**CHAPTER 1**

**INTRODUCTION**

**CHAPTER 1**

# INTRODUCTION

# 1.1 PATTERN RECOGNITION

In machine learning, pattern recognition is the assignment of a label to a given input value. An example of pattern recognition is classification, which attempts to assign each input value to one of a given set of classes (for example, determine whether a given email is "spam" or "non-spam"). However, pattern recognition is a more general problem that encompasses other types of output as well. Other examples are regression, which assigns a real-valued output to each input; sequence labelling, which assigns a class to each member of a sequence of values (for example, part of speech tagging, which assigns a part of speech to each word in an input sentence); and parsing, which assigns a parse tree to an input sentence, describing the syntactic structure of the sentence.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform "most likely" matching of the inputs, taking into account their statistical variation. This is opposed to pattern matching algorithms, which look for exact matches in the input with pre-existing patterns. A common example of a pattern-matching algorithm is regular expression matching, which looks for patterns of a given sort in textual data and is included in the search capabilities of many text editors and word processors. In contrast to pattern recognition, pattern matching is generally not considered a type of machine learning, although pattern-matching algorithms (especially with fairly general, carefully tailored patterns) can sometimes succeed in providing similar-quality output to the sort provided by pattern-recognition algorithms.

Pattern recognition is studied in many fields, including psychology, psychiatry, ethology, cognitive science, computer science and traffic flow.

**1.2 MOTIVATION**

Computer vision aims at building machines able to view and to recognize objects and features as humans are able to do. In recent years, computer vision and psychology have extensively studied visual perception and object recognition and made significant progress, but current techniques for object recognition and shape classification do not yet provide entirely satisfactory solutions. Objects have several properties that can be used for recognition and categorization, like shape, color, texture and brightness. The property of shapes is used to provide a well performing algorithm for shape classification.

**1.3 SHAPE CLASSIFICATION**

There are two major approaches for shape representation in the literature: one approach is boundary based and uses contour information, and the second approach needs a holistic representation, requiring general information about the shape. Shape-based methods primarily analyze silhouettes, i.e boundaries of objects present in images. Silhouettes do not have holes or internal markings, and therefore are represented by a single closed curve, parameterized by its arc-length. Shape context is a descriptor developed for finding correspondences between point sets. Given a set of points from an image (e.g. extracted from a set of detected edges), the shape context captures the relative distribution of points in the plane relative to each point on the shape. A shape is represented by a discrete set of points sampled from the internal or external contour on the object. The shape contexts have been used as attributes for a weighted bipartite matching problem. Attalla and Siy have presented a polygonal approximation of shape contours that divide it into equal segments and all segments will serve as local features that will represent the shape. The method of Fourier descriptor is extended to object recognition to produce a set of normalized coefficients which are invariant under any affine transformation. The method is based on a parameterized boundary description which is transformed into the Fourier domain and normalized to eliminate dependencies on the affine transformation and on the starting point. The inner-distance between landmarks, i.e. the shortest path between these landmarks, has also been used to build shape descriptors. A Dynamic Programming (DP) approach has been used for shape matching and retrieval. The basic idea behind this approach is to represent each shape by a sequence of convex and concave segments using the inflection points extracted from the curvature and to allow the matching of merged sequences of small segments in a noisy shape with larger segments in the other shape. This procedure is obtained by a DP algorithm searching for the least cost match in a DP table.

Some of the object recognition methods rely on holistic representation of shapes. Geometric invariants are shape descriptors that remain unchanged under geometric transformations such as projections or changes in point of view. These invariant descriptors can be obtained locally or globally.

In the algorithm described here, we use contour of shapes and shape context descriptor. Using DP, we classify the shapes. We have tested for set of databases like kimia-25, kimia-99, kimia-216 and also for complex database like MPEG-7.

**1.4 PROBLEM STATEMENT**

Suppose there is a set of shapes in a database, which is divided into N classes and n shapes in each class. Given an input shape from the database, the algorithm should match the input shape to the class to which it belongs.

**CHAPTER 2**

**LITERATURE SURVEY**

**CHAPTER 2**

# LITERATURE SURVEY

This section enlists the title of the papers along with the brief description about them. The same were referenced during the course of completion of our project. These papers provided us with initial impetus and required information during the project completion.

In this paper, Shape Matching And Object Recognition Using Shape Contexts [1] Serge Belongie, Jitendra Malik and Jan Puzicha published in 2002, a new shape descriptor, the shape context for correspondence recovery and shape based object recognition is introduced.

Shape context greatly simplify recovery of correspondences between points of two given shapes. Moreover shape context leads to a robust score for measuring shape similarity, once shapes are aligned. A novel approach to measuring similarity between shapes and exploit it for object recognition. The measurement of similarity is preceded by (1) solving for correspondences between points on the two shapes, (2) using the correspondences to estimate an aligning transform. In order to solve the correspondence problem, we attach a descriptor, the shape context, to each point. The shape context at a reference point captures the distribution of the remaining points relative to it, thus offering a globally discriminative characterization. Corresponding points on two similar shapes will have similar shape contexts, enabling to solve for correspondences as an optimal assignment problem. Given the point correspondences, we estimate the transformation that best aligns the two shapes; regularized thin plate splines provide a flexible class of transformation maps for this purpose. The dissimilarity between the two shapes is computed as sum of matching errors between corresponding points, together with a term measuring the magnitude of aligning transform. We treat recognition in a nearest neighbour classification framework as the problem of finding the stored prototype shape that is maximally similar to that in the image.

In this paper, A survey of shape analysis techniques [2] by Sven Loncaric, a review of shape analysis methods is provided. Shape analysis methods play an important role in systems for object recognition, matching, registration, and analysis. Research in shape analysis has been motivated, in part, by studies of human visual form perception systems. Several theories of visual form perception are briefly mentioned. Shape analysis methods are classified into several groups. Classification is determined according to the use of shape boundary or interior, and according to the type of result. An overview of the most representative methods was presented.

In this paper, Partial Shape Matching (PSM) of Spine X-Ray Shapes Using Dynamic Programming [3] by Sameer Antani, L.Rodney Long, PSM methods for matching shapes with variable number of points and with different data point distributions is presented. Dynamic Programming (DP) is proposed for matching partial shapes by allowing merging of the data points in the process of PSM. DP is implemented based on two shape representation methods: line segments and multiple open triangles. The performance evaluation, which is based on human relevance judgments of these two shape representations in PSM, is also presented. DP searches for all possible matching paths and selects the most promising one with the minimum distance. The distance consists of the differences between the corresponding shape features extracted from both query shape and the candidate shape and the merging cost associated with a merging process. Since a shape matching algorithm must be based on the properties of its underlying representation, DP was implemented slightly different for these two shape representations. In partial shape matching, matching for variable number of points is as important as matching for fixed number of points. We have applied DP to two shape representations: line segments and multiple open triangles. The results show potential for solving problems with shape descriptions with difference point distributions.

In this paper, a survey of shape feature extraction techniques by Yang Mingqiang, Kpalma Kidiyo, Ronsin Joseph published in july, 2008 [4], the existing approaches of shape-based feature extraction is focused. Efficient shape features must present some essential properties such as

**Identifiability:** Shapes which are found perceptually similar by human have the same feature different from the others.

**Translation, rotation and scale invariance:** The location, rotation and scaling changing of the shape must not affect the extracted features.

**Affine invariance:** The affine transform performs a linear mapping from 2D coordinates to other 2D coordinates that preserves the "straightness" and "parallelism" of lines. Affine transform can be constructed using sequences of translations, scales, flips, rotations and shears. The extracted features must be as invariant as possible with affine transforms.

**Noise resistance:** Features must be as robust as possible against noise, i.e., they must be the same whichever be the strength of the noise in a give range that affects the pattern.

**Occultation invariance:** When some parts of a shape are occulted by other objects, the feature of the remaining part must not change compared to the original shapes.

**Statistically independent:** Two features must be statistically independent. This represents compactness of the representation.

**Reliable:** As long as one deals with the same pattern, the extracted features must remain the same.

In general, shape descriptor is some set of numbers that are produced to describe a given shape feature. A descriptor attempts to quantify shape in ways that agree with human intuition. Good retrieval accuracy requires a shape descriptor to be able to effectively and perceptually similar shapes from a database. Usually, the descriptors are in the form of a vector. Shape descriptors should meet the following requirements:

The descriptors should be as complete as possible to represent the content of the information items. The descriptors should be represented and stored compactly. The size of descriptor vector must not be too large. The computation of distance between descriptors should be simple; otherwise the execution time would be too long.

Shape feature extraction and representation plays an important role in the following categories of application:

**Shape retrieval**: Searching for all shapes in a typically large database of shapes that are similar to a query shape. Usually all shapes within a given distance from the query are determined or the first few shapes that have the smallest distance.

**Shape recognition and classification:** Determining whether a given shape matches a model sufficiently or which of representative class is the most similar.

Shape alignment and registration: transforming or translating one shape so that it best matches another shape, in whole or in part.

**Shape approximation and simplification:** Constructing a shape of fewer elements (points, segments, triangles, etc.), that is still similar to the original.

Many shape description and similarity measurement techniques have been developed in the past. A number of new techniques have been proposed in recent years. There are 3 main different classification methods as follows:

**Contour-based methods and region-based methods**: This is the most common and general classification and it is proposed by MPEG-7. It is based on the use of shape boundary points as opposed to shape interior points. Under each class, different methods are further divided into structural approaches and global approaches. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections (primitives).

**Space domain and transform domain**: Methods in space domain match shapes on point (or point feature) basis, while feature domain techniques match shapes on feature (vector) basis.

**Information preserving (IP) and non-information preserving (NIP):** IP methods allow an accurate reconstruction of a shape from its descriptor, while NIP methods are only capable of partial ambiguous reconstruction. For object recognition purpose, IP is not a requirement.

In this paper, Shape Retrieval Based on Dynamic Programming [5] by Evangelos Milios and Euripides G.M. Petrakis published in 2001, a shape matching algorithm for deformed shapes based on dynamic programming is proposed. The algorithm is capable of grouping together segments at finer scales in order to come up with appropriate correspondences with segments at coarser scales. It illustrates the effectiveness of the algorithm in retrieval of shapes by content on two different two dimensional datasets, one of static hand gesture shapes and another of marine life shapes. They also demonstrate the superiority of the approach over traditional approaches to shape matching and retrieval, such as Fourier descriptors, Geometric and Sequential moments. The evaluation is based on human relevance judgments following a well established methodology from the information retrieval field. The shape matching algorithm that lies at the core of the methodology takes in two shapes and computes (a) their distance and (b) the correspondences between similar parts of the two shapes. In retrievals, only the distances are used. However, the correspondences help assess the plausibility of the distance computation, if necessary. The algorithm is based on dynamic programming, performs implicitly at multiple scales and allows the matching of deformed shapes. The superiority over traditional approaches to shape matching and retrieval (Fourier descriptors, Geometric and Sequential moments) using two different datasets with 980 and 1,100 shapes respectively. The paper also introduces to the Computer Vision community a well-established methodology for the evaluation of the retrieval results obtained by more than one competing methods. Current research is directed towards extending the matching algorithm for open curves. Future work includes the experimentation with more datasets and methods, and the handling of combined queries involving more than one feature (e.g., shape, color, text).

In this paper, Using Dynamic Programming for Solving Variational Problems on Vision [6] by Amir A Amini , Terry E Weymouth, Ramesh C Jain, variational approaches have been proposed for solving many inverse problems in early vision, such as in the computation of optical flow, shape from shading, and energy-minimizing active contour models. In general however, variational approaches do not guarantee global optimality of the solution, require estimates of higher order derivatives of the discrete data, and do not allow direct and natural enforcement of constraints. In this paper dynamic programming is proposed as a novel approach for solving variational problem in vision. Dynamic programming ensures global optimality of the solution, it is numerically stable, and it allows for the hard constraints to be enforced on the behavior of the solution within a natural and straight forward structure. As a specific example of the efficacy of the proposed approach, application of dynamic programming to the energy minimizing active contours is described. The optimization problem is set up as a discrete multistage decision process and is solved by a “time delayed” discrete dynamic programming algorithm. A parallel procedure is discussed that can result in savings in computational costs.

In the paper, Retrieval Of Deformed and Occluded Shapes Using Dynamic Programming [7] by Zusheng Rao, Euripides G.M. Petrakis, Evangelos Milios published in 1999, an approach for matching deformed shapes using dynamic programming (DP) is proposed. The algorithms handle noise and shape distortions by allowing matching of merged sequences of consecutive small segments in a shape, with larger segments of another shape. The proposed algorithms handle occlusion while being invariant to translation, scale and orientation transformations of shapes. It illustrates the effectiveness of our algorithms in retrieval of shapes on two different two-dimensional datasets, one of static hand gesture shapes and another of marine life shapes. The evaluations are based on human relevance judgments and the results are a good support to our claims of accuracy. In this work they focus on DP methods and they propose shape matching algorithms for occluded and deformed shapes. If the two shapes are scaled with respect to each other, the algorithm determines the appropriate scale for matching. The algorithms find the best association between segments of one shape and segments of the other shape. This is formulated as a minimization problem which is solved efficiently by dynamic programming: A table of partial costs is built and the optimal complete matching is searched in the form of a path in the DP table that minimizes a total dissimilarity cost. The algorithms are optimal, that is, they always find the least cost path. They report experimental results of the algorithms on a data set of 980 two-dimensional hand gesture shapes and on a marine life database with 1,500 shapes.

In the paper, New Method For Shape Recognition Using Dynamic Programming[8] by Noredinne Gherabi and Mohamed Bahaj, a new method for shape recognition is presented based on dynamic programming. First, each contour of shape is represented by a set of points. After alignment and matching between two shapes, the outline of the shape is divided into parts according to N angular and M radial sectors, Each Sector contains a portion of the contour; this portion is divided at the inflexion point into convex and concave sections, and the information about sections are extracted in order to provide a semantic content to the outline shape, then this information are coded and transformed into a string of symbols. Finally the best alignment of two complete strings and computes the optimal cost of similarity. In this paper, a new approach to finding the best matching between two shapes using the technique of dynamic programming is presented. A fast and efficient algorithm to find a good similarity between the shapes is developed. The use of a dynamic programming approach greatly accelerates the similarity research, and makes it possible to handle an extremely large amount of similarity matching. Application of the approach to shape recognition and shape retrieval is demonstrated, and obtained a better performance in comparison with some previous methods.

**CHAPTER 3**

**PROPOSED METHOD**

**CHAPTER 3**

**PROPOSED METHOD**

**3.1 OUTLINE OF PROPOSED APPROACH**

The proposed algorithm for shape retrieval and recognition is based on several steps summarized in Fig.3.1. The algorithm is based on the analysis of the contour of the pair of shapes under consideration. Its contour is recovered and mapped into a pair of N points. The cost of matching between points pi and qj from the two shapes is evaluated by the shape context which we will briefly describe further. Having the cost of matching each point from one shape with all the points in the other shape, by using dynamic programming we obtain the best matching between the point sets of the two shapes. Dynamic programming not only recovers the best matching between points, but also identifies outliers, i.e. points in the two shapes which cannot be properly matched. This step is essential for identifying partial occlusions in the two shapes. The cost of matching obtained from dynamic programming is used for classification. The proposed approach has been tested on most public shape database.

**3.2 Shape context**

Shape context has been introduced by Belongie. An object is represented by a discrete set of points sampled regularly along the contour. For every point, a log-polar histogram, the shape context is computed approximating the distribution of adjacent point locations relative to a reference point. In order to achieve scale invariance, the outer radius for the histograms is set equal to the mean distance between all the pair points.

 ****

**Shape classification based on minimum cost of matching**

**Class: Hand**

**Fig.3.1** Outline of proposed approach

For a point pi i=1….Nof the shape, the shape context is a coarse histogram hiof the relative coordinates of the remaining qj N−1 points:



The bins are uniform in log-polar space, making the descriptor more sensitive to positions of nearby points than those of more distant points. The cost of matching a pair of points piand qj from two shapes is computed as

****

Where hi(k) and hj(k) denote the K-bin normalized histogram at pi and qj , respectively. This definition of shape context is not rotationally invariant, but can be easily modified to be a completely rotational invariant descriptor. There are several ways to obtain rotational invariance. One possibility is to make the reference angle for shape context equal to the local tangent angle on the shape. Another option is to align the shape context axis with the principal axis of the shape. Shape context is not invariant for mirror transformation; therefore we consider also the mirrored shape when computing the shape context.



**Fig 3.2** A scheme of representation of point sets using shape context .A shape is represented by a discrete set of points sampled regularly along the contours. For every point, a log-polar histogram, the shape context is computed which approximates the distribution of adjacent point locations relative to the reference point.

**3.3 Dynamic programming for point correspondence**

The origin of the term dynamic programminghas very little to do with writing code. It was first coined by Richard Bellman in the 1950s, a time when computer programming was an esoteric activity practiced by so few people as to not even merit a name. Back then programming meant .planning, and dynamic programming was conceived to optimally plan multistage processes. Dynamic programmingis a very powerful algorithmic paradigm in which a problem is solved by identifying a collection of sub problems and tackling them one by one, smallest first, using the answers to small problems to get the solution for larger ones, until the whole lot of them is solved.

Dynamic Programming (DP) has already been used for many applications especially for contour matching. These applications consider sequences of convex/concave segments—separated by inflection points forming the two shapes under consideration with the goal of finding the best match of segments in shapes A and B. This is formulated as a minimization problem which is solved efficiently by DP. The algorithm builds a DP table or matrix, where rows and columns correspond to inflection points of the two shapes A and B, respectively.

The Dynamic Programming (DP) algorithm has two main steps providing the correspondence between the point sets and removing the outliers (and occlusions). The first step consists in filling cell by cell a DP table identifying possible outliers and occlusions. The second step tracks the minimum path over the DP table. In our case, we have two point sets describing the contour of the two shapes and we want to find the best correspondence between these two point sets and identify outliers. Given two shapes A and B, each described by a set of Npoints, given piin shape A and qjin shape B, the cost of matching piand qjis given by Eq. (2), which are the entries of a suitable cost matrix C(i, j). This step is based on a DP algorithm based on the following procedure:

Let us suppose we have a penalty threshold pfor an occlusion (i.e. an outlier) and we have another suitable DP(i, j )matrix with the same size of the cost matrix Cwith zero entries. We fill each cell in the table by considering the penalty value of pand three previous entries in the DP table, namely left cell, upper cell and the diagonal cell, and we fill it. During this process, at each step, we keep track of the minimum among the three previous entries so as to find the minimum path after filling the matrix. Indeed, we start from the last entry and, based on the conditions that have given the minimum values at each step, we track back and find the complete path that gives us the corresponding points and outlier.

The basic idea for dynamic programming is as given below:

Let DP be the dynamic programming table and C be the cost matrix obtained after shape context. The entries to Dynamic Programming (DP) table are as shown in below figure 3.3. Entries DP(i-1, j-1), DP(i-1,j), and DP(i, j -1) are needed to fill in DP(i, j).



**Fig 3.3** DP table

Any order is fine, as long as DP(i-1, j), DP(i,j-1), and DP(i-1,j-1) are handled beforeDP(i,j). For instance, we could fill in the table one row at a time, from top row to bottom row, and moving left to right across each row or alternatively, we could fill it in column by column. Both methods would ensure that by the time we get around to computing a particular table entry, all the other entries we need are already filled in. After filling all the entries, the goal i.e DP(m,n) represents the minimum cost of matching. Shape classification is done using this data for all point sets.

**CHAPTER 4**

**IMPLEMENTATION**

**CHAPTER 4**

**IMPLEMENTATION**

**FLOW DIAGRAM OF THE PROPOSED METHOD**

Number of classes, number of images per class. Set the parameters as bin distance = 5, bin angle=12, no of samples =100

START

Extract the longest contour of the image.

Down sample the contour for no of samples.

N

All images of all classes are done?

Y

Save contour data of all images in file

Calculate Shape Context for each point

B

All images of all classes except IClass are done?

Using DP, find the cost of matching.

Get the image which has minimum cost of matching.

Calculate the cost matrix using the SC of all points of image IClass and other image. The cost matrix contains cost of matching one point on first image and another point on second image.

Consider a image to be classified(IClass). Get its contour and SC from the files created in previous steps.

Save SC data of all images in file

Y

All images of all classes are done?

B

D

Display the class to which the input shape belongs. After shape classification if it belongs to same class then it is classified perfectly, else it is a mismatch.

**Fig4** Flow diagram of the proposed method

Consider a shape database. Identify the number of classes and number of objects per class. Set the parameters such as bin distance, bin angle which is used in shape context. Find the longest contour for each of the shapes in the database and mark 100 points on the contour uniformly.

**Calculating shape context**

* To calculate shape context for a shape do the following

Consider a point in the shape; with respect to that point calculate the bin count of all other 99 points. The bin count is calculated using Euclidean distance from the point under reference to all other remaining points.

The Euclidean distance or Euclidean metric is the "ordinary" [distance](http://en.wikipedia.org/wiki/Distance) between two points that one would measure with a ruler, and is given by the [Pythagorean formula](http://en.wikipedia.org/wiki/Pythagorean_theorem). By using this formula as distance, Euclidean space (or even any [inner product space](http://en.wikipedia.org/wiki/Inner_product_space)) becomes a [metric space](http://en.wikipedia.org/wiki/Metric_space). The associated [norm](http://en.wikipedia.org/wiki/Norm_%28mathematics%29) is called the [Euclidean norm](http://en.wikipedia.org/wiki/Norm_%28mathematics%29#Euclidean_norm). Older literature refers to the metric as Pythagorean metric.

If **p** = (p1, p2) and **q** = (q1, q2) then the distance is given by

\mathrm{d}(\mathbf{p},\mathbf{q})=\sqrt{(p_1-q_1)^2 + (p_2-q_2)^2}.

Also, the angle from the line joining the point under reference and all other 99 points and the X-axis should be measured using the formula

Angle= atan2(

Now normalize all the distances and angles to determine which bin it belongs to. For example bin1=0>=angle>=30 and 0>=d(p,q)>=r1 where r1 is the first radius considered. Since there are 5 distance bins and 12 angle bins, we get a total of 60 bin count for the point under reference. Hence, we get number of points belonging to each bin with reference to point under reference.

* We repeat the procedure for all the points in shape and normalize it by dividing by number of sample points considered i.e 100.
* Repeat same procedure for all the shapes in database and store it in a file.

**Cost of matching**

* Consider an input shape. In order to calculate the cost of matching for the input shape and every reference shape we use the formula:



Where, pi- point i of input shape where i is ranging from 1 to 100.

qj- point j of reference shape where j is ranging from 1 to 100.

hi(k)- count with respect to point i in input shape in kth bin.

hj(k)- count with reference to point j in reference shape in kth bin

k- number of bins(i.e 1 to 60)

* This will result in 100X100 matrix for every pair of input and reference shape.

**Dynamic programming (calculating DP table)**

* Given the cost of matching dynamic programming finds the minimum cost of matching for every input and reference shapes.
* The size of the DP table is same as cost matrix.
* The algorithm for dynamic programming is as follows

The penalty is given by average of cost of matching \* threshold. Here threshold is considered as 0.6.

Average of cost matching is given by sum of values in cost matrix C divided by number of values in C.

Let Dynamic Programming (DP) matrix represent the DP table

Initialize:

DP(1,1) = min [ C(1,1) , penalty ]

For the first row DP table is filled as:

For j: 2 to 100

DP( 1,j ) = min [ C(1,j) + (j-1) \* penalty , penalty + DP(1,j-1) ]

For the first column DP table is filled as:

For i: 2 to 100

DP(i,1) = min [ C(i,1) + (i-1) \* penalty , penalty +DP(i-1,1) ]

For remaining cell DP table is filled as:

For i: 2 to 100

For j: 2 to 100

DP(i,j) = min [ C(i,j) + DP(i-1,j-1) , penalty + DP(i-1,j) , penalty + DP(i,j-1)

After filling all the cells DP(100,100) gives the minimum cost for input and reference shape considered.

**Shape classification**

* Calculate DP for input and all reference shape.
* The reference shape which gives minimum cost for input shape is considered as the best match and classified to the class to which the reference shape belongs.

**CHAPTER 5**

**RESULTS**

**CHAPTER 5**

**RESULTS**

In our experiments the contour of each shape is represented by 100 points. During the computation of the shape context we used 5 and 12 bins for computing the histograms for distances and angles, respectively. The penalty value for detecting the outliers was set equal to 0.6 of the mean cost C.

**5.1 Kimia-25 shape data set**

The fig 5.1 shows kimia-25 data set.

****

**Fig 5.1** The kimia-25 data set consists of 25 shapes from six different categories: each row shows a different object category.

Kimia-25 provided by Sharvit consists of 25 shapes from six different categories as shown in Fig. 5.1. In each category we have four images. Since our algorithm works for equal number of objects in each class, we have taken only 24 images. Out of 24 images, 20 images have properly matched which gives a performance of 83.33%.

**5.2 Kimia-99 data set**

Kimia-99 dataset is shown in Fig 5.2.

****

**Fig. 5.2** Kimi-99 database: Each row shows a different object category.

This database consists of 99 shapes from nine different categories as shown in Fig. 4.2. In each category we have eleven images. Out of 99 images, 91 images have properly matched which gives a performance of 91.9%.

**5.3 Kimia-216**

The figure 5.3 shows the examples of shapes in Kimia-216 data set.

****

**Fig 5.3** Examples of kimia-216 data set.

This database consists of 216 shapes from eighteen different categories as shown in Fig. 5.3. In each category we have twelve images. Out of 216 images, 201 images have properly matched which gives a performance of 93%.

**5.4 MPEG-7**

Figure 5.4 shows examples of shapes in the MPEG-7 data set.

This database consists of 1400 shapes from 70 different categories as shown in Fig. 5.4. In each category we have twenty images. Out of 1400 images, 1269 images have properly matched which gives a performance of 90.64%.



**Fig 5.4** Examples of shapes in the MPEG-7 Database. One object from each one of the seventy categories is shown

***Global: L = 30, x = 0***

***L = 47, x = 0***

***Global: L = 30, x = 0***

**5.5 Final summary of performance**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Databases** | **Number of images taken** | **Number of classes** | **Number of objects** | **Number of images matched** | **Number of mismatches** | **Performance** |
| Kimia-25 | 24 | 6 | 4 | 20 | 4 | 83.33% |
| Kimia-99 | 99 | 9 | 11 | 91 | 8 | 91.9% |
| Kimia-216 | 216 | 18 | 12 | 201 | 15 | 93% |
| MPEG-7 | 1400 | 70 | 20 | 1269 | 131 | 90.64% |

**Table 5.5** Performance measure

**CHAPTER 6**

**APPLICATIONS**

**CHAPTER 6**

**APPLICATIONS**

**6.1 Military Applications**

Detection and classification of suspicious objects such as camouflaged bombs placed at roadside and in airports. Military uses pattern recognition to detect bomb planting activity.

A very simple implementation of this technique is to take a picture on Day 1, then another picture from the same location and in the same direction on Day 2. Next subtract the two images from each other and see what is left over. All the things that did not change between the two images will disappear and only those things that changed will be left! In this case, what will be see on the "difference image" might be some disturbed gravel, tire tracks off the side of the road (assuming it is a road-side bomb emplacement) or, if the military guys are really lucky, what will be left is a bunch of guys standing around with shovels and a bomb as they work to put it in place.

**6.2 Industrial Applications**

Include those in factories(inspection and robotics), in offices(optical character recognition) and in exotic niches such as remote sensing and biomedicine.

**6.3 Protein Classification**

Given a new protein sequence its functions and properties can be predicted by pairwise comparisons with the sequences of other proteins whose properties are already known. The study of proteins, generally under the heading of proteomics, is a vast and complex subject, and much effort has been made to classify and categorize, according to the many specific fields of investigation under which they come.

**6.4 Multimedia**

In multimedia it is used for segmentation, recognition, indexing of speech, audio, image and video data. It is also used to analyze social media and investigate how people interact in a networked world.

**6.5 Document Analysis**

One of the largest application areas of pattern recognition methods is the field of document analysis and recognition. The problems addressed range from optical character recognition (i.e. reading machine printed texts), postal address reading, handwriting recognition, processing of historical documents, and document image retrieval to pen computing for man-machine interaction.

**6.6 Food Analysis**

Supervised pattern recognition is used in food analysis i.e. food authentication (verification that a food stuff comes from a alleged origin.Data analysis has become a fundamental task in analytical chemistry due to the great quantity of analytical information provided by modern analytical instruments. Supervised pattern recognition aims to establish a classification model based on experimental data in order to assign unknown samples to a previously defined sample class based on its pattern of measured features. The basis of the supervised pattern recognition techniques mostly used in food analysis are reviewed, making special emphasis on the practical requirements of the measured data and discussing common misconceptions and errors that might arise. Applications of supervised pattern recognition in the field of food chemistry appearing in bibliography in the last two years are also reviewed.

**CHAPTER 7**

**CONCLUSION**

**CHAPTER 7**

**CONCLUSION**

The algorithm for shape classification described has been tested on a variety of shape databases and most of them provide better performances. Indeed for the Kimia-25, kimia-99, kimia-216 and MPEG-7. The proposed algorithm provides good results for classification. The algorithm here is based on two major properties: shape context i.e distribution of adjacent point locations relative to a reference point. Shape context descriptor is considered to give a better performance when compared to the previous fourier descriptor method. Second property is the dynamic programming which illustrates the capability of best matching between two shapes using dynamic programming. Dynamic programming is considered as the best method for finding correspondences between two shapes and also in detecting outliers. The minimum cost of matching obtained from dynamic programming is used to classify shapes. We have tested for a few standard databases which gives performance approximately equal to 91. When tested for simple databases like kimia-99, kimia-216, a better performance is obtained compared to complex databases like MPEG-7.

**CHAPTER 8**

**SCOPE FOR FUTURE WORK**

**CHAPTER 8**

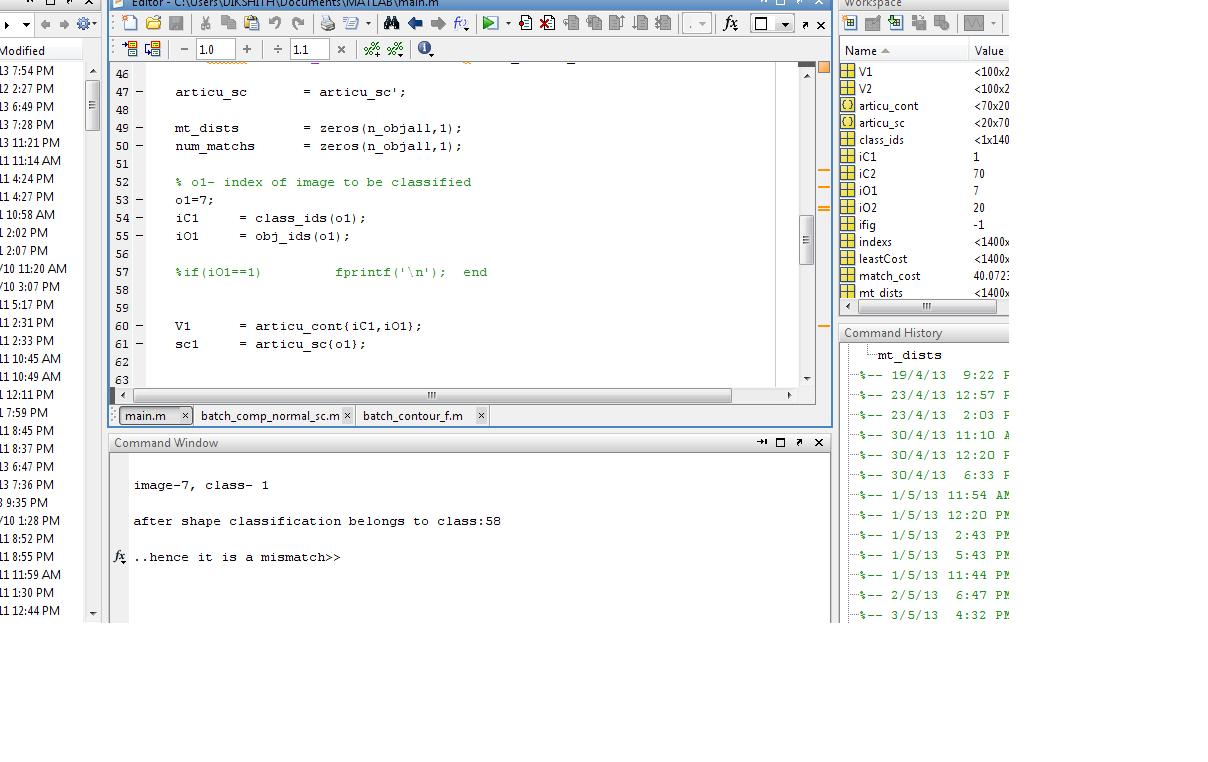
**SCOPE FOR FUTURE WORK**

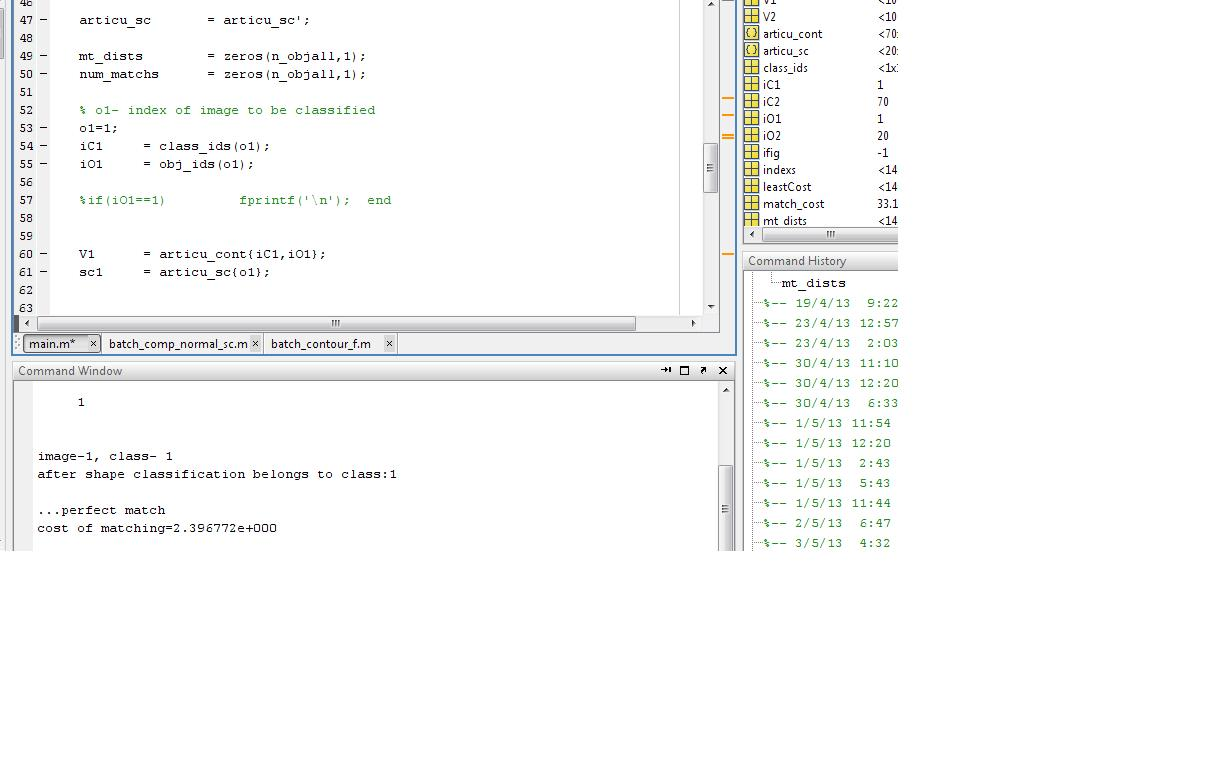
Further recognition and retrieval rate can be improved by using hybrid feature extractors and by using different distance classifiers.

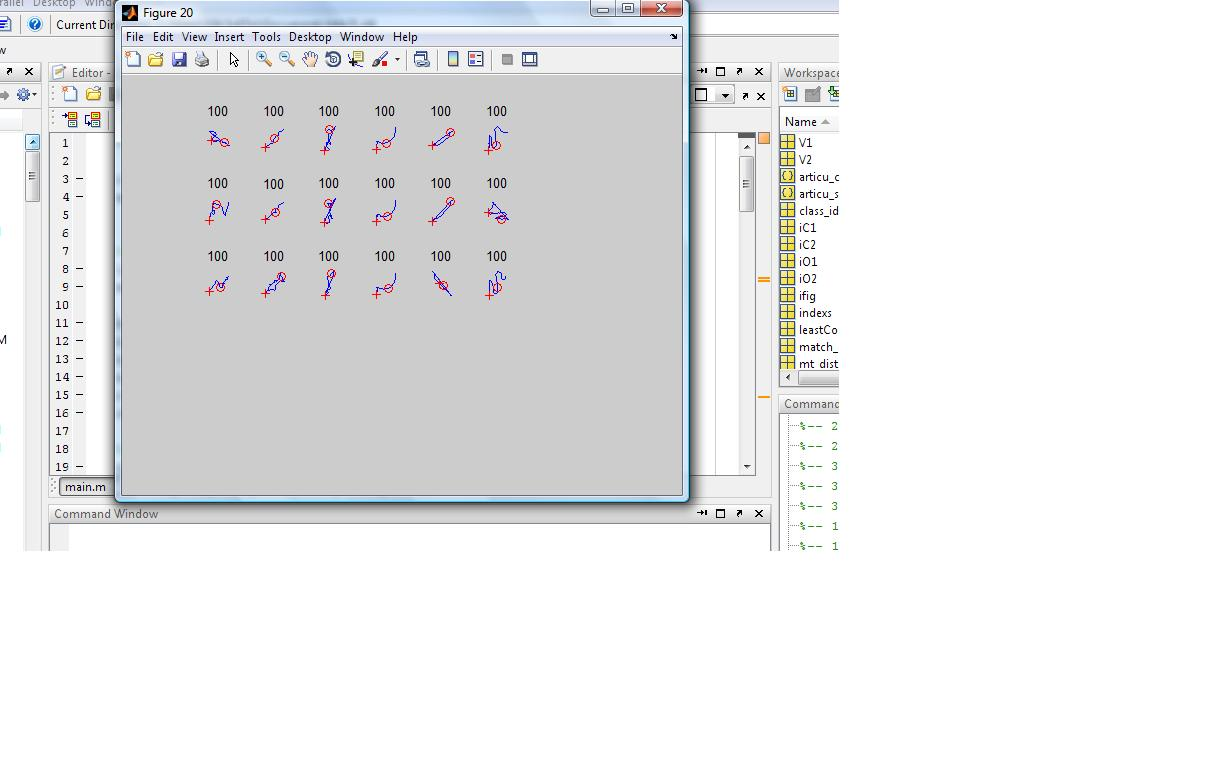
The concept of string matching can be used after dynamic programming. Several algorithms can be used, such as the edit or Levenstein distance . The edit distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is either an insertion or a deletion, or a substitution of a single character. The edit distance is a generalization of the usual Hamming distance for strings of the same length, where only substitutions are considered. A DP algorithm is used for computing the edit distance. Finally, recognition and retrieval are obtained by a simple nearest-neighbor procedure. The symbolic representation of a shape is based on the combination of local and global information. Comparison among string of symbols is easy and fast and does not require any sophisticated computation. With the symbolic representation, shapes can be easily analyzed at a multi-resolution, allowing a higher accuracy. It seems also to be able to recognize shapes extracted from real and cluttered images and therefore is a useful tool for pattern recognition.

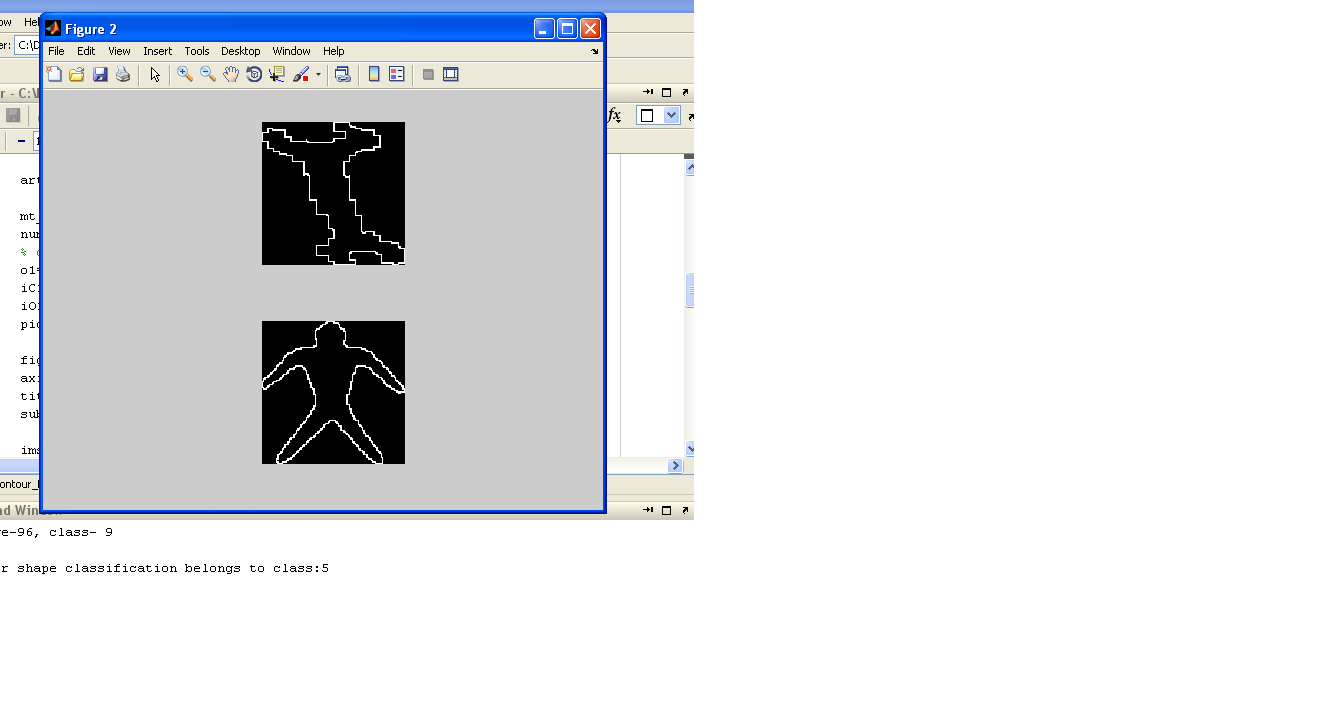
**SCREEN SHOTS**

SCREEN SHOTS

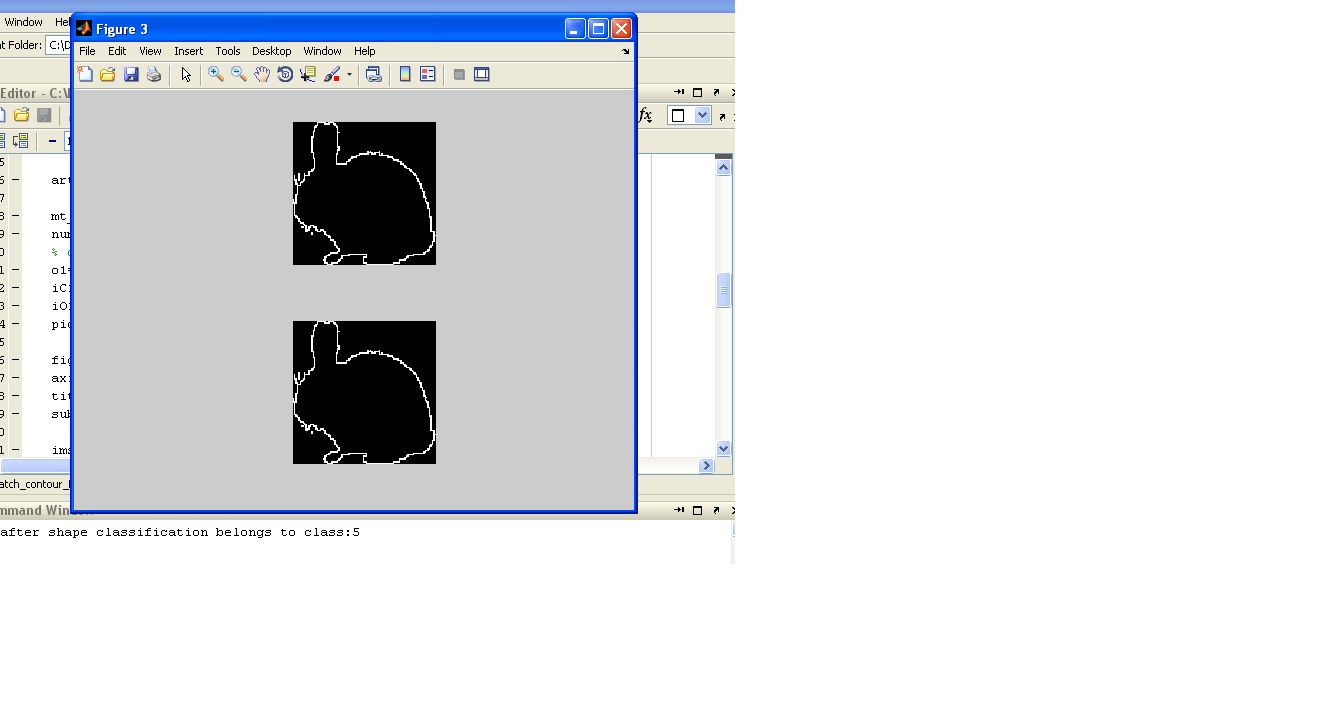
**Figure a**. Screenshot of the output screen When the input shape doesnot match to the class to which it belongs, the output is a mismatch.

**Figure b.** Screenshot of output screen If a input shape matches correctly to the class to which it belongs, then the output is a perfect match

**Figure c.** Snapshot showing contours of shapes in kimia-25

****

**Figure d.** Snapshot showing input and output images during a mismatch in kimia-99 shape dataset

****

**Figure e.** Snapshot showing input and output images during a match in kimia-99 shape dataset

**REFERENCES**

**REFERENCES**

1. Serge Belongie, Jitendra Malik and Jan Puzicha, “Shape Matching And Object Recognition Using Shape Contexts “, 2002.
2. Sven Loncaric, “A survey of shape analysis techniques”.
3. Sameer Antani and L.Rodney Long, “Partial Shape Matching of Spine X-Ray Shapes Using Dynamic Programming”,
4. Yang Mingqiang, Kpalma Kidiyo, Ronsin Joseph, “A survey of shape extraction techniques”, july 2008.
5. Evangelos Milios and Euripides G.M. Petrakis, “Retrieval Based on Dynamic Programming”, 2001
6. Amir A Amini , Terry E Weymouth, Ramesh C Jain, “Using Dynamic Programming for Solving Variational Problems on Vision.
7. Zusheng Rao, Euripides G.M. Petrakis, Evangelos Milios, “Retrieval Of Deformed and Occluded Shapes Using Dynamic Programming”, 1999.
8. Noredinne Gherabi and Mohamed Bahaj, “Method For Shape Recognition Using Dynamic Programming”.
9. T. Sebastian, P.N. Klein, B. Kimia, “Recognition of shapes by editing their shock graphs”, IEEE Trans. Pattern Anal. Mach. Intell. 26(5) (2004) 550–571.
10. J.W. Gorman, O.R. Mitchell, F.P. Kuhl, “Partial shape recognition using dynamic programming”, IEEE Trans. Pattern Anal. Mach. Intell. 10 (2)(1988) 257–266.
11. T.H. Cormen, C.E. Leiserson, R.L. Rivest, ”Introduction to Algorithms”, MIT Press, Cambridge, MA, 2001.
12. B.Leibe, B.Schiele, “Analyzing appearance and contour based methods for object categorization”, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, 2003, 409–415.
13. M. Bober, “MPEG-7 visual shape descriptors”, IEEE Trans. Circuits Syst. Video Technol. 11 (6) (2001) 716–719.
14. G. McNeill, S. Vijayakumar, “A probabilistic approach to robust shape matching”, in: International Conference on Image Processing (ICIP), 2006
15. T. Sebastian, P.N. Klein, B. Kimia, “Recognition of shapes by editing their shock graphs”, IEEE Trans. Pattern Anal. Mach. Intell. 26(5) (2004) 550–571.
16. H. Ling, D.W. Jacobs, “Shape classification using the inner-distance”, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 286–299.
17. H. Freeman, “Computer processing of line-drawing images”, Comput. Surv. 6 (1) (1974) 57–97.
18. Biederman, G.Ju, ”Surface versus edge-based determinants of visual recognitions”, Cognit. Psychol. 20 (1988) 38–64.
19. S. Ullman, “High-Level Vision”, MIT Press, Cambridge, MA, 1996.
20. E. Rivlin, I. Weiss, “Local invariants for recognition”, IEEE Trans. Pattern Anal Mach. Intell. 17 (3) (1995) 226–238.

.

`

**APPENDIX**

**APPENDIX**

**IMPORTANT FUNCTIONS**

* Clear ALL: removes all variables, global, functions and MEX links.

* Sum: sum of elements.

S= sum(X) is the sum of elements of the vector X. If X is a matrix, S is a row vector with the sum over each column. For N-D arrays, sum(X) operates along the first non-singleton dimensions. If X is the floating point, that is double or single, S is a accumulated in double has class double.

* Imread: Read image from graphics file.

A = imread (FILENAME, FMT) reads a gray scale or color image from the file specified by the string FILENAME. If the file is not in the directory, or in a directory on the MATLAB path, specify the full pathname**.**

* Imshow: Display image in Handle Graphics figure.

Imshow (I) displays the gray scale image I.

.

* Subplot: Create axes in tiled positions.

H = subplot (m, n, p), or subplot (mnp), breaks the Figure window Into an m-by-n matrix of small axes, selects the pth axes for the current plot, and returns the axis handle. The axes are counted along the top row of the Figure window, then the second row, etc.

* Double: Convert to double precision.

Double (X) returns the double precision value for X. If X is already a double precision array, double has no effect.

.

* Min: Smallest component.

For vectors, min(X) is the smallest element in X. For matrices, Min (X) is a row vector containing the minimum element from each Column. For N-D arrays, min(X) operates along the first non-singleton dimension.

* Figure: Create figure window.

Figure, by itself creates a new figure window and returns its handle.

* Max: Largest component.

For vectors, max(X) is the largest element in X. For matrices, Max(X) is a row vector containing the maximum element from each column. For N-D arrays, max(X) operates along the first non-singleton dimension.

* Sort: Sort array elements in ascending or descending order.

sort(A) sorts the elements along different dimensions of an array, and arranges those elements in ascending order.

* Addpath : Adds the specified directory.

addpath('directory') adds the specified directory to the top (also called front) of the current MATLAB search path. Use the full pathname for directory.

* Fprintf: Write text to device.

fprintf(obj,'cmd') writes the string cmd to the device connected to the serial port object , obj. The default format is %s\n. The write operation is synchronous and blocks the command line until execution is complete.

* Contourc: Low-level contour plot computation.

contourc(Z) computes the contour matrix from data in matrix Z, where Z must be at least a 2-by-2 matrix. The contours are isolines in the units of Z. The number of contour lines and the corresponding values of the contour lines are chosen automatically.

* Find:Find indices and values of nonzero elements.

find(X) locates all nonzero elements of array X, and returns the linear indices of those elements in vector ind. If X is a row vector, then ind is a row vector; otherwise, ind is a column vector. If X contains no nonzero elements or is an empty array, then ind is an empty array.

* Plot: 2-D line plot.

plot(Y) plots the columns of Y versus their index if Y is a real number. If Y is complex, plot(Y) is equivalent to plot(real(Y),imag(Y)). In all other uses of plot, the imaginary component is ignored.

* Floor: Round toward negative infinity

floor(A) rounds the elements of A to the nearest integers less than or equal to A. For complex A, the imaginary and real parts are rounded independently.

* Ceil: Round toward positive infinity.

ceil(A) rounds the elements of A to the nearest integers greater than or equal to A. For complex A, the imaginary and real parts are rounded independently.

.