**Chapter 1**

**Introduction**

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

Image processing basically includes the following three steps:

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

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

Applications where image processing is widely used are as follows:

1. Image sharpening and restoration
2. Medical field
3. Remote sensing
4. Transmission and encoding
5. Machine/Robot vision
6. Color processing
7. Pattern recognition
8. Video processing
9. Microscopic Imaging

In medical field, image processing is used to a wide extent.

MRI (Magnetic Resonance Imaging), an advanced medical imaging technique, is basically used in the biomedical to detect and visualize finer details in the internal structure of the body. This technique is basically used to detect the differences in the tissues which have a far better technique as compared to computed tomography. So this makes this technique a very special one for the brain tumor detection and cancer imaging.

A brain tumor occurs when abnormal cells form within the brain. There are two main types of tumors: malignant or cancerous tumors and benign tumors. Cancerous tumors can be divided into primary tumors that start within the brain, and secondary tumors that have spread from somewhere else, known as brain metastasis tumors.

a) Benign Tumor: A benign tumor is a tumor is the one that does not expand in an abrupt way; It doesn’t affect its neighboring healthy tissues and also does not expand to nonadjacent tissues. Moles are the common example of benign tumors.

b) Malignant Tumor: Malignancy is the type of tumor that grows worse with the passage of time and ultimately results in the death of a person. Malignant is basically a medical term that describes a severe progressing disease. Malignant tumor is a term which is typically used for the description of cancer.

According to the Central Brain Tumor Registry of the United States (CBTRUS), there were 64,530 new cases of primary brain and central nervous system tumors diagnosed by the end of 2014. This necessitates greater effort in the field of brain tumor diagnosis.

Hence, many image processing techniques and methodologies are used in the present day to detect brain tumor, its size and location.

In this project, we use Artificial Bee Colony algorithm along with Fuzzy C-means to detect brain tumor.

* 1. **Artificial Bee Colony (ABC) algorithm:**

**1.1.1 Taxonomy**

The Bees Algorithm beings to Bee Inspired Algorithms and the field of Swarm Intelligence [4], and more broadly the fields of Computational Intelligence and Metaheuristics. The Bees Algorithm is related to other Bee Inspired Algorithms, such as Bee Colony Optimization, and other Swarm Intelligence algorithms such as Ant Colony Optimization and Particle Swarm Optimization.

**1.1.2 Inspiration**

The Bees Algorithm is inspired by the foraging behavior of honey bees. Honey bees collect nectar from vast areas around their hive (more than 10 kilometers). Bee Colonies have been observed to send bees to collect nectar from flower patches relative to the amount of food available at each patch. Bees communicate with each other at the hive via a waggle dance that informs other bees in the hive as to the direction, distance, and quality rating of food sources.

**1.1.3 Concept**

Honey bees collect nectar from flower patches as a food source for the hive. The hive sends out scout’s that locate patches of flowers, who then return to the hive and inform other bees about the fitness and location of a food source via a waggle dance. The scout returns to the flower patch with follower bees. A small number of scouts continue to search for new patches, while bees returning from flower patches continue to communicate the quality of the patch.

**Swarm Intelligence**- Any attempt to design algorithms or distributed problem solving devices inspired by the collective behavior of social insect colonies and other animal societies.

Most fundamental concepts that are necessary and sufficient properties to obtain swarm intelligence behavior are:

1. Self-Organization
2. Division of labor

**Self-Organization:** can be defined as a set of dynamical mechanisms that establish basic rules for interactions between the components of the system. The rules ensure that the interactions are executed on the basis of purely local information without any relation to the global pattern [4].

**Division of labor:** In swarm behavior different tasks are performed simultaneously by specialized individuals which is referred to as division of labor. It enables swarm to respond to changed conditions in the search space [2].

**1.1.4 Strategy**

The information processing objective of the algorithm is to locate and explore good sites within a problem search space [1]. Scouts are sent out to randomly sample the problem space and locate good sites. The good sites are exploited via the application of a local search, where a small number of good sites are explored more than the others. Good sites are continually exploited, although many scouts are sent out each iteration always in search of additional good sites [4].

**1.2 Fuzzy C Means**

Clustering objects or patterns into several groups. It attempts to organize unlabeled input objects into clusters or “natural groups” such that data points within a cluster are more similar to each other than those belonging to different clusters, i.e., to maximize the intra-cluster similarity while minimizing the inter-cluster similarity. In the field of clustering analysis, a number of methods have been put forward and many successful applications have been reported. Clustering algorithms can be loosely categorized into the following categories: hierarchical, partition-based, density-based, grid-based and model-based clustering algorithms. Among them, partition-based algorithms which partition objects with some membership matrices are most widely studied. Traditional partition-based clustering methods usually are deterministic clustering methods which usually obtain the specific group which objects belong to, i.e., membership functions of these methods take on a value of 0 or 1. We can accurately know which group that the observation object pertains to. This characteristic brings about these clustering methods’ common drawback, that we cannot clearly know the probability of the observation object being a part of different groups, which reduces the effectiveness of hard clustering methods in many real situations. For this purpose, fuzzy clustering methods which incorporate fuzzy set theory have emerged. Fuzzy clustering methods quantitatively determine the affinities of different objects with mathematical methods, described by a member function, to divide types objectively [2].

Among the fuzzy clustering method, the fuzzy c-means (FCM) algorithm is the most well-known method because it has the advantage of robustness for ambiguity and maintains much more information than any hard clustering methods. The algorithm is an extension of the classical and the crisp k-means clustering method in fuzzy set domain. It is widely studied and applied in pattern recognition, image segmentation and image clustering, data mining, wireless sensor network and so on.

**Chapter 2**

**Problem Statement**

A research problem aimed at

* Enhancement of image through noise removal
* Segmentation of brain tumor from MRI image using an optimization method, Artificial Bee Colony, with an image based clustering methodology, Fuzzy C-means.
* Detection of tumor using water shed algorithm.

**Chapter 3**

**Literature Survey**

**3.1 Concepts**

**3.1.1 Thresholding**

Thresholding is the simplest method of image segmentation. It is a non-linear operation that converts a gray-scale image into a binary image where the two levels are assigned to pixels that are below or above the specified threshold value. In other words, if pixel value is greater than a threshold value, it is assigned one value (may be white), else it is assigned another value (may be black). In OpenCV, we use cv2.threshold () function:

cv2.threshold (src, thresh, maxval, type [, dst])

This function applies fixed-level thresholding to a single-channel array. The function is typically used to get a bi-level (binary) image out of a grayscale image (compare () could be also used for this purpose) or for removing a noise, that is, filtering out pixels with too small or too large values. There are several types of thresholding supported by the function [3].

Thresholding is the simplest non-contextual segmentation technique. With a single threshold, it transforms a greyscale or color image into a binary image considered as a binary region map. The binary map contains two possibly disjoint regions, one of them containing pixels with input data values smaller than a threshold and another relating to the input values that are at or above the threshold.

**3.1.2 Clustering**

**Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups.

Clustering can be considered the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a *structure* in a collection of unlabeled data. A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”.

A *cluster* is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters [3].

So, the goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. But how to decide what constitutes a good clustering? It can be shown that there is no absolute “best” criterion which would be independent of the final aim of the clustering. Consequently, it is the user which must supply this criterion, in such a way that the result of the clustering will suit their needs.

**3.1.3 Segmentation**

Segmentation is the most important part in image processing. Fence off an entire image into several parts which is something more meaningful and easier for further process. These several parts that are rejoined will cover the entire image. Segmentation may also depend on various features that are contained in the image. It may be either color or texture. Before denoising an image, it is segmented to recover the original image. The main motto of segmentation is to reduce the information for easy analysis. Segmentation is also useful in Image Analysis and Image Compression.

Segmentation can be classified as follows [3]:

* Region Based
* Edge Based
* Threshold
* Feature Based Clustering
* Model Based

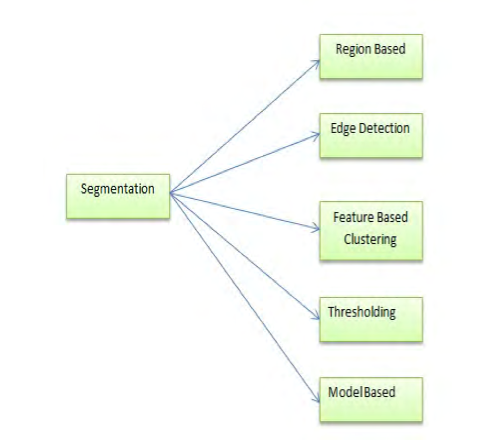


Fig 3.1.1 Classification of segmentation

**3.2 Importance of Brain Tumor**

The brain is the anterior most part of the central nervous system. The location of tumors in the brain is one of the factors that determine how a brain tumor affects an individual's functioning and what symptoms the tumor causes. Along with the Spinal cord, it forms the Central Nervous System (CNS).

Brain tumor is an abnormal growth caused by cells reproducing themselves in an uncontrolled manner. Magnetic Resonance Imager (MRI) is the commonly used device for diagnosis. In MR images, the amount of data is too much for manual interpretation and analysis. During past few years, brain tumor segmentation in magnetic resonance imaging (MRI) has become an emergent research area in the field of medical imaging system. Accurate detection of size and location of brain tumor plays a vital role in the diagnosis of tumor.

A tumor also known as neoplasm is a growth in the abnormal tissue which can be differentiated from the surrounding tissue by its structure. A tumor may lead to cancer, which is a major leading cause of death and responsible for around 13% of all deaths worldwide. Cancer incidence rate is growing at an alarming rate in the world.

Great knowledge and experience on radiology are required for accurate tumor detection in medical imaging. Automation of tumor detection is required because there might be a shortage of skilled radiologists at a time of great need. We propose an automatic brain tumor detection and localization framework that can detect and localize brain tumor in magnetic resonance imaging.

Now days, one of the main cause for increasing mortality among children and adults is brain tumor. It has been concluded from the research of most of the developed countries that number of people suffering and dying from brain tumors has been increased to 300 per year during past few decades. The National Brain Tumor Foundation (NBTF) for research in United States estimates the death of 13000 patients while 29,000 undergo primary brain tumor diagnosis. This high mortality rate of brain tumor greatly increases the importance of Brain Tumor detection. Real time diagnosis of tumors by using more reliable algorithms has been the main focus of the latest developments in medical imaging and detection of brain tumor in MR images and CT scan images has been an active research area.

**3.3 Pre-processing**

Preprocessing indicates that the same tissue type may have a different scale of signal intensities for different images. Preprocessing functions involve those operations that are normally required prior to the main data analysis and extraction of information and are generally grouped as radiometric or geometric corrections. Radiometric corrections include correcting the data for sensor irregularities and unwanted sensor or atmospheric noise, removal of non-brain voxels and converting the data so they accurately represent the reflected or emitted radiation to find out a transformation between two images precisely. The preprocessed images will have some noise which should be removed for the further processing of the image. Image noise is most apparent in image regions with low signal level such as shadow regions or under exposed images. There are so many types of noise like salt – and – pepper noise, film grains etc., All these noise are removed by using algorithms. There are several filters that can be used.

**Removal of unwanted parts from the brain MRI image:**

In preprocessing module image acquired will be processed for correct output. Medical images surely will have some Film Artifacts like labels, marks and unwanted or critical parts which are detected and removed for better result. Pre-processing was done by using some algorithm. For all images the pre-processing should be done so that the result can be obtained in the better way. To find out the transformation between two images precisely they should be preprocessed to improve their quality and accuracy of result. If these images are too noisy or blurred, they should be filtered and sharpened.

**3.3.1 Gaussian Filter**

Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function. It is a widely used effect in graphics software, typically to reduce [image noise](https://en.wikipedia.org/wiki/Image_noise) and reduce detail. The visual effect of this blurring technique is a smooth blur resembling that of viewing the [image](https://en.wikipedia.org/wiki/Image) through a translucent screen, distinctly different from the [bokeh](https://en.wikipedia.org/wiki/Bokeh) effect produced by an out-of-focus lens or the shadow of an object under usual illumination. Gaussian smoothing is also used as a pre-processing stage in [computer vision](https://en.wikipedia.org/wiki/Computer_vision) algorithms in order to enhance image structures at different scales.

Mathematically, applying a Gaussian blur to an image is the same as [convolving](https://en.wikipedia.org/wiki/Convolution) the image with a [Gaussian function](https://en.wikipedia.org/wiki/Gaussian_function). This is also known as a two-dimensional [Weierstrass transform](https://en.wikipedia.org/wiki/Weierstrass_transform). By contrast, convolving by a circle (i.e., a circular [box blur](https://en.wikipedia.org/wiki/Box_blur)) would more accurately reproduce the [bokeh](https://en.wikipedia.org/wiki/Bokeh) effect. Since the [Fourier transform](https://en.wikipedia.org/wiki/Fourier_transform) of a Gaussian is another Gaussian, applying a Gaussian blur has the effect of reducing the image's high-frequency components; a Gaussian blur is thus a [low pass filter](https://en.wikipedia.org/wiki/Low_pass_filter).

**3.3.1.1 Implementation**

A Gaussian blur effect is typically generated by convolving an image with a kernel of Gaussian values. In practice, it is best to take advantage of the Gaussian blur’s separable property by dividing the process into two passes. In the first pass, a one-dimensional kernel is used to blur the image in only the horizontal or vertical direction. In the second pass, another one-dimensional kernel is used to blur in the remaining direction. The resulting effect is the same as convolving with a two-dimensional kernel in a single pass, but requires fewer calculations.

Discretization is typically achieved by sampling the Gaussian filter kernel at discrete points, normally at positions corresponding to the midpoints of each pixel. This reduces the computational cost but, for very small filter kernels, point sampling the Gaussian function with very few samples leads to a large error. In these cases, accuracy is maintained (at a slight computational cost) by integration of the Gaussian function over each pixel's area.

When converting the Gaussian’s continuous values into the discrete values needed for a kernel, the sum of the values will be different from 1. This will cause a darkening or brightening of the image. To remedy this, the values can be normalized by dividing each term in the kernel by the sum of all terms in the kernel.

**3.3.1.2 Common uses**

Gaussian smoothing is commonly used with [edge detection](https://en.wikipedia.org/wiki/Edge_detection). Most edge-detection algorithms are sensitive to noise; the 2-D Laplacian filter, built from a discretization of the [Laplace operator](https://en.wikipedia.org/wiki/Laplace_operator), is highly sensitive to noisy environments. Using a Gaussian Blur filter before edge detection aims to reduce the level of noise in the image, which improves the result of the following edge-detection algorithm. This approach is commonly referred to as [Laplacian of Gaussian](https://en.wikipedia.org/wiki/Laplacian_of_Gaussian), or LoG filtering**.**

**3.3.1.3 Blur Control**

The blurring is controlled by two parameters:

1) The box size, described by (2·n+1) pixels in one direction

2) The radius r The Gaussian bell in one direction delivers: x/r 0 1 2 3 w(x) 1.0 0.6065 0.1353 0.0111 We can choose r=0.465·n. This results in a weight factor 0.1 at the outermost pixel, at x=n , which seems to be reasonable. Less than 0.1 doesn´t make much sense. For pixels on the diagonal corners of the xy-box the value is smaller.

Weak blur: small box n=1 r=0.465 Weight factors 0.1 1.0 0.1

Strong blur: large box n=3 r=1.398

Weight factors 0.1 0.358 0.773 1.0 0.773 0.358 0.1.

**3.3.2 Median Filter**

In medical image processing, it is necessary to perform a high degree of noise reduction in an image before performing high-level processing steps. So the noise can be removed through Median Filter high frequency components from MRI without disturbing the edges and it is used to reduce ‘salt and pepper’ noise. This technique calculates the median of the surrounding pixels to determine the new demonized value of the pixel.

A median is calculated by sorting all pixel values by their size, then selecting the median value as the new value for the pixel. The amount of pixels which should be used to calculate the median. For each pixel, a 3 x 3, 5 x 5, 7 x 7, 9 x 9, 11 x 11 window of neighborhood pixels are extracted, and the pixel intensity values are arranged in ascending order and the median value is calculated for that window. The intensity value of the center pixel is replaced with the median value. This procedure is done for all the pixels in the image to smoothen the edges of Magnetic Resonance Image.

High Resolution Image was obtained when using 3 x 3 than 5 x 5 and so on. Median Filter can remove the noise, high frequency components from MRI without disturbing the edges and it is used to reduce salt and pepper noise. This technique calculates the median of the surrounding pixels to determine the new demonized value of the pixel. A median is calculated by sorting all pixel values by their size, then selecting the median value as the new value for the pixel. The amount of pixels which should be used to calculate the median. Example i3=medfilt2(i3,[3 3]); i3 is filtered image its return from medfilt2 function. Noise is like interferences which present as an irregular granular pattern. This random variation in signal intensity degrades image information.

The main source of noise in the image is the patient's body RF emission due to thermal motion. The whole measurement chain of the MR scanner also contributes to the noise. This noise corrupts the signal coming from the transverse magnetization variations of the intentionally excited spins on the selected slice plane. Four filters in the Enhancement phase are designed to enhance the appearance of images, primarily by sharpening edges, corners, and line detail. Several of the new enhancement filters also incorporate a noise reduction component.

**The disadvantage of the median filter**

Although median filter is a useful non-linear image smoothing and enhancement technique. It also has some disadvantages. The median filter removes both the noise and the fine detail since it can't tell the difference between the two. Anything relatively small in size compared to the size of the neighborhood will have minimal affect on the value of the median, and will be filtered out. In other words, the median filter can't distinguish fine detail from noise.

**3.3.3 Adaptive median Filter**

The adaptive median filtering has been applied widely as an advanced method compared with standard median filtering [7]. The Adaptive Median Filter performs spatial processing to determine which pixels in an image have been affected by impulse noise. The Adaptive Median Filter classifies pixels as noise by comparing each pixel in the image to its surrounding neighbor pixels.

The size of the neighborhood is adjustable, as well as the threshold for the comparison. A pixel that is different from a majority of its neighbors, as well as being not structurally aligned with those pixels to which it is similar, is labeled as impulse noise. These noise pixels are then replaced by the median pixel value of the pixels in the neighborhood that have passed the noise labeling test.

A new type of adaptive center filter is developed for impulsive noise reduction of an image without the degradation of an original image. The image is processed using an adaptive filter. The shape of the filter basis is adapted to follow the high contrasted edges of the image. In this way the artifacts introduced by a circularly symmetric filter at the border of high contrasted areas are reduced.

**Purpose**

1. Remove impulse noise
2. Smoothing of other noise
3. Reduce distortion, like excessive thinning or thickening of object boundaries

**How it works?**

● Adaptive median filter changes size of Sxy (the size of the neighborhood) during operation.

● Notation

 Zmin = minimum gray level value in Sxy

 Zmax = maximum gray level value in Sxy

 Zmed = median of gray levels in Sxy

 Zxy = gray level at coordinates (x, y)

 Smax = maximum allowed size of Sxy

● Algorithm

Level A: A1 = Zmed - Zmin

         A2 = Zmed - Zmax

         if A1 > 0 AND A2 < 0, go to level B

else increase the window size

if window size < Smax, repeat level A

else output Zxy

Level B: B1 = Zxy - Zmin

B2 = Zxy - Zmax

if B1 > 0 AND B2 < 0, output Zxy

else output Zmed

● Explanation

Level A: IF Zmin < Zmed < Zmax, then

              • Zmed is not an impulse

              (1) go to level B to test if Zxy is an impulse

              ELSE

              • Zmed is an impulse

              (1) the size of the window is increased and

              (2) level A is repeated until

                 (a) Zmed is not an impulse and go to level B or

                 (b) Smax reached: output is Zxy

Level B: IF Zmin < Zxy < Zmax, then

              • Zxy is not an impulse

              (1) output is Zxy (distortion reduced)

              ELSE

              • either Zxy = Zmin or Zxy = Zmax

              (2) output is Zmed (standard median filter)

              • Zmed is not an impulse (from level A)

**Advantages**

The standard median filter does not perform well when impulse noise is

Greater than 0.2, while the adaptive median filter can better handle these noises.

The adaptive median filter preserves detail and smooth non-impulsive noise, while the standard median filter does not.

**3.3.4 Bilateral Filter**

The bilateral filter is a non-linear technique that can blur an image while respecting strong edges [7]. Its ability to decompose an image into different scales without causing haloes after modification has made it ubiquitous in computational photography applications such as tone mapping, style transfer, relighting, and denoising. This text provides a graphical, intuitive introduction to bilateral filtering, a practical guide for efficient implementation and an overview of its numerous applications, as well as mathematical analysis.

Bilateral filtering is a technique to smooth images while preserving edges. It can be traced back to 1995 with the work of Aurich and Weule on nonlinear Gaussian filters. It was later rediscovered by Smith and Brady as part of their SUSAN framework, and Tomasi and Manduchi who gave it its current name. Since then, the use of bilateral filtering has grown rapidly and is now ubiquitous in image processing applications .It has been used in various contexts such as denoising, texture editing and relighting, tone management , DE mosaicking , stylization, and optical-flow estimation . The bilateral filter has several qualities that explain its success:

Its formulation is simple: each pixel is replaced by a weighted average of its neighbors. This aspect is important because it makes it easy to acquire intuition about its behavior, to adapt it to application-specific requirements, and to implement it.

It depends only on two parameters that indicate the size and contrast of the features to preserve.

Bilateral Filter can be used in a non-iterative manner. This makes the parameters easy to set since their effect is not cumulative over several iterations. It can be computed at interactive speed even on large images, thanks to efficient numerical schemes, and even in real time if graphics hardware is available. In parallel to applications, a wealth of theoretical studies explain and characterize the bilateral filter’s behavior. The strengths and limitations of bilateral filtering are now fairly well understood. As a consequence, several extensions have been proposed.

**3.3   Fuzzy C Means Clustering**

**3.3.1 Why it so popular in medical field**

Medical image segmentation plays an important role in a variety of biomedical-imaging applications, such as the quantification of tissue volumes, diagnosis, localization of pathology, study of anatomical structure, treatment planning, and computer-integrated surgery. However, segmentation of medical images involves three main image-related problems. First, images contain noise that can alter the intensity of a pixel such that its classification becomes uncertain. Second, images exhibit intensity inhomogeneity where the intensity level of a single tissue class varies gradually over the extent of the image. Third, images have finite pixel size and are subject to partial volume averaging where individual pixel volumes contain a mixture of tissue classes so that the intensity of a pixel in the image may not be consistent with any one class. To overcome these problems, many segmentation techniques have been proposed in the past decades, such as the expectation maximization (EM) algorithm, level set method, clustering, and so on.

Clustering for image segmentation usually classifies image pixels into *c*-clusters such that members of the same cluster are more similar to one another than to members of other clusters, where the number, *c*, of clusters is usually predefined or set by some validity criterion or a priori knowledge . In the clustering methods, fuzzy *c*-means (FCM) based algorithms have been widely used in medical image segmentation. Such a success chiefly attributes to the introduction of fuzziness for the belongingness of each image pixel. This enables the clustering methods to retain more information from the original image than the crisp or hard segmentation methods [2].

**3.3.2 Algorithm**:

Fuzzy clustering is a powerful unsupervised method for the analysis of data and construction of models. In many situations, fuzzy clustering is more natural than hard clustering. Objects on the boundaries between several classes are not forced to fully belong to one of the classes, but rather are assigned membership degrees between 0 and 1 indicating their partial membership.

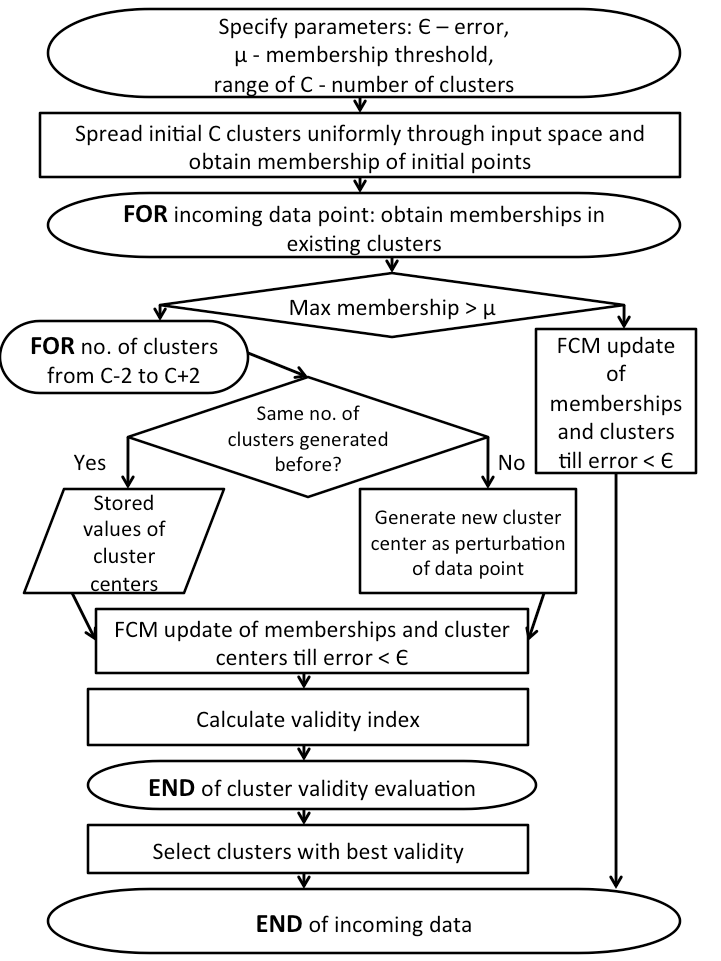


Fig 3.3.1 Flow Chart of Fuzzy C Means

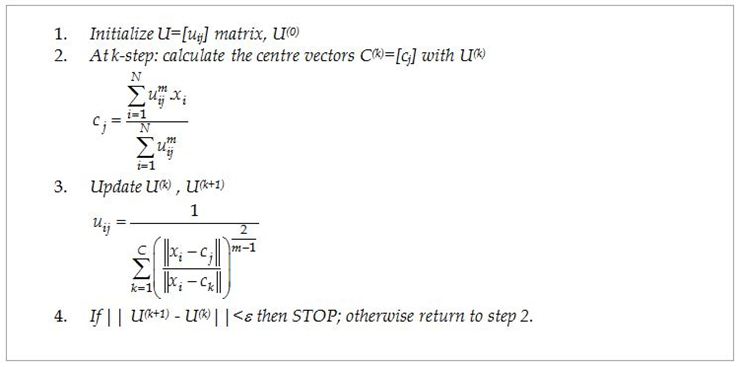


Fig: 3.3.2 Fuzzy C Means Algorithm

Here m is any real number greater than 1, uij is the degree of membership of xi in the cluster j, xi is the ith of d-dimensional measured data, cj is the d-dimension center of the cluster. This algorithm works by assigning membership to each data point corresponding to each cluster center on the basis of distance between the cluster center and the data point. More the data is near to the cluster center more is its membership towards the particular cluster center. Clearly, summation of membership of each data point should be equal to one. After each iteration membership and cluster centers are updated according to the formula mentioned in the algorithm.

**3.3.3 Benefits of FCM**

* Gives best result for overlapped data set and comparatively better than k-means algorithm.
* Unlike k-means where data point must exclusively belong to one cluster center here data point is assigned membership to each cluster center as a result of which data point may belong to more than one cluster center.

**3.3.4 Difference between FCM and K means**

|  |  |
| --- | --- |
| **Fuzzy C-means Clustering** | **K-Means Clustering** |
| 1. In fuzzy clustering, each point has a  probability of belonging to each cluster. | 1. Each point completely belongs to just one cluster in the traditional k-means. |
| 2. Fuzzy c-means specifically tries to deal with the problem where points are  somewhat in between centers or  otherwise ambiguous by replacing  distance with probability. | 2. K-means uses a weighted  centroid based on those probabilities. |
| 3.The resulting clusters are best analyzed as probabilistic distributions. | 3. The resulting clusters are best analyzed as a hard assignment of labels. |
| 4. With regards to performance, the FCM therefore needs to perform k (i.e. number of clusters) multiplications for each point, for each dimension hence it is slow. (full inverse-distance weighting). | 4. K means has to perform only k  multiplications, hence is faster (distance calculation). |

**3.4 Natural explanation of bee**

**3.4.1 Artificial bee algorithm explanation**

[4]Three essential components of forage selection:

**Food Sources**: The value of a food source depends on many factors such as its proximity to the nest, its richness or concentration of its energy and the ease of extracting this energy.

**Employed Foragers**: They are associated with a particular food source which they are currently exploiting or are employed at. They carry with them information about this particular source, its distance and direction from the nest, the profitability of the source and share this information with a certain probability.

**Unemployed Foragers**: They are continually at look out for a food source to exploit. There are two types of unemployed foragers: Scouts, searching the environment surrounding the nest for new food sources and onlookers waiting in the nest and establishing a food source through the information shared by employed foragers.

The model defines two leading modes of the behavior

1. Recruitment to a nectar source
2. The abandonment of the source

**3.4.2 Exchange of information among bees**

1. The exchange of information among bee is the most important occurrence in the formation of collective knowledge.
2. The most important part of the hive with respect to exchanging information is the dancing area.
3. Communication among bees related to the quality of food source takes place in the dancing area.
4. This dance is called a Waggle dance
5. Employed foragers share their information with a probability proportional to the profitability of the food source, and the sharing of this information through waggle dancing is longer n duration.
6. An onlooker on the dance floor, probably she can watch numerous dances and decides to employ herself at the most profitable source.
7. There is a greater probability of onlooker choosing more profitable sources since more information is circulated about the more profitable source.

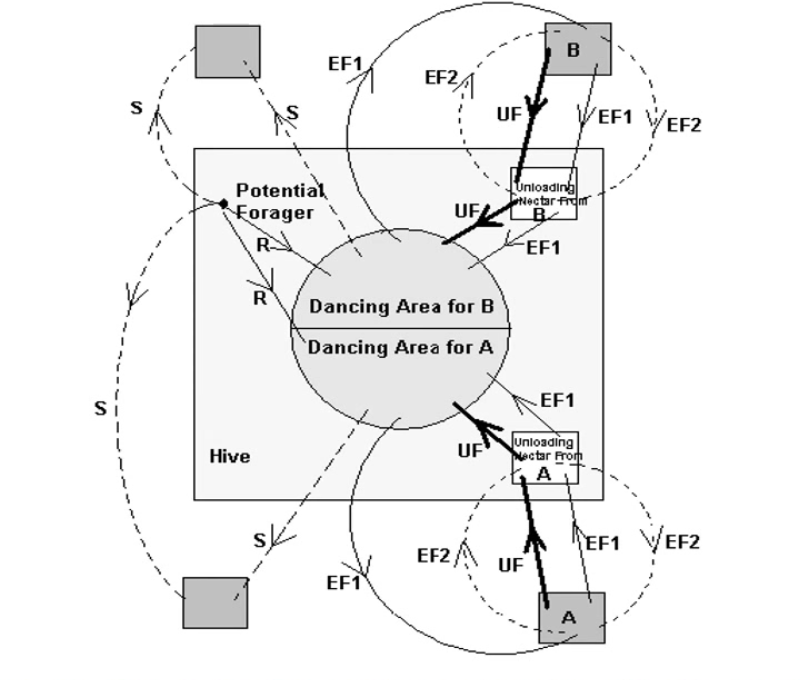


Fig 3.4.1 The Behaviour of Honey Bee foraging for nectar

**3.4.3 Basic Self Organization properties**

1. Positive Feedback- As the nectar amount of food sources increases, the number of onlookers visiting them increase too.
2. Negative Feedback- The exploitation process of poor food sources is stopped by bees.
3. Fluctuations: The scouts carry out a random search process for discovering new food sources.
4. Multiple interactions: Bees share their information about food sources with their nest mates in the dance area [6].

**3.4.4 Artificial Bee Colony Algorithm**

1. Each cycle of search consists of three steps: moving the employed and onlooker bees onto the food source and calculating their nectar amounts and determining the scout bees and directing them onto possible food sources.
2. A food source position represents a possible solution to the problem solution to the problem to be optimized.
3. The amount of nectar of a food source corresponds to the quality of the solution
4. Onlookers are placed on the food sources by using a probability based selection process.
5. As the nectar amount of a food source increases, the probability value with which the food source is preferred by onlookers increases too
6. The scouts are characterized by low search costs and a low ayerage in food source quality. One bee is selected as the scout bee.
7. The selection is controlled by a control parameter called “limit”.
8. If a solution representing a food source is not improved by a predetermined number of trials, then that food source is abandoned and the employed bee is converted to a scout

**3.4.5 Control Parameters of ABC Algorithm**

1. Swarmsize [1]
2. Limit
3. Number of onlookers: 50% of the swarm
4. Number of employed bees: 50% of the swarm
5. Number of scouts : 1

**3.4.6 Exploration VS Exploitation**

* 1. Onlookers and employed bees carry out the exploitation process in the search space.
  2. Scouts control the exploitation process.

**3.4.7 Explanation Of code**

1. bee.initial() :  Number of food sources are initialized  randomly and fitness of each food source is computed. Tis task is performed by the initial scout bee and hence supports exploration
2. bee.Memorize() : The best fodd source is memorized by the bee.
3. bee.sendEmployedBees(): In the function an artificially bee generates a random solution that is a mutant of the original solution.
4. The formula for producing a candidate solution from the existing is described below:

V_{i_{k}} = X_{i_{k}}+\Phi_{i_{k}}\times (X_{i_k}-X_{j_k})  

        Where j :{1,2….. Number of Employed Bees}

      k: {1,2,.....D} are randomly chosen index

     D: Number of parameters to optimize.

     j <> i : both are randomly chose.

\Phi_{i_{k}}

     Random number [-1 ,  +1]

     Solution[ param2change] = Foods [i][param2change] + (Foods [i] [param2change] - Foods[neighbour][param2change])\*(r-0,5)\*2

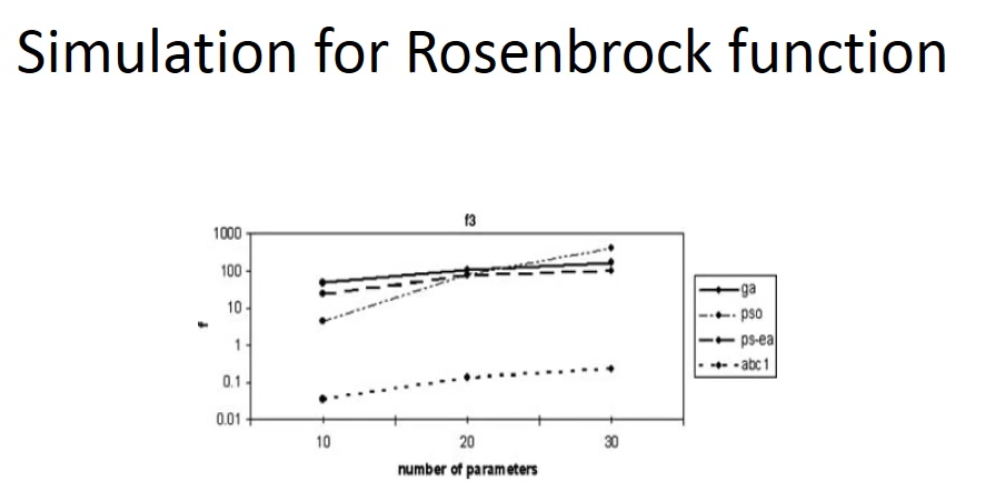
     5.     After each candidate source position vi,j is produced and then evaluated by the artificial bee, its     performance is compared with that of xi,k . If the new food has equal or better nectar than the old source, it is replaced with the old one in the memory. Otherwise , the old one is retained. In other words, a greedy selection mechanism is employed as the selection operation between the old and the current food sources.

    6. bee.CalulateProbabilities(): An onlooker bee chooses a food source depending on the probability   value associated with that food source , pi, calulated by:

P_i=\frac{fit_i}{\sum_j{fit_j}}

7. bee.SendScoutBees(): The trial parameter is defined for those solutions that are exhausted and not changing. This function determines those food sources and abandons them.

**3.4.8 Simulation of Rosenbrock function:**



The mean best value obtained for f3 by GA, PSO, PS-EA and ABCI after 500, 750 and 1000 cycles dimensions 10, 20, 30.

**3.4.9 Conclusion**

ABC algorithm in fact employs four different selection processes:

1. A global selection process used by the artificial onlooker bees for discovering promising regions as
2. A local selection process carried out in a region by the artificial employed bees and the onlookers depending on local information.
3. A local selection process called greedy selection process carried out by all bees in that if the nectar amount of candidate source. Otherwise the bee keeps the present one in the memory.
4. A random selection process carried out by scouts.

**3.5 Watershed Algorithm**

A watershed is a basin-like landform defined by highpoints and ridgelines that descend into lower elevations and stream valleys. In the study of image processing a watershed of a grayscale image is analogous to the notion of a catchment basin of a height map. In short, a drop of water following the gradient of an image flows along a path to finally reach a local minimum. Intuitively, the watershed of a relief correspond to the limits of the adjacent catchment basins of the drops of water.  
 There are different technical definitions of a watershed. In graphs, watershed lines may be defined on the nodes, on the edges, or hybrid lines on both nodes and edges. Watersheds may also be defined in the continuous domain. There are also many different algorithms to compute watersheds [9]. Watershed algorithm is used in image processing primarily for segmentation purposes.  
 There are mainly three methods to implement watershed.  
They are listed below:  
-Distance Transform Approach  
-Gradient method  
-Marker Controlled Approach   
  
**3.5.1 Gradient Method**

The gradient magnitude is used to preprocess a gray-scale image prior to using the watershed transform for segmentation. The gradient magnitude image has high pixel  
values along object edges and low pixel values everywhere else. Watershed transform would result in watershed ridge lines along object edges. There is a problem of over  
segmentation in this method [8].

The topological gradient provides a global analysis of the image then the almost unwanted contours due to the noise added to a given image can be significantly reduced by our approach. The experimental results show that the over segmentation problem, which usually appears with the watershed technique, can be attenuated, and the segmentation results can be performed using the topological gradient approach. Another advantage of this method is that it splits the segmentation  
process into two separate steps: first we detect the main edges of the image processed, and then we compute the watershed of the gradient detected

**3.5.2 Inter-pixel watershed**

S. Beucher and F. Meyer introduced an algorithmic inter-pixel implementation of the watershed method, given the following procedure:  
1. Label each minimum with a distinct label. Initialize a set S with the labeled nodes.  
2. Extract from S a node x of minimal altitude F, that is to say F(x) = min{F(y)|y ∈ S}. Attribute the label of x to each non-labeled node y adjacent to x, and insert y in S.  
3. Repeat Step 2 until S is empty.  
  
**3.5.3 Principle**

Any greytone image can be considered as a topographic surface.   
If we flood this surface from its minima and, if we prevent the merging of the waters coming from different sources, we partition the image into two different sets: the catchment basins and the watershed lines. However, in practice, this transform produces an important over-segmentation due to noise or local irregularities in the gradient image.   
  
**3.5.4 Applications of watershed algorithm**  
1. Traffic monitoring  
2. Road segmentation  
3. Coffee bean separation  
4. Silver grains on a photographic plate

It is one of the best methods to group pixels of an image on the basis of their intensities. Pixels falling under similar intensities are grouped together. It is a good segmentation technique for dividing an image to separate a tumor from the image Watershed is a mathematical morphological operating tool. Watershed is normally used for checking output rather than using as an input segmentation technique because it usually suffers from over segmentation and under segmentation.  
 For using watershed segmentation different methods are used. Two basic principle methods are given below:

1) The computed local minima of the image gradient are chosen as a marker. In this method an over segmentation occurs. After choosing marker region merging is done as a second step.

2) Watershed transformation using markers utilizes the specifically defined marker positions. These positions are either defined explicitly by a user or they can be determined automatically by using morphological tools.

**Chapter 4**

**Project requirement specification**

**4.1.1 Performance Requirements**

User Satisfaction:  The project is such that it stands up to the user expectation, user friendly GUI is built for easier application of algorithms.

Response Time: The response of all the operations are good, however fcm applied on the image takes time due to the high pixel intensity of TIFF images.

Error Handling: Response to user errors and undesired situations has been taken care of to ensure that the system operates without halting. Once the image is uploaded, user just needs to select the next algorithm to be applied hence there is no room for error.

Portable:  The system is portable

* Windows 95/98/Me/NT/2000/XP
* Linux
* LinuxPPC
* Mac OS X
* Unix

To any of this operating system provided opencv Python is installed and running.

**4.1.2 User Requirements:**

The system is easy to learn and understand. A regular user can also use the system efficiently, without any difficulties.

**4.1.3   Efficiency Requirement**

The system should respond to user action within 20ms. The system handles one user and one image at a time, multiple images cannot be checked for tumor at the same time.

**4.1.4   Storage Requirement**

The space provision for a single image is 266 KB to 350 KB.

The space provision for opencv is 4MB disk space.

**4.1.5  Functional Requirements**

The system processes images in acceptable image formats (JPG, PNG, TIFF etc.).

The system shall detect brain tumor present in the image and display it with different colours.

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Chapter 5**

**System Requirement Specification**

**5.1 Hardware Requirements**

Processor                  : Intel Pentium III or later

Main Memory (RAM): 4MB (disk space)

Cache Memory          : 512 KB

Monitor                      : 4-17 inches or more

**5.1.1 Laptop specification**

The OpenCV software runs on personal computers that are based on Intel architecture processors and running Microsoft\* Windows\* 95, Windows 98, Windows 2000, or Windows NT\*. The OpenCV integrates into the customer’s application or library written in C or C++.

**5.2 Software Requirements**

**5.2.1 Opencv requirements**

The OpenCV software run on Windows platforms. The code and syntax used for function and variable declarations in this manual are written in the ANSI C style. However, versions of the OpenCV for different processors or operating systems may, of necessity, vary slightly.

OpenCV algorithms are developed in C++. Also wrappers for languages such as Python and Java have been developed. OpenCV runs on both desktop (Windows, Linux, Android, MacOS, FreeBSD, OpenBSD) and mobile (Android, Maemo, iOS).

**5.2.2 Python requirements**

The [Python libraries](http://www.python.org/downloads/) are required to build the *Python interface* of OpenCV. We have used the version 2.7.*x*. This is also a must if you want to build the *OpenCV documentation*.

[Numpy](http://numpy.scipy.org/) is a scientific computing package for Python. Required for the *Python interface*.

[Intel © Threading Building Blocks (*TBB*)](http://threadingbuildingblocks.org/file.php?fid=77) is used inside OpenCV for parallel code snippets. Using this will make sure that the OpenCV library will take advantage of all the cores you have in your systems CPU.

   yum install tbb-devel

[Intel © Integrated Performance Primitives (*IPP*)](http://software.intel.com/en-us/articles/intel-ipp/) may be used to improve the performance of color conversion, Haar training and DFT functions of the OpenCV library.

[Sphinx](http://sphinx.pocoo.org/) is a python documentation generator and is the tool that will actually create the *OpenCV documentation*. This on its own requires a couple of tools installed.

* Sphinx-1.3.6-py2.py3-none-any.whl
* Sphinx-1.4.1-py2.py3-none-any.whl
* yum install python-sphinx

**5.3  Assumptions**

1. User is digital literate
2. Knows how to open application, use navigation, follow the exact order of the algorithms to be implemented on the image.
3. Knows the structure of the Brain and possible places a tumor can occur.
4. Has opencv 3 installed and running.

**5.4 languages used**

Python

**5.5 Use cases**

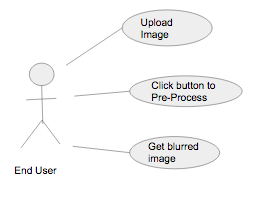


Fig 5.5.1 User Action

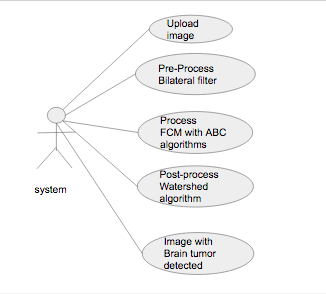


Fig 5.5.2 System

**5.5.1 User Module**

****

Fig 5.5.3

|  |  |
| --- | --- |
| User Case | Description |
| Actor | User |
| Precondition | Input image should be loaded to the System. |
| Main Scenario | User uploads the image using UI. |
| Post Condition | Image successfully uploaded. |

**5.5.2 Pre-processing Module**

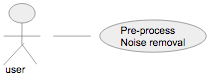
****

Fig 5.5.4

|  |  |
| --- | --- |
| User Case | Description |
| Actor | System |
| Precondition | Uploaded input image. |
| Main Scenario | Bilateral filter is applied. Image is blurred, hence noise removed. |
| Post Condition | Image with no noise is delivered. |

**5.5.3 Process Module**

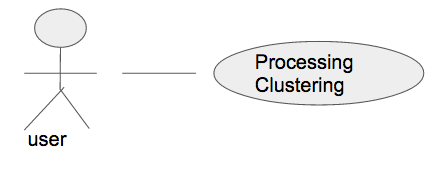
****

Fig 5.5.5

|  |  |
| --- | --- |
| User Case | Description |
| Actor | System |
| Precondition | Uploaded input image. |
| Main Scenario | FCM and optimization technique ABC is applied |
| Post Condition | Image with brain tumor detected. |

**5.5.4 Post-processing Module**

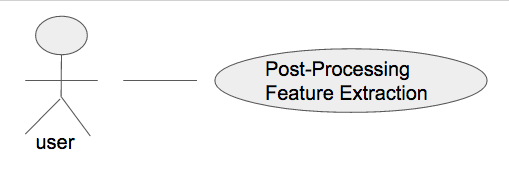
****

Fig 5.5.6

|  |  |
| --- | --- |
| User Case | Description |
| Actor | System |
| Precondition | Uploaded input image. |
| Main Scenario | Watershed algorithm is applied, feature extraction is done. |
| Post Condition | Image with brain tumor detected clearly. |

**5.5.5 Testing Module**

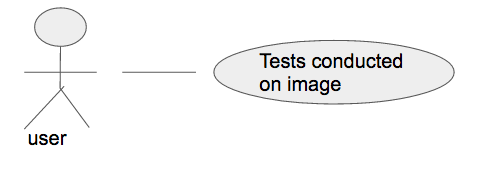
****

Fig 5.5.7

|  |  |
| --- | --- |
| User Case | Description |
| Actor | System |
| Precondition | Running of Brain tumor detection code on images |
| Main Scenario | Compare and note the true positives, true negatives, false positives and false negatives. |
| Post Condition | Derive an accuracy percentage. |

**5.6 Gantt Chart**

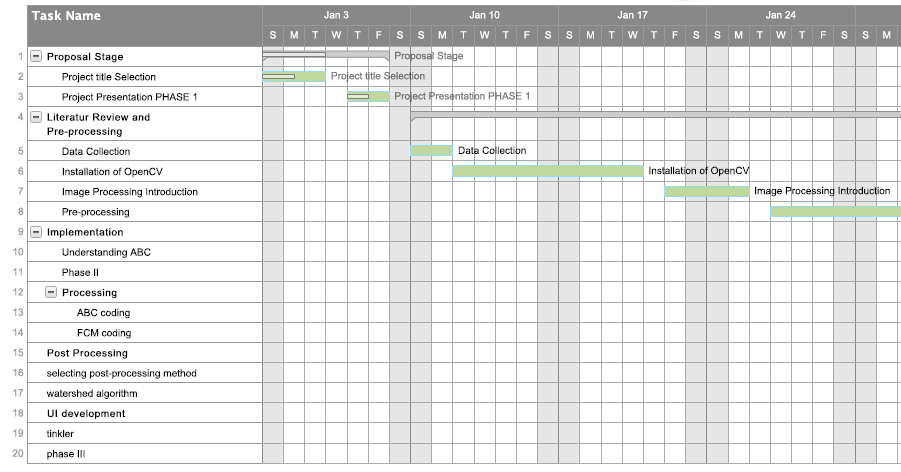
****

Fig 5.6.1 Gantt Chart 1

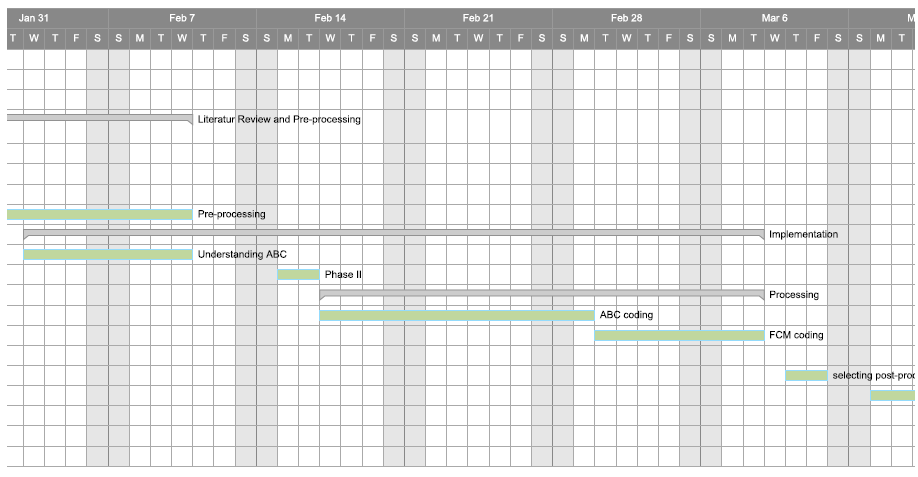
****

Fig 5.6.2 Gantt Chart 2

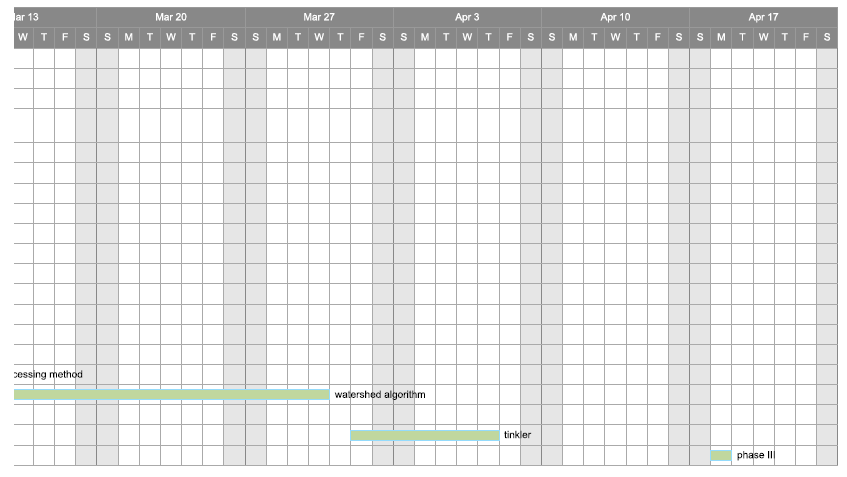
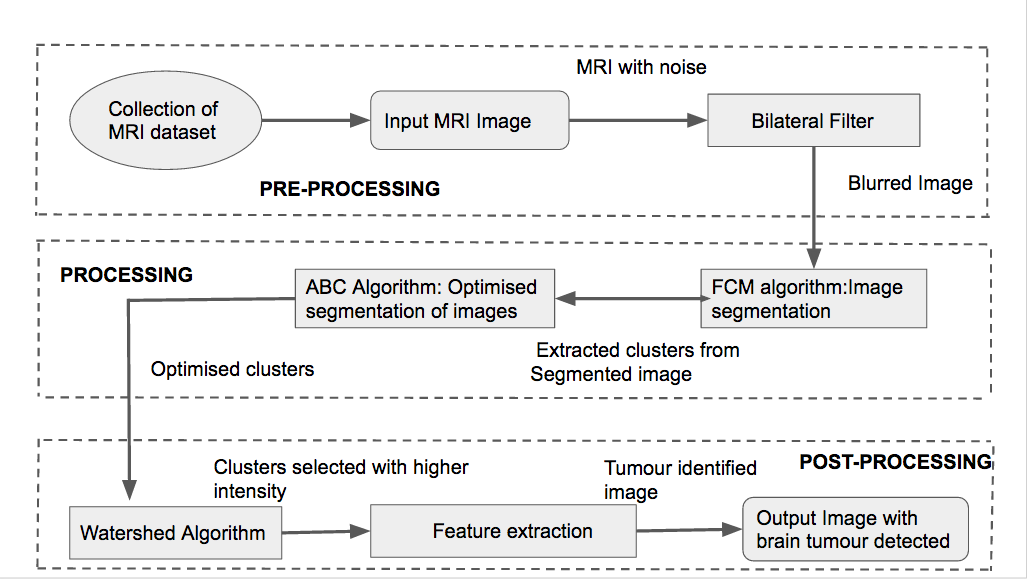
****

Fig 5.6.3 Gantt Chart 3

**Chapter 6**

**System Design**

**6.1 Block diagram**

****

**6.2 Modules**

**6.2.1 User Module**

Here the user uploads an image, which is either in jpg, bmp, png or tiff. Python library is used to convert image to 2 dimensional matrix which can be fed to our algorithms.

**6.2.2 Preprocessing Module**

The image may have noise, disturbances. Therefore we preprocess the image by removing noise, converting it to grayscale, smoothing the image. We do this using Bilateral filter.

**6.2.3 Processing Module**

Here the enhanced image which has been Pre-processed, is fed into Fuzzy C means algorithm first and then to the optimization technique of Artificial Bee algorithm. The output is an image with detection of high density clusters, predicting them to be tumor.

**6.2.4 Post-Processing Module**

The image after undergoing processing, is fed to feature extraction algorithm. Hence, here Watershed algorithm is applied, which colours the image with different colour in different density, thus providing the possible location of brain tumor.

openCV is used in all of these modules.

**6.3 Data Flow Diagram**

**6.3.1 DFD Pre-processing**

Module name: Pre-processing Module

Input: Brain MRI images

Output: Enhanced Image

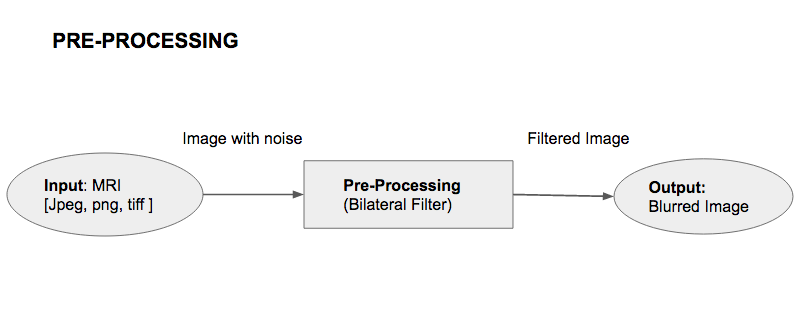


Fig 6.3.1 Pre-processing

**6.3.2 DFD Processing**

Module name: Processing Module

Input: Brain MRI images

Output: Image after applying FCM and ABC

**6.3.2.1 DFD Processing level 0**

Module name: FCM algorithm

Input: Brain MRI images

Output: Image after applying FCM

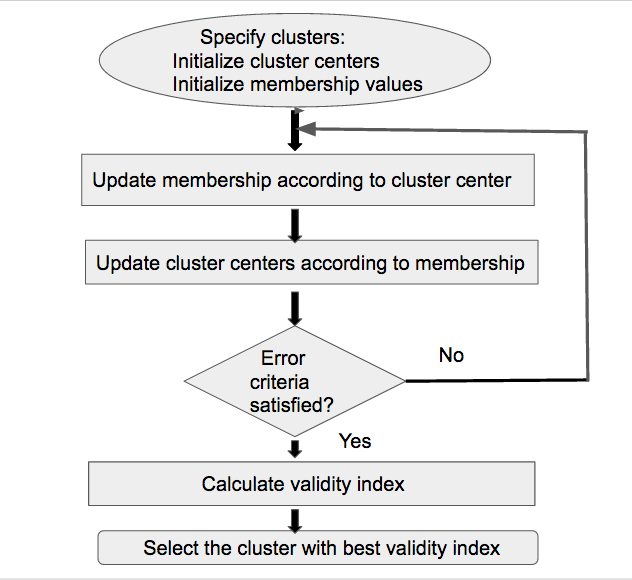


Fig 6.3.2 FCM

**6.3.2.2 DFD Processing level 1**

Module name: ABC algorithm

Input: Brain MRI images

Output: Image after applying ABC

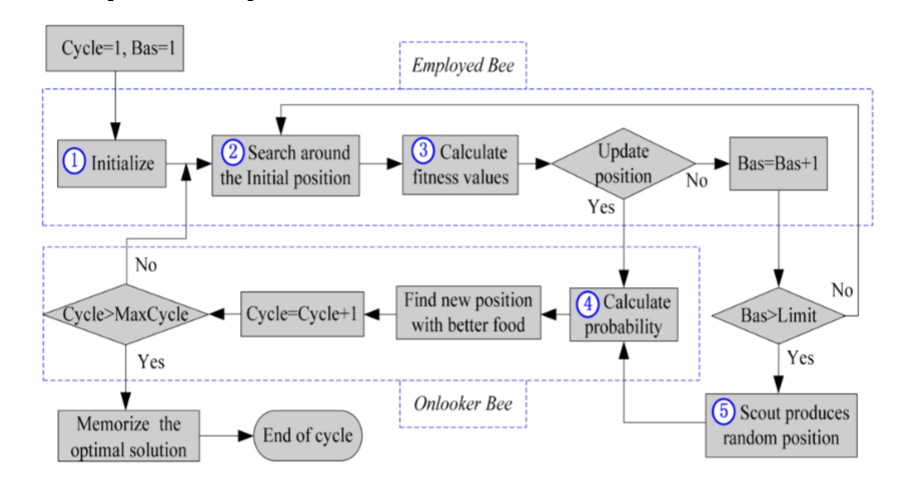
****

Fig 6.3.3 ABC

**6.3.3 DFD Post-processing**

Module name: Post-processing Module

Input: Brain MRI images

Output: Brain tumor detected image

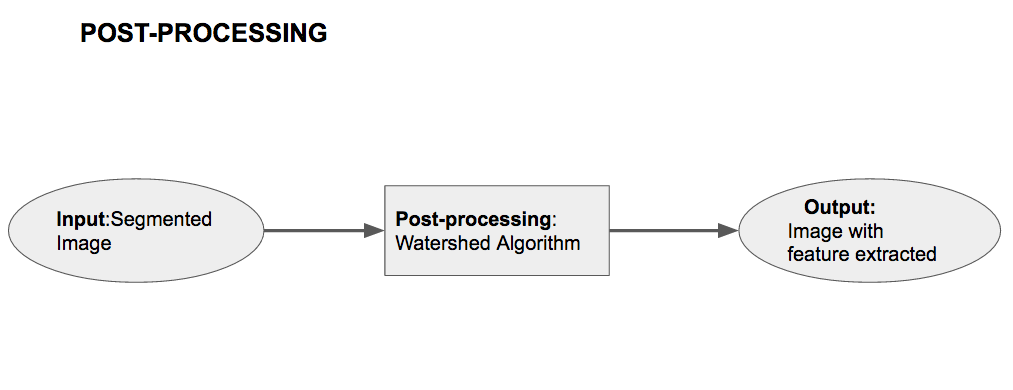
****

Fig 6.3.4 Post-processing

**Chapter 7**

**Implementation**

**7.1 DATA COLLECTION**

It is observed that, to validate and verify the results of the proposed algorithm the standard databases of MRI images are not available. Hence, the data collection is made under the assistance of our guilde from open source datasets.

• http://www.oasis- brains.org/app/action/BundleAction/bundle/O AS1\_CROSS

• http://www.medinfo.cs.ucy.ac.cy/index.php/do wnloads/datasets

**7.2 Pre-processing**

In this stage, the main purpose is to eliminate noise in order to prepare it for segmentation, since noise appearing in the image might ruin segmentation quality.

Various filters such as

1. Gaussian filter

2. Median filter

3. Adaptive Median filter

4. Bilateral filter are implemented.

A bilateral filter is a non-linear, edge-preserving and noise reducing smoothing filter for images. The intensity value at each pixel in an image is replaced by a weighted average of intensity values from nearby pixels. This weight can be based on a Gaussian distribution.

This preserves sharp edges by systematically looping through each pixel and adjusting weights to the adjacent pixels accordingly.

**7.2.1 Pseudo code of Bilateral Filter**

img = cv2.imread('opencv\_logo.png')

blur = cv2.bilateralFilter(img,9,75,75)

plt.subplot(121),plt.imshow(img),plt.title('Original')

plt.xticks([]), plt.yticks([])

plt.subplot(122),plt.imshow(blur),plt.title('Blurred')

plt.xticks([]), plt.yticks([])

plt.show()

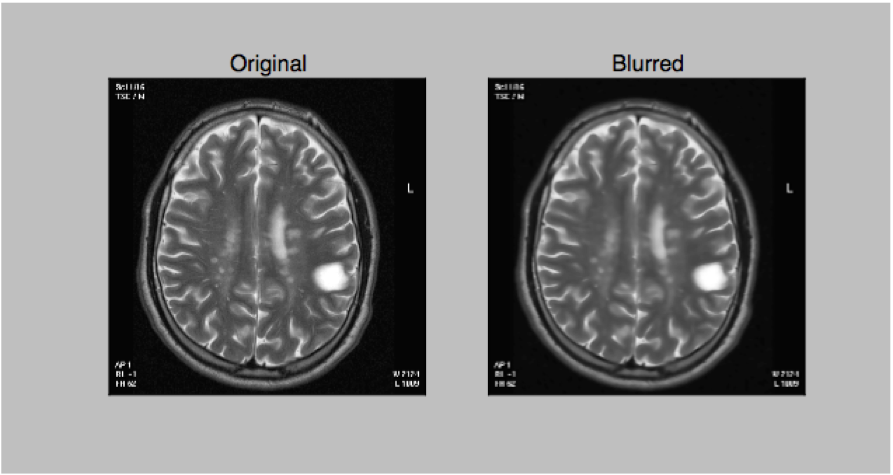


Fig 7.2.1 Bilateral Filter applied on Brain MRI

**7.3 Processing**

The segmentation of an image involves the division or separation of the image into regions of similar attribute. The ultimate aim in a large number of image processing applications is to extract important features from the image data. Here FCM and Artificial Bee Algorithms are used respectively.

**7.3.1 Pseudo code for FCM**

The FCM algorithm attempts to partition a finite collection of nelements X = \{ \mathbf{x}_1, . . . , \mathbf{x}_n \}into a collection of c fuzzy clusters with respect to some given criterion.

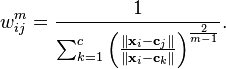
Given a finite set of data, the algorithm returns a list of ccluster centres C = \{ \mathbf{c}_1, . . . , \mathbf{c}_c \}and a partition matrix

W = w_{i,j} \in[0, 1],\; i = 1, . . . , n,\; j = 1, . . . , c, where each element, w_{ij}, tells the degree to which element, \mathbf{x}_i, belongs to cluster \mathbf{c}_j.

The FCM aims to minimize an objective function:

\underset{C} {\operatorname{arg\,min}}  \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij}^m \left\|\mathbf{x}_i - \mathbf{c}_j \right\|^2,

where:



FCM pseudo code:

Step 1: Generates brain portin only dataset x=[x1,x2……,xn] of MR Brain image.

Step2: Set various parameters like [the scalar weighting exponent m) and the termination condition that is the maximum no of iteration.

Step 3: Select the number of clusters c

Step 4: Get intial set of random cluster centers z=[z1,z2….zc]

Step 5: Calculate euclidean distance for c clusters

Step 6: calculate membership matrix

Step 7: Update the cluster center zj using the membership matrix.

Step 8: if the termination condition is not met go to Step 5

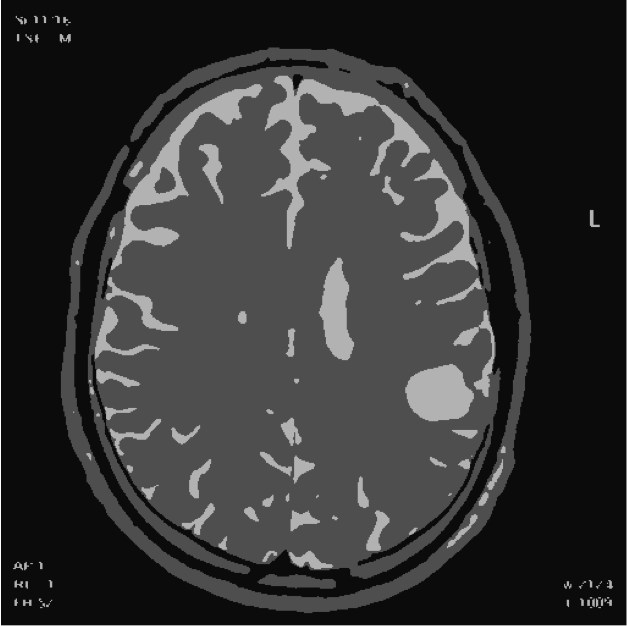


Fig 7.3.1 Image after applying FCM

**7.3.2 Pseudo code for ABC**

Input: Problemsize, Beesnum, Sitesnum, EliteSitesnum, PatchSizeinit, EliteBeesnum, OtherBeesnum

Output: Beebest

Population ← InitializePopulation(Beesnum, Problemsize);

while ¬StopCondition() do

EvaluatePopulation(Population);

Beebest ←GetBestSolution(Population);

NextGeneration ← ∅;

Patchsize ← ( PatchSizeinit × PatchDecreasefactor);

Sitesbest ← SelectBestSites(Population, Sitesnum);

foreach Sitei ∈ Sitesbest do

RecruitedBeesnum ← ∅;

if i < EliteSitesnum then

RecruitedBeesnum ← EliteBeesnum;

else

RecruitedBeesnum ← OtherBeesnum;

 end

Neighborhood ← ∅;

for j to RecruitedBeesnum do

Neighborhood ← CreateNeighborhoodBee(Sitei, Patchsize);

end

NextGeneration ← GetBestSolution(Neighborhood);

end

RemainingBeesnum ← (Beesnum- Sitesnum);

for j to RemainingBeesnum do

NextGeneration ← CreateRandomBee();

end

Population ← NextGeneration;

end

return Beebest;

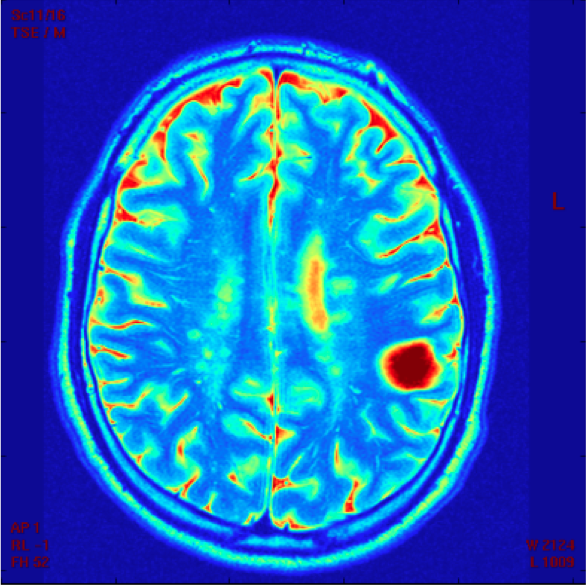


Fig 7.3.2 Image after applying ABC

**7.4 Post-processing**

After segmentation, tumor extraction process is carried out. Firstly, segmented image is filtered if it includes a great amount of noise.

• Gray level thresholding is employed to convert that image to the binary image.

• Watershed of the image is retrieved to extract the tumor from the segmented image.

**7.4.1 Pseudo code for Watershed algorithm**

image=skimage.io.imread('IM\_00086.TIF', as\_grey=False)  
# denoise image  
denoised = rank.median(image, disk(10))  
   
# find continuous region (low gradient) --> markers  
markers = rank.gradient(denoised, disk(5)) < 10  
markers = ndimage.label(markers)[0]  
   
#local gradient  
gradient = rank.gradient(denoised, disk(2))  
   
# process the watershed  
labels = watershed(gradient, markers)  
   
# display results  
fig, axes = plt.subplots(ncols=4, figsize=(8, 2.7))  
ax0, ax1, ax2, ax3 = axes  
   
ax0.imshow(image, cmap=plt.cm.gray, interpolation='nearest')  
ax1.imshow(gradient, cmap=plt.cm.spectral, interpolation='nearest')  
ax2.imshow(markers, cmap=plt.cm.spectral, interpolation='nearest')  
ax3.imshow(image, cmap=plt.cm.gray, interpolation='nearest')  
ax3.imshow(labels, cmap=plt.cm.spectral, interpolation='nearest', alpha=.7)  
   
for ax in axes:  
   ax.axis('off')  
   
plt.subplots\_adjust(hspace=0.01, wspace=0.01, top=1, bottom=0, left=0, right=1)  
plt.show()

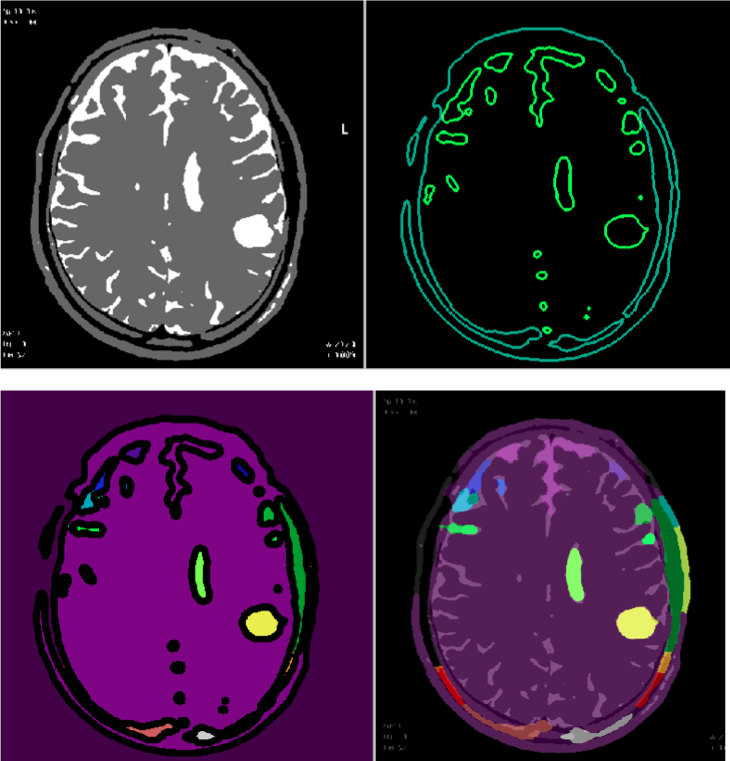
****

Fig 7.4.1 Watershed algorithm applied on image

**Chapter 8**

**Results & Discussions**

**8.1  Testing and Test Cases:**

**Testing methods used**

Black box testing

User Interface testing

**8.1.1 Black box testing**

Black-box testing is a method of software testing that examines the functionality of an application based on the specifications.

Applied on 30 images with brain tumor

|  |  |  |
| --- | --- | --- |
| Input Image | Brain Tumor Detected | Brain Tumor Not detected |
| Image with Brain tumor | 24 | 6 |

This indicates 80% accuracy

Applied on 30 images without brain tumor

|  |  |  |
| --- | --- | --- |
| Input Image | Brain tumor not detected | Brain Tumor wrongly Detected |
| Image without Brain tumor | 27 | 3 |

This indicates 10% error

**User Interface testing**

User interface testing, a testing technique used to identify the presence of defects is a software under test by using Graphical user interface.

|  |  |  |
| --- | --- | --- |
|  | Yes | No |
| Is the correct window type used? |  |  |
| If the screen requires data entry on a specific field |  |  |
| If the screen contains text boxes that allow data entry, ensure that the width of data entered does not exceed the width of the table field |  |  |
| If the buttons function properly |  |  |
| Does the application crash? |  |  |

**8.2 Results Obtained**

Image with Brain Tumor detected and highlighted with a yellowish colour is obtained. The entier image is coloured with different colour intensities due to the feature extraction by watershed algorithm and clusterization by Fuzzy C means along with optimization Technique of Artificial Bee Algorithm.

**Chapter 9**

**Screenshots**

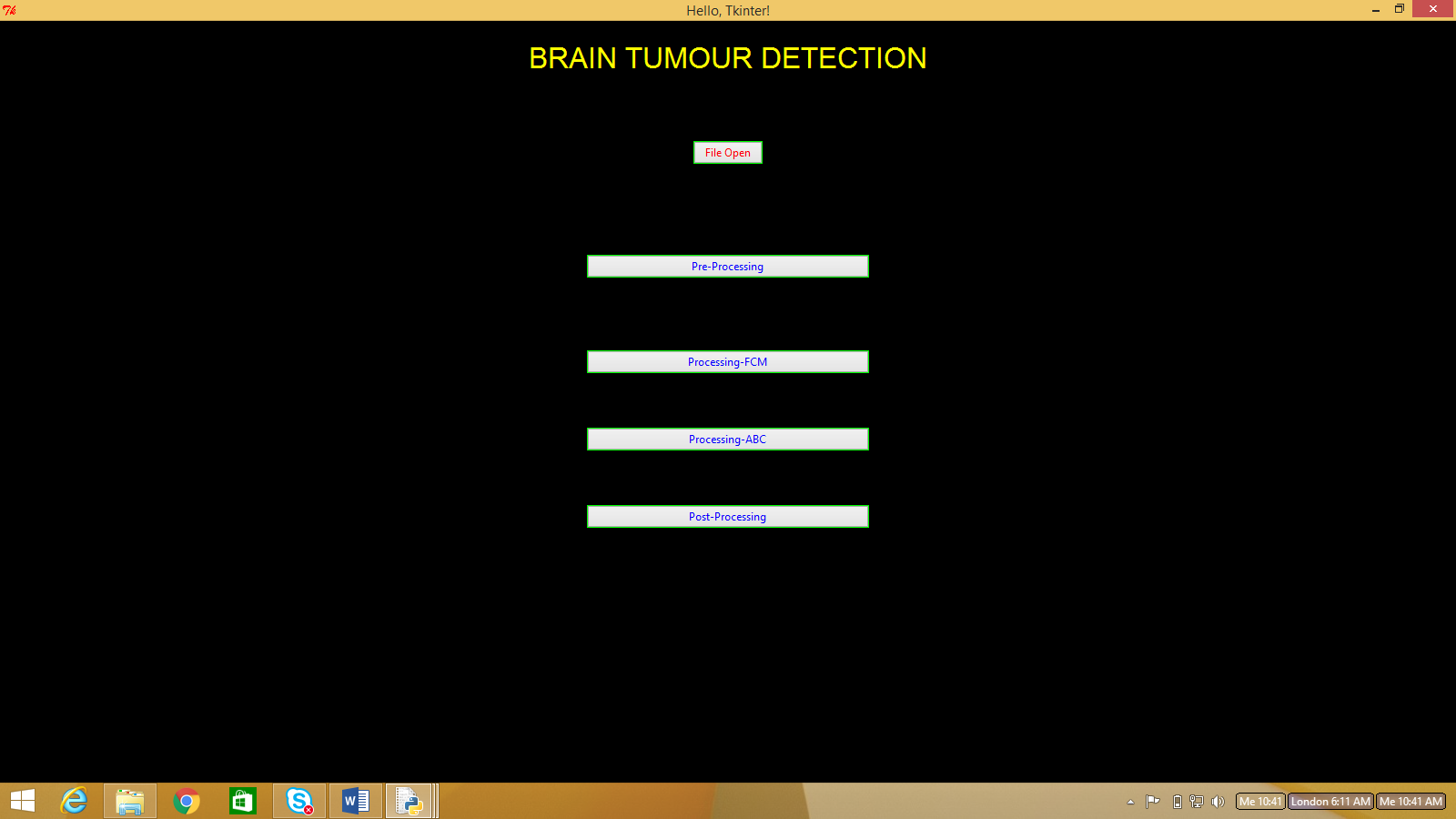


Fig 9.1 Initial Page

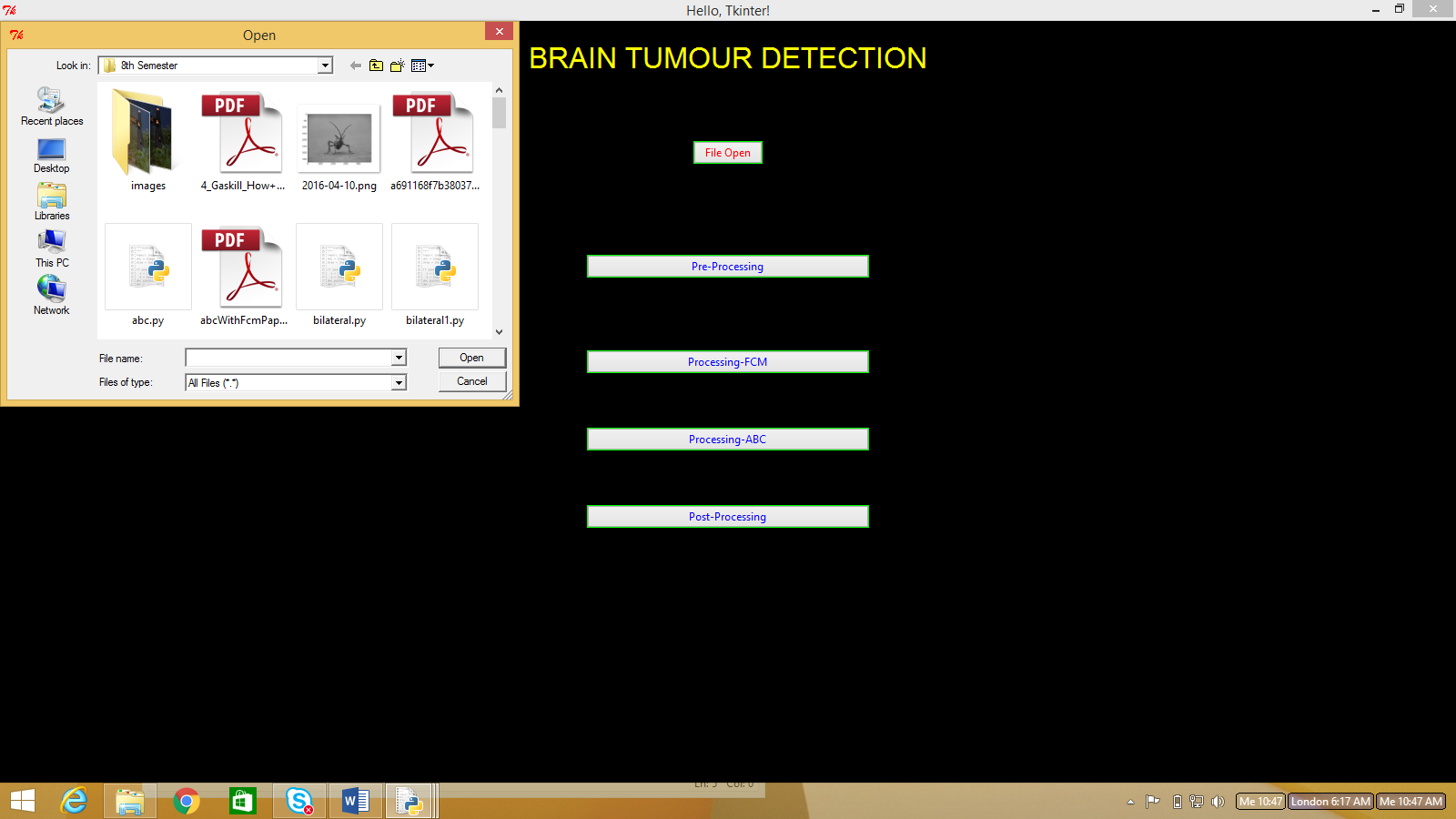


Fig 9.2 Choosing MRI Image

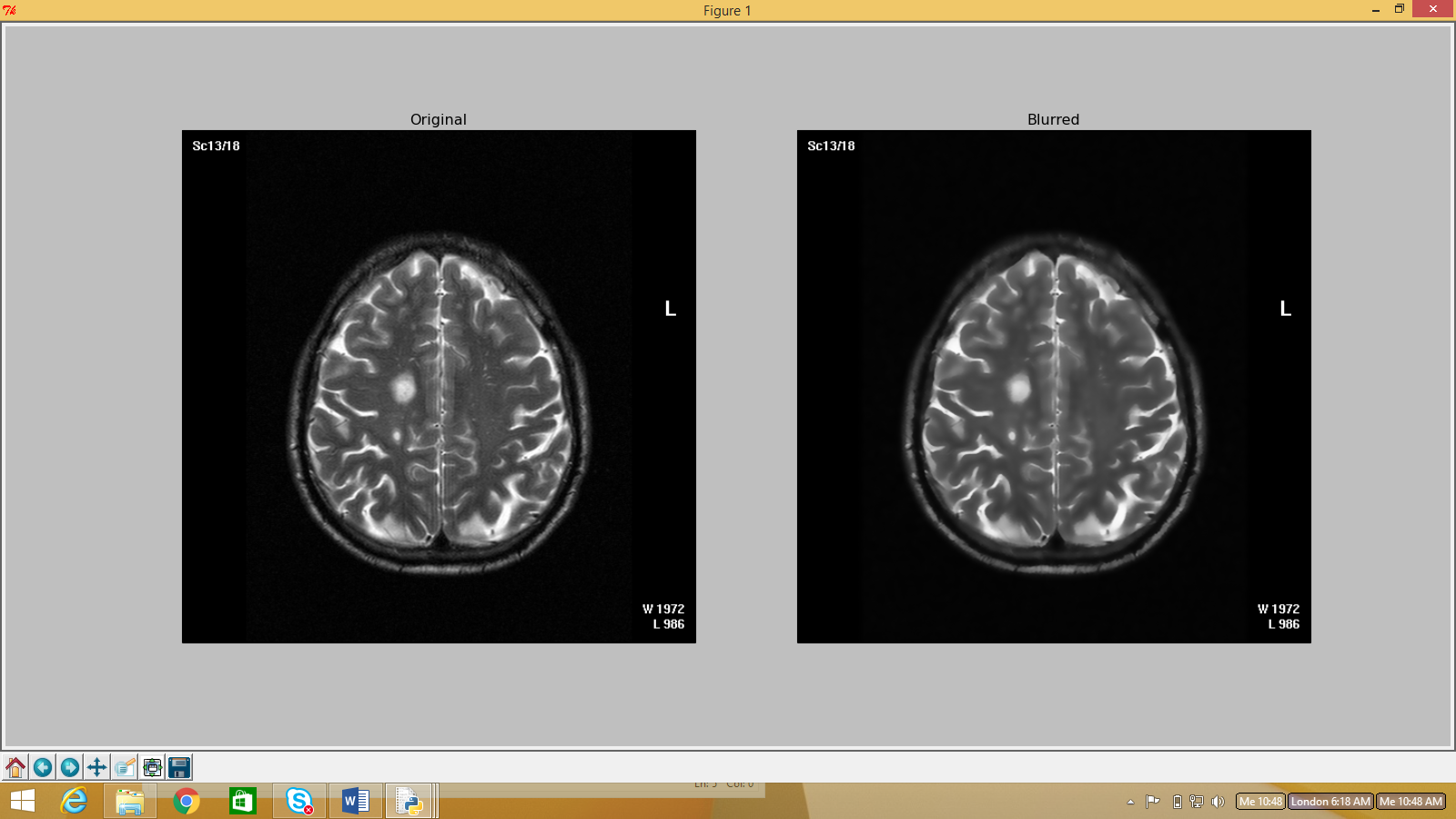


Fig 9.3 Pre-processing

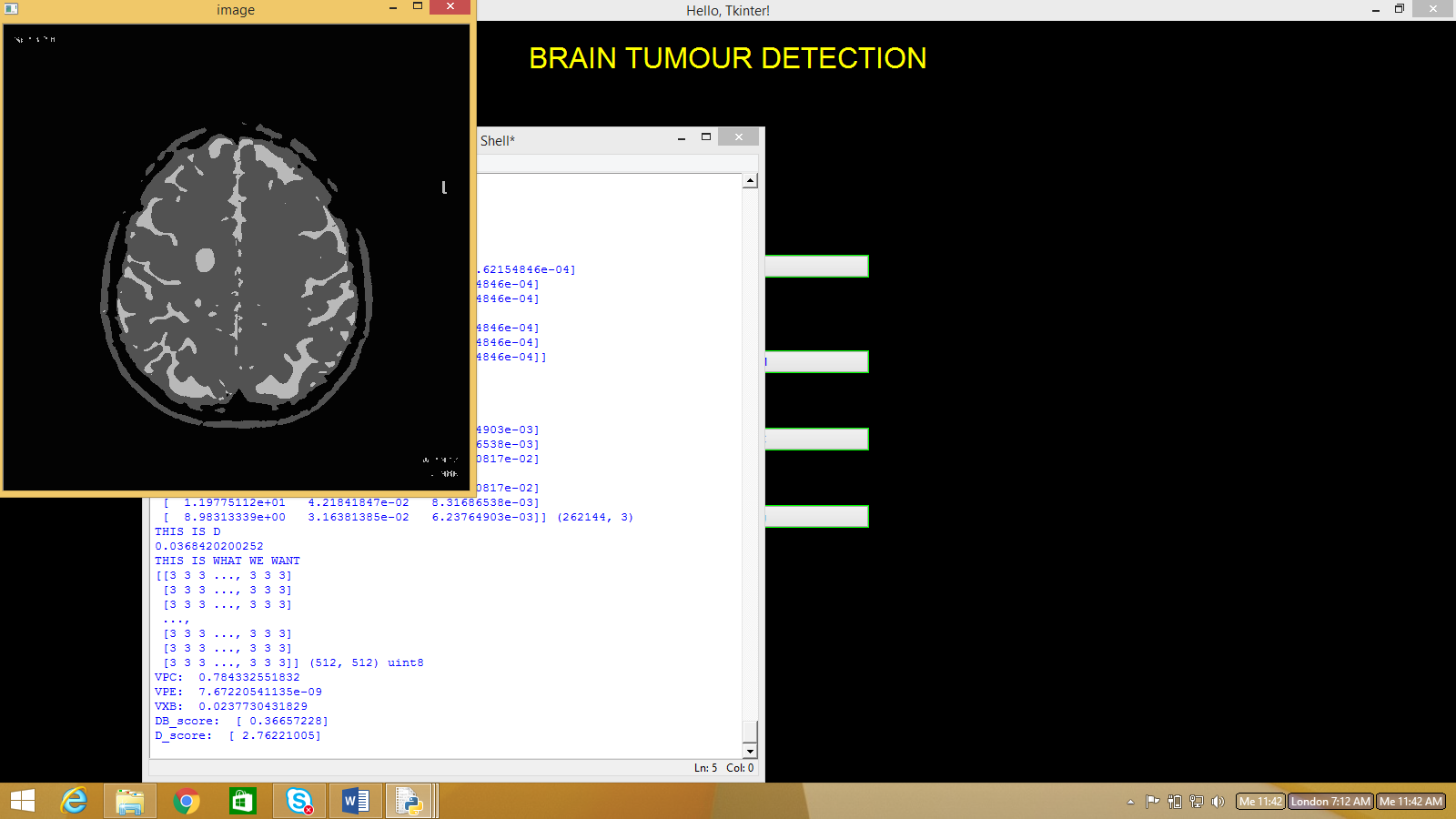


Fig 9.4 Processing FCM

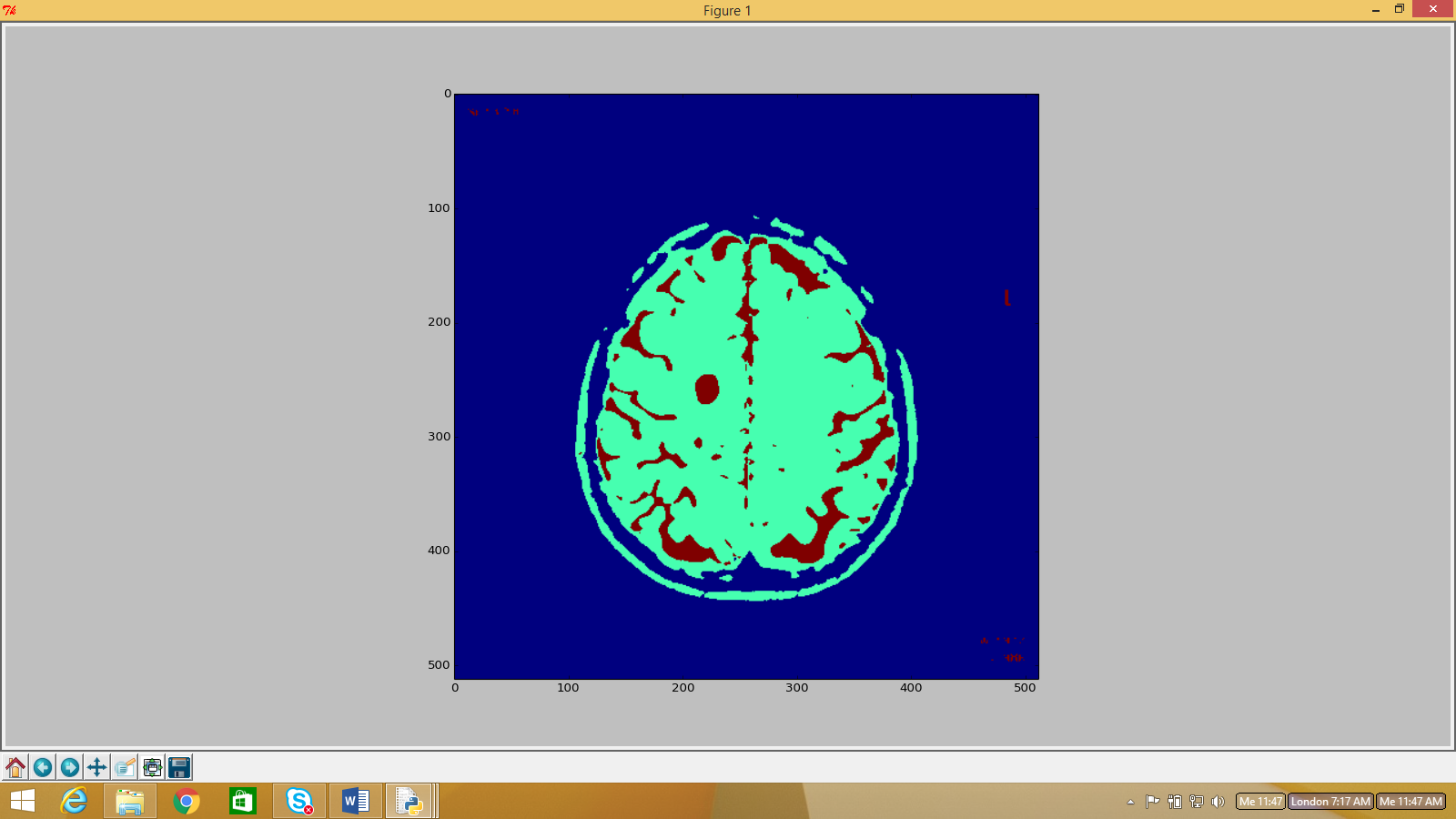


Fig 9.5 Processing ABC

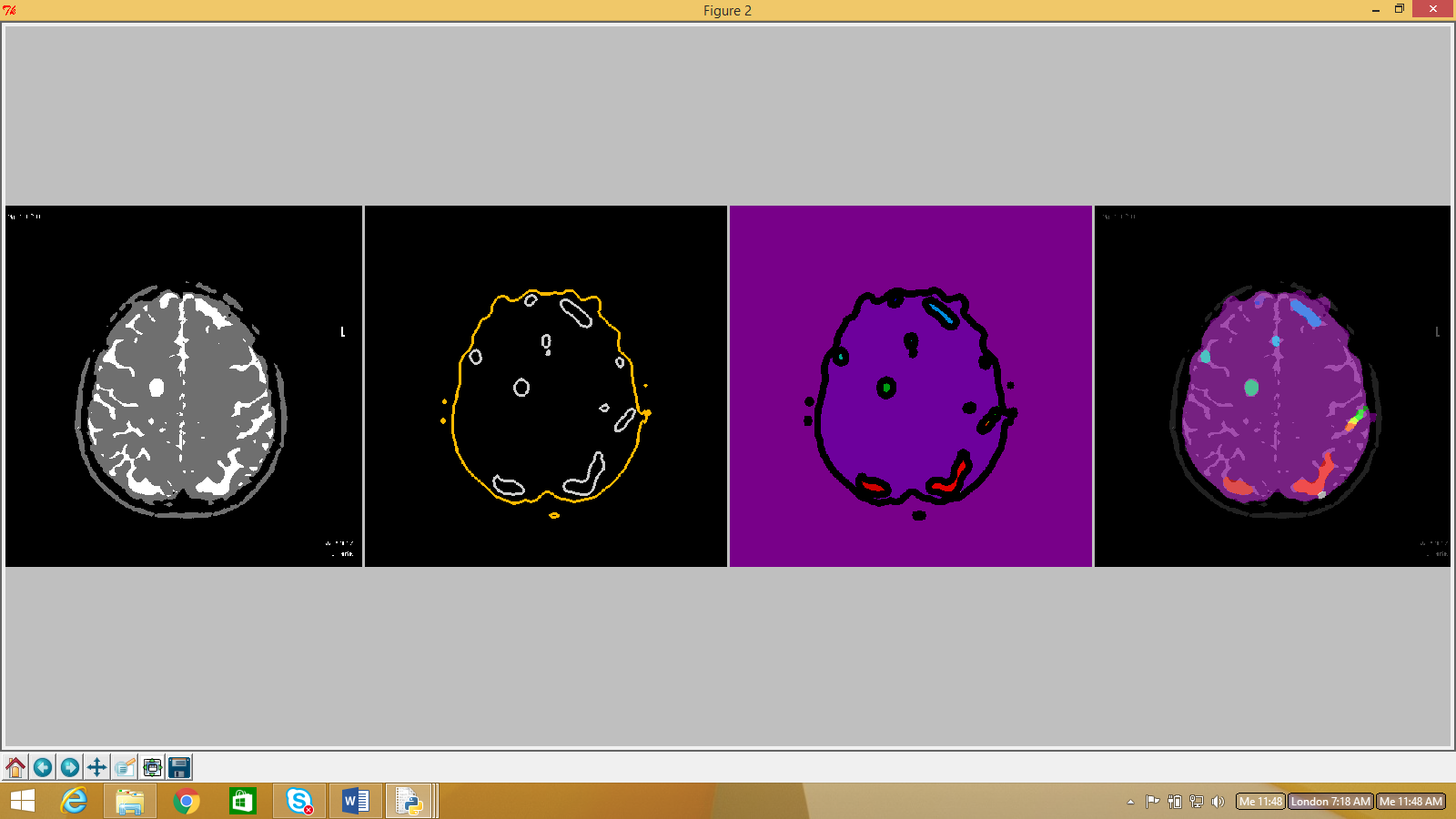


Fig 9.6 Watershed

**Chapter 10**

**Conclusion and Future Work**

**10.1 Conclusion**

In pre-processing, Gaussian, Median, Adaptive Median and Bilateral filters were analyzed and compared. Both visual and numerical results indicate that Bilateral filter outperforms the others.

Initially, the enhanced MRI image was run on FCM. The output of which was run on ABC. The visual results showed that the combination of the

two algorithms gave better results.

Project accuracy is 75-80%.

**10.2 Future Work**

The future work will be to optimize the algorithms better, and to try and make the code more efficient and fast. Also improving the accuracy further.

**Chapter 11**

**REFERENCES**

[1] Karaboga D. An Idea Based On Honey Bee Swarm For Numerical Optimization, Technical Report-TR06, Erciyes university, Engineering Faculty, Computer Engineering Department, 2005.

[2]  Dervis Karaboga and Celal Ozturk,”Fuzzy Clustering with Artificial Bee Colony Algorithm”, Scientific Research and Essays, Vol 5.

[3] T. Logeswari and M. Karnan, An improved implementation of brain tumor detection using segmentation based on soft computing, International Journal of Computer Theory and Engineering, Vol. 2, No. 4, August, 2010.

[4] E. Bonabeau, M. Dorigo, G. Theraulaz, “Swarm intelligence: From Natural to Artificial Systems” New York, NY:Oxford University Press, 1999

[5] Emrah Hancer, Celal Ozturk and Dervis Karaboga “Extraction of brain tumor from MRI images with artificial bee colony segmentation methodology” Department of Computer Engineering, Erciyes University, Turkey.

[6] Bahriye, A. and D. Karaboga, 2012. A modified artificial bee colony algorithm for real-Parameter optimization. Int. J. Inform. Sci., 192: 120-142. DOI: 10.1016/j.ins.2010.07.015

[7] http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html

[8] <http://docs.opencv.org/3.1.0/d3/db4/tutorial_py_watershed.html>

[9] Anam Mustaqeem ,Ali Javed, Tehseen Fatima, An Efficient Brain Tumor Detection Algorithm Using Watershed & Thresholding Based Segmentation, I.J. Image, Graphics and Signal Processing, 2012, 10, 34-39 Published Online September 2012 in MECS