**Chapter – 1**

**MATLAB**

* 1. **Introduction to MATLAB**

MATLAB is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, you can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java.

You can use MATLAB for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology. More than a million engineers and scientists in industry and academia use MATLAB, the language of technical computing.

**Numeric Computation**

MATLAB provides a range of numerical computation methods for analyzing data, developing algorithms, and creating models. The MATLAB language includes mathematical functions that support common engineering and science operations. Core math functions use processoroptimized libraries to provide fast execution of vector and matrix calculations.

Available methods include:

• Interpolation and regression

• Differentiation and integration

• Linear systems of equations

• Fourier analysis

• Eigenvalues and singular values

• Ordinary differential equations (ODEs)

• Sparse matrices

MATLAB add-on products provide functions in specialized areas such as statistics, optimization, signal analysis, and machine learning.

* 1. **Data Analysis and Visualization**

MATLAB provides tools to acquire, analyze, and visualize data, enabling you to gain insight into your data in a fraction of the time it would take using spreadsheets or traditional programming languages. You can also document and share your results through plots and reports or as published MATLAB code.

**Acquiring Data**

MATLAB lets you access data from files, other applications, databases, and external devices. You can read data from popular file formats such as Microsoft Excel; text or binary files; image, sound, and video files; and scientific files such as netCDF and HDF. File I/O functions let you work with data files in any format.Using MATLAB with add-on products, you can acquire data from hardware devices, such as your computer’s serial port or sound card, as well as stream live, measured data directly into MATLAB for analysis and visualization. You can also communicate with instruments such as oscilloscopes, function generators, and signal analyzers.

**Analyzing Data**

MATLAB lets you manage, filter, and preprocess your data. You can perform exploratory data analysis to uncover trends, test assumptions, and build descriptive models. MATLAB provides functions for filtering and smoothing, interpolation, convolution, and fast Fourier transforms (FFTs). Add-on products provide capabilities for curve and surface fitting, multivariate statistics, spectral analysis, image analysis, system identification, and other analysis tasks.

**Visualizing Data**

MATLAB provides built-in 2-D and 3-D plotting functions, as well as volume visualization functions. You can use these functions to visualize and understand data and communicate results. Plots can be customized either interactively or programmatically.

The MATLAB plot gallery provides examples of many ways to display data graphically in MATLAB. For each example, you can view and download source code to use in your MATLAB application.

* 1. **Documenting and Sharing Results**

You can share results as plots or complete reports. MATLAB plots can be customized to meet publication specifications and saved to common graphical and data file formats. You can automatically generate a report when you execute a MATLAB program. The report contains your code, comments, and program results, including plots. Reports can be published in a variety of formats, such as HTML, PDF, Word, or LaTeX.

**Programming and Algorithm Development**

MATLAB provides a high-level language and development tools that let you quickly develop and analyze algorithms and applications.The MATLAB Language. The MATLAB language provides native support for the vector and matrix operations that are fundamental to solving engineering and scientific problems, enabling fast development and execution.With the MATLAB language, you can write programs and develop algorithms faster than with traditional languages because you do not need to perform lowlevel administrative tasks such as declaring variables, specifying data types, and allocating memory. In many cases, the support for vector and matrix operations eliminates the need for for-loops. As a result, one line of MATLAB code can often replace several lines of C or C++ code.

MATLAB provides features of traditional programming languages, including flow control, error handling, and object-oriented programming (OOP). You can use fundamental data types or advanced data structures, or you can define custom data types.

You can produce immediate results by interactively executing commands one at a time. This approach lets you quickly explore multiple options and iterate to an optimal solution. You can capture interactive steps as scripts and functions to reuse and automate your work.

MATLAB add-on products provide built-in algorithms for signal processing and communications, image and video processing, control systems, and many other domains. By combining these algorithms with your own, you can build complex programs and applications.

* 1. **Development Tools**

MATLAB includes a variety of tools for efficient algorithm development, including:

Command Window – Lets you interactively enter data, execute commands and programs, and display results.

MATLAB Editor – Provides editing and debugging features, such as setting breakpoints and stepping through individual lines of code.

Code Analyzer – Automatically checks code for problems and recommends modifications to maximize performance and maintainability.

MATLAB Profiler – Measures performance of MATLAB programs and identifies areas of code to modify for improvement.

Additional tools compare code and data files, and provide reports showing file dependencies, annotated reminders, and code coverage.

**Integration with Other Languages and Applications**

You can integrate MATLAB applications with those written in other languages. From MATLAB, you can directly call code written in C, C++, Java, and .NET. Using the MATLAB engine library, you can call MATLAB code from C, C++, or Fortran applications.

**Performance**

MATLAB uses processor-optimized libraries for fast execution of matrix and vector computations. For general-purpose scalar computations, MATLAB uses its just-intime (JIT) compilation technology to provide execution speeds that rival those of traditional programming languages.To take advantage of multicore and multiprocessor computers, MATLAB provides many multithreaded linear algebra and numerical functions. These functions automatically execute on multiple computational threads in a single MATLAB session, enabling them to execute faster on multicore computers.

You can take further advantage of multicore desktop and other high-performance computing resources such as GPUs and clusters with add-on parallel computing products. These products provide highlevel constructs that let you parallelize applications with only minor changes to MATLAB code.

**Chapter-2**

**Digital Image Processing & Segmentation**

**2.1) Digital Image**

A **digital image** is a numeric representation (normally binary) of a two-dimensional image. Depending on whether the image resolution is fixed, it may be of vector or raster type. Without qualifications, the term "digital image" usually refers to raster images also called bitmap images.

[Raster images](http://en.wikipedia.org/wiki/Raster_image) have a finite set of digital values called *picture elements* or pixels. The digital image contains a fixed number of rows and columns of pixels. Pixels are the smallest individual element in an image, holding quantized values that represent the brightness of a given color at any specific point. Raster images can be created by a variety of input devices and techniques, such as digital cameras, scanners, coordinate-measuring machines, seismographic profiling, airborne radar, and more.

Vector images resulted from mathematical geometry (vector). In mathematical terms, a vector consists of point that has both direction and length.Often, both raster and vector elements will be combined in one image; for example, in the case of a billboard with text (vector) and photographs (raster).

**2.2) Digital Image Processing**

**Digital image processing** is the use of computer algorithms 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 modeled in the form of multidimensional systems.

Digital image processing allows the use of much more complex algorithms, and hence, can offer both more sophisticated performance at simple tasks, and the implementation of methods which would be impossible by analog means.

In particular, digital image processing is the only practical technology for:

* Classification
* Feature extraction
* Pattern recognition
* Projection
* Multi-scale signal analysis

Some techniques which are used in digital image processing include:

* Pixelation
* Linear filtering
* Principal components analysis
* Independent component analysis
* Hidden Markov models
* Anisotropic diffusion
* Partial differential equations
* Self-organizing maps

**2.3) Point Detection**

**Interest point detection** is a recent terminology in computer vision that refers to the detection of interest points for subsequent processing. An interest point is a point in the image which in general can be characterized as follows:

* it has a clear, preferably mathematically well-founded, definition,
* it has a well-defined *position* in image space,
* the local image structure around the interest point is rich in terms of local *information contents*, such that the use of interest points simplify further processing in the vision system,
* it is *stable* under local and global perturbations in the image domain, including deformations as those arising from perspective transformations (sometimes reduced to affine transformations, scale changes, rotations and/or translations) as well as illumination/brightness variations, such that the interest points can be reliably computed with high degree of *reproducibility*.
* Optionally, the notion of interest point should include an attribute of *scale*, to make it possible to compute interest points from real-life images as well as under scale changes.

Today, a main application of interest points is to signal points/regions in the image domain that are likely candidates to be useful for *image matching* and *view-based object recognition*. For this purpose, several types of corner detectors and blob detectors have been demonstrated to be highly useful in practical applications (see respective articles for references). Blob detectors and corner detectors have also been used as primitives for texture recognition, texture analysis and for constructing object models from multiple views of textured objects.

If one aims at drawing a distinction between corner detectors and blob detectors, this can often be done in terms of their localization properties at corner structures. For a junction structure in the image domain that corresponds to an intersection of physical edges in the three-dimensional world, the localization properties of a corner detector will in most cases be much better than the localization properties that would be obtained from a blob detector. Hence, for the purpose of computing structure and motion from multiple views, corner detectors will in many cases have advantages compared to blob detectors in terms of smaller localization error. Notwithstanding this, blob descriptors have also been demonstrated to be useful when relating object models to temporal imagery.

**2.4) Edge Detection**

**Edge detection** is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction, which aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The same problem of finding discontinuities in 1D signals is known as step 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:

* discontinuities in depth,
* discontinuities in surface orientation,
* changes in material properties and
* 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.

Edge detection is one of the fundamental steps in image processing, image analysis, image pattern recognition, and computer vision techniques. During recent years, however, substantial (and successful) research has also been made on computer vision method  that do not explicitly rely on edge detection as a pre-processing step.

**Chapter-3**

**Segmentation**

**3.1) Introduction**

In computer vision, **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. 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 visual 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 as colour, intensity and  texture of the digital image. Adjacent regions are significantly different with respect to the same characteristic(s).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.

Image segmentation is an important technology for image processing. There are

many applications whether on synthesis of the objects or computer graphic images require precise segmentation. With the consideration of the characteristics of each object composing images in MPEG4, object-based segmentation cannot be ignored. Nowadays, sports programs are among the most popular programs, and there is no doubt that viewers’ interest is concentrated on the athletes. Therefore, demand for image segmentation of sport scenes is very high in terms of both visual compression and image handling using extracted athletes. In this project, we introduce a basic idea about color information and edge extraction to achieve the image segmentation. The color information helps obtain the texture information of the target image while the edge extraction detects the boundary of the target image. By combining these, the target image can be correctly segmented and represent. Besides, because color information

and edge extraction can use basic image processing methods, they can not only

demonstrate what textbook claims but also make us realize their function works. We expect that we can extract most part of the target.

**3.2) Techniques for Segmentation**

Several general-purpose algorithms and techniques have been developed for image segmentation. Since there is no general solution to the image segmentation problem, these techniques often have to be combined with domain knowledge in order to effectively solve an image segmentation problem for a problem domain.

**3.2.1) Thresholding**

The simplest method of image segmentation is called the [thresholding](http://en.wikipedia.org/wiki/Thresholding_(image_processing)" \o "Thresholding (image processing)) method. This method is based on a clip-level (or a threshold value) to [turn](http://en.wikipedia.org/wiki/Segmentation_(image_processing)) a gray-scale image into a binary image.

The key of this method is to select the threshold value (or values when multiple-levels are selected). Several popular methods are used in industry including the maximum entropy method, [Otsu's method](http://en.wikipedia.org/wiki/Otsu%27s_method) (maximum variance), and [k-means](http://en.wikipedia.org/wiki/K-means) clustering.

Recently, methods have been developed for thresholding computed tomography (CT) images. The key [idea](http://en.wikipedia.org/wiki/Segmentation_(image_processing)) is that, unlike [Otsu's method](http://en.wikipedia.org/wiki/Otsu%27s_method), the thresholds are derived from the radiographs instead of the (reconstructed) image.

**3.2.2) Clustering Methods**

The [K-means algorithm](http://en.wikipedia.org/wiki/K-means_algorithm) is an [iterative](http://en.wikipedia.org/wiki/Iterative) technique that is used to [partition an image](http://en.wikipedia.org/wiki/Cluster_analysis) into *K* clusters. The basic [algorithm](http://en.wikipedia.org/wiki/Algorithm) is:

1. Pick *K* cluster centers, either [randomly](http://en.wikipedia.org/wiki/Random) or based on some [heuristic](http://en.wikipedia.org/wiki/Heuristic)
2. Assign each pixel in the image to the cluster that minimizes the [distance](http://en.wikipedia.org/wiki/Distance) between the pixel and the cluster center
3. Re-compute the cluster centers by averaging all of the pixels in the cluster
4. Repeat steps 2 and 3 until convergence is attained (e.g. no pixels change clusters)

In this case, [distance](http://en.wikipedia.org/wiki/Distance) is the squared or absolute difference between a pixel and a cluster center. The difference is typically based on pixel [color](http://en.wikipedia.org/wiki/Hue" \o "Hue), [intensity](http://en.wikipedia.org/wiki/Brightness), [texture](http://en.wikipedia.org/wiki/Texture_(computer_graphics)), and location, or a weighted combination of these factors. *K* can be selected manually, [randomly](http://en.wikipedia.org/wiki/Random), or by a heuristic.

This algorithm is guaranteed to converge, but it may not return the optimal solution. The quality of the solution depends on the initial set of clusters and the value of *K*.

In statistics and machine learning, the k-means algorithm is a clustering algorithm to partition n objects into k clusters, where k < n. It is similar to the [expectation-maximization algorithm](http://en.wikipedia.org/wiki/Expectation-maximization_algorithm) for mixtures of [Gaussians](http://en.wikipedia.org/wiki/Gaussian_filter) in that they both attempt to find the centers of natural clusters in the data. The model requires that the object attributes correspond to elements of a vector space. The objective it tries to achieve is to minimize total intra-cluster variance, or, the squared error function. The k-means clustering was invented in 1956. The most common form of the algorithm uses an iterative refinement heuristic known as [Lloyd's algorithm](http://en.wikipedia.org/wiki/Lloyd%27s_algorithm). Lloyd's algorithm starts by partitioning the input points into k initial sets, either at random or using some heuristic data. It then calculates the mean point, or centroid, of each set. It constructs a new partition by associating each point with the closest centroid. Then the centroids are recalculated for the new clusters, and algorithm repeated by alternate application of these two steps until convergence, which is obtained when the points no longer switch clusters (or alternatively centroids are no longer changed). Lloyd's algorithm and k-means are often used synonymously, but in reality Lloyd's algorithm is a heuristic for solving the k-means problem, as with certain combinations of starting points and centroids, Lloyd's algorithm can in fact converge to the wrong answer. Other variations exist, but Lloyd's algorithm has remained popular, because it converges extremely quickly in practice. In terms of performance the algorithm is not guaranteed to return a global optimum. The quality of the final solution depends largely on the initial set of clusters, and may, in practice, be much poorer than the global optimum. Since the algorithm is extremely fast, a common method is to run the algorithm several times and return the best clustering found. A drawback of the k-means algorithm is that the number of clusters k is an input parameter. An inappropriate choice of k may yield poor results. The algorithm also assumes that the variance is an appropriate measure of cluster scatter.

**3.2.3) Otsu’s Method**

In computer vision and image processing, **Otsu's method** is used to automatically perform histogram shape-based image thresholding or, the reduction of a graylevel image to a binary image. The algorithm assumes that the image to be thresholded contains two classes of pixels or bi-modal histogram (e.g. foreground and background) then calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal. The extension of the original method to multi-level thresholding is referred to as the Multi Otsu method. Otsu's method is named after Nobuyuki Otsu.

**3.2.4) Compression-based Methods**

Compression based methods postulate that the optimal segmentation is the one that minimizes, over all possible segmentations, the coding length of the data.[[5]](http://en.wikipedia.org/wiki/Segmentation_(image_processing)#cite_note-4)[[6]](http://en.wikipedia.org/wiki/Segmentation_(image_processing)#cite_note-5) The connection between these two concepts is that segmentation tries to find patterns in an image and any regularity in the image can be used to compress it. The method describes each segment by its texture and boundary shape. Each of these components is modeled by a probability distribution function and its coding length is computed as follows:

1. The boundary encoding leverages the fact that regions in natural images tend to have a smooth contour. This prior is used by [Huffman coding](http://en.wikipedia.org/wiki/Huffman_coding) to encode the difference [chain code](http://en.wikipedia.org/wiki/Chain_code) of the contours in an image. Thus, the smoother a boundary is, the shorter coding length it attains.
2. Texture is encoded by [lossy compression](http://en.wikipedia.org/wiki/Lossy_compression" \o "Lossy compression) in a way similar to [minimum description length](http://en.wikipedia.org/wiki/Minimum_description_length) (MDL) principle, but here the length of the data given the model is approximated by the number of samples times the [entropy](http://en.wikipedia.org/wiki/Entropy) of the model. The texture in each region is modeled by a [multivariate normal distribution](http://en.wikipedia.org/wiki/Multivariate_normal_distribution) whose entropy has closed form expression. An interesting property of this model is that the estimated entropy bounds the true entropy of the data from above. This is because among all distributions with a given mean and covariance, normal distribution has the largest entropy. Thus, the true coding length cannot be more than what the algorithm tries to minimize.

For any given segmentation of an image, this scheme yields the number of bits required to encode that image based on the given segmentation. Thus, among all possible segmentations of an image, the goal is to find the segmentation which produces the shortest coding length.

**3.2.5) Histogram-based Methods**

Histogram-based methods are very efficient when compared to other image segmentation methods because they typically require only one pass through the pixels. In this technique, a histogram is computed from all of the pixels in the image, and the peaks and valleys in the histogram are used to locate the clusters in the image. Colour or intensity can be used as the measure.

A refinement of this technique is to recursively apply the histogram-seeking method to clusters in the image in order to divide them into smaller clusters. This is repeated with smaller and smaller clusters until no more clusters are formed.

One disadvantage of the histogram-seeking method is that it may be difficult to identify significant peaks and valleys in the image. In this technique of image classification distance metric and integrated region matching are familiar.

Histogram-based approaches can also be quickly adapted to occur over multiple frames, while maintaining their single pass efficiency. The histogram can be done in multiple fashions when multiple frames are considered. The same approach that is taken with one frame can be applied to multiple, and after the results are merged, peaks and valleys that were previously difficult to identify are more likely to be distinguishable.

**3.2.6) Region Growing Methods**

The first [region-growing](http://en.wikipedia.org/wiki/Region-growing) method was the seeded region growing method. This method takes a set of seeds as input along with the image. The seeds mark each of the objects to be segmented. The regions are iteratively grown by comparing all unallocated neighbouring pixels to the regions. The difference between a pixel's intensity value and the region's mean, \delta, is used as a measure of similarity. The pixel with the smallest difference measured this way is allocated to the respective region. This process continues until all pixels are allocated to a region.

Seeded region growing requires seeds as additional input. The segmentation results are dependent on the choice of seeds. Noise in the image can cause the seeds to be poorly placed. Unseeded region growing is a modified algorithm that doesn't require explicit seeds. It starts off with a single region A_1 – the pixel chosen here does not significantly influence final segmentation. At each iteration it considers the neighbouring pixels in the same way as seeded region growing. It differs from seeded region growing in that if the minimum \delta is less than a predefined threshold T then it is added to the respective region A_j. If not, then the pixel is considered significantly different from all current regions A_i and a new region A_{n+1} is created with this pixel.

One variant of this technique, proposed by [Haralick](http://en.wikipedia.org/wiki/Haralick" \o "Haralick) and Shapiro (1985),[[1]](http://en.wikipedia.org/wiki/Image_segmentation" \l "cite_note-computervision-1) is based on pixel [intensities](http://en.wikipedia.org/wiki/Brightness). The [mean](http://en.wikipedia.org/wiki/Arithmetic_mean) and [scatter](http://en.wikipedia.org/wiki/Scatter) of the region and the intensity of the candidate pixel is used to compute a test statistic. If the test statistic is sufficiently small, the pixel is added to the region, and the region’s mean and scatter are recomputed. Otherwise, the pixel is rejected, and is used to form a new region.

A special region-growing method is called \lambda-connected segmentation (see also [lambda-connectedness](http://en.wikipedia.org/wiki/Lambda-connectedness)). It is based on pixel [intensities](http://en.wikipedia.org/wiki/Brightness) and neighborhood-linking paths. A degree of connectivity (connectedness) will be calculated based on a path that is formed by pixels. For a certain value of \lambda, two pixels are called \lambda-connected if there is a path linking those two pixels and the connectedness of this path is at least \lambda. \lambda-connectedness is an equivalence relation.

**3.2.7) Split-And-Merge Method**

Split-and-merge segmentation is based on a [quadtree](http://en.wikipedia.org/wiki/Quadtree" \o "Quadtree) partition of an image. It is sometimes called quadtree segmentation.

This method starts at the root of the tree that represents the whole image. If it is found non-uniform (not homogeneous), then it is split into four son-squares (the splitting process), and so on so forth. Conversely, if four son-squares are homogeneous, they can be merged as several connected components (the merging process). The node in the tree is a segmented node. This process continues recursively until no further splits or merges are possible.[[10]](http://en.wikipedia.org/wiki/Image_segmentation#cite_note-split-and-merge1-10)[[11]](http://en.wikipedia.org/wiki/Image_segmentation#cite_note-split-and-merge2-11) When a special data structure is involved in the implementation of the algorithm of the method, its time complexity can reach O(n\log n), an optimal algorithm of the method.

**3.3) Applications Of Segmentation**

Some of the practical applications of image segmentation are:

* Medical imaging
* Locate tumors and other pathologies
* Measure tissue volumes
* Computer-guided surgery
* Diagnosis
* Treatment planning
* Study of anatomical structure
* Locate objects in satellite images (roads, forests, etc.)
* Face recognition
* Iris recognition
* Fingerprint recognition
* Traffic control systems

**Chapter-4**

**Text-Information-Extraction**

**(TIE)**

**4.1) Introduction**

The usual process of a business entity in replicating files for safe-keeping is to use the scanning software. Once the images are captured, you now have this file that you can easily view from any computer system. The problem is that the images are meant to be viewed as pictures. If you need some sort of data from the files you still need to record manually and copy down the details on another piece of paper. This tedious and time consuming task that can take hours to complete is made easy with the text extraction from image program.

The presence of the text extraction from image software will ease the burden of manual tasks related to data gathering and selection. Captured data from the photos can now easily be retrieved with the help of the text extraction from image system. Archiving documents are not only for keeping the files safe but rather for keeping the information from the files accessible to all the authorized users in the area. The text extraction from image program is used to assist the operational workflow of the business.

The text extraction from image software should always be equipped with three factors that make it a worthwhile device to have. First, it should be able to retain text and data formatting during the conversion process. This means that any printed form remains and looks the same, even if it is viewed from the electronic database. Such aspects will ensure the credibility of the actual document in paper form. Second aspect of the text extraction from image program would be the extract data for content aggregation. This means that the text extraction from image details can then easily be posted to another spreadsheet automatically. You no longer have to do the actual posting process yourself. As automated systems, they should have this ability and feature to do so. Third and last, the text extraction from image software should be able to preserve in full the text quality from the printed document. Again, such an effort is to ensure that any details from the files are kept honest and objective. Legible texts that are maintained from the scanning down to the retrieval are what most users are looking for nowadays.

There have been attempts in the past for extraction of textual component of an image by analyzing the edges of candidate regions or homogeneous color/gray scale components that contain the characters. A number of page segmentation methods have been studied and compared objectively in. The BESUS method, for example, is constructed using a number of morphology-based modules.

In computer vision and image processing, **Otsu's method** is used to automatically perform histogram shape-based image thresholding,or, the reduction of a graylevel image to a binary image. The algorithm assumes that the image to be thresholded contains two classes of pixels or bi-modal histogram (e.g. foreground and background) then calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal. The extension of the original method to multi-level thresholding is referred to as the Multi Otsu method. Otsu's method is named after Nobuyuki Otsu.

**4.2) Method**

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^2_w(t)=\omega_1(t)\sigma^2_1(t)+\omega_2(t)\sigma^2_2(t)

Weights \omega_i are the probabilities of the two classes separated by a threshold t and \sigma^2_ i variances of these classes.

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

\sigma^2_b(t)=\sigma^2-\sigma^2_w(t)=\omega_1(t)\omega_2(t)\left[\mu_1(t)-\mu_2(t)\right]^2

which is expressed in terms of class probabilities \omega_i and class means \mu_i.

The class probability \omega_1(t) is computed from the histogram as t:

\omega_1(t)=\Sigma_0^t p(i)

while the class mean \mu_1(t) is:

\mu_1(t)=\Sigma_0^t p(i)\,x(i)

where x(i) is the value at the center of the ith histogram bin. Similarly, you can compute \omega_2(t) and \mu_t on the right-hand side of the histogram for bins greater than t.

**4.3) Algorithm**

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

**Algorithm:-**

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 \ldots  maximum intensity
   1. Update \omega_i and \mu_i
   2. Compute \sigma^2_b(t)
4. Desired threshold corresponds to the maximum \sigma^2_b(t)
5. You can compute two maximums (and two corresponding thresholds). \sigma^2_{b1}(t) is the greater max and \sigma^2_{b2}(t) is the greater or equal maximum
6. Desired threshold = \frac{threshold_1 + threshold_2 }{2}

**4.4) Scope**

There are several information sources for text information extraction in images (e.g., color, texture, motion, shape, geometry, etc). It is advantageous to merge various information sources to enhance the performance of a text information extraction system. It is, however, not clear as to how to integrate the outputs of several approaches. There is a clear need for a public domain and representative test database for objective benchmarking. The lack of a public test set makes it difficult to compare the performances of competing algorithms, and creates difficulties when merging several approaches.

For caption text, significant progress has been made and several applications, such as an automatic video indexing system, have already been presented. However, their text extraction results are inappropriate for general OCR software: text enhancement is needed for low quality video images and more adaptability is required for general cases (e.g., inverse characters, 2D or 3D deformed characters, polychrome characters, and so on). Very little work has been done on scene text. Scene text can have different characteristics from caption text. For example, part of a scene text can be occluded or it can have complex movement, vary in size, font, color, orientation, style, alignment, lighting, and transformation.

Although many researchers have already investigated text localization, text detection and tracking for video images is required for utilization in real applications (e.g., mobile handheld devices with a camera and real-time indexing systems). A text-image-structure-analysis, analogous to a document structure analysis, is needed to enable a text information extraction system to be used for any type of image, including both scanned document images and real scene images through a video camera. Despite the many difficulties in using Text Information Extraction systems in real world applications, the importance and usefulness of this field continues to attract much attention.

**4.5) Applications**

Text extraction from images and video sequences finds many useful applications in document processing , detection of vehicle license plate, analysis of technical papers with tables, maps, charts, and electric circuits, identification of parts in industrial automation , and content- based image/video retrieval from image/video databases. Educational and training video and TV programs such as news contain mixed text-picture-graphics regions. Region classification is helpful in object-based compression, manipulation and accessibility. Also, text regions may carry useful information about the visual content. However due to the variety of fonts, sizes, styles, orientations, alignment effects of uncontrolled illuminations, reflections, shadows, the distortion due to perspective projection as well as the complexity of image background, automatic localizing and extracting text is a challenging problem. Characters in a text are of different shapes and structures.

There are numerous applications of a text information extraction system, including document analysis, vehicle license plate extraction, technical paper analysis, and object-oriented data compression. In the following, we briefly describe some of these applications.

− **Wearable or portable computers:** with the rapid development of computer hardware technology, wearable computers are now a reality. A TIE system involving a hand-held device and camera was presented as an application of a wearable vision system. Watanabe’s translation camera can detect text in a scene image and translate Japanese text into English after performing character recognition. Haritaoglu also demonstrated his TIE system on a hand-held device.

− **Content-based video coding or document coding**: The MPEG-4 ( Moving Picture Experts Group)standard supports object-based encoding. When text regions are segmented from other regions in an image, this can provide higher compression rates and better image quality. Feng et al. and Cheng et al. apply adaptive dithering after segmenting a document into several different classes. As a result, they can achieve a higher quality rendering of documents containing text, pictures, and graphics.

− **License/container plate recognition**: There has already been a lot of work done on vehicle license plate and container plate recognition. Although container and vehicle license plates share many characteristics with scene text, many assumptions have been made regarding the image acquisition process (camera and vehicle position and direction, illumination, character types, and color) and geometric attributes of the text. Cui and Huang model the extraction of characters in license plates using Markov random field. Meanwhile, Park et al. use a learning-based approach for license plate extraction, which is similar to a texture-based text detection method . Kim et al. [88] use gradient information to extract license plates. Lee and Kankanhalli [34] apply a connected component-based method for cargo container verification.

− **Text-based image indexing**: This involves automatic text-based video structuring methods using caption data .

− **Texts in WWW images**: The extraction of text from WWW images can provide relevant information on the Internet. Zhou and Lopresti use a CC-based method after color quantization.

− **Video content analysis**: Extracted text regions or the output of character recognition can be useful in genre recognition . The size, position, frequency, text alignment, and OCR-ed results can all be used for this.

− **Industrial automation**: Part identification can be accomplished by using the text information on each part.

**Chapter-5**

**Phases Of Software Development**

**5.1) Planning Phase**

**Problem Recognition**

A problem is well defined very rarely. It corps out with a vague feeling of some statements lead to vague conclusions. So the first task is to get more crucial information by interviewing and meeting concerned people. It clarifies how the problem is felt, how often it occurs, how it affects the business and which departments are suffering with this. This phase consists of the following tasks.

By the end of this phase, idea as to how the information enters the system, how it is stored, how it is processed, how information changes affects the working of the system and finally the output format required by the end-user was collected. All the information generated from this phase acted as an input to the next phase.

* **Feasibility study**.

A feasibility study is an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation.

A feasibility study is a test of a system proposal according to its workability impact on organization, ability to meet user needs and effective use of resources. The objective of a feasibility study is not to solve a problem but to acquire a sense of its scope. In other words, a feasibility study is an evaluation or analysis of the potential impact of a proposed project. During the study, the problem definition is crystallized and the aspects of the problem to be included in the system are determined. After the initial investigation of system, this helped to have in-depth study of existing system, understanding its strength and weaknesses & requirements for new proposed system.

Feasibility study was done in three phases documented below:

* **Behavioral feasibility:** People are inherently resistant to change and computers have been known to facilitate change. There is always some reluctance among the users against the introduction of new system but they were told that this system would eliminate the unnecessary overhead of database migration and conversion, which presently had to be carried out on daily basis to facilitate transactions between the different departments. The objective this feasibility phase is to take the operational staff into confidence. As the success of a good system depends upon the willingness of the operating staff, they were taken into full confidence that the new proposed system would make their jobs easier, relieve them from the unnecessary overheads and reduce the possibility of errors creeping into the system.
* **Economic feasibility:** Economic feasibility is the most frequently used method for evaluating the effectiveness of the candidate system. More commonly known as cost\benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with the costs. If benefits outweigh the costs, then the decision is made to design and implement the system. A cost\benefit analysis was done for the proposed system to evaluate whether it would be b economically viable or not. The organization has in store many machines with high processing power necessary to implement the system. Also the organization has necessary software i.e. Visual Studio, SQL Server etc. or hardware to support the system. Considering the programmer time and the negligible hardware/software cost required for developing the system, it was found that the benefits in terms of reduced overhead as a result of elimination of the requirement of database migration and conversion was more than the cost.
* **Cost-based study**: It is important to identify cost and benefit factors, which can be categorized as follows: 1. Development costs; and 2. Operating costs. This is an analysis of the costs to be incurred in the system and the benefits derivable out of the system.
* **Time-based study**: This is an analysis of the time required to achieve a return on investments. The future value of a project is also a factor.
* **Technical feasibility:** Technical feasibility centers on the existing computer system. (Hardware/software) and to what extent it can support the proposed addition also the organization already has sufficient high-end machines to serve the processing requirements of the proposed system. So there is no need to purchase new software as the organization has necessary software i.e. Visual Studio, SQL Server etc or hardware to support the proposed system. Having gone through the steps of the overall analysis and feasibility study the next step was to carry out a detailed system analysis. The project analysis phase was conducted to learn about the proposed system, to study the problems and finally select a system that would take care of the problems identified in the working of the present system.

**5.2) Analysis Phase**

Systems analysis is the study of sets of interacting entities, including computer systems analysis. This field is closely related to operations research. It is also "an explicit formal inquiry carried out to help someone (referred to as the decision maker) identify a better course of action and make a better decision than he might otherwise have made."

*Analysis is defined as the procedure by which we break down an intellectual or substantial whole into parts so that we can achieve our end goals.*

This approach breaks systems analysis into 5 phases:

* Scope definition
* Problem analysis
* Requirements analysis
* Logical design
* Decision analysis

**Detailed Study of the Existing System**

This phase provides the overall requirement for the system what is to be done. Input for this phase is the information collected through several data collecting schemes such as survey, cross-questioning-answering etc and the raw data obtained which is not properly ordered and not in the precise manner. So here this raw data is converted into useful information written in precise manner and thus output is a formal document. After collecting all the information and requirements, they were verified from the concerned persons by presenting a diagrammatic version of the proposed system. The points missing were added to the system specifications for the desired system. So this final document provides the system requirement specifications for the desired system. It helps in reducing the total development cost and also establishes the various points for validation and verification.

**5.3) Design Phase**

After the analysis phase we have with us the details of the existing system and the requirements of the user for the new system. This phase diverts focus from the problem domain to the solution domain. It acts as a bridge between the requirement phase and its solution. The design phase focuses on the detailed implementation of the system recommended in the feasibility study. Emphasis is on translating performance specifications into design specifications.

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

*Object-oriented analysis and design (OOAD)* methods are becoming the most widely used methods for computer system design. The UML has become the standard language used in Object-oriented analysis and design. It is widely used for modeling software systems and is increasingly used for high designing non-software systems and organizations.

**The External Design**

External design consists of conceiving, planning out and specifying the externally observable characteristics of the software product. These characteristics include user displays or user interface forms and the report formats, external data sources and the functional characteristics, performance requirements etc. External design begins during the analysis phase and continues into the design phase.

**Logical design**

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, which involves a simplistic (and sometimes graphical) representation of an actual system. In the context of systems design, modelling can undertake the following forms, including:

* Data flow diagrams
* Entity Life Histories
* Entity Relationship Diagrams

**Physical design**

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed as output.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc.

**Design Methodology: Rapid Application Development (RAD)**

Rapid Application Development (RAD) is a methodology in which a systems designer produces prototypes for an end-user. The end-user reviews the prototype, and offers feedback on its suitability. This process is repeated until the end-user is satisfied with the final system.

**5.4) Implementation and Testing**

**1. Implementation Issues**

Implementation phase of the software development is concerned with translating the design specifications into the source code. After the system has been designed, and arrives the stage of putting it into actual usage known as the implementation of the system. This involves putting up of actual practical usage of the theoretically designed system. The primary goal of implementation is to write the source code and the internal documentation so that conformance of the code to its specifications can easily be verified and so the debugging, modifications and testing are eased. This goal can be achieved by making the source code as clear and as straightforward as possible. Simplicity, Elegance and Clarity are the hallmarks of good programs whereas complexity are indications of inadequate design and misdirected thinking. The system implementation is a fairly complex and expensive task requiring numerous inter-dependent activities. It involves the effort of a number of groups of people: user and the programmers and the computer operating staff etc. This needs a proper planning to carry out the task successfully. Thus it involves the following activities:

* Writing and testing of programs individually
* Testing the system as a whole using the live data
* Training and Education of the users and supervisory staff

1. **Testing**

The most important activity at the implementation stage is the system testing with the objective of validating the system against the designed criteria. During the development cycle, user was involved in all the phases that are analysis, design and coding. After each phase the user was asked whether he was satisfied with the output and the desired rectification was done at the moment. During coding, generally bottom up technique is used. Firstly the lower level modules are coded and then they are integrated together.

Thus before implementation, it involves the testing of the system. The testing phase involves testing first of separate parts of the system and then finally of the system as a whole. Each independent module is tested first and then the complete system is tested. This is the most important phase of the system development. The user carries out this testing and test data is also prepared by the user to check for all possible combinations of correct data as well as the wrong data that is trapped by the system. So the testing phase consists of the following steps:

* 1. **Unit testing:**

In the bottom of coding technique, each module is tested individually. Firstly the module is tested with some test data that covers all the possible paths and then the actual data was fed to check for results.

* 1. **Integration testing:**

After all the modules are ready and duly tested, these have to be integrated into the application. This integrated application was again tested first with the test data and then with the actual data.

* 1. **Parallel testing:**

The third in the series of tests before handling over the system to the user is the parallel processing of the old and the new system. At this stage, complete and thorough testing is done and supports out the event that goes wrong. This provides the better practical support to the persons using the system for the first time who may be uncertain or even nervous using it.

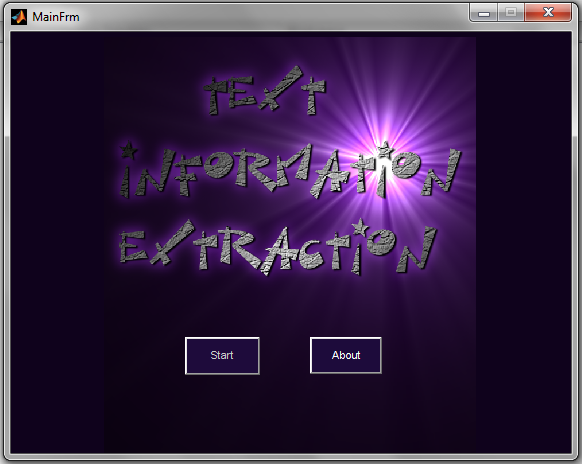
**The testing will be performed considering the following points:**

* Clerical procedure for collection and disposal of results
* Flow of data within the organization
* Accuracy of report output
* Software testing which involves testing of all the programs together. This involves the testing of system software utilities being used and specifically develops application software.
* Incomplete data formats
* Halts due to various reasons and the restart procedures.
* Range of items and incorrect formats
* Invalid combination of data records.
* Access control mechanism used to prevent unauthorized access to the system

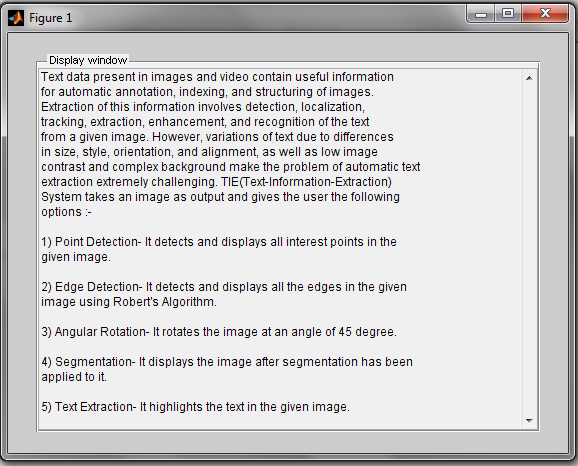
**Chapter-6**

**Snapshots**

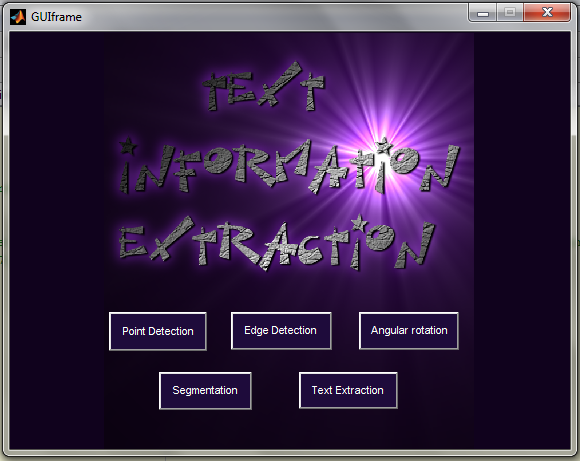
**6.1) Main Form**

****

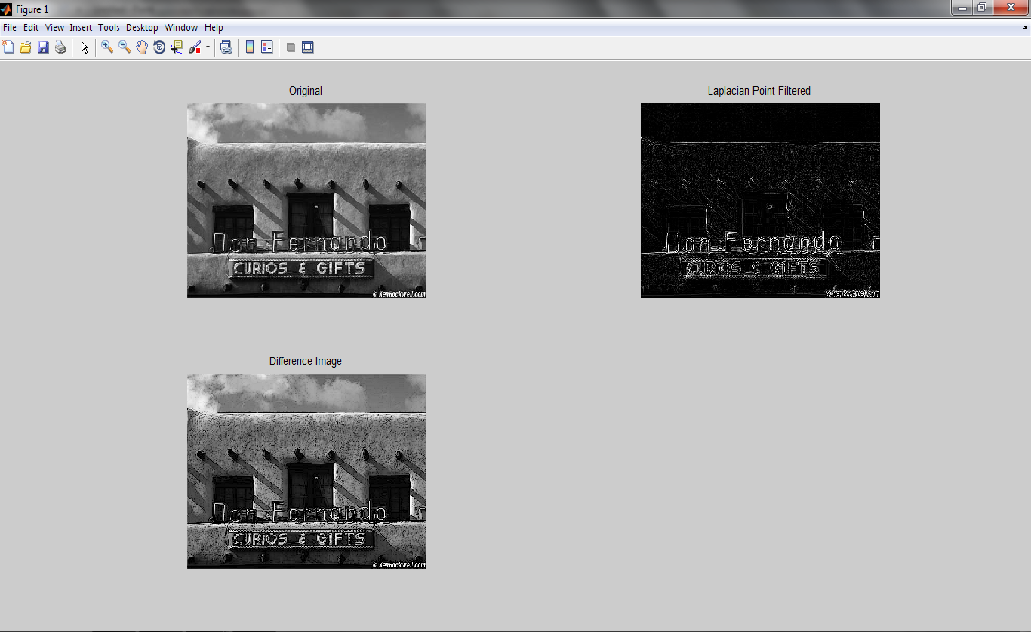
**6.2) About Form**

****

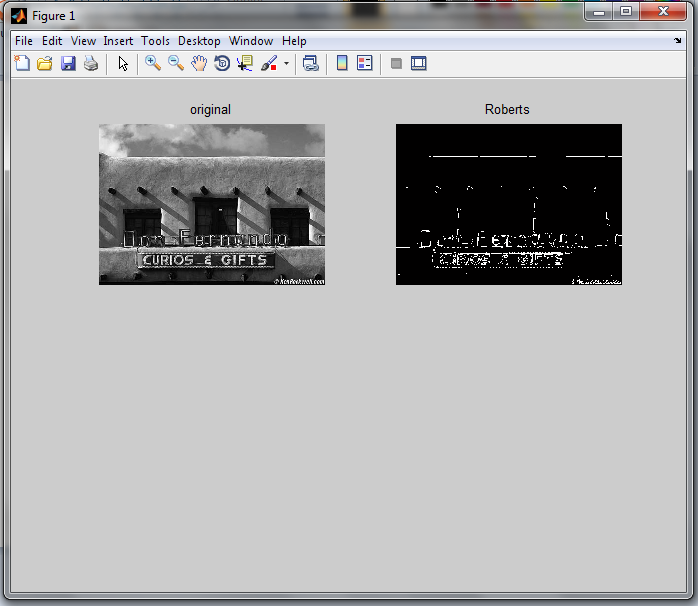
**6.3) Functionalities**

****

**6.4) Point Detection**

****

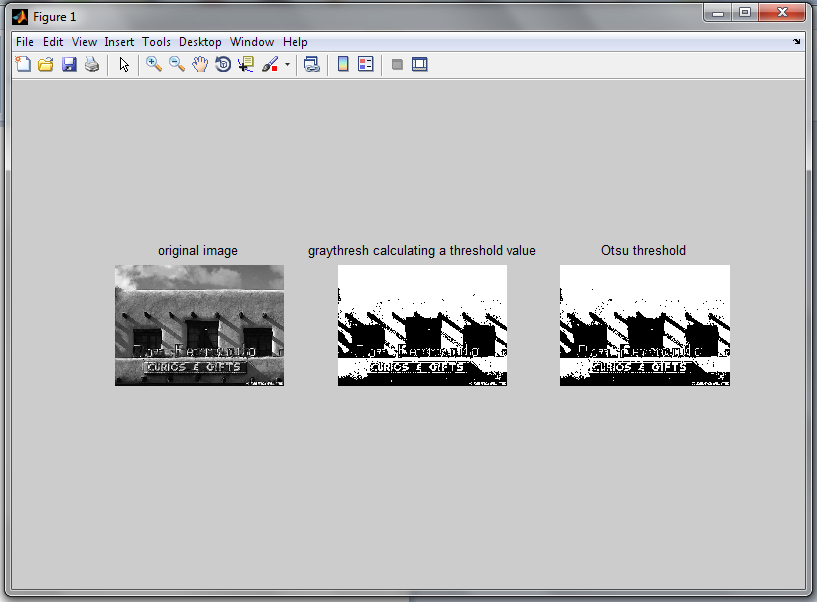
**6.5) Edge Detection**

****

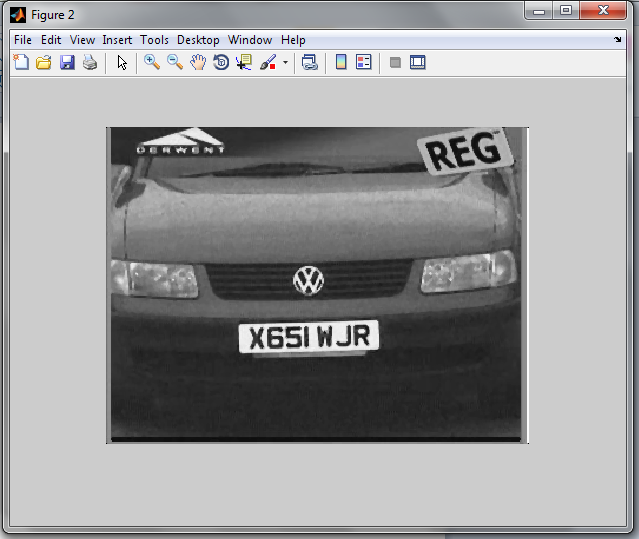
**6.6) Angular Rotation**

****

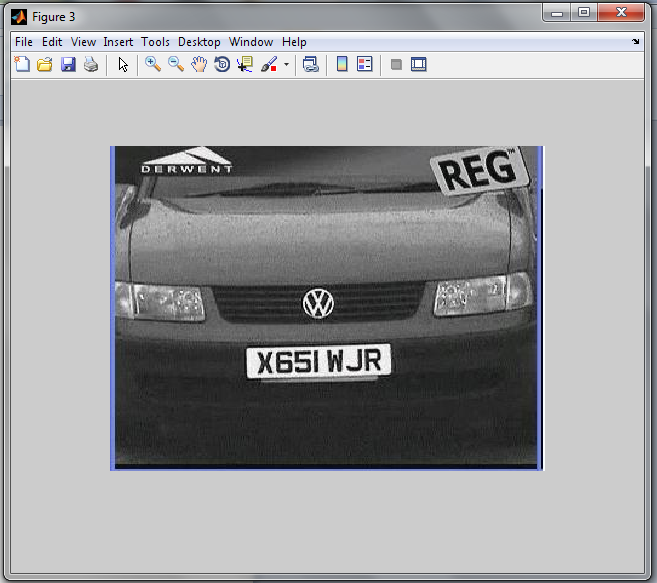
**6.7) Segmentation**

****

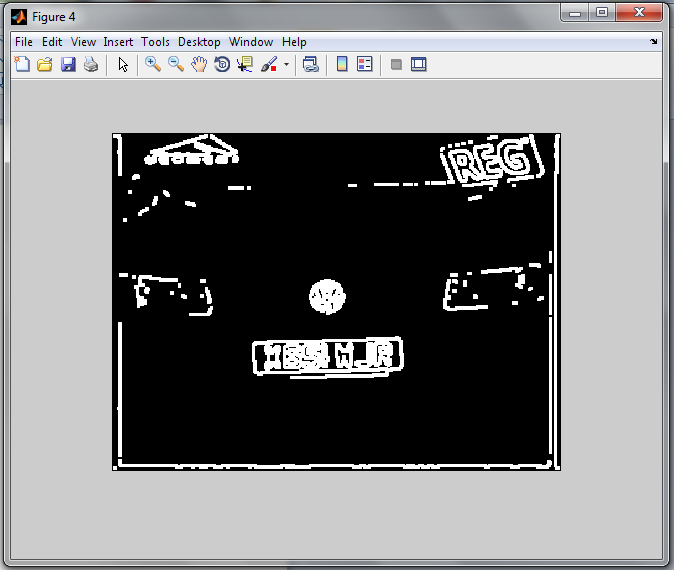
**6.8) Text Extraction - Original Image**

****

**6.9) Text Extraction – Text outlined**

****

**6.10) Text Extraction – Text Highlighted (Output Image )**

****

**Chapter-7**

**Bibliography**

**Bibliography**

**Books Referred:-**

* Digital Image Processing  
   - Rafael C. González and Richard Eugene Woods

**Sites surfed:-**

* Journal of Theoretical and Applied Informatin Technology -<http://www.jatit.org/volumes/research-papers/Vol16No1/6Vol16No1.pdf>
* Text Informatin Extraction In Images - <http://www.cse.msu.edu/prip/Files/TextDetectionSurvey.pdf>
* Wikipedia -   
  [www.wikipedia.org/](http://www.wikipedia.org/)