Application of Type-2 Fuzzy Logic to remove noise from colour images by using Bacterial Foraging

Major project submitted in partial fulfilment of the

Requirement for the award of degree of

Master of Technology

in

Information Systems

Submitted by

Minal Bansal

(2K11/ISY/15)

Under the guidance of

Prof. O. P. Verma

(HOD, IT Department)

Department of Information Technology

Delhi Technological University

Bawana Road, Delhi – 110042

(2011-2013)

CERTIFICATE

This is to certify that **Ms. Minal Bansal (2K11/ISY/15)** has carried out the major project titled "**Application of Type-2 Fuzzy Logic to remove noise from colour images by using Bacterial Foraging** " as a partial requirement for the award of Master of Technology degree in Information Systems by Delhi Technological University.

The major project is a bonafide piece of work carried out and completed under my supervision and guidance during the academic session **2011-2013**. The matter contained in this report has not been submitted elsewhere for the award of any other degree.

(Project Guide)
Prof. O. P. Verma
Head of Department
Department of Information Technology
Delhi Technological University
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I express my gratitude to my major project guide Prof. O. P. Verma, HOD, IT Dept., Delhi Technological University, for the valuable support and guidance he provided in making this major project. It is my pleasure to record my sincere thanks to my respected guide for his constructive criticism and insight without which the project would not have shaped as it has.

I humbly extend my words of gratitude to other faculty members of this department for providing their valuable help and time whenever it was required.

Minal Bansal

Roll No. 2K11/ISY/15

M.Tech (Information Systems)

E-mail: minalbansal.hr@gmail.com

ABSTRACT

Usually during acquisition or transmission of an image, it gets contaminated with the impulse noise. In this paper, we present a novel application of type-2 fuzzy logic to the design of an image processing operator called an impulse detector to remove this impulse noise from images. The type-2 fuzzy logic based impulse detector can be used to guide impulse noise removal filters to significantly improve their filtering performance and enhance their final images. The structure of the proposed impulse detector is based on two 3-input and 1-output first order Sugeno type, interval type-2 fuzzy inference system. The values of the internal parameters of the type-2 fuzzy membership functions of the systems are determined by experiment. When a noisy image is passed through the detector, its job is to determine which of the pixels are noisy and which are not and then filter is applied on these noisy pixels to output a restored image. Two advantages are addressed through the use of detector. One is, not all the pixels of an image are noisy. So, it helps in recognizing the noisy pixels. Other is, filter need not to be applied on complete image but is applied only on noisy pixels. Application of filter on complete image degrades the quality of image. So, the usage of detector before filtering helps in restoring the quality of image as well as it cuts cost. The performance of the impulse detector is evaluated by using it in combination with median impulse noise filter on four different popular test images under realistic noise condition of 30%. The results demonstrate that the type-2fuzzy logic based impulse detector can be used as an efficient tool to effectively improve the performances of impulse noise filters and reduce the impulse noise undesirable distortion effects. But it has been shown that better performance can be expected if the internal parameters of the fuzzy inference system are optimized. The internal parameters of the type-2 fuzzy membership functions of the systems are determined by training. Bacterial Foraging Optimization Technique has been used for the training purpose. In order to determine the ideal behaviour of the impulse detector, an ideal detector has been developed which outputs the 100% accurate results which can be used by the optimization algorithm for the computation purpose. Also, it helps in comparing the results by the detector with and without optimization.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction

During acquisition or transmission of a digital image, it usually gets contaminated by impulse noise due to imperfections in used image sensors or in communication channels. It is important to remove this impulse noise from the image as different image processing tasks like edge detection, object recognition, image segmentation etc. are affected by noise. While removing noise from a digital image, the details in the image should be preserved.

In the field of image processing, several techniques exist for filtering images corrupted by impulse noise. Most of these methods focus on removal of noise irrespective of its effect on quality of the image. When an image is contaminated by impulse noise, only a subset of pixels are corrupted and rest of the pixels are unaffected by noise. Application of noise filter on complete image is not desirable as it degrades the details in the image. So, various methods were proposed to develop an impulse detector which detects the corrupted pixels in the image. If the pixel is found corrupted, then its value is restored by applying certain technique on pixels in a given filtering window. If the pixels is found uncorrupted by impulse noise, then its value is left unaltered.

The extent of having advantage of using impulse detector before filtering directly depends on the performance of the detector. The performance of the detector can be measured by counting whether the pixels determined by detector as corrupted are actually corrupted as well as the pixels determined as uncorrupted are actually uncorrupted. Better is the performance of impulse detector; higher is the quality of the image being restored.

With the growing interest in applications of fuzzy logic, detectors were developed which use *type-2 fuzzy logic systems (FLS)* to determine the corruptness of a pixel. The membership functions of a type-2 fuzzy logic system are fuzzy which helps in handling higher level of uncertainty usually found in noisy environments.

Here, we present a novel impulse detector based on first order type-2 Sugeno fuzzy logic to restore impulse noise corrupted images and preserving the details and texture of image at the same time. Experimental results show that our method performs superior than the competing methods in terms of noise removal as well as details preservation.

1.2 Image Processing

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

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

Image processing basically includes the following three steps:

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

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

1. Visualization - Observe the objects that are not visible.
2. Image sharpening and restoration - To create a better image.
3. Image retrieval - Seek for the image of interest.
4. Measurement of pattern – Measures various objects in an image.
5. Image Recognition – Distinguish the objects in an image.

The two types of methods used for Image Processing are Analog and Digital Image Processing. Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that has to be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing. Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from satellite platform contains deficiencies. To get over such flaws and to get originality of information, it has to undergo various phases of processing. The three general phases that all types of data have to undergo while using digital technique are Pre- processing, enhancement and display, information extraction.

There are wide applications of image processing. Some of them are listed below:

1. Intelligent Transportation Systems – This technique can be used in Automatic number plate recognition and Traffic sign recognition.

2. Remote Sensing – For this application, sensors capture the pictures of the earth's surface in remote sensing satellites or multi – spectral scanner which is mounted on an aircraft. These pictures are processed by transmitting it to the Earth station. Techniques used to interpret the objects and regions are used in flood control, city planning, resource mobilization, agricultural production monitoring, etc.

3. Moving object tracking – This application enables to measure motion parameters and acquire visual record of the moving object. The different types of approach to track an object are:

   i) Motion based tracking
   ii) Recognition based tracking

4. Defence surveillance – Aerial surveillance methods are used to continuously keep an eye on the land and oceans. This application is also used to locate the types and formation of naval vessels of the ocean surface. The important duty is to divide the various objects present in the water body part of the image. The different parameters such as length, breadth, area, perimeter, compactness are set up to classify each of divided objects. It is important to recognize the distribution of these objects in different directions that are east, west, north, south, northeast, northwest, southeast and south west to explain all possible formations of the vessels. We can interpret the entire oceanic scenario from the spatial distribution of these objects.

5. Biomedical Imaging techniques – For medical diagnosis, different types of imaging tools such as X- ray, Ultrasound, computer aided tomography (CT) etc. are used.

Some of the applications of Biomedical imaging applications are as follows:

i) Heart disease identification– The important diagnostic features such as size of the heart and its shape are required to know in order to classify the heart diseases. To improve the diagnosis of heart diseases, image analysis techniques are employed to radiographic images.

ii) Lung disease identification – In X- rays, the regions that appear dark contain air while region that appears lighter are solid tissues. Bones are more radio opaque than tissues. The ribs, the heart, thoracic spine, and the diaphragm that separates the chest cavity from the abdominal cavity are clearly seen on the X-ray film.

iii) Digital mammograms – This is used to detect the breast tumour. Mammograms can be analysed using Image processing techniques such as segmentation, shape analysis, contrast enhancement, feature extraction, etc.

6. Automatic Visual Inspection System – This application improves the quality and productivity of the product in the industries.

· Automatic inspection of incandescent lamp filaments – This involves examination of the bulb manufacturing process. Due to no uniformity in the pitch of the wiring in the lamp, the filament of the bulb gets fused within a short duration. In this application, a binary image slice of the filament is created from which the silhouette of the filament is fabricated. Silhouettes are analysed to recognize the non-uniformity in the pitch of the wiring in the lamp. This system is being used by the General Electric Corporation.

i) Automatic surface inspection systems – In metal industries it is essential to detect the flaws on the surfaces. For instance, it is essential to detect any kind of aberration on the rolled metal surface in the hot or cold rolling mills in a steel plant. Image processing techniques such as texture identification, edge detection, fractal analysis etc. are used for the detection.

ii) Faulty component identification – This application identifies the faulty components in electronic or electromechanical systems. Higher amount of thermal energy is generated by these faulty components. The Infra-red images are produced from the distribution of thermal energies in the assembly. The faulty components can be identified by analysing the Infra-red images.

A wide research is being done in the Image processing technique:

1. Cancer Imaging – Different tools such as PET, MRI, and Computer aided Detection helps to diagnose and be aware of the tumour.
2. Brain Imaging – Focuses on the normal and abnormal development of brain, brain ageing and common disease states.
3. Image processing – This research incorporates structural and functional MRI in neurology, analysis of bone shape and structure, development of functional imaging tools in oncology, and PET image processing software development.
4. Imaging Technology – Development in image technology have formed the requirement to establish whether new technologies are effective and cost beneficial. This technology works under the following are:
   i) Magnetic resonance imaging of the knee
   ii) Computer aided detection in mammography
   iii) Endoscopic ultrasound in staging the oesophageal cancer
   iv) Magnetic resonance imaging in low back pain
   v) Ophthalmic Imaging
5. Development of automated software- Analyses the retinal images to show early sign of diabetic retinopathy.
6. Development of instrumentation – Concentrates on development of scanning laser ophthalmoscope.

We all are in midst of revolution ignited by fast development in computer technology and imaging. Against common belief, computers are not able to match humans in calculation related to image processing and analysis. But with increasing sophistication and power of the modern computing, computation will go beyond conventional, Von Neumann sequential architecture and would contemplate the optical execution too. Parallel and distributed computing paradigms are anticipated to improve responses for the image processing results.

1.3 Noise

Image noise is random (not present in the object imaged) variation of brightness or color information in images, and is usually an aspect of electronic noise. It can be produced by the sensor and circuitry of a scanner or digital camera. Image noise can also originate in film grain and in the unavoidable shot noise of an ideal photon detector. It is an undesirable by-product of image capture that adds spurious and extraneous information.

The original meaning of "noise" was and remains "unwanted signal"; unwanted electrical fluctuations in signals received by AM radios caused audible acoustic noise ("static"). By analogy unwanted electrical fluctuations themselves came to be known as "noise". Image noise is, of course, inaudible. The magnitude of image noise can range from almost imperceptible specks on a digital photograph taken in good light, to optical and radio astronomical images that are almost entirely noise, from which a small amount of information can be derived by sophisticated processing (a noise level that would be totally unacceptable in a photograph since it would be impossible to determine even what the subject was).

Different types of noise are:

1. Gaussian noise - Gaussian noise is statistical noise that has a probability density function of the normal distribution (also known as Gaussian distribution). In other words, the values that the noise can take on are Gaussian-distributed. It is most commonly used as additive white noise to yield additive white Gaussian noise (AWGN).
2. Poisson noise - Poisson noise has a probability density function of a Poisson distribution.
3. Salt & pepper noise - It represents itself as randomly occurring white and black pixels. An effective noise reduction method for this type of noise involves the usage of a median filter. Salt and pepper noise creeps into images in situations where quick transients, such as faulty switching, take place. The image after distortion from salt and pepper noise looks like the image attached.

   Fat-tail distributed or **impulsive noise** is sometimes called salt-and-pepper noise or spike noise. An image containing salt-and-pepper noise will have dark pixels in bright regions and bright pixels in dark regions. This type of noise can be caused by Analog-to-digital converter errors, bit errors in transmission, etc. It can be mostly eliminated by using dark frame subtraction and interpolating around dark/bright pixels.
4. Speckle noise - Speckle noise is a granular noise that inherently exists in and degrades the quality of images. Speckle noise is a multiplicative noise, i.e. it is in direct proportion to the local grey level in any area. The signal and the noise are statistically independent of each other.

These types of noise lead to the development in the field of image processing by the introduction of various techniques to remove this noise. Some of them are:

1. Arithmetic mean filter: This filter computes the average value of the pixels intensity values in the sub-image (of size m x n).
2. Geometric mean filter: It is similar to an arithmetic mean filter, but it tends to lose less detail in the process.
3. Harmonic mean filter: This filter computes the harmonic mean of the pixels intensity values.
4. Contraharmonic mean filter: This filter computes the contraharmonic mean of the pixels intensity values. Note that the contraharmonic filter reduces to the mean filter for $Q = 0$, and to the harmonic mean filter for $Q = -1$.
5. Median filter: Replaces the value of the pixel by the median of the pixels in the sub-image.
6. Max filter: For $d = 1$, replaces the value of the pixel by the maximum of the pixel intensity values in the sub-image. For $d>1$, uses the mean of the top d values.
7. Min filter: For $d = 1$, replaces the value of the pixel by the minimum of the pixel intensity values in the sub-image. For $d>1$, uses the mean of the lowest d values.
8. Mid-point filter: Replaces the value of the pixel by the mid-point of the pixels in the sub-image.
9. Alpha trimmed mean filter: Replaces the value of the pixel by the mean of the remaining pixel intensity values after discarding the top $d/2$ and lowest $d/2$ intensity values.

1.4 Fuzzy Logic

Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based. The idea of fuzzy logic was first advanced by Dr. Lotfi Zadeh of the University of California at Berkeley in the 1960s. Dr. Zadeh was working on the problem of computer understanding of natural language. Natural language (like most other activities in life and indeed the universe) is not easily translated into the absolute terms of 0 and 1. (Whether everything is ultimately describable in binary terms is a philosophical question worth pursuing, but in practice much data we might want to feed a computer is in some state in between and so, frequently, are the results of computing.)

Fuzzy logic includes 0 and 1 as extreme cases of truth (or "the state of matters" or "fact") but also includes the various states of truth in between so that, for example, the result of a comparison between two things could be not "tall" or "short" but ".38 of tallness." Fuzzy logic seems closer to the way our brains work. We aggregate data and form a number of partial truths which we aggregate further into higher truths which in turn, when certain thresholds are exceeded, cause certain further results such as motor reaction. A similar kind of process is used in artificial computer neural network and expert systems. It may help to see fuzzy logic as the way reasoning really works and binary or Boolean logic is simply a special case of it. It is a type of logic that recognizes more than simple true and false values. With fuzzy logic, propositions can be represented with degrees of truthfulness and falsehood. For example, the statement, *today is sunny,* might be 100% true if there are no clouds, 80% true if there are a few clouds, 50% true if it's hazy and 0% true if it rains all day.

Fuzzy logic has proved to be particularly useful in expert system and other artificial intelligence applications. It is also used in some spell checkers to suggest a list of probable words to replace a misspelled one.

CHAPTER 2

BACTERIAL FORAGING

2.1 Introduction

As the studies tell, natural selection works to eliminate animals with poor foraging strategies i.e. methods for locating, handling, and ingesting food and favour the propagation of genes of those animals that have successful foraging strategies because they are more likely to enjoy reproductive success as they obtain enough food to enable them to reproduce. After generations and generations, poor foraging techniques are either eliminated or redesigned into good ones. After clear examination of these evolutionary principles, scientists in the field of "foraging theory" have tried to hypothesize that it is appropriate to model the activity of foraging as an optimization process. The process says that a foraging animal takes actions to maximize the energy obtained per unit time spent in foraging, under constraints presented by its own physiology and environment where physiology means sensing and cognitive capabilities and environment covers the density of prey and risks from predators as well as physical characteristics of the search area. These constraints have been balanced by evolution and then, essentially engineered what is sometimes called as an optimal policy of foraging. Such a term is especially justified in cases where the models and policies have been validated. Some optimization models have also been valid for social foraging where groups of animals cooperatively forage.

2.2 Elements of Bacterial Foraging

The theory of foraging is based on the assumption that animals search for nutrients and obtain them in a way that tends to maximize their energy intake E per unit time T spent foraging. It can be somewhat presented in a function like

$$\frac{E}{T}$$

Maximizing such a function provides nutrient sources to survive and additional time for other important activities like fighting, fleeing, mating, reproducing, sleeping, or shelter building. Sometimes shelter-building and mate-finding activities bear similarities to foraging. Clearly, foraging is very different for different species. Generally herbivores find food easily but must eat a lot of it. On the contrary, carnivores generally find it difficult to locate food but do not have to eat much as their food is of high energy value. The pattern of available nutrients is established by environment like through what other organisms can be nutrients available or geological constraints like rivers and mountains, and weather patterns and then it places some constraints on obtaining that food like small portions of that food may be separated bylarge distances. Also, risk is involved during foraging due to predators, the prey may be mobile and hence it must be chased and then its ultimate success is dependent on physiological characteristics of the forager.

In some cases, the food (nutrients) is distributed in patches. Now it is the work of animal to find these patches and then decide whether to enter a particular patch and search for food and then continue in this direction or move and find another patch depending upon its characteristics like the quality and quantity of nutrients present in the patch. Generally the patches are encountered sequentially, and many times risk and great effort are needed to move from one patch to the other. Past experience plays an important role as if an animal finds a nutrient-poor patch, but on the basis of past experience, it expects that there should be better patch somewhere else, then it will take risk and efforts to locate another patch, and if it finds that acceptable then, it will seek another patch. Moreover, if for some time an animal has been in a patch, it can start to deplete its resources, so that there should be an optimal time after which animal can leave the patch and venture out to try to locate a richer one.

Optimal foraging theory states the foraging problem as an optimization problem and via computational or analytical methods can provide an optimal foraging policy that specifies how foraging decisions are made. Generally, dynamic programming formulations have been used [1].

2.3 Search Techniques of Foraging Animals

There are various techniques of searching for foraging animals. In one case, the predation is broken into various components which are similar for most of the animals. The predation components are locating the prey, attacking it, handling it and ingesting it. The ease of foraging depends on the relationship between the size of predator and prey. If the size of the prey is large in comparison to predator, then it becomes easy for predator to locate the prey. Then the only problem lies in handling the prey. On the contrary, if the size of the prey is smaller than that of predator, then it is easy to handle and ingest it, effort is only applied in locating and attacking the prey. In general, the size of prey is smaller than predator and hence effective search techniques are required to locate the prey.

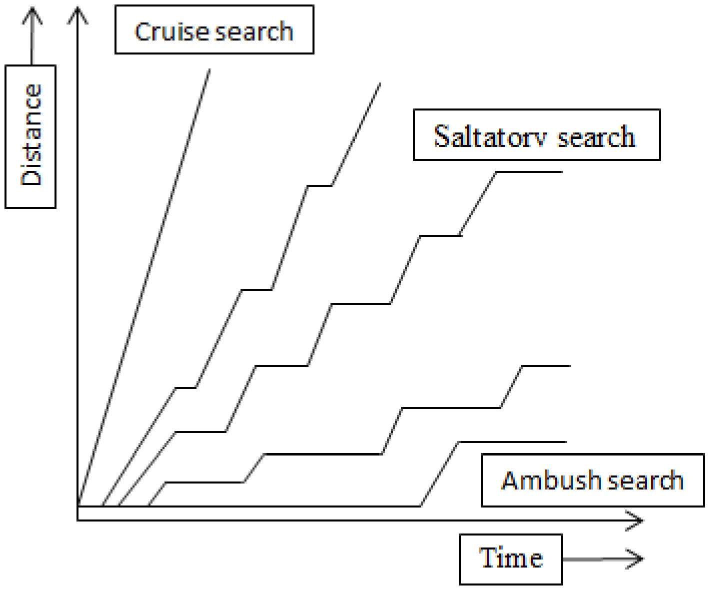


Figure 1: *Search Techniques of foraging animals* [2]

Various types of search strategies are being followed by different animals. Two such search strategies are cruise and ambush search. In “cruise” search, the forager moves through the entire environment continuously and the distance constantly increases with time. In “ambush” search, the forager waits for the prey to cross the strike area of forager. Most foraging animals have strategies lying between ambush and cruise search. Many saltatory search techniques are possible with an alternative sequence of cruising and waiting. Saltatory search can be adjusted to suit the environment.

2.4 Social and Intelligent Foraging

The above discussed foraging and search strategies were for individual animals. However, there can be more advantages to group (or social) foraging. Some means of communication is required for group foraging. Like in humans, this could be language. It might be certain movements or noises or trail-laying mechanisms in other animals. The advantages of group foraging include:

• Since more animals are searching for nutrients, the likelihood of finding nutrients increases. When one animal finds nutrients, it can tell others in the group where they are. Such a joining to a group provides access to an information centre that helps in survival.

• It also helps in increasing capability to cope with larger prey. The group can join up on a large prey and attack (kill) and ingest it.

•Also, protection from predators can be provided by members of the group to each other.

Sometimes it is also useful to think of a group of animals as a single living creature and grouping and communication as a collective intelligence that actually results in more successful foraging for each individual of the group.

For social foraging, you may consider how a pack of wolves hunts or a flock of birds, colony of ants, swarm of bees, or school of fish behave. Connections between foraging behaviour of colonies of ants and optimization engineering applications have been studied. Bonabeau et al. [3] explain how colonies of ants can solve minimum spanning-tree problems, shortest-path problems, and traveling salesperson problems etc. These ants use communication which is indirect called "stigmergy", in which one ant can change its environment and later another ant can modify its behaviour because of that modification. The process can be understood with an example, if an ant goes out for foraging, it may search a lot in a relatively random pattern. And once it locates a food source, it gets back to the anthill leaving a trail of "pheromone" which can stay there for up to a few months. Then, when other ants go out for foraging, they follow the pheromone trail and locate food more easily. We can consider the first ant as having recruited additional foragers and the trails as a type of memory for the whole ant colony. Memory, communications and learning result in more intelligent and efficient foraging for the group.

Likewise, other social insects use different communication methods. For example, it has been found that after successful foraging, a bee will come back to the hive and tells the quality and location of the food source through different types of dances. Group behaviour of different organisms through computer simulations can be studied in these books [4], [5], [6]. Cooperation in groups can help lower life forms achieve higher forms of foraging intelligence.

2.5 Bacterial Foraging: Algorithm

The three basic iterative steps in bacterial foraging algorithm are:

1. Chemotaxis and Swarming: Suppose m1,m2,m3 are the parameters to be optimized. They represent axis of space coordinates (like x,y,z axis in rectangular coordinate system). Now let fi,j,k,l(m1,m2,m3) represents position of ith bacterium at a point in m1,m2,m3 coordinate system, in jth Chemotaxis, kth reproduction and lth elimination and dispersal step. Also let C(i) represents unit run-length of a bacterium. Let del(i) is three elements direction vector (because position of bacteria being represented by three coordinates which are basically optimization parameters). Each element of del (i) is a random number lying between [-1, 1]. If ith bacterium meets favourable environment which is represented by less value of cost function at that point in space coordinates, it swims which means direction vector will remain same as was in previous (j-1th) chemotaxis step. Otherwise, del (i) is assigned with a new value which is a random number lying between [-1,1]. After each chemotaxis step, the bacteria move and reach new points in space (whose coordinate axis are optimization parameters). Each point represents a set of optimization parameters. Here, at these present locations, fitness of each bacterium is evaluated which further decides next movement of the bacterium. Fitness of ith bacterium is represented by Cost function Pi,j,k,l. Better fitness mean less value of Cost function.

2. Reproduction: Health status (fitness) of each bacterium is calculated after each complete chemotaxis process. It is overall sum of cost function , where NC is total number of steps in a complete chemotaxis process. Locations of healthier bacteria represent better sets of optimization parameters. Then, to further speed up and refine the search, more number of bacteria is required to be placed at these locations in the optimization domain. This is done in reproduction step. Healthiest half of bacteria (with minimum value of cost function) are let to survive, while the weaker half die. Each surviving bacterium splits up into two and these two are placed at the same location. In this way population of bacteria remains constant.

3. Elimination and Dispersal Event: The chemotaxis process performs local search and reproduction speeds up convergence of search parameters. But, chemotaxis and reproduction may not be enough to reach the global minimum point (best optimized set of parameters). The bacteria may also get trapped in local minima assuming it to be the best fitness position in the surrounding patch. To avoid this to happen, elimination and dispersal event is

performed. The bacterium having probability Ped (probability of elimination and dispersion) is eliminated from present location and one bacterium is placed (dispersion) at a random location so as to realize global search. The population of bacteria still remains constant.

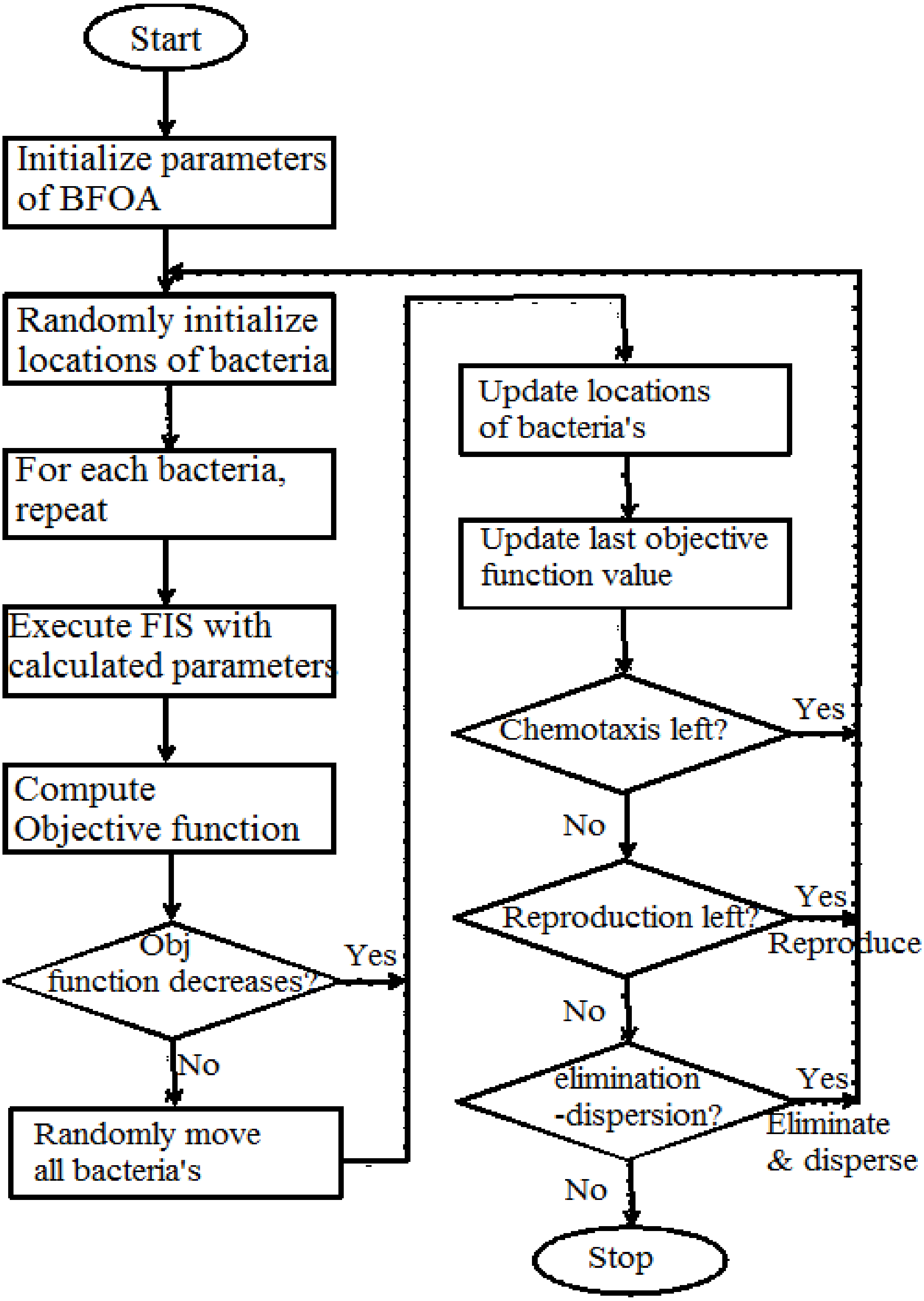


Figure 2: *Bacterial Foraging Optimization Algorithm*

2.6 E.coli

The *Escherichia.coli* bacterium has a plasma membrane, cell wall, and capsule which contains the cytoplasm and nucleoid. Thepiliare used as a type of gene transfer toother *E. coli* bacteria, and locomotion is done with the help of flagella. The cell is approximately 1 μmin diameter and2 μm in length. Weight of the *E. coli* cell about 1picogram and contains about 70% water. It is probably the best understood microorganism. The sequencing of its entire genome has been done. It contains about 4 million of the A, G, C, and T letters which refers to adenosine, guanine, cytosine and thymine—arranged into a total of about 4 thousand genes. Mutation rate in *E. coli* is about 10−7per gene, per generation, and can affect its physiological aspects. *E. coli* bacteria generally engage in a type of sex called conjugation in which small gene sequences are uni directionally transferred from one bacterium to another through an extended pilus.

When *E. coli* grows, gets longer and divides in the middle into two daughters. When sufficient food is given and is held at the temperature of the human gut of 37 ° C, it can synthesize and replicate to make a copy of itself in approximately 20 min and hence growth of a population of bacteria is exponential with a relatively short time to double. For example, following[8], if today you start with one cell and sufficient food, by same time tomorrow there will be272=4.7×1021 cells, which is enough to pack a cube 17 m on one side.

Also, the *E. coli* bacterium has a control system that enables it to search for food and try to avoid noxious substances. For example, it swims away from alkaline and acidic environments and move toward more neutral ones. To explain the behaviour of the *E. coli* bacterium, we will explain its actuator i.e. the flagellum, sensors, and closed-loop behaviour. You will see that *E. coli* performs a kind of saltatory search. This section is based on the work [7]-[15].

2.7 Swimming and Tumbling

Locomotion is achieved through a set of rigid flagella which enable the bacterium to swim through each of them rotating in the same direction at approximately 100-200 revolutions per second. Each flagellum is left-handed helixes configured so that as the base of the flagellum rotates counter clockwise, if viewed from the free end of the flagellum looking toward the cell, produces a force against the bacterium so it pushes the cell. Each flagellum can be considered as a type of propeller. If a flagellum rotates clockwise, it will pull at the cell. From an engineering view, the rotating shaft at the base of the flagellum is an interesting contraption that appears to use what biologists call a universal joint. Also, the mechanism that creates the rotational forces to spin the flagellum in either direction is explained by biologists as a biological motor [8], [9], [10], [11], [12], [13], [14], [15]. The motor is quite efficient as it makes a complete revolution using only about 1,000 protons, and thus *E. coli* spends less than 1% of its energy budget for motility. An *E. coli* bacterium can move in two different ways. One is it can run (swim for a period of time) and second is it can tumble, and it alternates between these two modes of operation its entire lifetime. If the flagella rotate in clockwise direction then every flagellum pulls towards the cell, and the net effect is that each flagellum operates relatively independently of the others, and so the bacterium tumbles about.

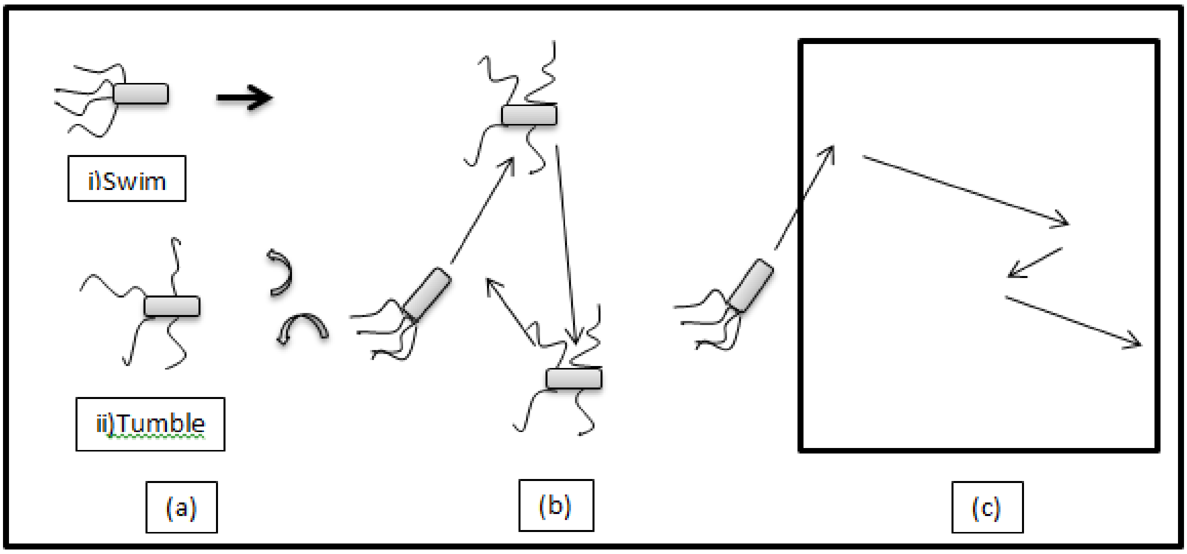


Figure 3:*Swim, tumble and chemotactic behaviour of Flagella i) Counterclockwise rotation of Flagella ii) Clockwise rotation of Flagella*

To tumble after a run, the cell slows down or stops first; since bacteria are so small, they experience almost no inertia, only viscosity, so when a bacterium stops swimming, it stops within the diameter of a proton [14]. We call the time interval during which a tumble occur a tumble interval.

2.8 Bacterial Motile Behaviour

The motion patterns that the bacteria will generate in the presence of chemical attractants and repellants are called chemotaxes. For *E. coli*, encounters with serine or aspartate result in attractant responses, whereas repellent responses result from the metal ions Ni and Co, hanges in pH, amino acids like leucine, and organic acids like acetate. What is the resulting emergent pattern of behaviour for a whole group of *E. coli* bacteria? Generally, as a group, they will try to find food and avoid harmful phenomena, and when viewed under a microscope, you will get a sense that a type of intelligent behaviour has emerged, since they will seem to intentionally move as a group. To explain how chemotaxis motions are generated, we must simply explain how the *E. coli* decides how long to run, since from the above discussion we know what happens during a tumble or run. First, note that if an *E. coli* is in some substance that is neutral in the sense that it does not have food or noxious substances, and if it is in this medium for a long time (e.g., more than 1 min), then the flagella will simultaneously alternate between moving clockwise and counter clockwise so that the bacterium will alternately tumble and run. This alternate on between the two modes will move the bacterium, but in random directions, and this enables it to search for nutrients. For example, inside the isotropic homogeneous environment as described above, the bacterium alternately tumbles and runs with the mean tumble and run lengths given above and at the speed that was already given. And if these bacteria are put in a homogeneous concentration of serine then a variety of changes occurs in the characteristics of their motile behaviour. For instance, mean run length and mean speed increase and mean tumble time decreases. They do still produce, however, a basic type of searching behaviour; even though the bacterium has some food, it persistently searches for more. Suppose that we call this its baseline behaviour. As an example of tumbles and runs in the isotropic homogeneous medium, in one try motility trial lasting 29.5 s there were 26 runs, the maximum run length was 3.6 s, and the mean speed was about 21 μm/s[8]-[12].

2.9 Sensing and Decision Making

The sensors are the receptor proteins that are signalled directly by external substances or through the periplasmic substrate-binding proteins. The sensor is very sensitive, in some cases requiring less than ten molecules of attractant to trigger a reaction, and attractants can trigger a swimming reaction in less than 200 ms. You can then think of the bacterium as having a high gain with a small attractant detection threshold. On the other hand, the corresponding threshold for encountering a homogeneous medium after being in a nutrient-rich one is larger. Also, a type of time averaging is occurring in the sensing process. The receptor proteins then affect signalling molecules inside the bacterium. Also, there is in effect an adding machine and an ability to compare values to arrive at an overall decision about which mode the flagella should operate in; essentially, the different sensors add and subtract their effects, and the more active or numerous have a greater influence on the final decision. The sensory and decision-making system in *E. coli* is probably the best understood one in biology; here, we are ignoring the underlying chemistry needed for a full explanation.

2.10 Elimination and Dispersion

It is possible that the local environment where a population of bacteria live changes either gradually or suddenly due to some other influence.

Events can occur such that all the bacteria in a region are killed or a group is dispersed into a new part of the environment. For example, local significant increases in heat can kill a population of bacteria that are currently in a region with a high concentration of nutrients. It may be that water or some animal will move populations of bacteria from one place to another in the environment. Over long periods of time, such events have spread various types of bacteria into virtually every part of our environment—from our intestines to hot springs and underground environments. What is the effect of elimination and dispersal events on chemotaxis? They have the effect of possibly destroying chemotactic progress, and they also have the impact of assisting during chemotaxis, since dispersal can place bacteria near better food sources. Through a broader perspective, elimination followed by dispersal are parts of the population-level long-distance motile behaviour.

2.11 Mobility and Swarming

Most bacteria are motile, and many types have analogoustaxes capabilities to *E. coli* bacteria. The specific sensing, actuation, and decision-making mechanisms are different [10]-[18]. Some bacteria can search for oxygen, and hence their motility behaviour is based on aerotaxis, whereas others search for desirable temperatures resulting in thermotaxis. Actually, the *E. coli* is capable of thermotaxis in that it seeks warmer environments with a temperature range of 20 to 37 ° C. Other bacteria search for or avoid light of certain wavelengths, and this is called phototaxis. Actually, the *E. coli* tries to avoid intense blue light, so it is also capable of phototaxis. Some bacteria swim along magnetic lines of force that enter the earth, so that in the northern hemisphere they swim toward the north magnetic pole and in the southern hemisphere they swim toward the south magnetic pole.

A particularly interesting group behaviour has been demonstrated for several motile species of bacteria, including *E.coli* and *S. typhimurium*, where intricate stable spatiotemporal patterns are formed in semisolid nutrient media[18]-[22].Here, we abuse the terminology and favour using the terminology that is used for higher forms of animals such as bees. When a group of *E. coli* cells is placed in the centre of a semisolid agar with a single nutrient chemo-effector, they move out from the centre in a travelling ring of cells by moving up the nutrient gradient created by consumption of the nutrient. Also, if the high amount of succinate are used as the nutrient, then the cells release the attractant aspartate so that they congregate into groups and hence move as concentric patterns of groups with high bacterial density. The spatial order results from outward movement of the ring and the local releases of the attractant; the cells provide an attraction signal to each other so they swarm together. Pattern formation can be suppressed by a background of aspartate. The pattern seems to form based on the dominance of the two stimuli.

CHAPTER 3

FUZZY INFERENCE SYSTEMS

3.1 Introduction

Fuzzy Logic (FL) is an approximate reasoning method for coping with life's uncertainties. Occasionally, the characteristics of various systems are very difficult to describe with mathematical equations because of their complexity [23] [24] [25]. Unlike the two value logic, FL is a set of mathematical based on degrees of membership rather than on the crisp membership for knowledge representation. In the essence of FL, the notion of membership in a fuzzy set is a continuous value rather than a "yes" or "no" decision. Fuzzy set is a set with uncleared boundaries in which a function is being used to assign a value for each element to show the degree of their membership. In fact, there are several types of membership functions, mostly the membership function used in the fuzzy logic are triangles, trapezoids, and Gaussians [26]. A continuous value between 0 and 1 provided by membership function is a measure for the likelihood that the instance will be in the set. Linguistic terms and numerical values may be defined by the characteristic functions as singletons, crisp sets and fuzzy sets. Alinguistic variable is used to describe a concept with the vague value and conditional statement are used to capture the human knowledge known as fuzzy rules shown as below:

$$if\ x\ is\ A\ then\ y\ is\ B$$

where x and y are linguistic variables and A and B are linguistic values determined by fuzzy sets. These rules will be used in an inference system.

Fuzzy Inference System (FIS) is the process of formulating of mapping the given input to the output which mainly includes four steps. The steps are fuzzification of input, rule evaluation folloed by aggregation of the rule outputs and then at last defuzzification [27] [28]. In the input fuzzification step, crisp inputs are converted to the fuzzy values based on the value of membership function. In the next step, using the previously fuzzified input to find out the rules and its antecedents. The process of unification of the output of all rules is called aggregation. Finally the result of the FIS is converted to the crisp value in the defuzzification phase. There are two FIS that are Mamdani and Sugeno which are varied somewhat in the way outputs are determined.

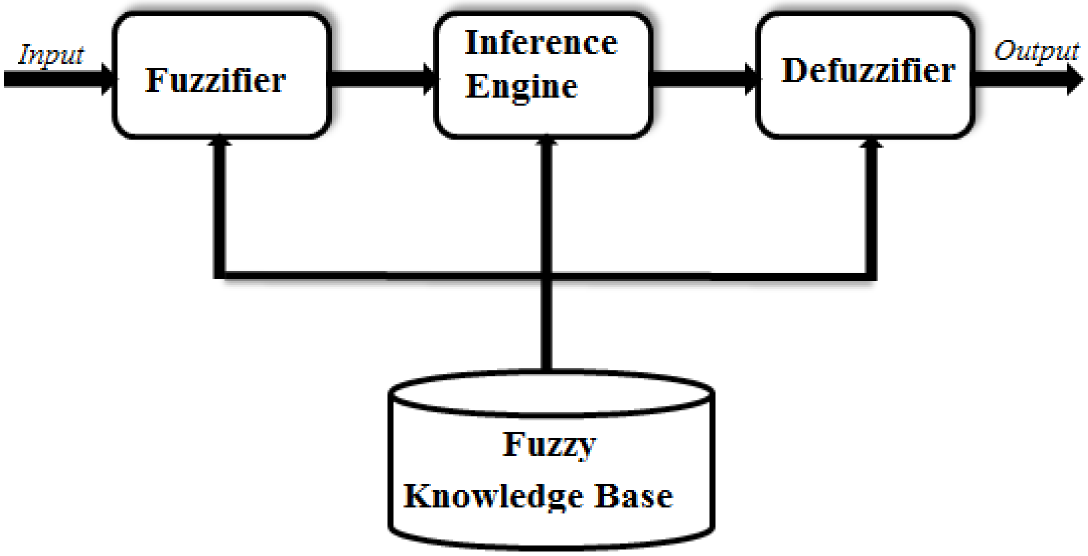


Figure 4: *Fuzzy Inference System*

One benefit of fuzzy systems is that the rule base can be created from expert knowledge, used to specify fuzzy sets to partition all variables and a sufficient number of fuzzy rules to describe the input/output relation of the problem at hand. However, a fuzzy system that is constructed by expert knowledge alone will usually not perform as required when it is applied because the expert can be wrong about the location of the fuzzy sets and the number of rules. A manual tuning process must usually be appended to the design stage which results in modifying the membership functions and/or the rule base of the fuzzy system [29] [30] [31] [32]. This tuning process can be very time consuming and error-prone. Also, in many applications expert knowledge is only partially available or not at all. It is therefore useful to support the definition of the fuzzy rule base by automatic learning approaches that make use of available data samples. This is possible since, once the components of the fuzzy system is kept in the parametric layout, the fuzzy inference system acts as a parametric model which can be tuned by a learning procedure.

3.2 Type-1 Fuzzy Set

A type-1 fuzzy set has a grade of membership that's crisp, whereas a type-2 fuzzy set has a grade of membership that's fuzzy. In fact, a type-2 fuzzy set could be called a "fuzzy-fuzzy set.” In this slide, when the primary variable called x has the value x' then the membership value for the type-1 fuzzy set is a single point (x') A μ. On the other hand, the membership value for the type-2 fuzzy set is an interval of values, as shown by the dark vertical line at x'. This interval lets us model uncertainties about the exact value of the membership function [33].

3.3 Type-2 Fuzzy Set

Let’s focus on WORDS, because from the very beginning, Zadeh used fuzzy sets to model them. Also “words mean different things to different people”. So, a type-1 fuzzy set is not able to capture the uncertainty about a word because once its membership function values are fixed there’s nothing uncertain about it. On the other hand, a type-2 fuzzy set is able to capture the uncertainty about a word because of its blurred membership function. Let’s get more specific about words and fuzzy sets [34][35].

Suppose you’d like to develop a fuzzy set model for a specific phrase, like some eye contact. It’s common to refer to such a phrase as a term or a word. Let’s begin by collecting data about this word from a group of subjects. An easy way to do this is to establish a scale—in this case zero-to-ten—and to then ask the subjects: “On a scale of zero-to-ten locate the end-points of an interval for some eye contact”. Also, all subjects won’t be giving the same end-points. Here, l and r are the sample averages of the left and right end-points for the data collected from a group of n subjects. The uncertainty intervals that are shown by the dark horizontal lines that are centred about l and r have lengths that are related to the sample standard deviations of the end-point data. Now, what do we do with all of this data? Using the uncertainty intervals, we can create a multitude of type-1 fuzzy sets. For example, let’s assume that the shape of the membership function for each type-1 fuzzy set is an isosceles triangle, and that there’s no uncertainty about the apex of the triangle, which is always located at l plus r divided by two.

Remember this is only an example, so at this time let’s not worry about triangle membership function assumption. Other choices could have been made for the shape of the type-1 fuzzy set. If we were to neglect the uncertainties about those end-points, this dashed triangle would be a very good candidate for a type-1 fuzzy set model for the word. Clearly, there are an infinite number of such triangle membership functions that would let us cover the two uncertainty bands.

3.4 Footprint of Uncertainty

So, what we do next is fill in all of the triangle membership functions. Doing this leads to the completely filled-in shaded-region that's shown on this slide, called footprint of uncertainty—FOU for short. The smaller the FOU is the less uncertainty there will be about the word. On the other hand, the larger the FOU is the more uncertainty there will be about the word. Observe that the FOU is bounded by lower and upper membership functions—LMF and UMF for short. These bounding membership functions are very important because they completely define the FOU. Each element of the FOU can be assigned a weight called a secondary grade. At a specific value of *x*, say *x'*, the collection of secondary grades over the possible membership function values is called a secondary membership function. Two very different kinds of secondary membership functions are shown on this slide—non-uniform and uniform.

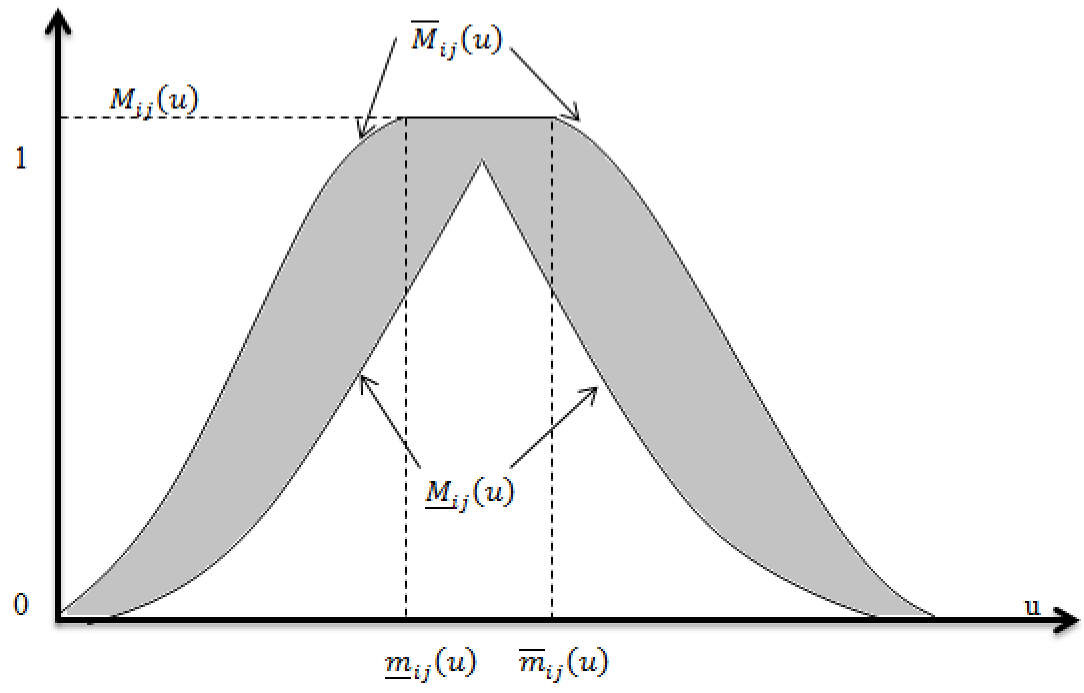


Figure 5:*A type-2 interval Gaussian membership function with uncertain mean. Shaded area is* ***footprint of uncertainty****(****FOU****)*[36]

A general type-2 fuzzy set has non-uniform secondary membership functions. Its overall membership function is three-dimensional and looks like a mountain range that sits atop its FOU. An interval type-2 fuzzy set has uniform secondary membership functions, and because the secondary grades are equal over the entire FOU they're set equal to 1 without any loss of generality. The membership function of an interval type-2 fuzzy set is also three dimensional but it looks like a plateau that sits atop its FOU. Because the third dimension of an interval type-2 fuzzy set conveys no useful new information, its FOU is a complete description for it. Yes, there is new terminology for type-2 fuzzy sets. We've already used some of it—FOU, LMF, UMF, secondary grade, and secondary membership function. The main variable of interest—for example, pressure, temperature—is called the primary variable. Its domain of possible membership values is called the secondary variable. At a specific value of x, say x' the entire domain of possible membership values is called the primary membership —$J_{x'}$. The insert shows that the secondary membership function sits atop the primary membership. The secondary membership function is also called a vertical slice. Finally, any type-1 fuzzy set that lies within the FOU is called an embedded fuzzy set. You probably should spend a few minutes reviewing this slide and the last two slides so that you'll remember the new terminology, because we'll be using lots of it in the rest of this module.

3.5 Interval Type-2 Fuzzy Set

Interval type-2 fuzzy sets are to-date the only kind of type-2 fuzzy sets that are actually being used in applications. Why is that? To most engineers and scientists, the phrase "To use" means the ability to compute quickly. As we know that this can be done for the interval type-2 fuzzy sets, it can't be done for general type-2 fuzzy sets [37]. It's the non-uniform secondary membership functions that make computations for general type-2 fuzzy sets very difficult. Fortunately, most of the calculations for interval type-2 fuzzy sets use interval arithmetic, which is very easy to use. Also, software is available for interval type-2 computations, so you don't have to re-invent the wheel. Finally, interval type-2 fuzzy sets have already been used in a large number of applications, so by reading articles about those applications you can get lots of helpful guidance. Just so you're not left with an incorrect impression that the only kind of FOU that people use is the triangular one we've already used, here are some other FOUs. Just as people use triangle, trapezoidal, Gaussian, bell, or piecewise linear membership functions for type-1fuzzy sets, these same kinds of functions can be used to model the lower and upper membership functions of an FOU. In the top figure on this slide there are so-called shoulder and interior FOUs. Notice that none of the lines on an FOU have to be parallel, although they could be. In the bottom figure on this slide is a Gaussian FOU, one that is not even symmetrical. To the left or right of *m* there may be a different standard deviation for both the LMF and UMF [38]. The parameters that describe the LMF and UMF of an FOU completely specify it. Numerical values for those parameters can be pre-specified by you or they can be optimized during the design of a fuzzy logic system by using application-data. Lots more FOUs are possible then the ones shown on this slide.

3.6 Rules

Rules are the heart of an FLS. They can be provided by experts, or extracted from data, or both. Each rule is an IF-THEN statement, an example of which is on this slide—IF pressure is low and temperature is high, THEN turn the valve a bit to the right. The IF part of a rule is called the antecedent. The THEN part of a rule is called the consequent. In general, each rule can have more than one antecedent but only one consequent. If a rule does have more than one consequent it can always be written as a collection of rules, each with the same antecedent but with only one consequent. In general, there will be *m* rules. However, an input to the FLS will only activate a small subset of these rules, usually much smaller than *m*. This is because each rule is associated with only a subset of its antecedent's domains, and an input to the FLS must reside at some point in those subset-domains [39].

A **fuzzy rule** can be defined as a "conditional statement" of the form:

IF *x is A* **THEN** *y is B*

Here, x and y are variables; A and B are linguistic values determined by fuzzy sets on the universe of discourse X and Y, respectively.

3.7 Output Processing

Recall that the Output Processing block of the interval type-2 FLS contains two subsystems—type-reducer and defuzzifier. The type-reducer transforms an interval type-2 fuzzy set into a type-1 fuzzy set, called the type-reduced set, through a process that is called type reduction. The type-reduced set is an interval-valued set. This set is completely described by it two end-points shown on this slide as *yl* and *yr*. These end-points depend on the inputs to the FLS, which is why they are shown as explicit functions of x. The defuzzifier maps the type reduced set into a crisp number, $y(x)$. It does this by computing the average value—the midpoint—of the type-reduced set. Type-reduction is a totally new concept to a FLS [40].

CHAPTER 4

PROPOSED METHOD

4.1 Structure of Impulse Noise Detector

The structure of the proposed impulse noise removal operator is as follows. The operator is constructed by combining four type-2 fuzzy logic filters, four defuzzifiers and a postprocessor. The operator processes the noisy pixels contained in its filtering window shown in figure and outputs the restored value of the centre pixel.

X(r-1,c-1)	X(r-1,c)	X(r-1,c+1)
X(r,c-1)	X(r,c)	X(r,c+1)
X(r+1,c-1)	X(r+1,c)	X(r+1,c+1)

Figure 6: *Filtering window*

All fuzzy logic filters employed in the structure of the operator are identical to each other and function as sub filters processing the horizontal, vertical, diagonal and the reverse diagonal pixel neighbour hoods in the filtering window, respectively. Hence, each of the four fuzzy logic filters accepts the centre pixel and two of its appropriate neighbouring pixels as input and then produces an output, which is a type-I interval fuzzy set representing the uncertainty interval for the restored value of the centre pixel. The four output fuzzy sets coming from the four NF filters are then fed to the corresponding defuzzifier blocks. Each defuzzifier defuzzifies the input fuzzy set and converts it into a single scalar value. The four scalar values obtained at the outputs of the four defuzzifiers represent four candidates for the restored value of the centre pixel of the filtering window. These four candidate values are finally evaluated by the postprocessor and converted into a single output value.

The output of the postprocessor is also the output of the proposed filtering operator and represents the restored value of the centre pixel of the filtering window. Each fuzzy logic filter employed in the structure of the proposed impulse noise removal operator is a type-2interval fuzzy inference system with 3-inputs and 1-output.

The internal structures of the fuzzy logic filters are identical to each other. The input-output relationship of any of the fuzzy logic filtersis as follows:

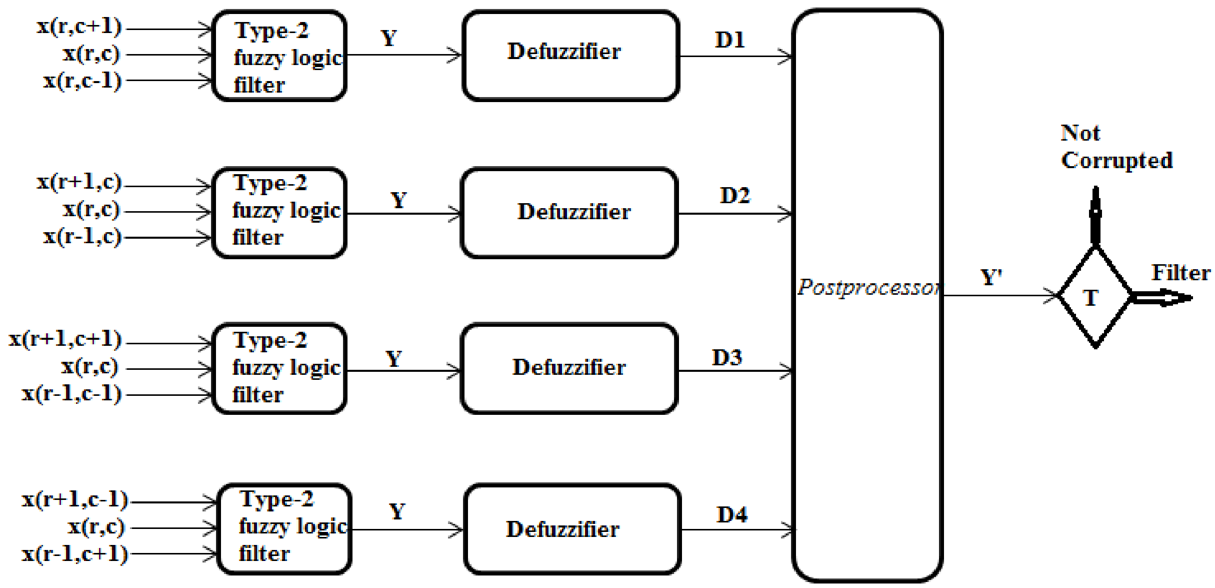


Figure 7: *Proposed impulse noise removal operator*

4.2 The Subdetectors

Let X1, X2, X3 denote the inputs of the fuzzy logic filter and Y denote its output. Each combination of inputs and their associated membership functions is represented by a rule in the rule base of the fuzzy logic filter. The rule base contains a desired number of fuzzy rules, which are as follows:

1. if$(X_1 \in M_{11})$ and $(X_2 \in M_{12})$ and $(X_3 \in M_{13})$, then (1)

$R_1 = k_{11}X_1 + k_{12}X_2 + k_{13}X_3 + k_{14}$

2. if$(X_1 \in M_{21})$ and $(X_2 \in M_{22})$ and $(X_3 \in M_{23})$, then

$R_2 = k_{21}X_1 + k_{22}X_2 + k_{23}X_3 + k_{24}$

3. if$(X_1 \in M_{31})$ and $(X_2 \in M_{32})$ and $(X_3 \in M_{33})$, then

$R_3 = k_{31}X_1 + k_{32}X_2 + k_{33}X_3 + k_{34}$

.

.

.

I. if ($X_1 \in M_{I1}$) and ($X_2 \in M_{I2}$) and ($X_3 \in M_{I3}$), then

$R_I = k_{I1}X_1 + k_{I2}X_2 + k_{I3}X_3 + k_{I4}$

.

.

.

N. if ($X_1 \in M_{N1}$) and ($X_2 \in M_{N2}$) and ($X_3 \in M_{N3}$),

$R_N = k_{N1}X_1 + k_{N2}X_2 + k_{N3}X_3 + k_{N4}$

where N is the number of fuzzy rules in the rule base, M denotes the I_{th} membership function of the J_{th}input andR_I denotes the output of the ith rule. The input membershipfunctions are type-2 interval Gaussian membership functionswith uncertain mean:

$$M_{ij}(u) = \exp\left[-\frac{1}{2}\left(\frac{u-m_{ij}}{\sigma_{ij}}\right)^2\right] \tag{2}$$

with i =1,2,…N and j=1,2,3. Here, the parameters$m_{ij}$and r_{ij} are the mean and the standard deviation of thetype-2 interval Gaussian membership function M_{ij} respectively, and the interval $[\underline{m}_{ij}, \overline{m}_{ij}]$ denote the lower and theupper bounds of the uncertainty in the mean. Since the membership functions M_{ij} are interval membershipfunctions, the boundaries of their FOU are characterizedby their upper and lower membership functions.

The output of the fuzzy logic filter is the weighted average of the individual rule outputs:

$$Y = \frac{\sum_{i=1}^{N} w_i R_i}{\sum_{i=1}^{N} w_i} \tag{3}$$

The weighting factor, $w_i$of thei_{th}rule is calculated byevaluating the membership expressions in the antecedent ofthe rule. This is accomplished by first converting the input values to fuzzy membership values by utilizing the input membership functions M_{ij} and then applying the "and"operator to these membership values. The "and" operatorcorresponds to the multiplication of the input membershipvalues:

$$W_i = M_{i1}(X_1).M_{i2}(X_2).M_{i3}(X_3) \tag{4}$$

Since the membership functions M_{ij} in the antecedentof the i_{th} rule are type-2 interval membership functions, the weighting factor w_i is a type-I interval set, i.e. $w_i = [\underline{w_i}, \overline{w_i}]$ whose lower and upper boundaries are determined by using the lower and the upper membership functions.

Therefore, there are 13 parameters determining the output of the i_{th} rule. Since the total number of rules in the rulebaseis N, then the total number of parameters in the rule base is 13N. Each fuzzy logic filter in the proposed operator has 30 rules and 390 parameters. The optimal values of these parameters are determined by training by using the least squares optimization algorithm. Once the training is completed, there is no need for further training.

After the weighting factors are obtained, the output Y of the fuzzy logic filter can be found by calculating the weighted average of the individual rule outputs. The output Y is also a type-I interval set, i.e. Y = $[\underline{Y}, \overline{Y}]$, since the$w_i$'s in the above equation are type-I interval sets andR_i's are scalars. The lower and the upper boundaries of Yare determined by using the iterative procedure proposed by[41].The information presented in this subsection is related with the input-output relationship of a type-2 interval fuzzy logic system with 3-inputs and 1-output.Readers interested in details of type-2 fuzzy inference systems as well as other type-2 fuzzy logic systems are referred to an excellent book on this subject [42].

4.3 Defuzzifier

The defuzzifier block takes the type-I interval fuzzy set obtained at output of the corresponding NF filter as input and converts it into a scalar value by performing centroid defuzzification [43]. Since the input set is a type-I interval fuzzy set, i.e. Y = $[\underline{Y}, \overline{Y}]$, its centroid is equal to the centre of the interval:

$$D = \frac{[\underline{Y} + \overline{Y}]}{2} \tag{5}$$

The postprocessor generates the final output of the proposed operator. It processes the four scalar values obtained at the outputs of the four defuzzifiers and produces a single scalar output, which represents the output of the proposed filter. The operation of the postprocessor may be described as follows ---

Let D1, D2, D3, D4 denote the outputs of the four defuzzifiers. The postprocessor first sorts these values, such that D’1 < D’2< D’3< D’4, where D’1, D’2, D’3, D’4 represent the output values of the defuzzifiers after sorting. Next, the lowest (D) and the highest (D) of the four values are discarded. Finally, the remaining two are averaged to obtain the postprocessor output, which is also the output of the proposed operator:

$$Y' = \frac{D'2+D'3}{2} \tag{6}$$

4.4 Postprocessor (Threshold)

The two scalar values obtained at the outputs of the two defuzzifiers are given to the inputs of the postprocessor, which converts them into a single scalar output representing the output of the impulse detector [44]. The postprocessor actually calculates the average value of the two defuzzifier outputs and then suitably maps this value to either 0 or 1 by converting it with a threshold corresponding to the half of the dynamic luminance range. Assuming that *dV* and *dH* denote the outputs of the defuzzifiers in the structure of the proposed detector, the input-output relationship of the postprocessor.

4.5 Training of Subdetectors

The internal parameters of the proposed impulse detection operator are optimized by training. The training of the proposed operator is accomplished by training the individual type-2 subdetectors in its structure. Each subdetector in the structure is trained individually and independently of the other.

The parameters of the subdetector under training are iteratively adjusted in such a manner that its output converges to the output of the ideal impulse detector. The ideal impulse detector is a conceptual operator representing the relationship between the input and the target training images. It does not necessarily exist in reality. It is only the output of the ideal impulse detector that is necessary for training and this is represented by a suitably designed target image.

4.6 Ideal Case

An image is obtained from the difference between the original training image and the noisy training image. For this purpose, the noisy training image is subtracted from the original training image. The pixels which are the same in the two images (the uncorrupted pixels) convert to zero values while the pixels which are different (the pixels corrupted due to noise) in the two images convert to nonzero values. The noise detection image is computed by replacing the zeros with black pixels and the non-zeros with white pixels. Therefore, locations of the white pixels in this image indicate the locations of the noisy pixels that need filtering while the locations of the black pixels indicate the uncorrupted pixels that need to be left unfiltered.

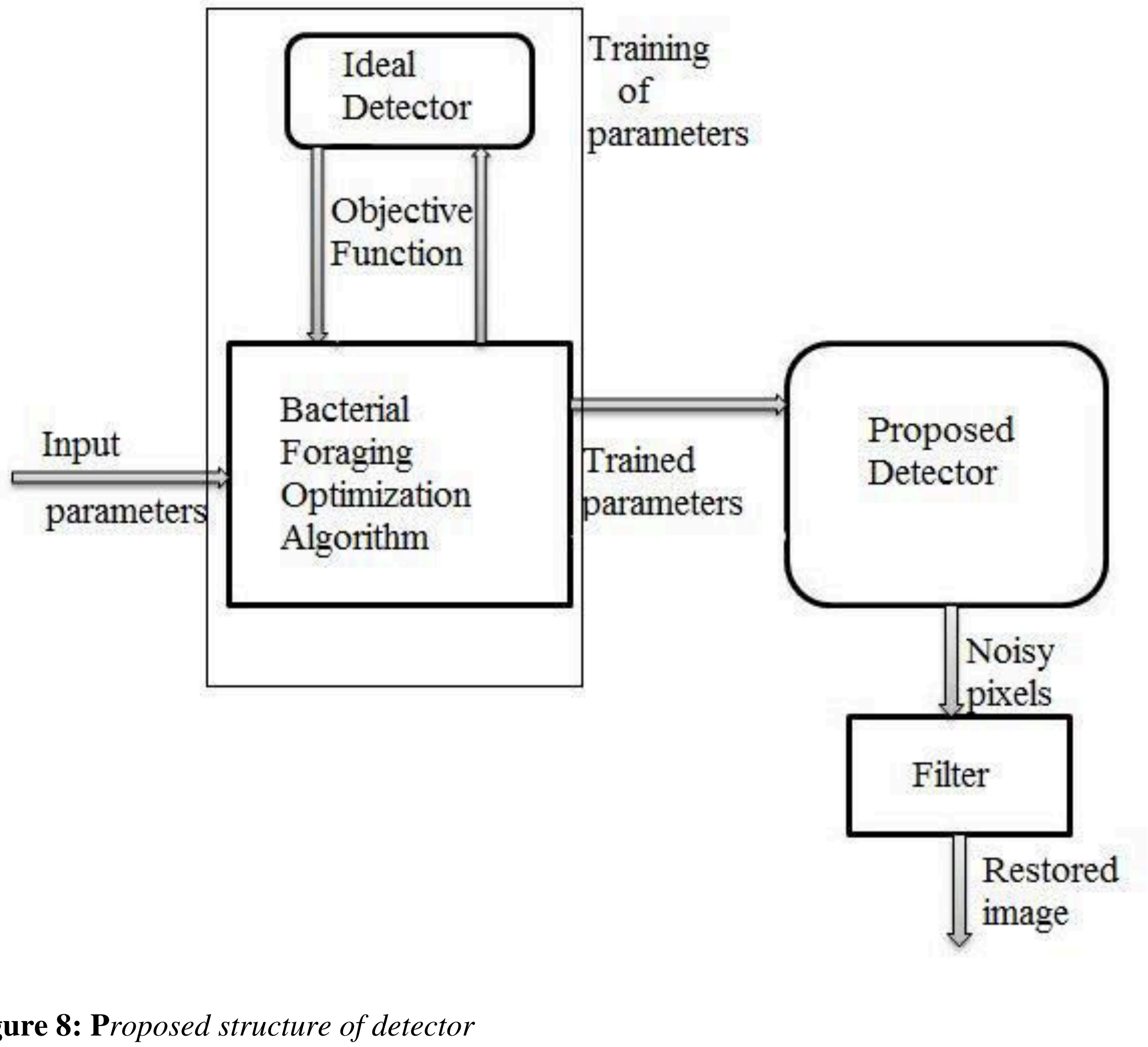


Figure 8: P*roposed structure of detector*

4.7 Bacterial Foraging Optimization Algorithm

The parameters of the proposed structure are optimized by using bacterial foraging optimization algorithm. The parameters refer to the different values of mean and standard deviation used in different rules of the fuzzy inference system. These parameters act as location of the bacteria's. One bacterium is assigned for each set of mean and standard deviation value pair. Initially the bacteria's are randomly initialized.

The objective function is defined by the difference in the number of pixels modified by the system and the actual number of pixels to be modified as per defined by the ideal case. The aim of the algorithm is to move bacteria's in a direction such that this difference tends to minimize. After certain specified number of iteration, the algorithm stops and output is the optimized vales for all sets of mean and standard deviation pairs.

In the proposed work, there are four fuzzy logic filters used. In each fuzzy logic filter, we can use any number of rules to create rule base. Increasing the number of rules used is chosen on the basis of performance yield by them. This number can be later increased to enhance the performance of the operator. However, increasing the number of rules adds to computational complexity. For best results, it is important to choose appropriate number of rules to balance between computational complexity and better performance. In the presented work, four rules have been used. And as we know, since it's a Sugeno model, each rule employs three membership functions. Hence for each filter, number of parameters are (3×4=) 12. And thus for four filters, number of parameters are (4×12=) 48. One bacterium is assigned each parameter. Hence, a total of 48 bacteria's are used.

Also, bacterial foraging optimization algorithm can be used in two ways. One is, run the three iterative steps until you get best solution and second is, choose an appropriate number of steps till when the loops will run. In the presented work, second way is used. The number of iterations has been limited to 35 for each step; chemotaxis, reproduction, elimination and dispersion.

The mean and standard deviation are stored in array thetai such that first dimension stores mean and second dimension stores standard deviation. These are initially randomly initialized and then these random values are optimized by bacterial foraging optiomization technique.

Θ (:, 1) = rand (S, 1) ; (7)
Θ (:, 2) = rand (S, 1) ;

In chemotaxis, the bacteria move in different directions to search for better values of Θ. It can be represented by, for i^{th} bacterium,

Θ (i, 1) = Θ (i, 1) + α (i) ; (8)

Θ (i, 2) = Θ (i, 2) + β (i) ;

Here, α (i) and β (i) refers to the size of the step taken by the bacteria in random direction. It can be easily specified by:

α (i) = randi ([-1 1], 1) /20 ; (9)
β (i) = randi ([-1 1], 1) /20 ;

The step size is taken between 0 to 1 which can be positive or negative which specifies the direction of movement.

The cost function can be minimized or maximized. Used bacterial foraging optimization algorithm minimizes the cost function to give best results. The ideal detector described earlier gives ideal result and hence can be used as standard. The impulse detector can be used to check how the BF algorithm affects the final output. Hence, the difference between ideal detector and output of impulse noise detector has been used as cost function. The aim is to minimize the value of cost function from iteration to iteration of bacterial foraging algorithm so that after specified number of iterations, we get good results. It can be expressed as an equation:

$$E = (J_{ideal} - J_{detector}) \tag{10}$$

Where J_{ideal} refers to the output of ideal detector and $J_{detector}$ refers to the output of impulse noise detector. The difference is stored in E, which is minimized iteration by iteration.

Chapter 5

EXPERIMENTAL RESULTS

5.1 Results for image restoration using Fuzzy Inference Detector

The type-2 fuzzy logic based impulse detector described in the previous section is implemented and the enhancements that it provides to the performance of a noise filter are evaluated by conducting a number of filtering experiments. The experiments are carefully designed so that the behaviour of the proposed impulse detector for different test images can be easily observed. Four popular test images from the literature are included in the filtering experiments. These are the *Baboon, House, Lena* and *Peppers* images, which are shown in Figure 8. These images are chosen to include different image properties in the experiments. All of these images are 24-bit colour images and they have the same size of 256-by-256 pixels.

The noisy test images used in the experiments are obtained by contaminating a given test image with an impulse noise of given noise density. The noise density being considered in the experiment is 30% representing an average realistic noise density. The performance enhancements obtained by using the proposed impulse detector with a noise filter is evaluated by using it with median impulse noise filter.

The median impulse noise filter operates on a minimal filtering window, which has a size of 3-by-3 pixels. The improvement contributed by the proposed detector to the performance of the noise filter is measured by using the *mean squared error (MSE)* criterion, defined as

$$\mathrm{MSE} = \frac{1}{LC}\sum_{l=1}^{L}\sum_{c=1}^{C}(s[l,c]-y[l,c])^2 \qquad (11)$$

where $s[l, c]$ and $y[l, c]$ represent the luminance value of the pixel at line l and column c of one of the three colour bands of the original and the restored versions of a corrupted test image respectively.

It should be noted that this definition of MSE is valid for gray level images only. Since the test images used in the filtering experiments reported in this section are colour images, this

MSE computation is performed for three times, one for each of the three colour bands (red, green and blue), and the three resulting MSE values are then averaged to obtain the representative MSE value for that image.

From a different point of view, a pixel in a colour image maybe considered as a point in the three dimensional R-G-B colour space. The contamination of this pixel by noise implies a change in at least one of the three colour components (red, green or blue) of this pixel, which corresponds to a shift of the location of the point that represents this pixel in the R-G-B colour space. Hence, the filtering operation performed on this pixel by a noise filter may be thought of as an attempt to move the point representing this pixel back to its original location in the R-G-B colour space.

(a)

(b)

(c)

(d)

Figure 9: *Original image a) lena b) baboon c) house d) peppers*

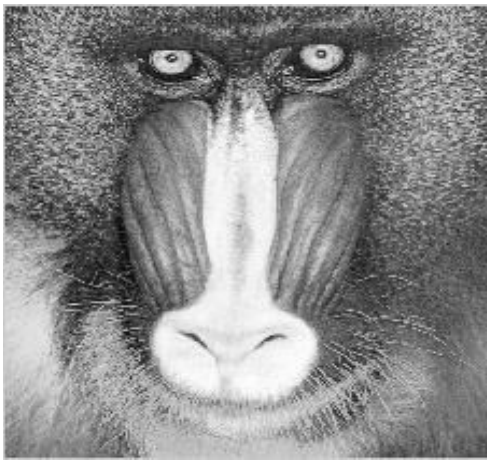
(a)

(b)

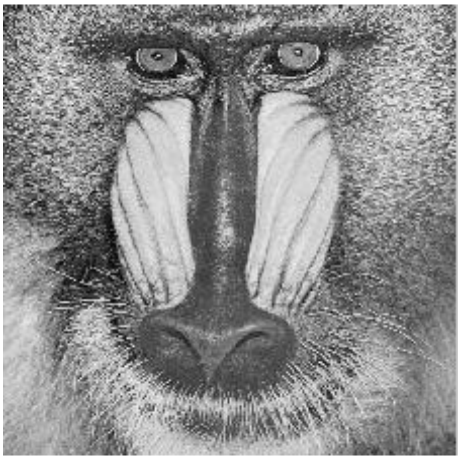
(c)

(d)

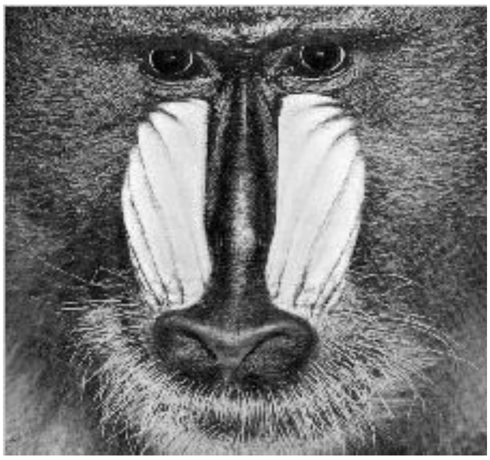
Figure 10: *Original lena image a) Red component b) Green component c) Blue component d) Gray image*



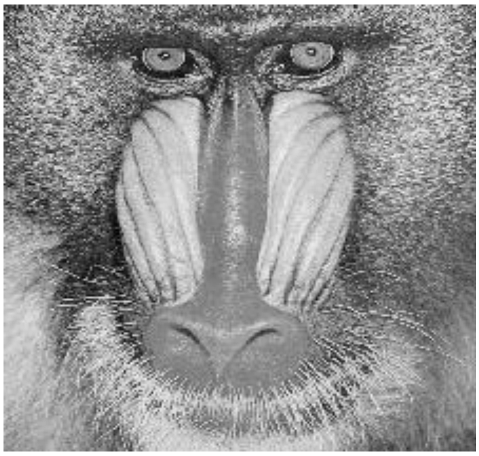
(a)



(b)



(c)



(d)

Figure 11: *Original baboon image a) Red component b) Green component c) Blue component d) Gray image*

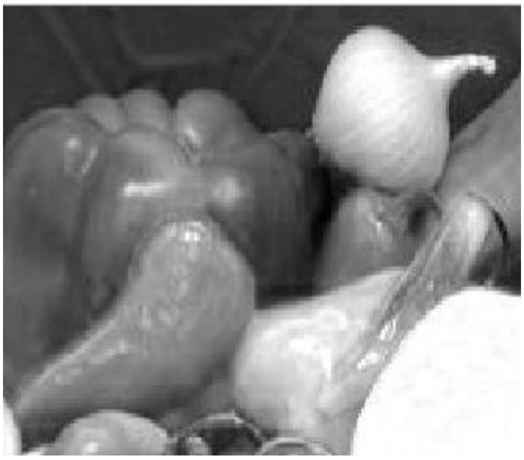
(a)

(b)

(c)

(d)

Figure 12: *Original house image a) Red component b) Green component c) Blue component d) Gray image*



(a)

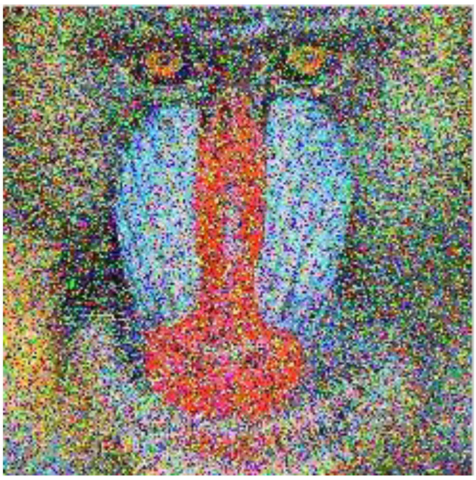
(b)

(c)

(d)

Figure 13: *Original peppers image a) Red component b) Green component c) Blue component d) Gray image*

(a)

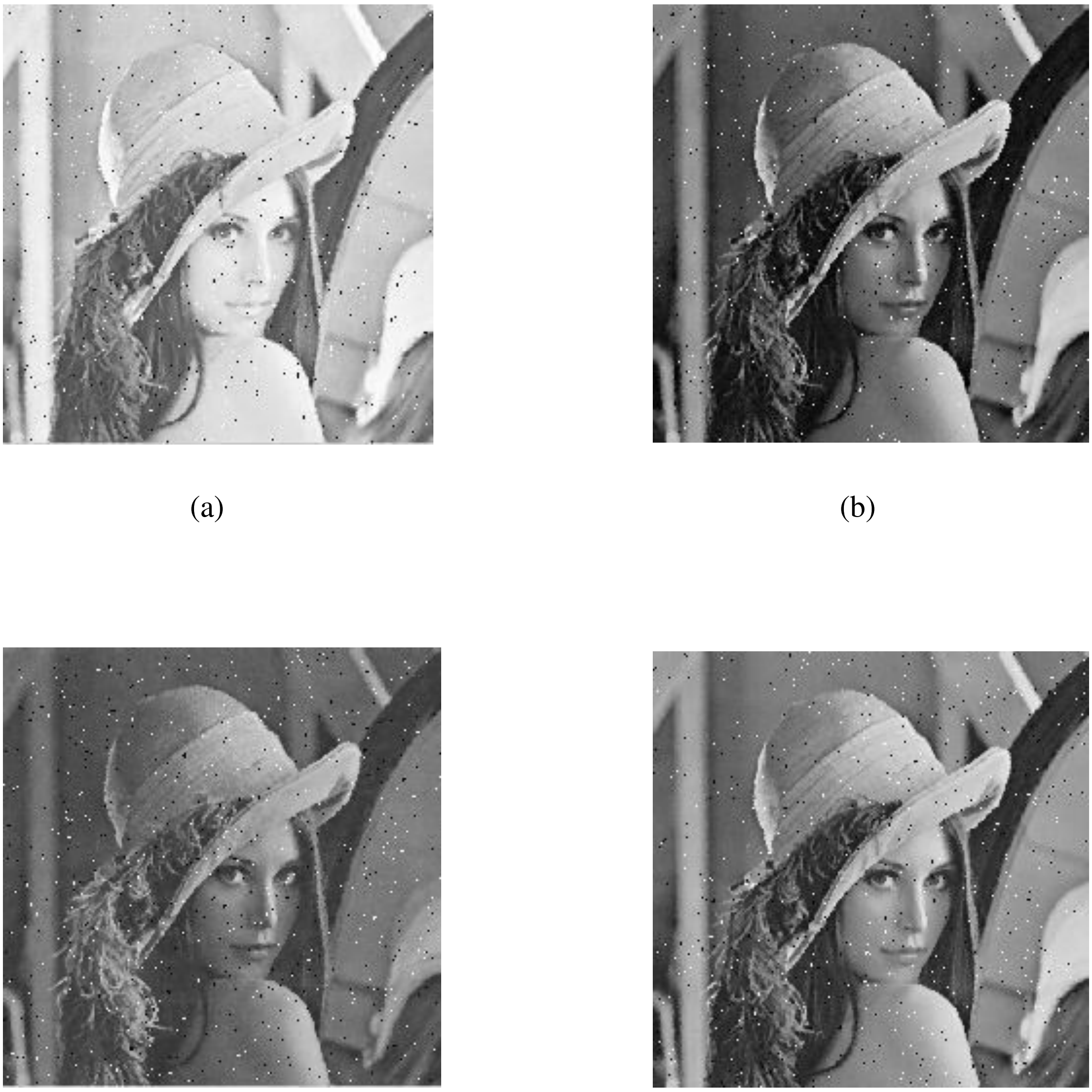


(b)

(c)

(d)

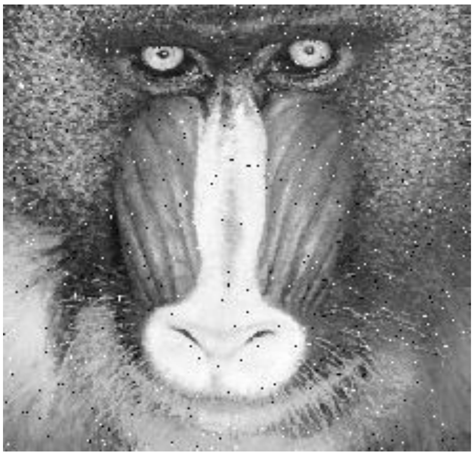
Figure 14: *Noisy images a) lena b) baboon c) house d) peppers*



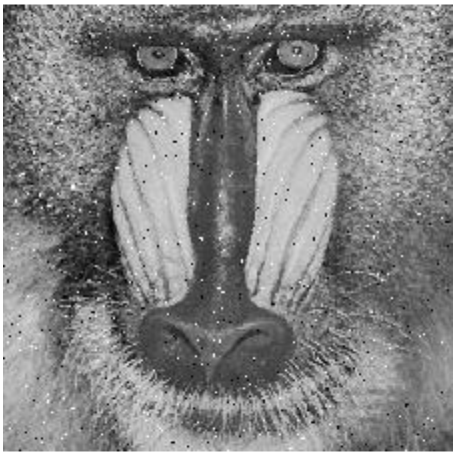
(a) (b)

(c) (d)

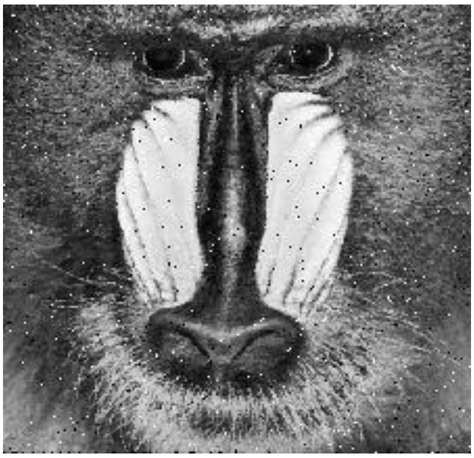
Figure 15: *Median Filterlena image a) Red component b) Green component c) Blue component d) Gray image*



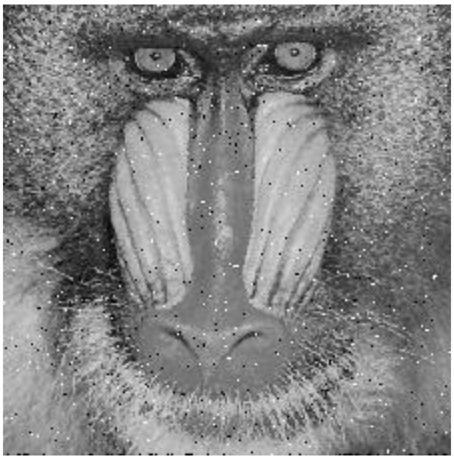
(a)



(b)



(c)



(d)

Figure 16: *Median Filter baboon image a) Red component b) Green component c) Blue component d) Gray image*

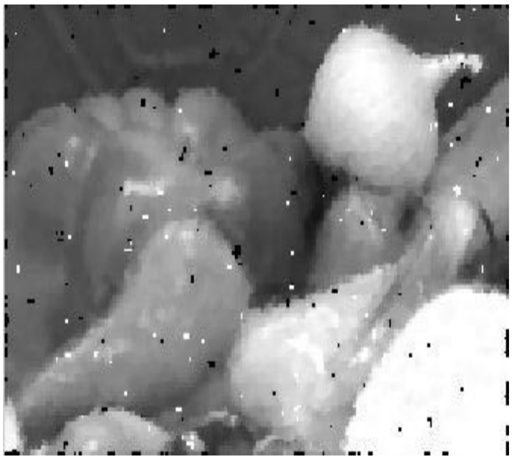
(a)

(b)

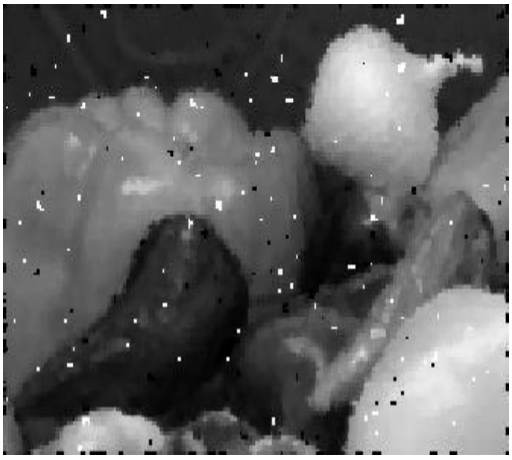
(c)

(d)

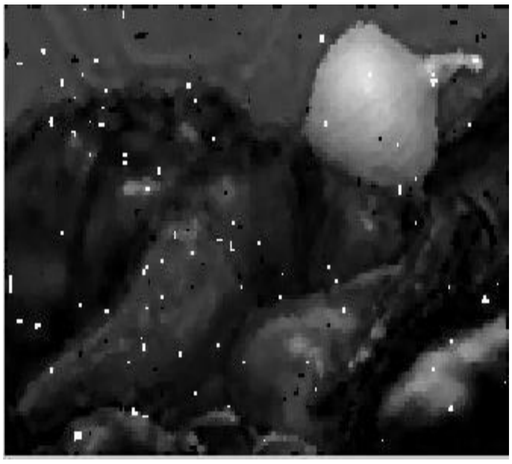
Figure 17: *Median Filter house image a) Red component b) Green component c) Blue component d) Gray image*



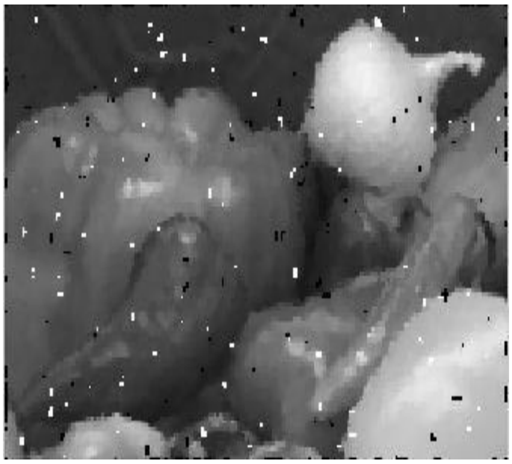
(a)



(b)



(c)



(d)

Figure 18: *Median Filter peppers image a) Red component b) Green component c) Blue component d) Gray image*

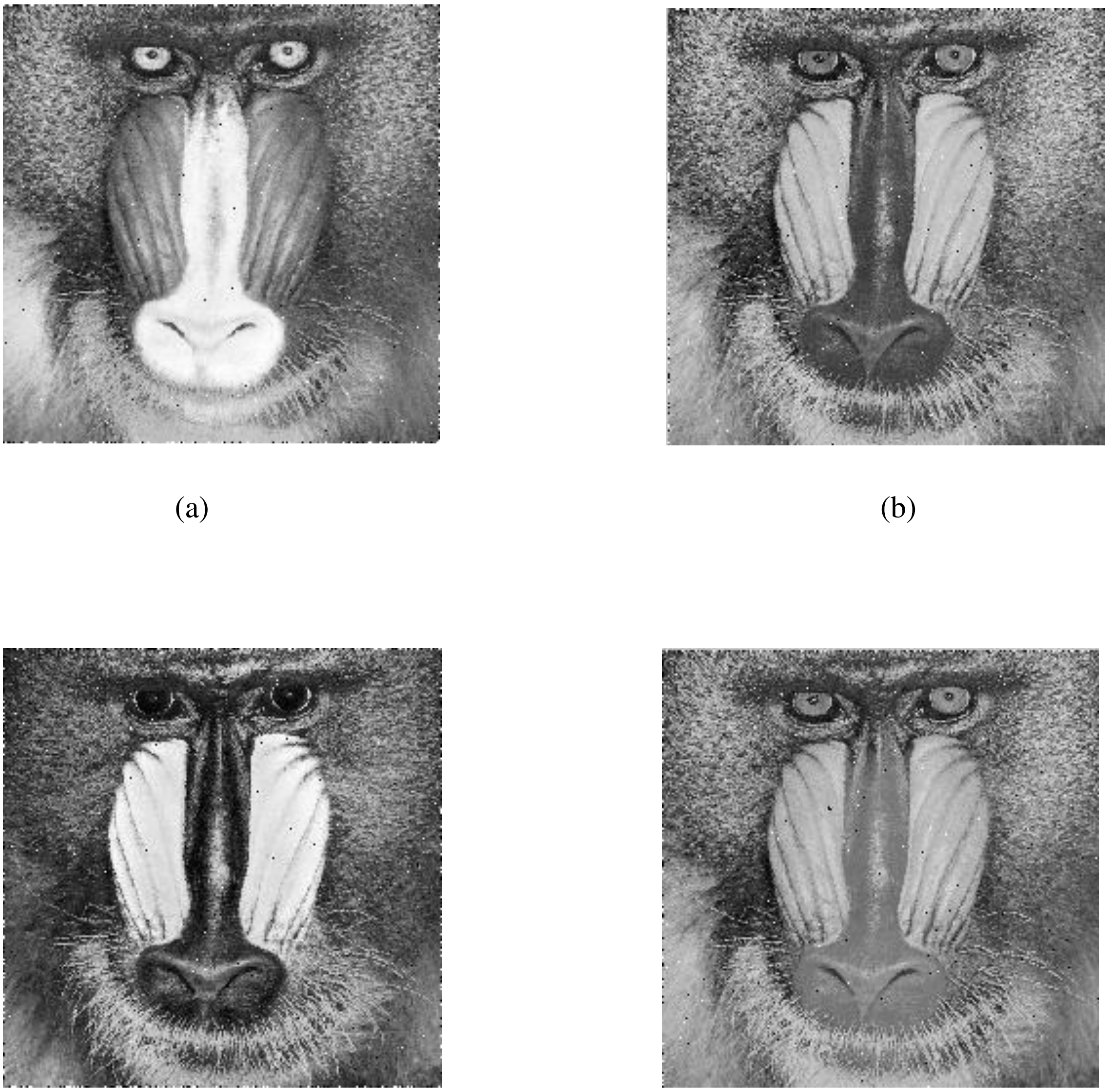
(a)

(b)

(c)

(d)

Figure 19: *Fuzzy Logic Filter lena image a) Red component b) Green component c) Blue component d) Gray image*



(a) (b) (c) (d)

Figure 20: *Fuzzy Logic Filter baboon image a) Red component b) Green component c) Blue component d) Gray image*

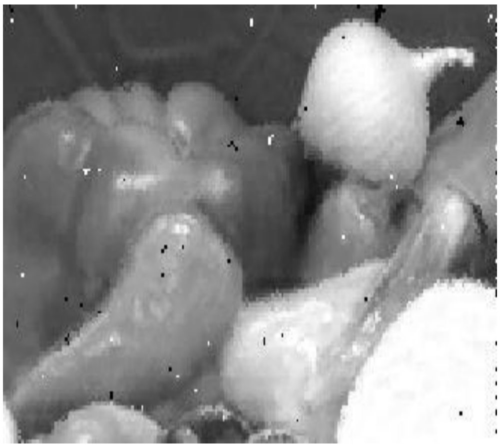
(a)

(b)

(c)

(d)

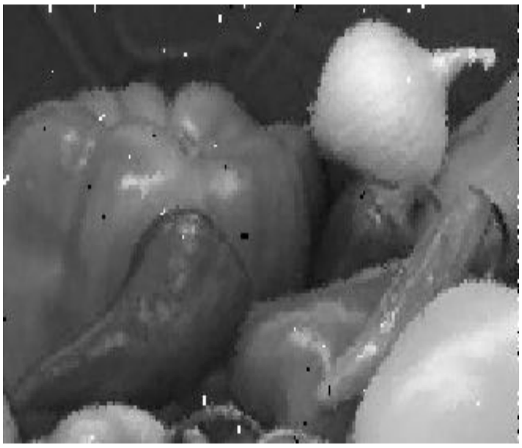
Figure 21: *Fuzzy Logic Filter house image a) Red component b) Green component c) Blue component d) Gray image*



(a)

(b)

(c)



(d)

Figure 22: *Fuzzy Logic Filter peppers image a) Red component b) Green component c) Blue component d) Gray image*

In order to determine the performance of the two filters, same noisy images are evaluated firstly without using the detector and then with detector. The results of the comparison of the mean squared error calculated without and with detector for different set of noisy images are given below:

Table 1: *Comparison of the MSE values calculated for the output images of the filters when used without and with fuzzy logic impulse detector.*

Image	***Lena***	***Baboon***	***House***	***Peppers***
Without Detector (Median Filtering)	279.67	779.88	198.37	307.00
With Fuzzy Logic Impulse Detector	144.03	308.63	107.34	151.91

The values for mean and standard deviation of the different Gaussian membership functions used to generate the rule set of the fuzzy inference system are randomly chosen. The results may vary with other sets of values of mean and standard deviation. Different set of values have been taken in the experiment, out of which the set of values giving best result is considered for the comparison with the standard median filter.

There are two values for each membership function, namely mean and standard deviation. Three membership functions are used to generate one rule. This gives ($3\times2 = 6$) parameters to generate one rule. We have considered rule base containing four rules only. Therefore for one class of pixels, ($6\times4 = 24$) parameters are required.

Now, we have four classes of pixels, namely horizontal, vertical, principal diagonal and other diagonal. Therefore, ($24\times4 = 76$) parameters are used in the detector.

The value of mean and standard deviation is between 0 and 1, inclusive. The parameters used in the experiment are:

Table 2: *Horizontal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$
$\boldsymbol{M_{i1}}$	0.31	0.90	0.70	0.67	0.24	0.56	0.50	0.50
$\boldsymbol{M_{i2}}$	0.24	0.56	0.50	0.90	0.70	0.67	0.45	0.67
$\boldsymbol{M_{i3}}$	0.40	0.80	0.35	0.68	0.15	0.95	0.45	0.45

Table 3: *Vertical class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$
$\boldsymbol{M_{i1}}$	0.34	0.90	0.25	0.97	0.74	0.57	0.52	0.55
$\boldsymbol{M_{i2}}$	0.21	0.56	0.72	0.70	0.90	0.61	0.43	0.63
$\boldsymbol{M_{i3}}$	0.70	0.80	0.59	0.18	0.35	0.96	0.45	0.45

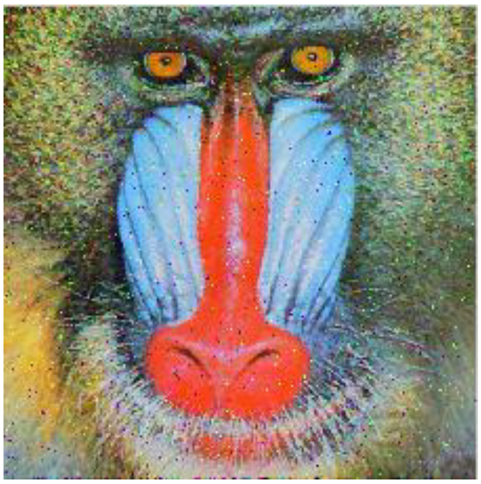
Table 4: *Principal Diagonal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\Sigma}$
$\boldsymbol{M_{i1}}$	0.56	0.90	0.70	0.67	0.14	0.56	0.70	0.53
$\boldsymbol{M_{i2}}$	0.90	0.56	0.70	0.50	0.40	0.67	0.65	0.67
$\boldsymbol{M_{i3}}$	0.56	0.80	0.35	0.48	0.65	0.95	0.55	0.46

Table 5: *Other diagonal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	$\boldsymbol{m}$	$\boldsymbol{\Sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$
$\boldsymbol{M_{i1}}$	0.37	0.60	0.72	0.67	0.25	0.56	0.52	0.50
$\boldsymbol{M_{i2}}$	0.28	0.56	0.58	0.50	0.73	0.67	0.49	0.61
$\boldsymbol{M_{i3}}$	0.42	0.50	0.32	0.68	0.17	0.95	0.43	0.40

(a)



(b)

(c)

(d)

Figure 23: *Median Filter a) lena b) baboon c) house d) peppers*

(a)

(b)

(c)

(d)

Figure 24: *Fuzzy Logic Filter a) lena b) baboon c) house d) peppers*

Based on these observations, a criterion is used termed as the *mean pixel distance* as an alternative performance measure that is better suited to colour images. The MPD criterion can mathematically be defined as follows:

$$\mathrm{MPD} = \frac{1}{LC}\sum_{l=1}^{L}\sum_{c=1}^{C}\|s[l,c] - y[l,c]\|^2 \qquad (12)$$

where **s**[l, c] and **y**[l, c] are vectorial quantities in the R-G-B colour space (i.e., **s**[l, c] = {sR[l, c], sG[l, c], sB[l, c]}) and represent the colour value of the pixel at line l and column c of the original and the restored versions of a corrupted colour test image respectively.

Here **s** [l, c],**y**[l, c] is the vectorial (Euclidian) distance between two points in the R-G-B colour space defined as:

$$\|s[l,c] - y[l,c]\| = \sqrt{\sum_{i\in\{R,G,B\}}\left(s_{i[l,c]} - y_{i[l,c]}\right)^2} \qquad (13)$$

In order to determine the performance of the two filters, same noisy images are evaluated firstly without using the detector and then with detector. The results of the comparison of the mean pixel distance calculated without and with detector for different set of noisy images are given below:

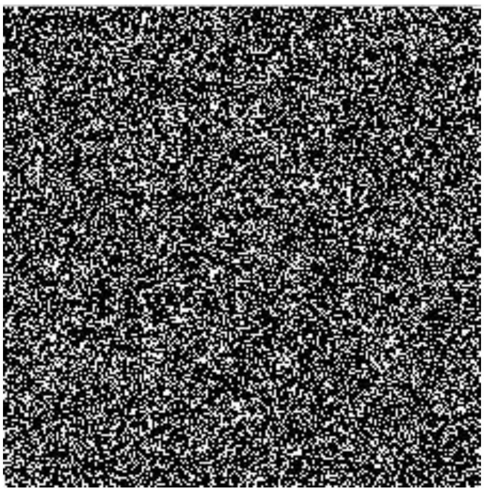
Table 6: *Comparison of the MPD values calculated for the output images of the filters when used without and with fuzzy logic impulse detector.*

Image	***Lena***	***Baboon***	***House***	***Peppers***
Without Detector (Median Filtering)	15.00	36.98	13.03	15.27
With Fuzzy Logic Impulse Detector	6.64	16.34	6.00	6.96

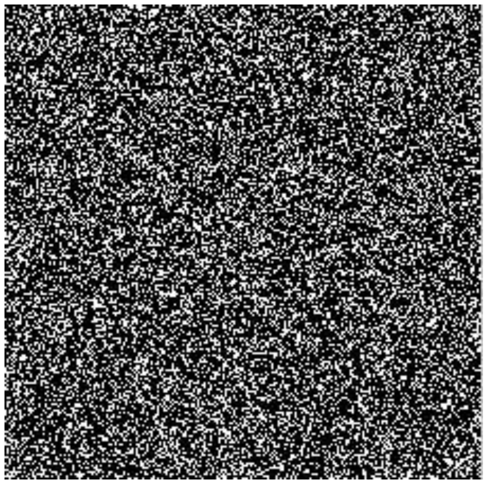
5.2 Results of image restoration using fuzzy logic detector with parameters optimized by bacterial foraging optimization algorithm

In each of the individual filtering experiments performed for a given noise filter / test image / noise density combination, the noise filter is combined with the detector and applied to the test image of that experiment corrupted with the noise density of that experiment. The performance of the noise filter is separately evaluated on the same noisy test image for the uses without and with the detector.

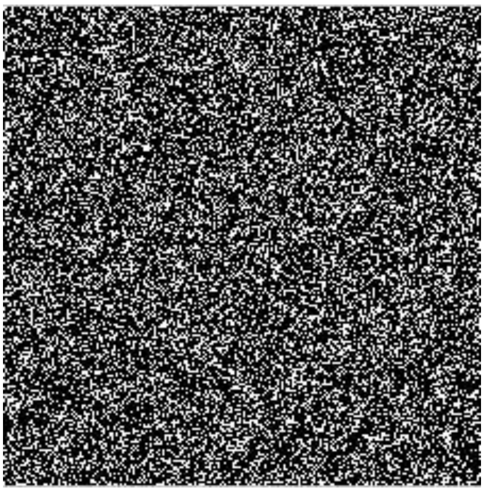
The same filtering procedure is followed in each of the individual filtering experiments. First, the noisy test image of the experiment is created as shown in the next figure. Then, the noisy test image is filtered by the noise filter. Next, the MSE and MPD values for the output image of the filter are calculated. Following this, the noisy test image is processed by using the proposed impulse detector. After that, the blur-reduced final output image is constructed from the selected pixels from the noisy input image and restored output image of the noise filter. The selection process is guided by the noise detector. Finally, the MSE and MPD values are calculated for the final output image for comparison with the previously calculated values.



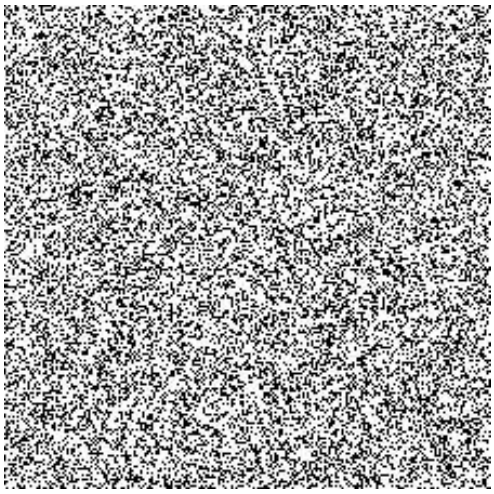
(a)



(b)

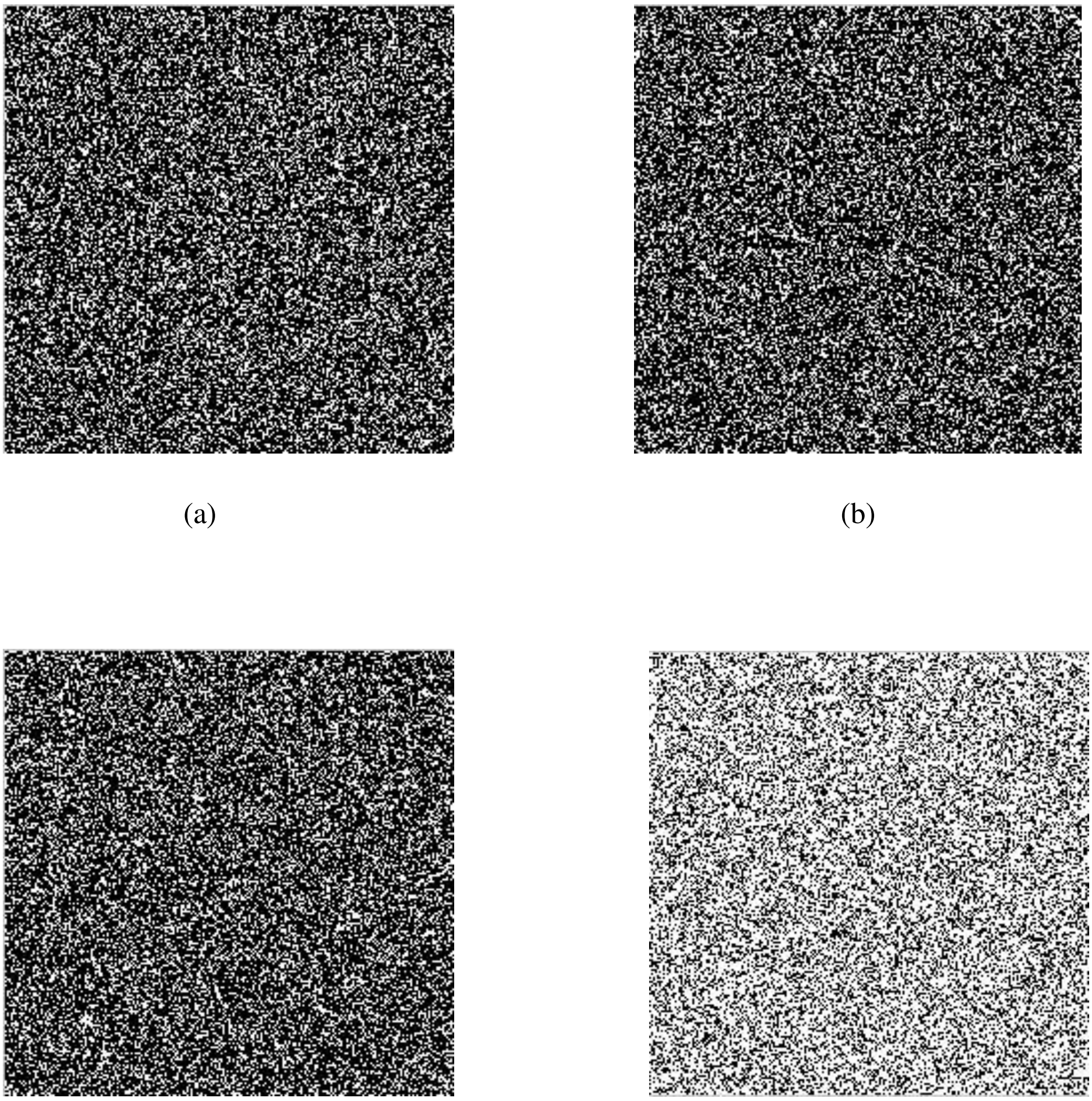


(c)



(d)

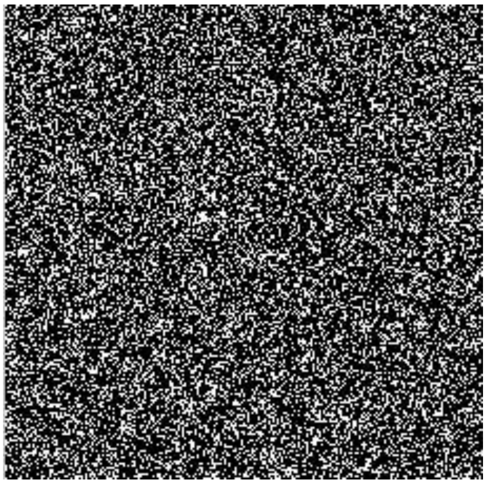
Figure 25: *Ideal Impulse Detector lena image a) Red component b) Green component c) Blue component d) Gray image*



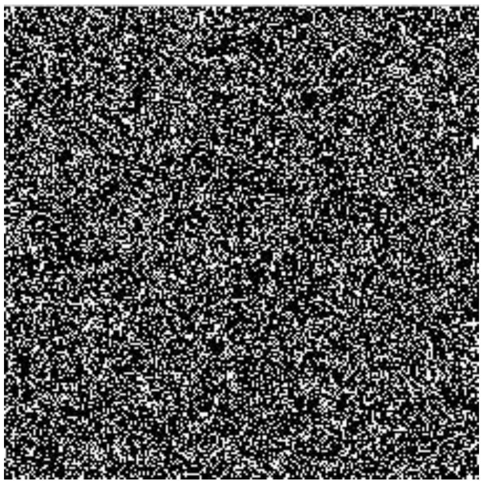
(a) (b)

(c) (d)

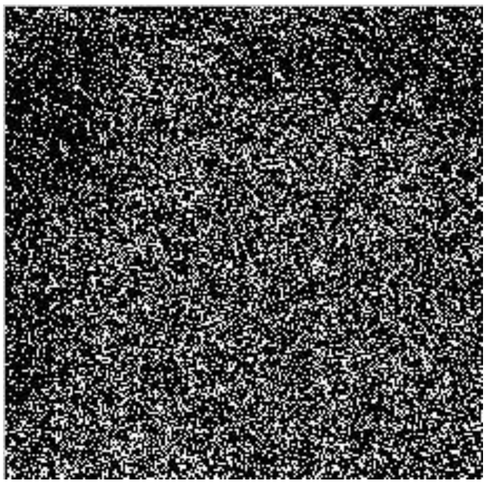
Figure 26: *Ideal Impulse Detector baboon image a) Red component b) Green component c) Blue component d) Gray image*



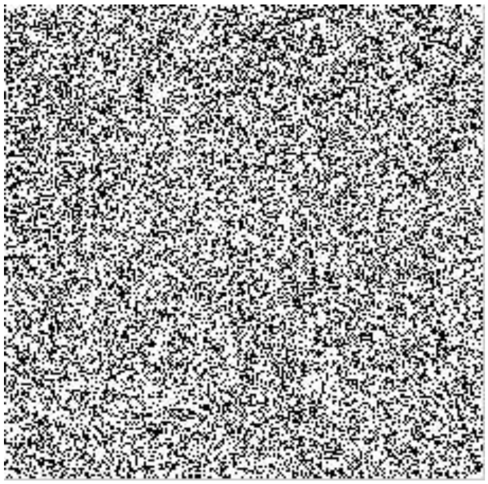
(a)



(b)

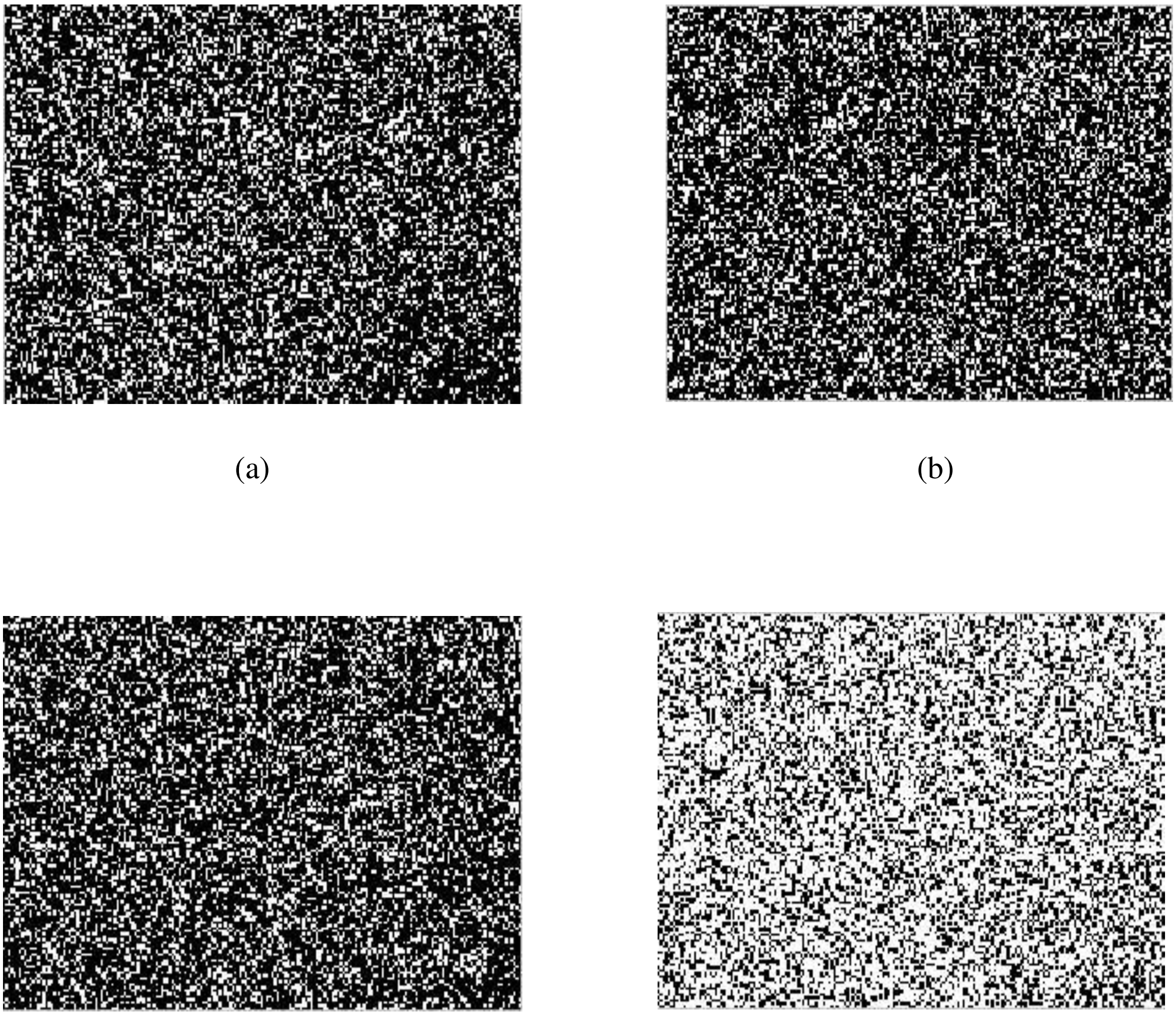


(c)



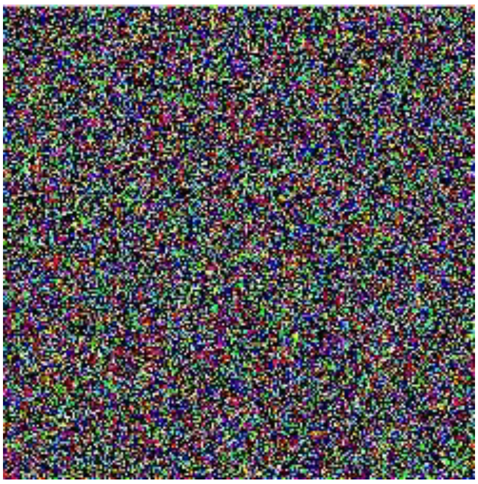
(d)

Figure 27: *Ideal Impulse Detector house image a) Red component b) Green component c) Blue component d) Gray image*

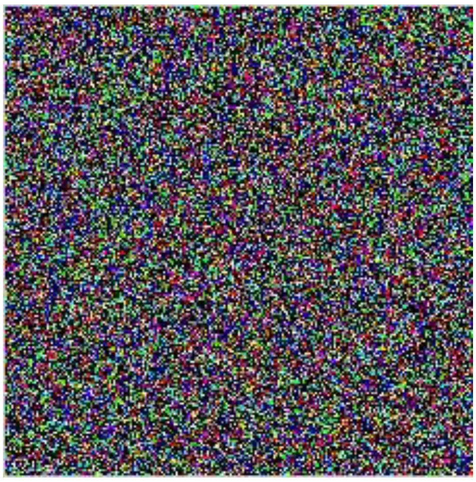


(a) (b) (c) (d)

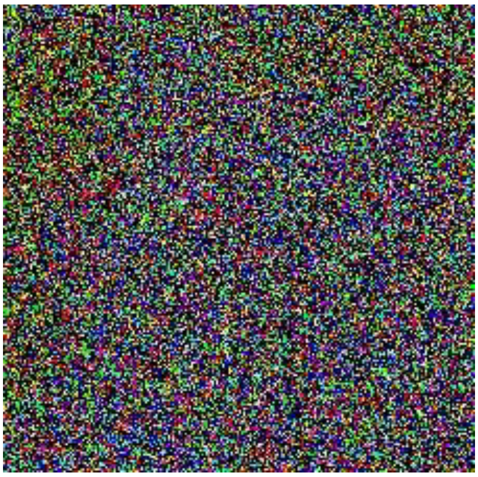
Figure 28: *Ideal Impulse Detector peppers image a) Red component b) Green component c) Blue component d) Gray image*



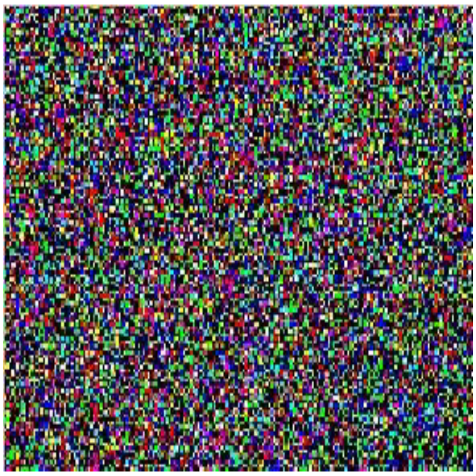
(a)



(b)



(c)



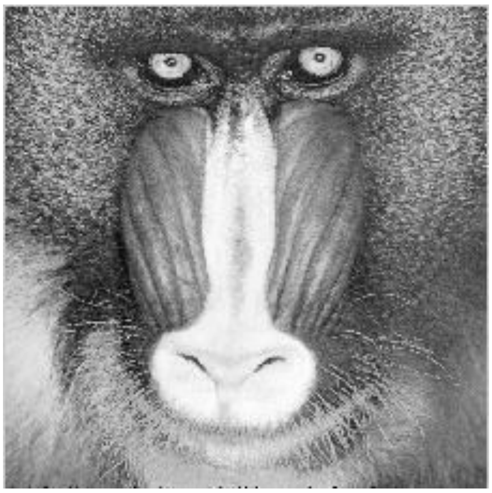
(d)

Figure 29: *Ideal Impulse Detector a) lena b) baboon c) house d) peppers*

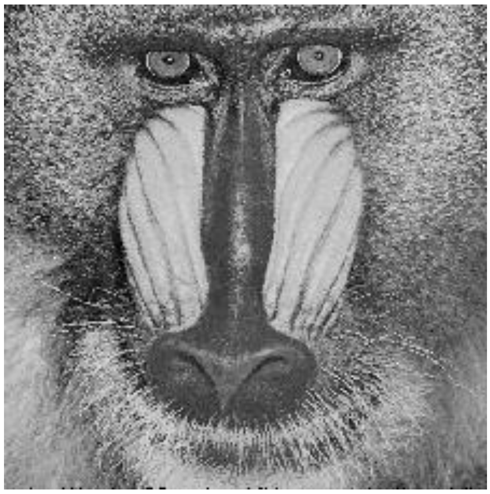
(a) (b)

(c) (d)

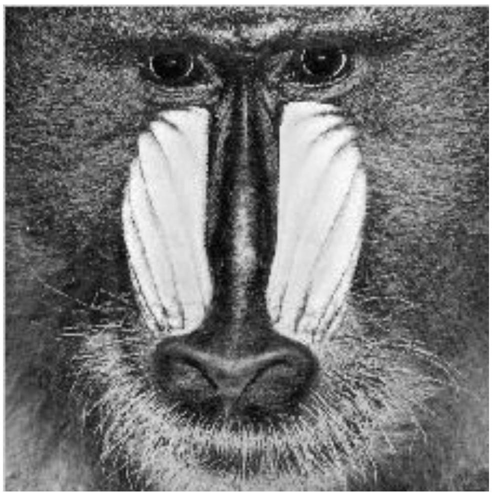
Figure 30: *Detector with BFOA lena image a) Red component b) Green component c) Blue component d) Gray image*



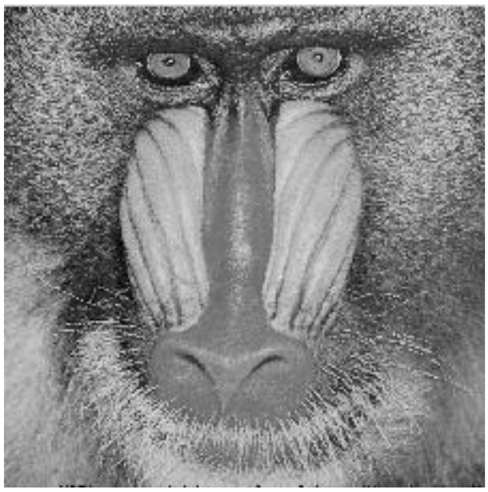
(a)



(b)

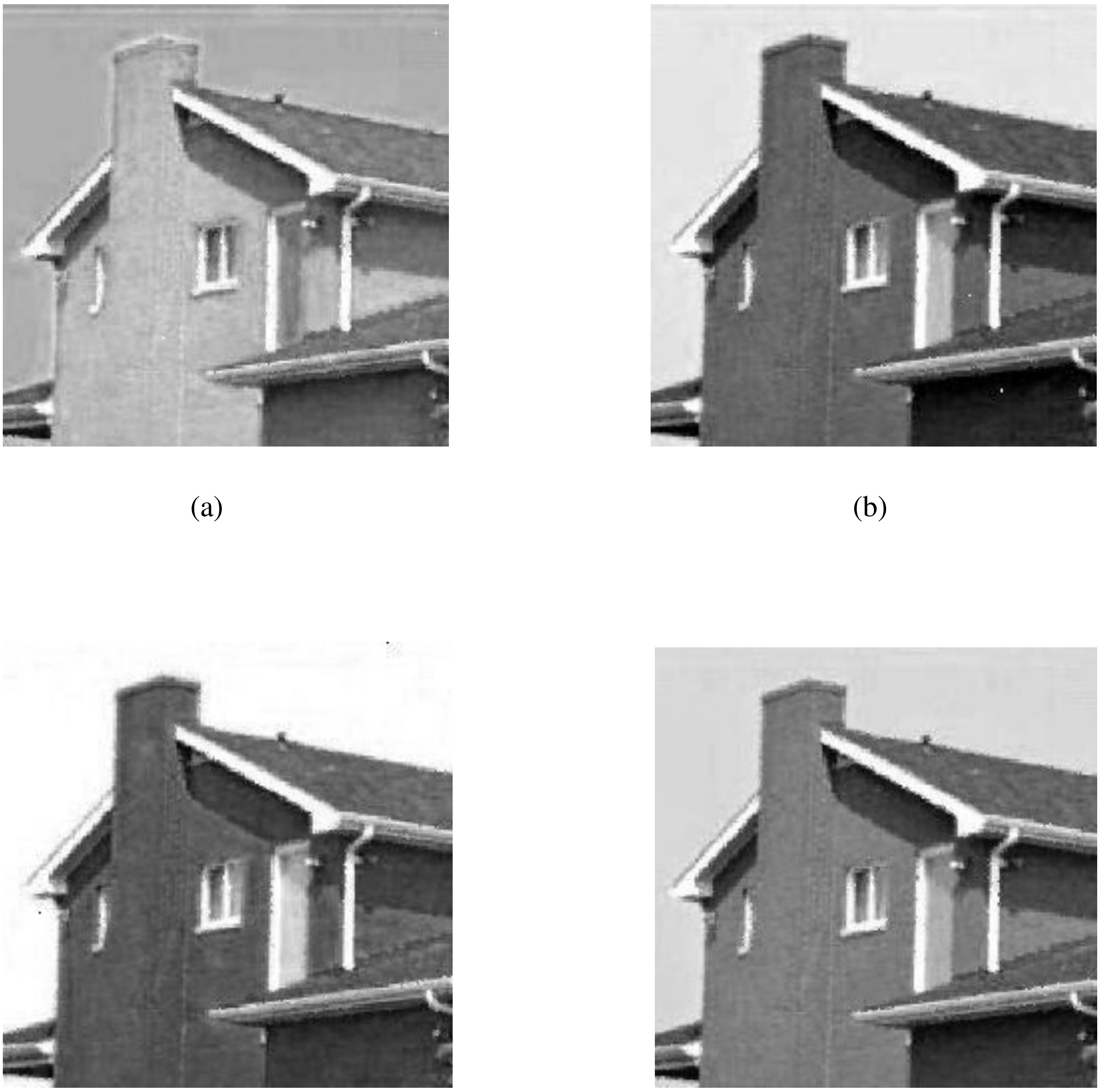


(c)



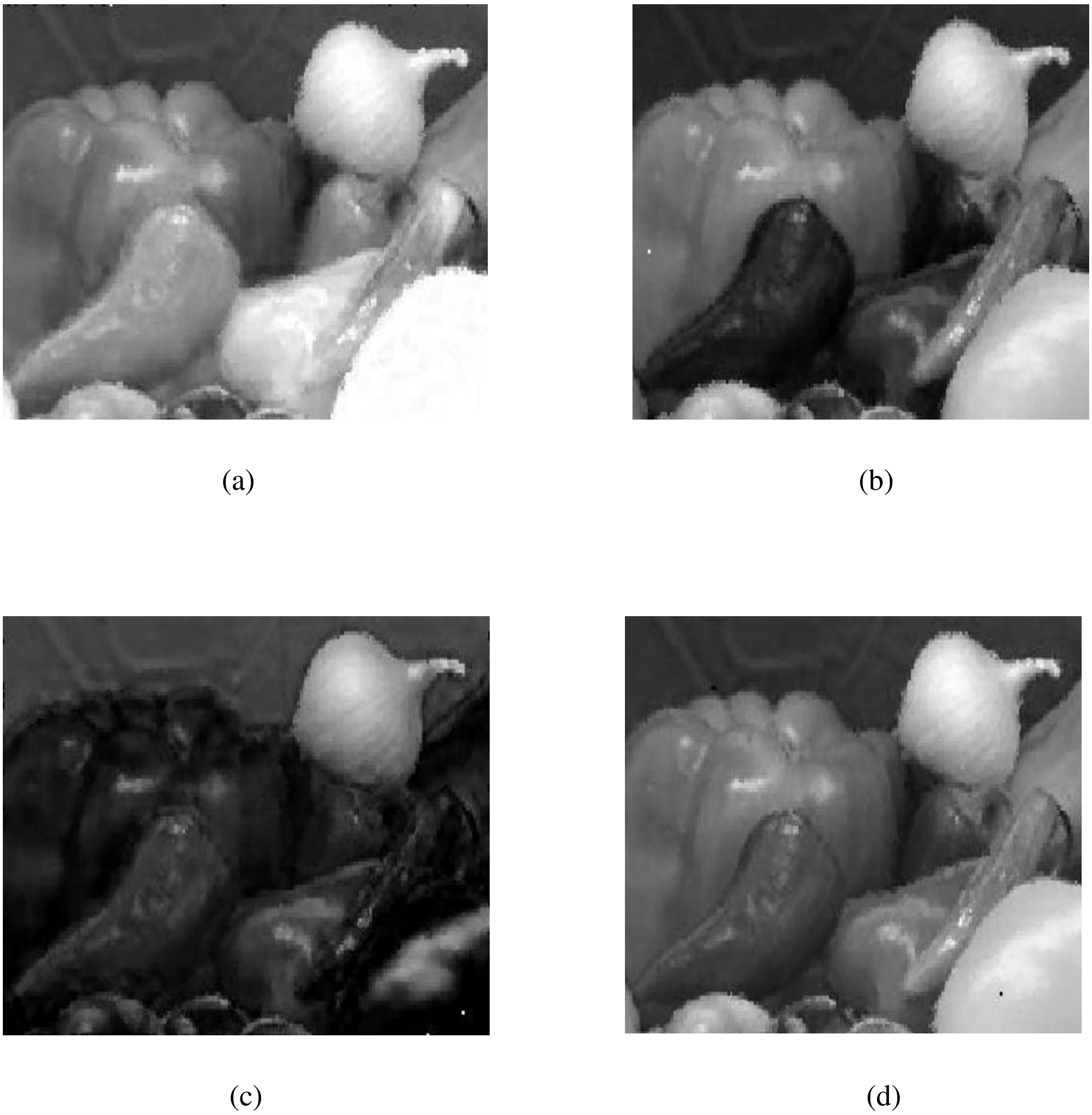
(d)

Figure 31: *Detector with BFOA baboon image a) Red component b) Green component c) Blue component d) Gray image*



(a) (b) (c) (d)

Figure 32: *Detector with BFOA house image a) Red component b) Green component c) Blue component d) Gray image*



(a)

(b)

(c)

(d)

Figure 34: *Detector with BFOA a) lena b) baboon c) house d) peppers*

The results of the comparison of the mean squared error calculated without and with optimization for different set of noisy images are given below:

Table 7: *Comparison of the MSE values calculated for the output images of the filters when used without and with optimization.*

Image	*Lena*	*Baboon*	*House*	*Peppers*
Without Optimization	144.03	308.63	107.34	151.91
With Optimization	17.44	95.59	33.32	57.97

The results of the comparison of the mean pixel distance calculated without and with optimization for different set of noisy images are given below:

Table 8: *Comparison of the MPD values calculated for the output images of the filters when used without and with optimization.*

Image	*Lena*	*Baboon*	*House*	*Peppers*
Without Optimization	6.64	16.34	6.00	6.96
With Optimization	1.40	5.74	1.18	1.58

The average MSE and MPD values for the filtering experiments performed have been shown in the table. Each of these tables comprises two sections, which are entitled "*Without optimization*" and "*With optimization*" respectively. The MSE or MPD values in the first section of the tables represent the results obtained by using the noise filters directly on the noisy test images with the fuzzy logic impulse detector; whereas the values in the second section represent the results obtained by using the noise filters with the proposed detector after optimized by BFOA.

It is easily demonstrated that the presented impulse detector with BFOA significantly improves the performance of all noise filters regarding both the MSE and the MPD criteria independent of the test image and the noise density.

The value of mean and standard deviation is between 0 and 1, inclusive. The parameters used in the experiment are:

Table 9: *Horizontal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	m	σ	m	σ	m	σ	m	σ
M_{i1}	0.41	0.90	0.50	0.97	0.64	0.86	0.50	0.80
M_{i2}	0.44	0.86	0.50	0.90	0.40	0.87	0.45	0.77
M_{i3}	0.40	0.80	0.55	0.88	0.45	0.95	0.45	0.95

Table 10: *Vertical class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	m	σ	m	σ	m	σ	m	Σ
M_{i1}	0.54	0.90	0.55	0.97	0.44	0.77	0.52	0.95
M_{i2}	0.51	0.86	0.52	0.70	0.50	0.81	0.43	0.83
M_{i3}	0.50	0.80	0.59	0.88	0.45	0.96	0.45	0.75

Table 11: *Principal diagonal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	m	σ	m	σ	m	σ	m	Σ
M_{i1}	0.56	0.90	0.50	0.77	0.54	0.76	0.50	0.93
M_{i2}	0.40	0.76	0.50	0.90	0.50	0.77	0.65	0.77
M_{i3}	0.46	0.80	0.55	0.88	0.65	0.75	0.55	0.86

Table 12: *Other diagonal class of pixels*

MF	*Rule 1*		*Rule 2*		*Rule 3*		*Rule 4*	
	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\sigma}$	$\boldsymbol{m}$	$\boldsymbol{\Sigma}$
$\boldsymbol{M_{i1}}$	0.57	0.70	0.42	0.87	0.55	0.76	0.52	0.90
$\boldsymbol{M_{i2}}$	0.48	0.86	0.58	0.80	0.53	0.87	0.49	0.81
$\boldsymbol{M_{i3}}$	0.42	0.90	0.42	0.88	0.47	0.95	0.43	0.90

Table 13: *Comparison of FCR values calculated for the detectors*

Detectors	Lena	Baboon	House	Peppers
Median Filter	5.51	8.27	2.71	5.77
FL impulse detector	3.98	6.52	1.75	4.73
Optimized impulse detector	0.12	0.01	0.08	2.53

This table presents the false classification ratio (FCR) values calculated for all impulse detectors. The FCR value is calculated as follows:

$$FCR = \frac{N_F}{N_T} \times 100 \tag{14}$$

where $N_F$denotes the number of falsely classified pixels of theinput image and $N_T$denotes the total number of pixels. It is easily observed from this table that the presented optimized impulse detector considerably outperforms the previous impulse detector.

Chapter 6

CONCLUSIONS

The current work has been focused on detection of impulse noise in gray scale as well as colour images with the help of fuzzy inference systems which consists of subdetectors whose parameters are optimized by bacterial foraging. The procedure for processing the noisy input image with a noise filter and applying the presented impulse detector for improving the output image of the noise filter is as follows:

1) A 3-by-3 pixel filtering window is slid over each colour band of the noisy input image one pixel at a time. The window is started from the upper-left corner of the colour band and moved sideways and progressively downwards in a *raster scanning* fashion until the bottom right corner position is reached.

2) For each filtering window position, the appropriate pixels of the filtering window representing the appropriate neighbourhoods of the centre pixel are fed to both the noise filter and the impulse detector inputs.

3) If the output of the impulse detector for the filtering window under concern is 0, which means that the centre pixel of the filtering window is uncorrupted and does not need restoration, the centre pixel of the filtering window is directly copied to the output image.

4) If the output of the impulse detector for the filtering window under concern is 1, which means that the centre pixel of the filtering window is corrupted and needs restoration, the pixel value obtained at the output of the noise filter is copied to the output image as the restored value of the centre pixel of the filtering window under concern.

5) This procedure is repeated until all pixels of the colour band under analysis and all colour bands of the noisy input image are covered.

Based on the results presented in the previous section and the remarks listed above, we conclude that the presented impulse detector can be used as an efficient tool for improving the output performance of a noise filter. The future scope of the work is to explore other evolutionary algorithms available in the literature for parameter optimization as well as to modify the proposed methods for the noisy images by means of changing the membership function and the fuzzy logic used in the sub detectors.

References

[1] D. Stephens and J. Krebs, *Foraging Theory*. Princeton, NJ: Princeton Univ.Press, 1986.

[2] W. O'Brien, H. Browman, and B. Evans, "Search strategies of foraging animals," *Amer. Scientist*, vol. 78, pp. 152-160, Mar./Apr. 1990.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Naturalto Artificial Systems.* New York: Oxford Univ. Press, 1999.

[4] C. Adami, *Introduction to Artificial Life*. New York: Springer-Verlag, 1998.

[5] M. Resnick, *Turtles, Termites, and Traffic Jams: Explorations in MassivelyParallel Microworlds*. Cambridge, MA: MIT Press, 1994.

[6] S. Levy, *Artificial Life: A Report from the Frontier where Computers Meet Biology.* New York: Vintage Books, 1992.

[7] T. Audesirk and G. Audesirk, *Biology: Life on Earth*, 5th ed. EnglewoodCliffs, NJ: Prentice Hall, 1999.

[8] H. Berg, "Motile behavior of bacteria," *Phys. Today*, pp. 24-29, Jan. 2000.

[9] M. Madigan, J. Martinko, and J. Parker, *Biology of Microorganisms*, 8th ed.Englewood Cliffs, NJ: Prentice Hall, 1997.

[10] F. Neidhardt, J. Ingraham, and M. Schaechter, *Physiology of the BacterialCell: A Molecular Approach*. Sunderland, MA: Sinauer, 1990.

[11] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J.Watson, *MolecularBiology of the Cell,* 2nd ed. New York: Garland Publishing, 1989.

[12] H. Berg and D. Brown, "Chemotaxis in escherichia coli analysed bythree-dimensional tracking," *Nature*, vol. 239, pp. 500-504, Oct. 1972.

[13] J. Segall, S. Block, and H. Berg, "Temporal comparisons in bacterialchemotaxis," *Proc. Nat. Acad. Sci.*, vol. 83, pp. 8987-8991, Dec. 1986.

[14] H. Berg, *Random Walks in Biology*. Princeton, NJ: Princeton Univ. Press,1993.

[15] D. DeRosier, "The turn of the screw: The bacterial flagellar motor," *Cell*,vol. 93, pp. 17-20, 1998.

[16] G. Lowe, M. Meister, and H. Berg, "Rapid rotation of flagellar bundles inswimming bacteria," *Nature*, vol. 325, pp. 637-640, Oct. 1987.

[17] T.-M. Yi, Y. Huang, M. Simon, and J. Doyle, "Robust perfect adaptation inbacterial chemotaxis through integral feedback control," *PNAS*, vol. 97, pp.4649-4653, April 25, 2000.

[18] J. Armitage, "Bacterial tactic responses," *Adv. Microbial Phys.*, vol. 41, pp.229-290, 1999.

[19] E. Budrene and H. Berg, "Dynamics of formation of symmetrical patternsby chemotactic bacteria," *Nature*, vol. 376, pp. 49-53, 1995.

[20] Y. Blat and M. Eisenbach, "Tar-dependent and -independent pattern formationby salmonella typhimurium," *J. Bacteriology*, vol. 177, pp. 1683-1691,Apr. 1995.

[21] E. Budrene and H. Berg, "Complex patterns formed by motile cells ofescherichia coli," *Nature*, vol. 349, pp. 630-633, Feb. 1991.

[22] D. Woodward, R. Tyson, M. Myerscough, J. Murray, E. Budrene, and H.Berg, "Spatio-temporal patterns generated by salmonella typhimurium,"*Biophysi. J.*, vol. 68, pp. 2181-2189, 1995.

[23] Zadeh, L.A., 1988. "Complexities inFuzzy logic", IEEE Computer., pp: 83-92, 1998.

[24] Serge, G., “Designing Fuzzy Inference Systems from Data:Interpretability oriented Review”, IEEETransactions on FuzzySystems., 9(3): 426-442, 2001.

[25] Wang, L.X., “Fuzzy systems are universal approximators”, proc. 1stIEEE conf. Fuzzy Systems, SanDiego, CA., 8(12): 1163-1170, 1992.

[26] Nayeripour, M., H. Khorsand, A.R. Roosta and T. Niknam, “A New Approach Based on FuzzyController for Volt/Var Control in Distribution System”, Australian Journal of Basic and Applied Sciences,4(3): 468-480, 2010.

[27]PejmanTahmasebi, ArdeshirHezarkhani; “Application of Adaptive Neuro-Fuzzy Inference System forGrade Estimation; Case Study, Sarcheshmeh Porphyry Copper Deposit, Kerman, Iran”, Australian Journal ofBasic and Applied Sciences, 4(3): 408-420, 2010.

[28] Akbarzadeh, A., R. TaghizadehMehrjardi, H. Rouhipour, M. Gorji and H.G. Refahi, “Estimating ofSoil Erosion Covered with Rolled Erosion Control Systems Using Rainfall Simulator (Neuro-fuzzy andArtificial Neural Network Approaches)”,Journal of Applied Scienes Research, 5(5): 505-514, 2009.

[29] F. Russo and G. Ramponi, “A fuzzy filter for images corrupted by impulse noise,”*IEEE Signal Processing Lett.*, vol. 3, pp. 168–170, 1996.

[30] Y. S. Choi and R. Krishnapuram, “A robust approach to image enhancement basedon fuzzy logic,” *IEEE Trans. Image Processing*, vol. 6, pp. 808–825, 1997.

[31] D. Van De Ville, M. Nachtegael, and D. Van der Weken, “Noise reduction by fuzzyimage filtering,” *IEEE Trans. Fuzzy Syst.*, vol. 11, pp. 429–436, 2003.

[32] S. Morillas, V. Gregori, and G. Peris-Fajarne, “A fast impulsive noise colour imagefilter using fuzzy metrics,” *Real-Time Imag.*, vol. 11, pp. 417–428, 2005.

[33] R. John and S. Coupland, “Type-2 fuzzy logic: A historical view,” *IEEE Comput.Intell. Mag.*, vol. 2, no. 1, pp. 57–62, 2007.

[34] J. M. Mendel, "Type-2 fuzzy sets and systems: An overview," *IEEE Comput. IntellMag.*, vol. 2, no. 2, pp. 20–29, 2007.

[35] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Trans. FuzzySyst.*, vol. 10, pp. 117–127, 2002.

[36] M. T. Yildirim, A. Bas¸türk, and M. E. Yüksel, "Impulse noise removal from digitalimages by a detail-preserving filter based on type-2 fuzzy logic," *IEEE Trans. Fuzzy Syst.*,vol. 16, pp. 920–928, 2008.

[37] H. Bustince, E. Barrenechea, and M. Pagola, "Interval-valued fuzzy sets constructedfrom matrices: Application to edge detection," *Fuzzy Sets Syst.*, vol. 160, pp. 1819–1840, 2009.

[38] P. Melin, O. Mendoza, and O. Castillo, "An improved method for edge detectionbased on interval type-2 fuzzy logic," *Expert Syst. Applicat.*, vol. 37, pp. 8527–8535, 2010.

[39] M. E. Yüksel and E. Bes¸dok, "A simple neuro-fuzzy impulse detector for efficientblur reduction of impulse noise removal operators for digital images," *IEEE Trans. FuzzySyst.*, vol. 12, pp. 854–865, 2004.

[40] N. N. Karnik and J. M. Mendel, "Centroid of a type-2 fuzzy set," *Inform. Sci.*, vol.132, pp. 195–220, 2001.

[41] L. Astudillo, O. Castillo, and P. Melin, "Intelligent control of an autonomous mobilerobot using type-2 fuzzy logic," *J. Eng. Lett.*, vol. 13, pp. 93–97, 2006.

[42] L. Gu and Y. Q. Zhang, "Web shopping expert using new interval type-2 fuzzy reasoning," *Soft Comput.*, vol. 11, pp. 741–751, 2007.

[43] N. N. Karnik and J. M. Mendel, "Centroid of a type-2 fuzzy set," *Inform. Sci.*, vol.132, pp. 195–220, 2001.

[44] H. R. Tizhoosh, “Image thresholding using type II fuzzy sets,” *Pattern Recognit.*, vol.38, pp. 2363–2372, 2005.