, 1997.
[61] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. In IEEE Patt. Anal. Mach. Intell., volume 20, pages 22–38, 1998.
[62] Robust Door Detection-Christopher Juenemann, Anthony Corbin, Jian Li.
[63] Damien Garcia, Assistant Professor, Department of Radiology, University of Montreal Hospital, Montreal, QC, Canada http://www.biomecardio.com/matlab/otsu.html.

Appendix

This project was applied for patent and has been indeed uploaded in the website of Indian Patent Search (ipindiaonline.gov.in). At the present time the patent is applied for scrutiny on behalf of IEM Kolkata.

Patent Application Number : 1159/KOL/2015

The Patent Details are given in the Appendix.

INTELLECTUAL PROPERTY INDIA
PATENTS | DESIGNS | TRADE MARKS
GEOGRAPHICAL INDICATIONS

सत्यमेव जयते
GOVERNMENT OF INDIA

Controller General of Patents,Designs and Trademarks
Department of Industrial Policy and Promotion
Ministry of Commerce and Industry

(12) PATENT APPLICATION PUBLICATION
(19) INDIA

(21) Application No. : 1159/KOL/2015

(22) Date of filing of Application :16/11/2015

(43) Publication Date : 29/04/2016
Journal No. - 18/2016

(54) Title of the invention : Smart Navigation System for visually challenged persons.

(51) International classification	:G01C21/00
(31) Priority Document No	:NA
(32) Priority Date	:NA
(33) Name of priority country	:NA
(86) International Application No	:PCT//
Filing Date	:01/01/1900
(87) International Publication No	: NA
(61) Patent of Addition to Application Number	:NA
Filing Date	:NA
(62) Divisional to Application Number	:NA
Filing Date	:NA

(71)**Name of Applicant :**
1)Institute of Engineering and Management
Address of Applicant :Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091 West Bengal India
(72)**Name of Inventor :**
1)Tuhin Utsab Paul (India)
2)Aninda Ghosh (India)
3)Nitish Kumar Thaklur (India)
4)Riya Sett (India)

(57) Abstract :
A Navigation System developed for visually impaired person, using Image Processing as the main tool, Supported by anEmbedded System in hand. Image processing with its increasing popularity is being used as a tool for social benefits. Along with microcomputer a spectacle mounted USB camera, Bluetooth earphone and a portable power source will form the whole invention.

Number of Pages = 9

InPASS Indian Patent Advanced Search System

Government of India
Controller General of Patents Designs and Trademarks
Department of Industrial Policy and Promotion
Ministry of Commerce and Industry

INTELLECTUAL PROPERTY INDIA

Patent Search | Patent E-register | Application Status | Help

Results

Application Number	Title	Application Date	Status
1159/KOL/2015	smart navigation system for visually challenged persons.	2015/11/16	Published
1159/KOL/2015	smart navigation system for visually challenged persons.	2015/11/16	Published

Invention Title	Smart Navigation System for visually challenged persons.
Publication Date	2016/04/29
Application Date	2015/11/16
Application Number	1159/KOL/2015
Patent Number	
Publication type	A
Field of Invention	FI13

Abstract: A Navigation System developed for visually impaired person, using Image Processing as the main tool, Supported by anEmbedded System in hand. Image processing with its increasing popularity is being used as a tool for social benefits. Along with microcomputer a spectacle mounted USB camera, Bluetooth earphone and a portable power source will form the whole invention.

Page 1 of 1 | Total Document(s): 2 | Displaying 1 - 2 of 2

Details

Bibliographic Data | Complete Specification | Application Status

Invention Title	Smart Navigation System for visually challenged persons.
Publication Number	18/2016
Publication Date	2016/04/29
Publication Type	INA
Application Number	1159/KOL/2015
Application Filing Date	2015/11/16
Priority Number	-
Priority Country	-
Priority Date	-
Field Of Invention	(FI13) COMPUTER SCIENCE
Classification (IPC)	G01C-21/00

Inventor

Name	Address	Country	Nationality
Tuhin Utsab Paul	Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091	IN	IN
Aninda Ghosh	Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091	IN	IN
Nitish Kumar Thakur	Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091	IN	IN
Riya Sett	Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091	IN	IN

Applicant

Name	Address	Country	Nationality
Institute of Engineering and Management	Institute of Engineering & Management, Saltlake Electronics Complex, Sector-V, Saltlake, Kolkata-700091	IN	IN

Abstract:
A Navigation System developed for visually impaired person, using Image Processing as the main tool, Supported by anEmbedded System in hand. Image processing with its increasing popularity is being used as a tool for social benefits. Along with microcomputer a spectacle mounted USB camera, Bluetooth earphone and a portable power source will form the whole invention.

Back | Search