MARMARA UNIVERSITY

FACULTY OF ENGINEERING

TYPE 2 FUZZY LOGIC APPLICATION OF GRAIN BOUNDARIES IN SEM PHOTOS

Deniz Erdem

GRADUATION PROJECT REPORT

Department of Mechanical Engineering

Supervisor

Prof. Dr. Aykut KENTLI

ISTANBUL, 2023

MARMARA NIVERSITY

FACULTY OF ENGINEERING

TYPE 2 FUZZY LOGIC APPLICATION TO RECOGNITION OF GRAIN BOUNDARIES IN SEM PHOTOS

by

Deniz ERDEM

June 2023, Istanbul

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

BACHELOR OF SCIENCE

AT

MARMARA UNIVERSITY

Signature of Author(s) ....

Department of Mechanical Engineering

Certified By ....

Project Supervisor, Department of Mechanical Engineering

Accepted By...

Head of the Department of Mechanical Engineering

ACKNOWLEDGEMENT

First of all, I would like to thank my supervisor **Prof.Dr. Aykut KENTLI**, for the valuable guidance and advice on preparing this thesis and giving me moral and material support.

June, 2023 **Deniz Erdem**

TABLE OF CONTENTS

ABSTRACT

Type-2 Fuzzy Logic Application to Recognition of Grain Boundaries in SEM Photos

A fuzzy logic computational strategy expands on conventional binary reasoning by introducing the idea of partial truth. It offers a structure for addressing ambiguity, uncertainty, and imprecision in information and making choices. Fuzzy logic lets in membership levels, emulating the fuzziness of real-world occurrences, in contrast to conventional logic, which depends on precise true or false values. Fuzzy sets, fuzzy rules, and linguistic variables are all used in fuzzy logic to simulate and deduce uncertain data. Fuzzy logic offers more adaptable and nuanced evaluation and choice-making in complex systems by collecting and analyzing ambiguity. Fuzzy logic is a helpful tool for handling real-world issues where defined limits and strong values might not apply because of its capacity to tolerate uncertainty. This study investigates fuzzy reasoning, particularly type-2 fuzzy logic, to identify grain boundaries in images taken with a scanning electron microscope (SEM).

SYMBOLS

∇ : Gradient Operator

Δ : Delta

ABBREVIATIONS

FIS **:** Fuzzy Inference Systems

FL **:** Fuzzy Logic

H **:** High

L **:** Low

M **:** Medium

MF **:** Membership Functions

SD **:** Standard Deviation

LIST OF FIGURES

1. INTRODUCTION

Problem Statement and Objectives of the Research

The precise identification of boundaries of grains in SEM images is the issue this study attempts to solve. Since they affect various material properties and behaviors, grain boundaries are vital to the Characterization and research of materials. Traditional approaches to grain boundary detection sometimes rely on subjective, time-consuming hand-annotation or thresholding methods techniques that might not fully capture complicated grain boundary patterns. For scientists to overcome current approaches' shortcomings, this study's primary goal is to create a grain boundary identification method employing type-2 fuzzy logic. Even in the face of noise, fluctuations in grain size, and unusual grain boundary shapes, the method seeks to deliver accurate and reliable grain boundary recognition in SEM images.

1.1.Brief Introduction to Fuzzy Logic

A mathematical framework called fuzzy logic enables the representation and management of erroneous or uncertain data. Fuzzy logic provides the idea of partial truth by giving various values varying degrees of membership, in contrast to standard binary logic, which is concerned with absolute values that are either true or false. It offers a versatile and understandable way to deal with vagueness, ambiguity, and unpredictability, rendering it appropriate for real-world applications (Panicker, 2023). The foundation of fuzzy logic is that things might have shades of truth instead of wholly true or false. Fuzzy logic facilitates the modeling and reasoning of complicated systems with ambiguous boundaries and uncertain data using multilingual factors, fuzzy sets, and rules (Angili et al., 2023). Fuzzy logic is a valuable tool in decision-making processes, machine control, detection of patterns, and other areas where vagueness and imperfection are prevalent because of its capacity to gather and interpret uncertain information.

1.2. Essentiality of Grain Boundary Detection in SEM Images

In material science and engineering, identifying the boundaries of grains in SEM (scanning electron microscope) pictures is of utmost significance. The surfaces between neighboring crystalline grains in an item are known as grain boundaries, and these interfaces significantly impact the material's physical, electrical, and chemical features. For various applications, including material design, manufacturing optimization, and failure analysis, it is essential to comprehend and characterize grain boundaries (Li et al., 2022). In SEM pictures, grain boundaries may be precisely identified and analyzed to provide researchers with important information about the architecture of materials. Grain boundaries impact essential material characteristics, including endurance, electrical conductivity, and corrosion resistance. Additionally, they are essential in procedures like grain development, recrystallization, and phase changes (Khalafe et al., 2022). Therefore, accurate grain boundary detection and Characterization help us better understand the behavior and performance of materials.

Detecting grain boundaries in SEM images also makes quantifying and analyzing the size of grains, shape, and dispersion easier. These factors are essential for determining material performance under various circumstances, appraising processing methods, and determining material quality. Defects, including dislocations, twins, and anomalies related to grain boundaries, can all be detected with the help of grain boundary information (Sifan, 2022). Grain boundary identification in SEM images is crucial because it can reveal vital information about the microstructure and characteristics of materials (Panicker, 2023). It helps scientists to build materials with improved performance and dependability by optimizing material design and fabrication techniques. Improvements in the study of materials are made possible by precise grain boundary determination in SEM pictures, a starting point for additional analysis and research.

2. FUZZY LOGIC

2.1.Basic Principles of Fuzzy Logic

Fuzzy logic differs from conventional binary logic in that it is based on many fundamental concepts. First, fuzzy logic recognizes that truth values might exist on an imaginary line spanning absolute truth (1) and absolute untruth (0). This idea of incomplete truth is captured using participation functions, which rank elements in a fuzzy set according to their degree of membership (Sifan, 2022). Given that they can take on quantities between 0 and 1, these membership functions enable a more sophisticated portrayal of uncertainty and ambiguity. The capability to deal with variables related to language and fuzzy sets is another tenet of fuzzy logic (Khalafe et al., 2022). Fuzzy sets enable limited membership, unlike crisp sets in standard logic, which classify elements as fully part of an ensemble or not at all. This implies that items can have varied degrees of membership in different fuzzy sets simultaneously. Fuzzy logic can more precisely simulate real-world circumstances where concepts and limits are frequently ill-defined, thanks to its adaptability in handling linguistic factors.

Fuzzy regulations and systems for inference are also included in fuzzy logic. Fuzzy rules use linguistic expressions to define the interactions between the inputs and outcomes in a system. Inference systems use these fuzzy rules to execute computations or make choices while considering the levels of participation linked to each language term (Fatimah et al., 2022). This enables the information's inherently unpredictable and imprecise nature to be considered throughout the deliberative and reasoning processes. Fuzzy logic offers a robust framework for handling imperfect, ambiguous, and uncertain information by adopting these ideas (Li et al., 2022). It allows a more adaptable and sophisticated representation of actual phenomena, enabling more reliable judgment calls, pattern identification, and system control.

2.2.Membership Functions and Fuzzy Sets

The extent to which the component conforms to a particular fuzzy set is determined by membership functions, which are a crucial component of fuzzy logic. The degree to which an element possesses the traits of the set is indicated by the membership degree (Song et al., 2022), which is represented by a value that ranges from 0 to 1 by a membership function. The type of application and the envisioned depiction of uncertainty determine the membership function's shape. It can be trapezoidal, gaussian, triangular, or sigmoidal, among other shapes. Fuzzy logic can efficiently handle and reason with unreliable facts thanks to these functions, which embody the haziness and inaccuracy inherent in real-world data (Fatimah et al., 2022). Essential elements of fuzzy logic are functions of membership and fuzzy sets.

In contrast, fuzzy sets classify elements according to how much they belong to a set. Fuzzy sets enable limited involvement compared to crisp classical sets, which assign entries to the set or its complement (Goktas et al., 2022). This implies that one element can simultaneously have varied degrees of membership in various fuzzy sets. Uncertainty, ambiguity, and shaky boundaries can all be represented and worked with using fuzzy sets, which offer a flexible foundation (Angili et al., 2023). They make it possible for real-world occurrences to be represented in a more sophisticated and detailed manner, allowing fuzzy logic to describe complicated structures and make judgments in the face of ambiguity. Fuzzy logic systems can be flexible and adaptable since the selection of functions for membership and fuzzy sets depends on the application in question and the desired way to express uncertainty.

2.3.Fuzzy Rules and Inference Systems

The fundamental components of fuzzy logic that support formulating decisions and reasoning in uncertain contexts include uncertain rules and inference systems. Fuzzy rules can capture these linkages by expressing the ambiguous relationships between input and output variables using linguistic terms, as shown in Figure 1 below. These guidelines specify how each linguistic term's degree of membership is considered when mapping the inputs to the outputs (Song et al., 2022). The inference system uses these fuzzy rules to generate calculations or choices. It combines the inputs and the rules, applying fuzzy logic operators like AND, OR, and NOT to provide clear outputs and establish the total degree of membership (Sifan, 2022). Fuzzy logic can handle and analyze detailed information thanks to fuzzy rules and inductive reasoning systems, offering a solid framework for modeling and resolving practical issues.

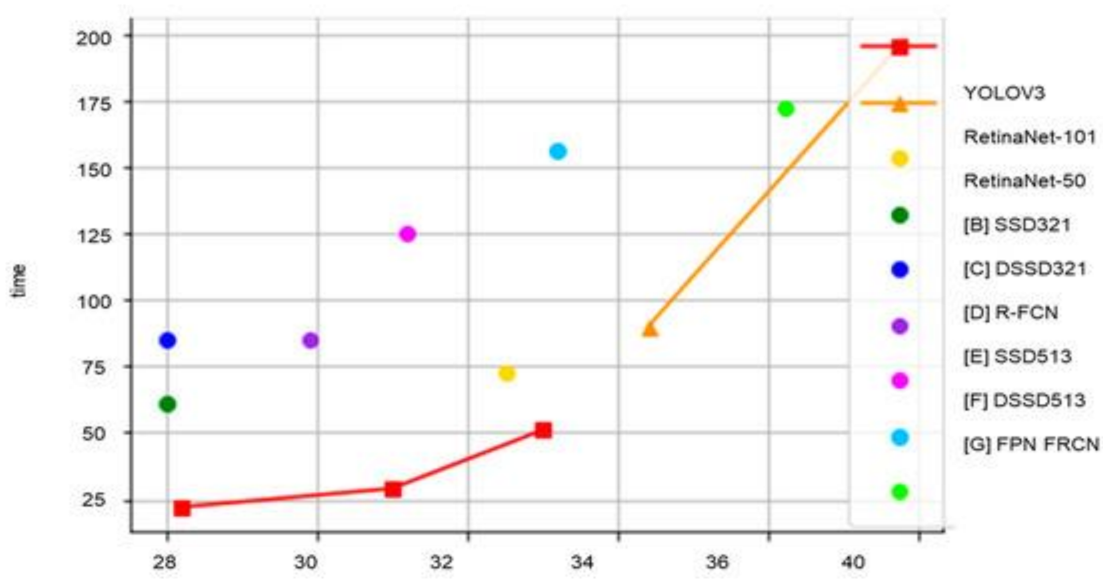

Figure 1: Rules for application of Fuzzy Logic

The process of creating fuzzy rules entails a thorough examination of the image characteristics that are important for detecting grain boundaries. This could involve edge information, texture, intensity fluctuations, or other pertinent properties. The project attempts to find the most informative elements and develop linguistic rules that describe the link between these qualities and the existence of grain boundaries by utilizing domain knowledge and skill (Angili et al., 2023). Considering the inherent inconsistencies and inaccuracies in the data, these fuzzy rules enable a more natural and understandable description of the detection process. To detect grain boundaries accurately and reliably, it is essential to construct strict fuzzy rules and reasoning algorithms (Mao et al., 2022). These elements enable fuzzy logic to make decisions based on imperfect and ambiguous data, enabling more exact grain boundary identification.

2.4.Type-2 Fuzzy Logic and its Advantages

Higher levels of ambiguity and unpredictability can be represented and handled using type-2 fuzzy logic, which expands the capabilities of classical fuzzy logic. In type-2 fuzzy logic, an additional degree of uncertainty is added when the functions of membership themselves turn into fuzzy sets (Li et al., 2022). This added level of uncertainty offers a more reliable and adaptable framework for handling complicated and inaccurate information. Type-2 fuzzy logic enables the modeling of uncertainty in shape and limits of the fuzzy sets and the membership degrees. The technique can handle image data's inherent unpredictability and uncertainty more efficiently by using type-2 fuzzy logic to detect grain boundaries in SEM images (Angili et al., 2023). Type-2 fuzzy logic has the advantage of being flexible in modeling complex and uncertain infrastructure, representing the logic and dealing with more significant levels of uncertainty, and having the potential to perform better in complex real-world settings.

Another benefit is the improved ability of type-2 fuzzy logic to handle intricate and nonlinear interactions. The ability of conventional type-1 fuzzy logic to capture complex relationships and interactions between parameters is constrained. Type-2 fuzzy logic, on the other hand, is better able to handle nonlinear and ambiguous interactions, enabling more precise simulation and forecast in intricate systems (Angili et al., 2023). Compared to probabilistic approaches, type-2 fuzzy logic provides more flexibility in expressing and handling uncertainty. It is suitable when there is a lack of data or only a small amount of objective information because it does not require accurate probability distributions (Jiang et al., 2022). Fuzzy logic's linguistic character enables intuitive and understandable illustrations of apprehension, making it simpler to explain decision-making processes and convey them to others.

3. GRAIN BOUNDARY DETECTION USING TYPE-2 FUZZY LOGIC

3.1.Preprocessing the SEM Photo

Preprocessing enhances picture quality and lowers noise when detecting grain boundaries in SEM images using type-2 fuzzy logic. Several preprocessing processes are applied to the SEM images to improve their applicability for precise grain boundary characterization (Song et al., 2022). The colored SEM picture is first transformed to grayscale to streamline further image processing while preserving the required intensity information. Then, applying appropriate filters, such as Gaussian or median filters, to minimize noise while accomplishing image smoothing removes undesirable artifacts and enhances overall image quality (Fatimah et al., 2022). The contrast is then adjusted to improve grain boundary visibility using enhancement techniques like histogram equalization or adaptive contrast stretching. In order to further minimize noise while keeping crucial image information, image-denoising algorithms like wavelet eliminating or double filtering are used.

3.2.Fuzzification of Image Pixels

The fuzzification of picture pixels is crucial in reflecting the unpredictability and inaccurate borders of the SEM images when utilizing type-2 fuzzy logic to detect grain boundaries. By converting the exact value of pixels to fuzzy sets, a process known as fuzzification, it is possible to characterize the degree to which each pixel belongs to specific categories (Jiang et al., 2022). Relevant member functions have been allocated to define the various grey levels or intensities contained in the image in order to fuzzify the image's pixels. These membership roles can be established using data-driven methods or domain expertise (Zhang et al., 2022). The particular SEM photo attributes and the desired depiction of uncertainty will determine which membership functions are used.

The approach allows for the gradual shift and unpredictability of pixel values by fuzzifying the graphic pixels, making it easier to apply type-2 fuzzy logic later. Considering the inherent diversity and inaccuracy in the SEM pictures offers a more realistic portrayal of the grain boundaries (Yu et al., 2022). The first phase of fuzzy inductive and rule-driven processing, fuzzification, provides the groundwork for enhanced grain boundary recognition and evaluation in SEM images (Zhang et al., 2022). The technique can manage the variation and unpredictability found in SEM photographs by assigning suitable member functions to the pictures' pixels, enabling precise and reliable grain line recognition and assessment.

3.3.Designing Fuzzy Rules for grain Boundary Detection

A critical component of using type-2 fuzzy reasoning for grain border recognition in SEM images is the creation of fuzzy rules. The connections between input parameters, such as intensity or texture properties gradients, and the outcome variable signifying the presence or lack of a grain boundary are captured by fuzzy rules. These regulations may be created using data-driven methods or expert knowledge (Chen et al., 2022). In order to create fuzzy rules, various stages of input, as well as output variables, are represented by linguistic phrases and variables. Fuzzy if-then rules explain the interactions between these variables by defining how the inputs correspond to the output (Ali et al., 2022). Combining linguistic concepts and fuzzy sets with fuzzy logic operators like AND, OR, and NOT enables flexible and accurate rule representation.

Given the unique properties of grain boundaries in SEM images and the available input information, fuzzy rules must be carefully designed. It entails specifying the proper membership functions, creating the rule base, and, if necessary, figuring out the rule weights. Consequently, type-2 fuzzy logic can detect grain boundaries accurately and reliably, facilitating better material analysis and comprehension in SEM imaging (Li et al., 2022). Additionally, the range and unpredictability of the SEM images and the particular needs of the grain boundary identification task should be considered while designing fuzzy rules (Zhang et al., 2022). The algorithm's efficiency can be optimized, resulting in precise detection and differentiation of grain boundaries, by adjusting the fuzzy rules through repeated modification and verification against ground truth data.

3.4. Defuzzification and obtaining the grain Boundary Map

Defuzzification is a crucial stage in the type-2 fuzzy logic method for detecting grain boundaries, followed by acquiring the grain boundary map. In order to appropriately depict the existence or inability of grain boundaries, the fuzzy output—which indicates the degrees to which pixels belong to the class of grain boundaries—must be defuzzified (Zhang et al., 2022). Various defuzzification techniques, including centroid-based and optimum inclusion methods, can be used to find a single representative value. The particular demands of the program should be addressed by the defuzzification technique of choice (Yu et al., 2022). This procedure makes sure that grain boundaries are clearly defined.

After defuzzification, the binary form of the boundaries of grains in the SEM image can be seen thanks to the sharp grain boundary map produced. This map aids in further investigation and characterisation of the microstructure of the materials by providing a visual and numerical representation of the identified grain boundaries (Li et al., 2022). Researchers can use the grain boundary map to analyze grain boundary characteristics, examine grain structure, and decide on material qualities and performance (Ali et al., 2022). The technique improves the understanding and investigation of material behavior and helps in numerous fields, such as materials in science, engineering, and production methods, by completing defuzzification and getting the grain boundary map.

4. MATLAB CODE EXPLANATION

Grain boundary is a basic feature of an image to obtain image characteristic. In a digital image, an edge occurs when the brightness changes sharply. Edges correspond to discontinuities in depth, or in surface orientation, changes in material properties, and variations in scene illumination. In the fuzzy logic system, we have used input and output variables and rule based algorithm for detecting edges. In the literature, there are some classical techniques to find sudden changes of discontinuities in the image is called as ‘edges’ such as Sobel, Robert, Prewitt, Laplacian of Gaussian, Canny, Kirsch. A novel approach to find edges in the image is fuzzy rule based algorithm. Mamdani fuzzy model is used and for this algorithm. Inputs and output of the system are defined using gradient and standard deviation. Fuzzy sets, membership functions and interval of them are selected intuitively.

4.1.Gradient and Standard Deviation Method

In this study, first both gradient and standard deviation values are computed, form two set of edges, utilized as inputs for our fuzzy system. Then fuzzy system decides on each pixel according to fuzzy rules. To obtain output value, ‘centroid’ method is used as defuzzification method and applying a proper threshold value. First step of the process it to find directional gradient of the gray scale image into the direction of x and y. After that gradient magnitude of the image is obtained.

$$Gx = \frac{dI(x,y)}{dx} = \lim_{\Delta x \to} \frac{I(x+n,y) - I(x-n,y)}{\Delta x}$$

$$Gy = \frac{dI(x,y)}{dy} = \lim_{\Delta y \to} \frac{I(x+n,y) - I(x-n,y)}{\Delta y}$$

By combining partial derivative of the image in the x and y direction, gradient vector of the image is written like that:

$$\nabla I = (Gx, Gy)$$

Using gradient vector, gradient magnitude and direction are calculated below equations:

$$g(x,y) = \sqrt{(\Delta x^2 + \Delta y^2)}$$

Gradient of an image and mapping into 0 to 100 and higher values of the gradient magnitude will be considered as edge candidate.

Second step is to find standard deviation of each pixel over adjacent neighborhood as shown below

P1	P2	P3
P4	P5	P6
P7	P8	P9

Figure 4.1: Adjacent neighborhood of center pixel

SD of each pixel and mapping into 0 to 100 and similarly pixels of standard deviation image, higher values will be correspond as edge candidate.

In the first two steps, inputs of the Fuzzy Inference System is defined and ready to create fuzzy set, membership functions, rules and the output of the system. As a third step, inputs, appropriate membership functions are defined for fuzzy system inputs. Then both of the mapped values are classified to one of the low, medium, or high classes. Using the classified values, specify and apply rules to the system. The output of fuzzy system explains to how extent a pixel could be edge. By the using fuzzy rules and centroid defuzzification method, the output of this fuzzy system is classified to one of three classes which are correspond to pixels with low, medium, high probability value to belong to edge pixels set.

In general, flow chart of the system is shown below

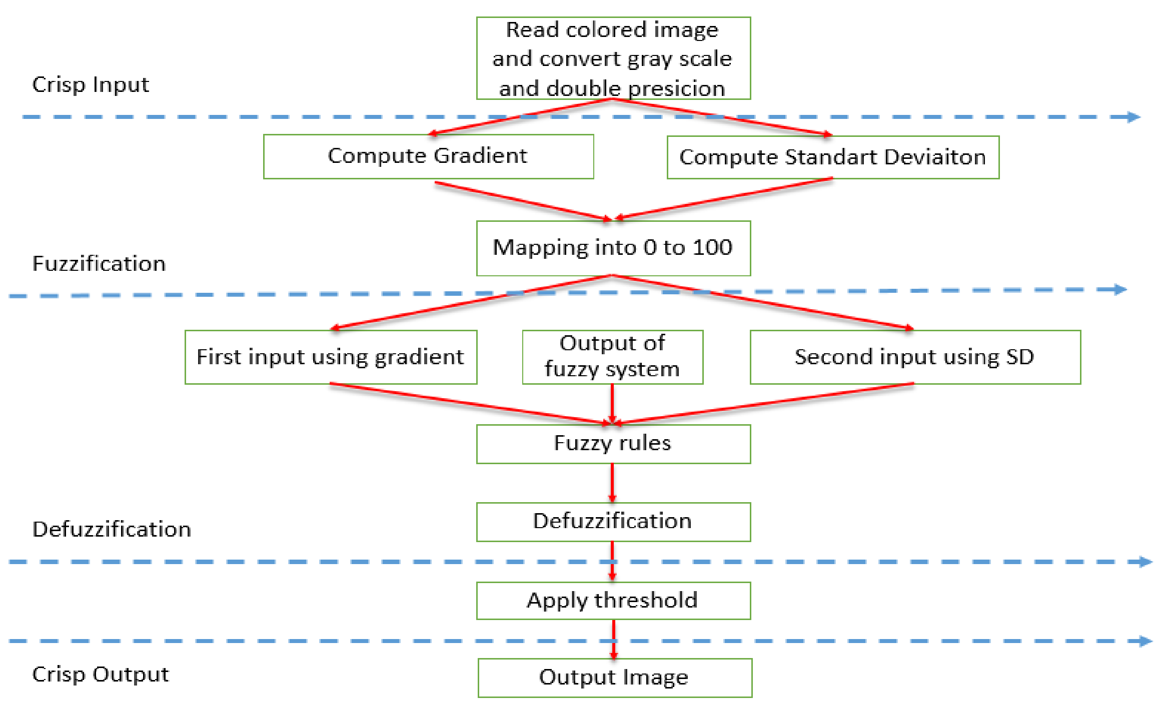

Figure 4.2: Flow chart of the system

4.2.Method Evaluation

A colored scale image is converted into the gray scale to intensity information and compute the directional change of the image in the x and y direction. Figure 4.2.1 and Figure 4.2.2 shows the colored and grayscale images, respectively.

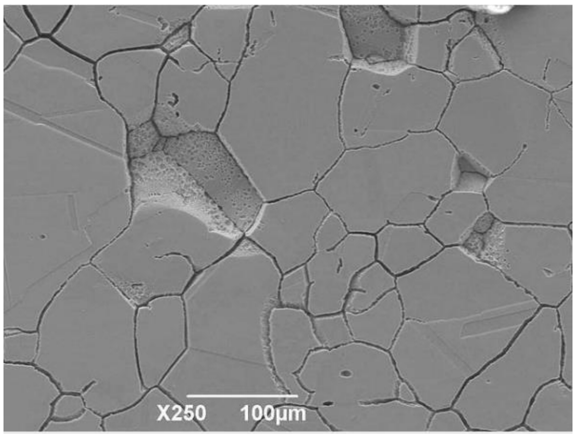

Figure 4.2.1: Original Image

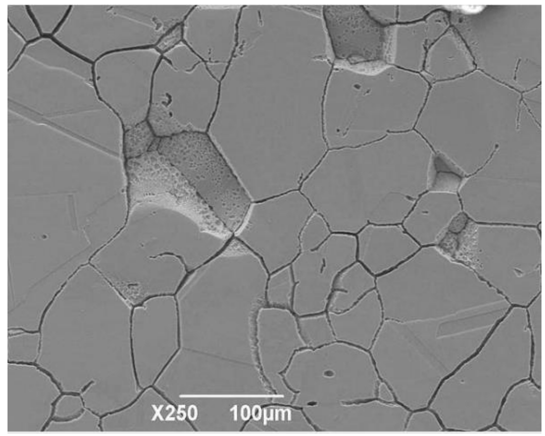

Figure 4.2.2: Grayscale Image

Directional derivatives of the grayscale image are computed by using Sobel gradient operator. The results are shown in the figures below, respectively. The figures shows how much the gray levels in image change in the positive x and y directions, this change in the intensity is encoded in the gray level of the image of the horizontal and vertical components.

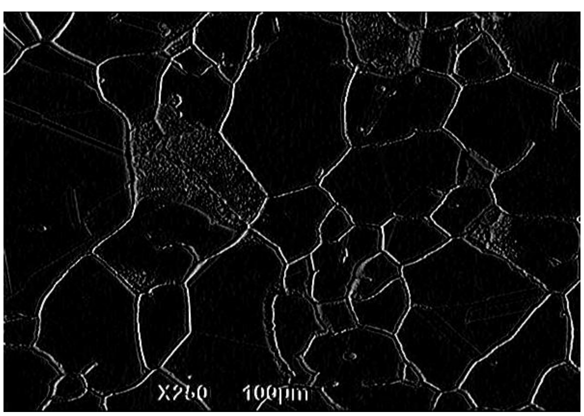

Figure 4.2.3: X direction gradient

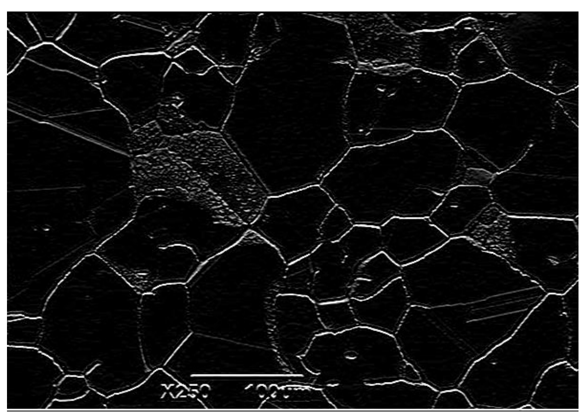

Figure 4.2.4: Y direction gradient

After directional changes of the image are computed, gradient magnitude and gradient direction can be found by using above mathematical equations. Again in the below figure, it represents magnitude values as an image in the range of 0 and 100. In the magnitude image grain boundary pixels has more value than background.

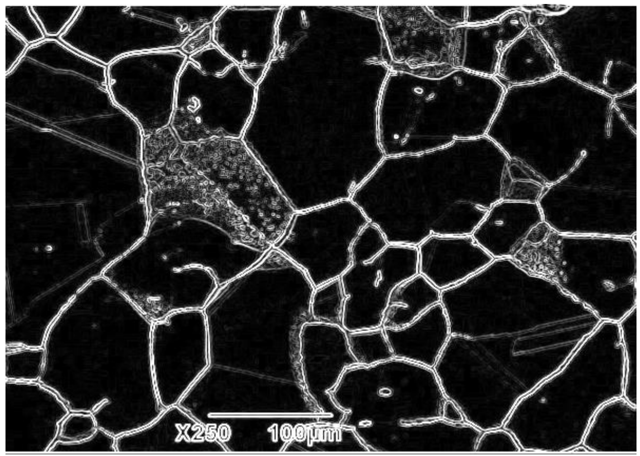

Figure 4.2.5: Gradient magnitude

As a second input, for each pixel, SD is computed by 3×3 mask and result below figure represents standard deviation of each pixel values as an image in the range of 0 and 100. Similarly, pixels with SD image edge pixels has more value than background.

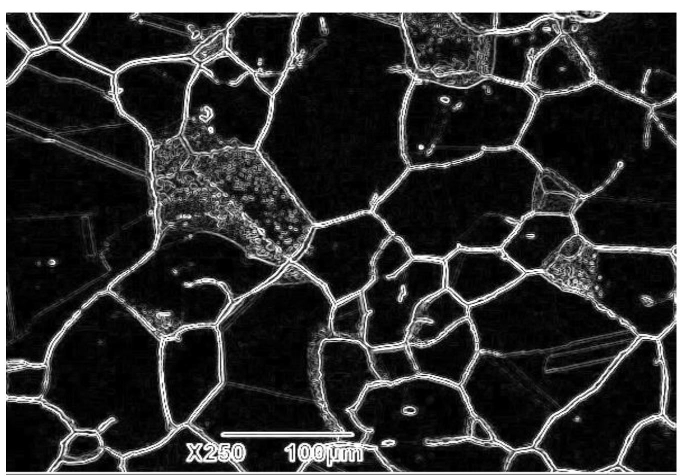

Figure 4.2.6: Standard Deviation of the image

Then both inputs gradient magnitude image and standard deviation of the mapped values and output of the system are classified to one of the low, medium, or high classes as shown in the below figure.

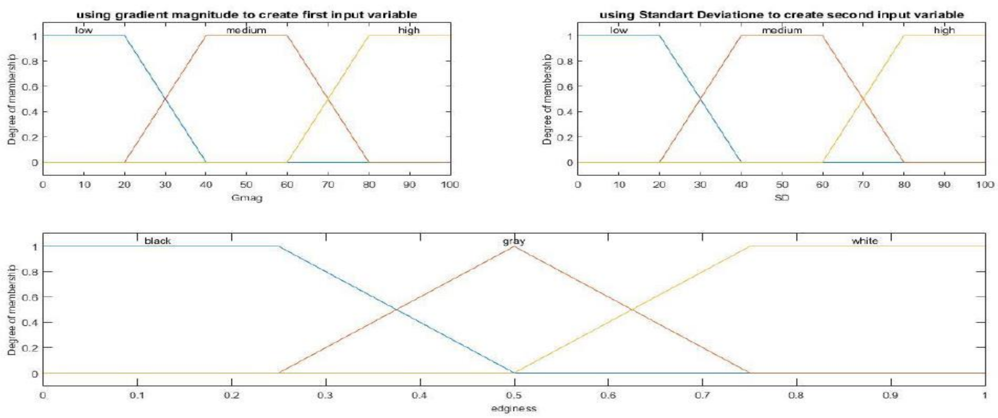

Figure 4.2.7: Inputs and Output of the Membership Functions

After specifying inputs and output, *IF-THEN* rules must be designed. In the figure below, inputs and output relations depending on the *IF-THEN* rules are represented. If-then rules are linguistic variables for membership functions.

Some of the rules like are:

1. If (Gmag is low) and (SD is low) then (edginess is black) (1)

2. If (Gmag is low) and (SD is medium) then (edginess is black) (1)

3. If (Gmag is low) and (SD is high) then (edginess is gray) (1)…

Inputs		Output
Gradient	SD	Edginess
L	L	L
L	M	L
L	H	M
M	L	L
M	M	M
M	H	H
H	L	M
H	M	H
H	H	H

L : low

M : medium

H: high

Figure 4.2.8: Rules for the Fuzzy Inference System

As a last step, centroid defuzzification is applied and output image is almost ready. Using output image and a proper threshold value, grain boundaries of the example image is shown below.

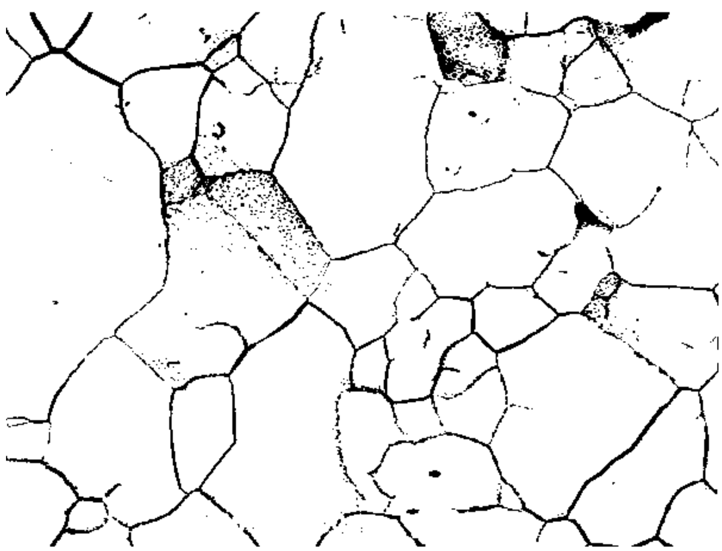

Figure 4.2.9: Grain Boundaries Output Image

4.3.Experimental Results

In the Experimental Results part, readers can see some SEM photos and their output data, where MATLAB determined the grain boundaries using the grain boundary detection code created in Type-2 Fuzzy Logic. All of the images below were created using the gradient method with a small amount of errors and noise, and the grain boundaries are successfully displayed with a high degree of clarity. Nine specimens were carefully selected to be used in the research and output images were created with the most possibly correct threshold factor.

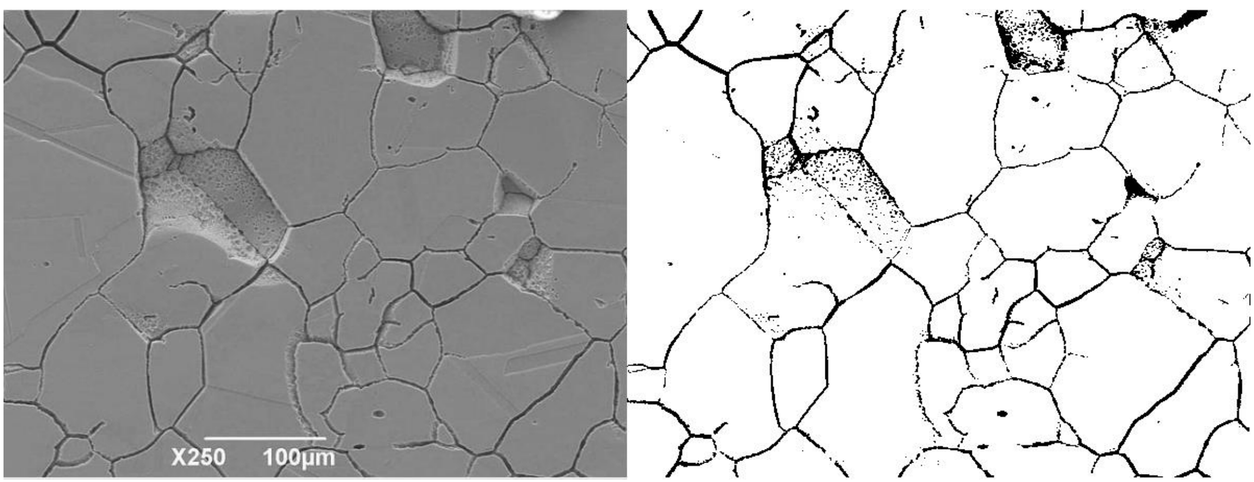

Figure 4.3.1: 310 stainless steel SEM image with its output image

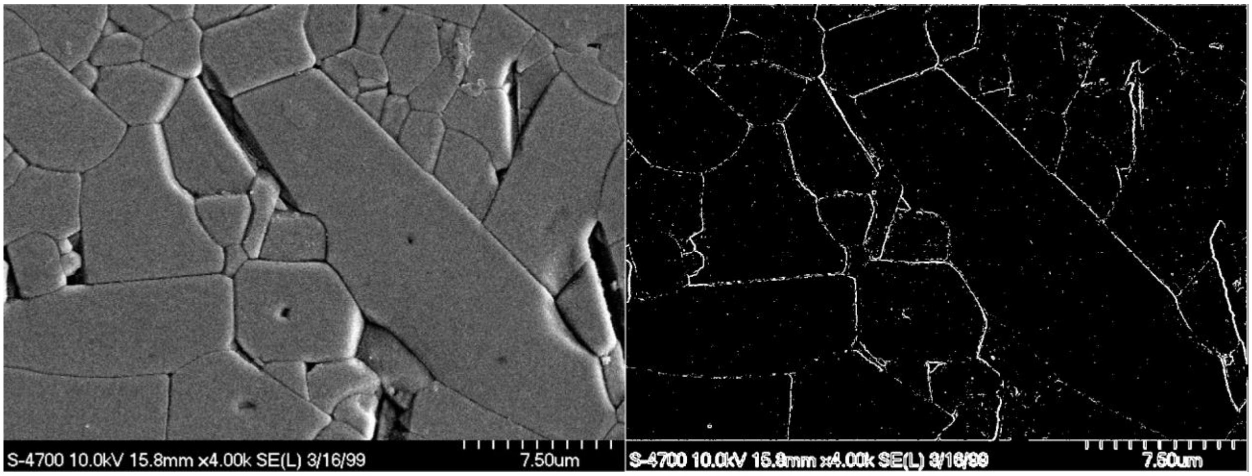

Figure 4.3.2: Alimuna with percentage 99.7 purity SEM image with its output image

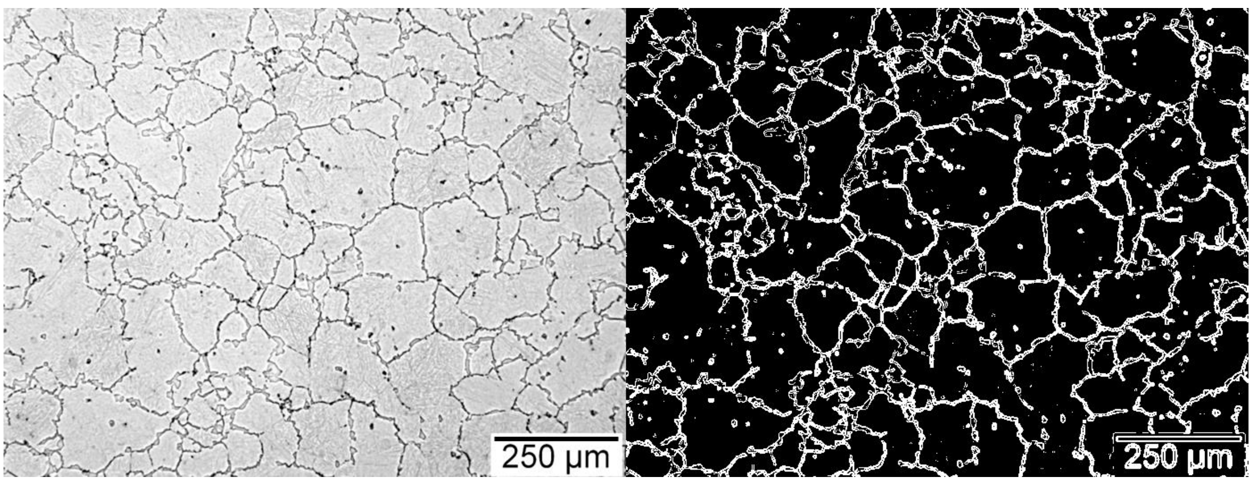

Figure 4.3.3: Austenite SEM image with its output image

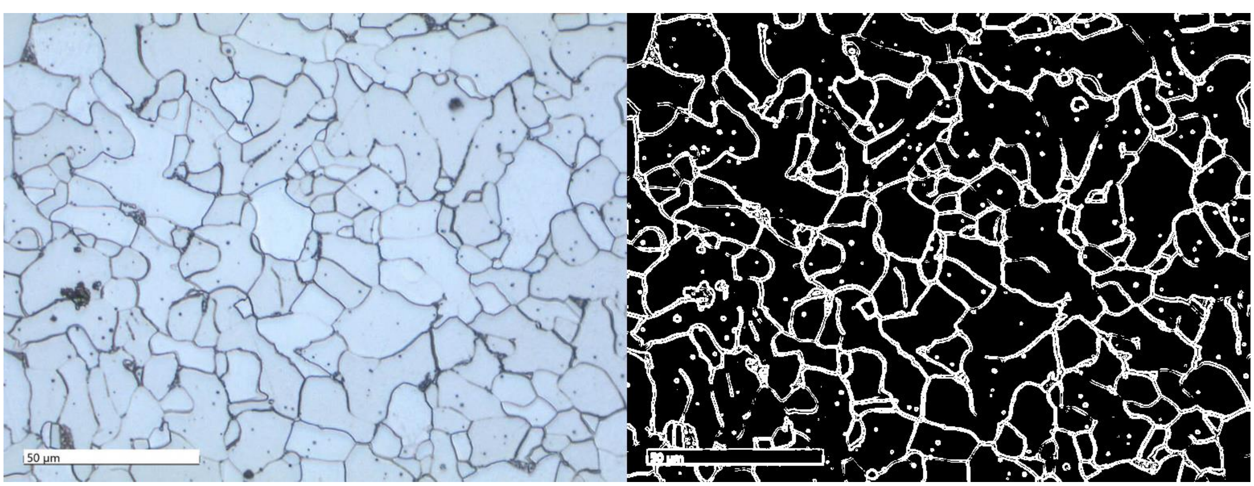

Figure 4.3.4: Ferritic Steel with % 0.1 C, etched with Nital SEM image with its output image

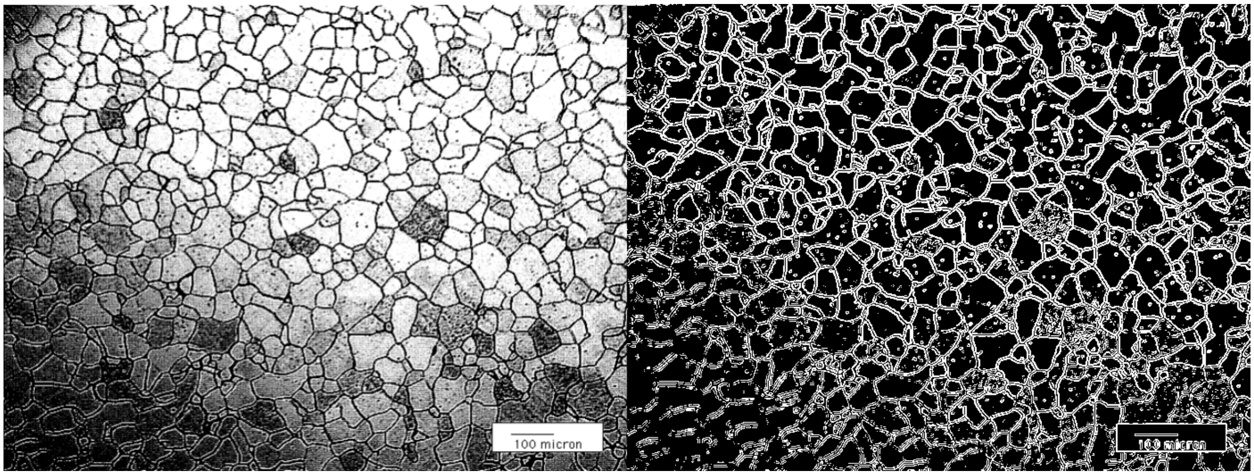

Figure 4.3.5: Grain Structure of Pre-Annealed Material SEM image with its output image

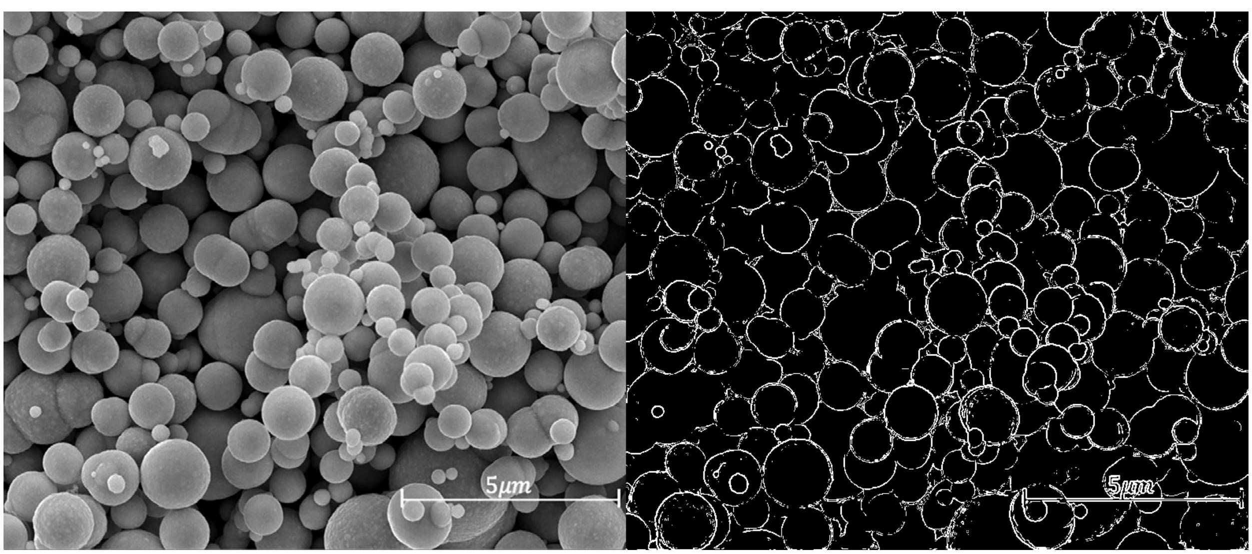

Figure 4.3.6: Iron Powder SEM image with its output image

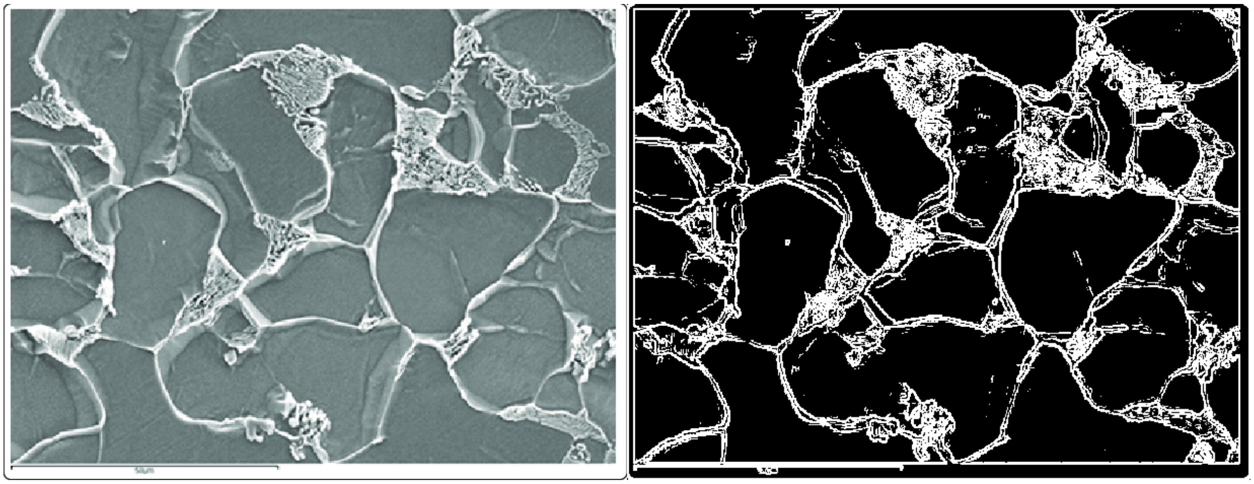

Figure 4.3.7: Laredo chain steel sample SEM image with its output image

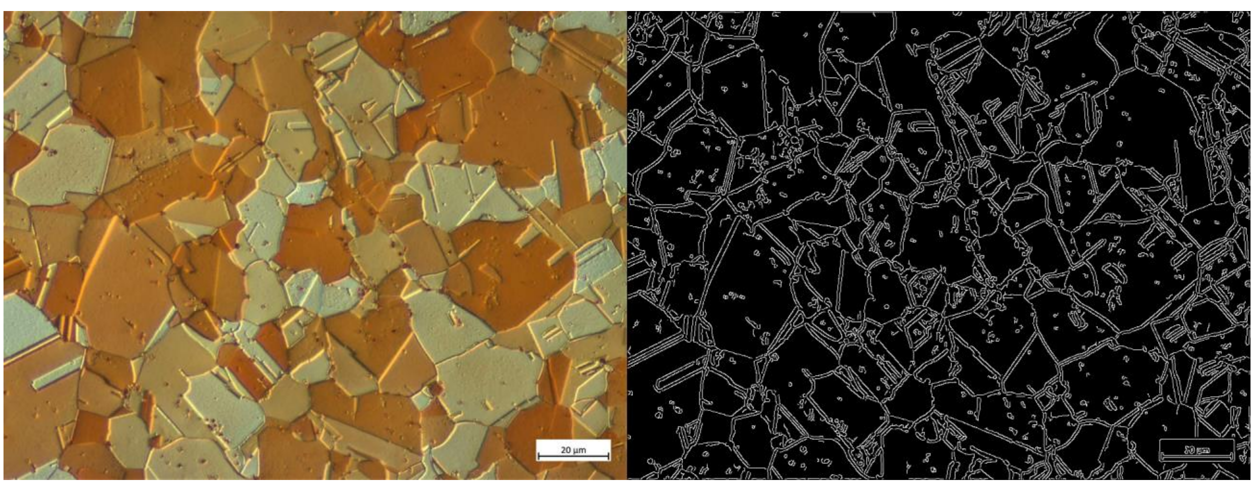

Figure 4.3.8: Pure Copper SEM image with its output image

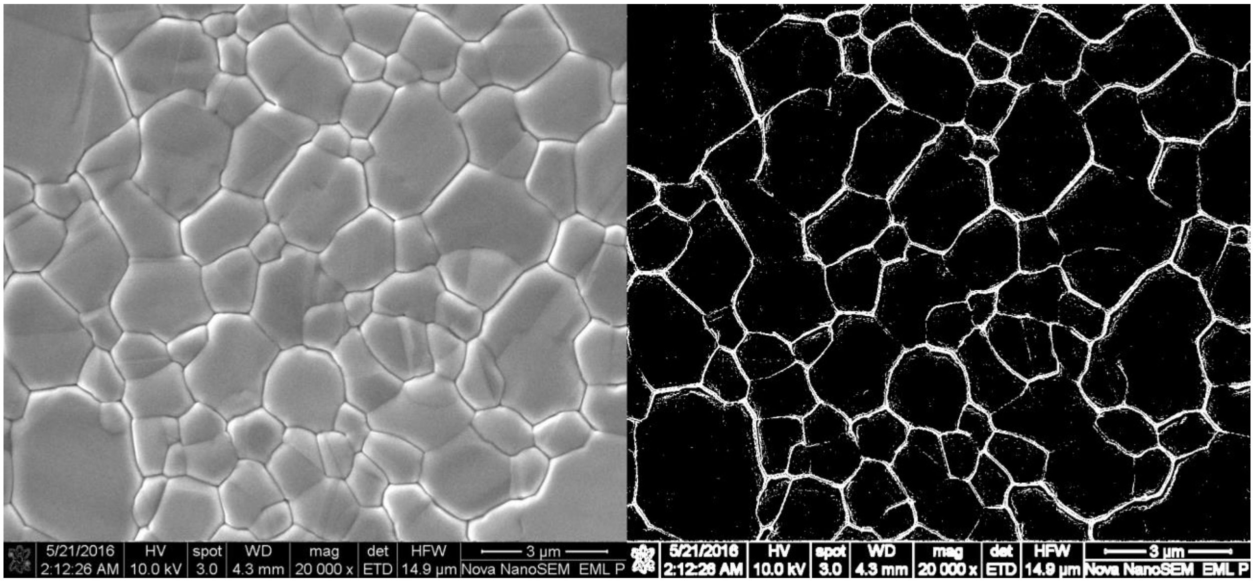

Figure 4.3.9: SiO_2 crystal SEM image with its output image

5. DISCUSSION

5.1.Description of the dataset used for evaluation

The dataset utilized for testing the in-depth evaluation of the type-2 fuzzy logic-based grain boundary identification using MATLAB code in SEM images is carefully selected to offer a thorough and varied representation. It consists of a selection of SEM photos that cover various grain boundaries and material kinds (Chen et al., 2022). This guarantees comprehensive coverage for substance compositions, including biological samples and mostly metals. The collection consists of SEM images taken at various magnifications and resolutions to capture grain structure features. The collection contains photos with various grain boundary difficulties to aid in a complete analysis (Ali et al., 2022). This includes SEM images with clearly discernible, straight-grain boundaries and those with uneven or curved boundaries. The dataset also considers grain boundary sizes and orientations, including smooth and coarse-grained structures.

The dataset is manually annotated in order to create a trustworthy foundation for purposes of assessment. Readers can thoroughly assess the efficacy of the grain boundary identification technique using this extensive dataset. The dataset gives researchers access to information about the algorithm's strengths and weaknesses, enabling a complete knowledge of how well it can identify grain boundaries in SEM images.

5.2.Comparison with other methods

The efficacy and distinction of the grain boundary recognition method using type-2 fuzzy logic in SEM images are evaluated against several widely used methods for grain boundary detection. The type-2 fuzzy reasoning methodology provides a more adaptable and flexible framework than thresholding techniques that depend on set intensity thresholds (Zhang et al., 2022). The approach effectively captures minor fluctuations and complicated grain-border structures that may be hard for thresholding approaches. The Canny or the Sobel method administrators, among other edge detection techniques, are frequently employed in image analysis to extract features. While they can recognize edges, noise or abnormalities may make it challenging to differentiate grain boundaries correctly (Ali et al., 2022). Contrarily, the type-2 fuzzy logic method can use linguistic conventions and contextual information to deliver more trustworthy and precise grain boundary recognition, even in the face of noise.

Deep learning models or alternative classifiers are used in machine learning-based methodologies, which have grown in popularity in recent years. These methods, however, frequently need a lot of instructional data and mighty computing power (Chen et al., 2022). On the other hand, the type-2 fuzzy logic approach provides a more comprehensible and transparent framework, enabling researchers to comprehend and optimize the rule-based system for grain boundary identification. The grain boundary detection technique employing type-2 fuzzy logic may show its benefits concerning accuracy, resilience, adaptability, and interpretability through careful comparisons with various existing methods (Yu et al., 2022). Readers can demonstrate the usefulness and promise of the type-2 fuzzy logic technique in SEM photo processing and grain boundary recognition activities by highlighting its better performance versus conventional thresholding, edge recognition, and data mining techniques.

5.3.Analysis of Results

Several evaluation measures are used to quantitatively evaluate the algorithm's performance in grain boundary detection utilizing type-2 fuzzy logic in SEM images. These measurements shed light on the algorithm's reliability, precision, recall, and general efficacy in identifying grain boundaries (Glushkov et al., 2022). The crossroads over Union (IoU), usually referred to as the Jaccard Index, is one evaluation metric that is frequently employed. It calculates the degree to which the ground reality markings and the observed grain boundaries overlap. The intersection of the union (IOU) between the reference and detected boundaries is determined as the ratio of the intersection region to the union area (Mao et al., 2022). A higher IoU value indicates excellent concordance between the identified borders and the surface reality.

5.4.Grain size statistical analysis and Characterization

The findings and analysis summarize the grain boundary criteria for every image in the dataset and are displayed in figures above. Statistical measurements like average, standard deviation, and confidence intervals might be produced to give a thorough insight into the codes' efficacy throughout the entire dataset (Glushkov et al., 2022). An in-depth examination of the type-2 fuzzy logic-based grain boundary detection technique is possible by examining assessment criteria and visual comparisons (Ali et al., 2022). It aids in understanding the codes' advantages and disadvantages, pointing out potential areas for development, and validating the algorithm's performance compared to existing truth annotations. Additionally, qualitative and quantitative evaluations thoroughly assess the algorithm's performance in precisely identifying grain boundaries in SEM images.

Materials research and engineering have devoted much attention to autonomous determining grain sizes. These approaches automate the procedures of grain size examination, doing away with the necessity for manual measurements by utilizing cutting-edge algorithms for image processing and computational instruments. These methods can precisely determine the diameters of every grain in a substance test by scrutinizing SEM pictures or other microscope images (Glushkov et al., 2022). Combining type-2 fuzzy logic with feature extraction and picture segmentation algorithms is one efficient method. This makes it possible to recognize and distinguish between individual grains in a picture automatically (Chen et al., 2022). After the grains have been divided into segments, the dimensions can be calculated based on other factors such as area, boundaries, or analogous diameter. The standard deviation of grain size, the distribution of size, and other important grain size properties are subsequently determined using statistical analysis.

A graph of Error (%) against picture identification can be drawn to assess how well the independent grain size measurement was performed. Each information point on the graph represents a particular image or specimen (Mao et al., 2022). The independent grain size evaluation approach can be further improved and refined by examining patterns and trends in the graph to find any differences or inconsistencies in the precision of the measurements.

5.5. Discussion on the Strengths and Limitations

5.5.1. Strengths

The type-2 fuzzy logic-based method for detecting grain boundaries in SEM images has many significant advantages, increasing its effectiveness and practical applicability. The algorithm's capacity to deal with the degree of ambiguity and inaccuracies that grain boundary identification entails is a major plus. In contrast to conventional approaches, the algorithm's type-2 fuzzy logic can capture the hazy borders and changes seen in SEM pictures, enabling more precise and reliable grain boundary detection (Chen et al., 2023). Complex materials with uneven grain boundary structures or intensity changes benefit the most from this strength. The algorithm's flexibility and adaptability are another asset. Researchers can modify relationships and fuzzy guidelines to adapt the algorithm to particular SEM images and grain boundary features (Li et al., 2022). Thanks to its flexibility, users can use their domain knowledge and skills to improve the algorithm's performance.

Additionally, the type-2 fuzzy logic approach's openness and interpretability enable greater comprehension and algorithmic improvement, resulting in more well-informed choices for grain boundary analysis problems. The proposed method also gains from picture preprocessing methods like noise elimination and hue enhancement. By reducing the effect of noise and boosting the apparent intensity of grain boundaries, these preprocessing techniques enhance the resolution of the SEM images (Peng et al., 2022). The algorithm can function better and more frequently across various SEM pictures with varied qualities and characteristics by solving common issues in SEM photography. The strengths of the suggested method increase its potential for use in various fields, such as the science of materials, metalworking, and biological research (Glushkov et al., 2022). The method provides a statistical evaluation of grain dimension, texture, and other features, improving an understanding of material performance and behavior.

5.5.2. Weaknesses

The computation complexity of the technique is one of its main flaws. When interpreting large-scale SEM pictures or vast datasets, applying fuzzy logic and including several preprocessing stages might add to the computational overhead. This can affect the algorithm's scalability and real-time performance, especially in applications that call for rapid evaluation or processing of many photos (Wang et al., 2022). The algorithm's dependence on accurate parameter adjustment is another drawback. Selecting suitable membership operations, fuzzy regulations, and threshold quantities significantly impacts the type-2 fuzzy reasoning approach's effectiveness (Mao et al., 2022). Determining the ideal variables for various grain boundary features and SEM photographs can take some effort and judgment. The algorithm may also be less reliable when used with various information or unusual data because of its susceptibility to parameter fluctuations.

The accuracy of recognizing grain boundaries in SEM images with low signal-to-noise ratios or complicated backgrounds may also provide difficulties for the suggested method. Erroneous positive or inaccurate negative findings may result from the existence of noise or disruption and also from the overlap of characteristics. Although preprocessing methods try to reduce these problems, they might not always be enough to fully get around the constraints of noisy or congested SEM pictures (Mao et al., 2022). More so, the effectiveness of the suggested approach depends on the caliber and exactness of the basis for truth annotations utilized for evaluation (Glushkov et al., 2022). Due to the subjective nature of hand annotation, biases, and inconsistencies are likely introduced, which could affect how well the algorithm performs.

Reliable evaluation requires guaranteeing exceptional authentic annotations and reducing human mistakes in the annotation process.

6. CONCLUSION

6.1.Summary of the research

The study proposed a type-2 fuzzy logic-based grain boundary detection method in SEM images. The goal was to solve the difficulties brought on by the uncertainty and imprecision inherent in jobs involving grain boundary identification. The study proved the efficiency and potential of the suggested strategy through a thorough, step-by-step description of the algorithm's execution in MATLAB. The study highlighted the type-2 fuzzy reasoning algorithm's benefits in capturing complex grain boundary structures and changes. The algorithm demonstrated improved adaptability and flexibility by adding uncertainty and imprecision compared to conventional approaches. Researchers could modify fuzzy rules and membership functions to adapt the algorithm to particular SEM images and grain boundary features.

The results showed how reliable and accurate the algorithm was in spotting grain boundaries, regardless of the face of noise or anomalies. Equally, the research proposes opportunities for additional research to improve the algorithm's effectiveness and applicability in various circumstances while noting some limits, such as computing cost, parameter adjustment, and sensitivity to noise. This study contributes slightly to the discipline of grain boundary recognition in SEM images by developing a new method that uses type-2 fuzzy logic. The results show the methodology's potential to enhance grain boundary studies by enabling quantitative evaluations of grain size, form, and other physical attributes.

It is impossible to exaggerate the significance of fuzzy logic in grain line identification since it provides essential benefits for dealing with this activity's inherent uncertainty and imprecision. Beyond the conventional binary approach, fuzzy logic offers a framework that enables the representation and processing of hazy and ambiguous data. Fuzzy logic offers a more effective and realistic method for conveying the intricate nature of grain structures in the framework of grain border recognition in SEM images, where borders may show changes, abnormalities, or overlapping features. The method can accurately describe the distinctions and levels of membership connected with grain boundaries by adding fuzzy relationships and fuzzy rules. As a result, borders can be categorized with varying levels of confidence rather than merely as binary distinctions, enabling a more sophisticated and flexible detection method.

Additionally, the interpretability and transparency of fuzzy logic are crucial for grain boundary analysis. Unlike black-box machine learning algorithms, fuzzy logic offers a rule-based framework that enables researchers to comprehend and analyze the decision-making process. This increases the researchers' trust in the findings and promotes information sharing and teamwork among subject matter experts. Researchers may improve and adjust the method by including domain expertise due to the comprehension of fuzzy logic, leading to superior grain boundary identification suited to certain materials and applications. Fuzzy logic can handle imperfect and ambiguous input, adapt to complicated grain structures, and improve interpretability, which are significant benefits of its usage in grain boundary detection. Using fuzzy logic in the suggested method demonstrates its significance in raising the precision and dependability of grain boundary recognition in SEM images.

6.2. Potential future direction

The research on type-2 fuzzy logic-based grain boundary detection in SEM images paves the way for future paths that could improve the suggested strategy and broaden its usefulness. Integrating type-2 fuzzy logic with deep learning methods like convolutional neural networks

(CNNs) is one area that needs more research. The comprehensibility and flexibility of fuzzy logic can be combined with the feature extraction skills of CNNs to increase the accuracy and robustness of grain boundary detection. The inclusion of specific domain expertise in the algorithm is another exciting direction. The algorithm can be tailored to various types of products and their distinctive grain boundary patterns by utilizing material-specific properties and information from domain specialists. Additionally, contrasting the outcomes with those of other grain boundary detection techniques already in use on other datasets can reveal important details about the advantages and disadvantages of the suggested strategy. Future research can concentrate on improving the method for the immediate processing of SEM images to satisfy the requirement for real-time analysis.

REFERENCES

Ali, A., Chiang, Y. W., & Santos, R. M. (2022). X-ray diffraction techniques for mineral characterization: A review of the fundamentals, applications, and research directions for engineers. *Minerals*, *12*(2), 205. https://www.mdpi.com/1485958

Angili, S. N., Morovvati, M. R., Kardan-Halvaei, M., Saber-Samandari, S., Razmjooee, K., Abed, A. M., ... & Khandan, A. (2023). Fabrication and finite element simulation of antibacterial 3D printed Poly L-lactic acid scaffolds coated with alginate/magnesium oxide for bone tissue regeneration. *International Journal of Biological Macromolecules*, *224*, 1152-1165. https://www.sciencedirect.com/science/article/pii/S0141813022024424

Chen, S., Qin, L., Xie, L., Yu, Q., Chen, Y., Chen, T., ... & Xie, J. (2022). Physicochemical characterization, rheological and antioxidant properties of three alkali-extracted polysaccharides from mung bean skin. *Food Hydrocolloids*, *132*, 107867. https://www.sciencedirect.com/science/article/pii/S0268005X22003873

Chen, X., Fan, L., Ji, D., & Lin, P. (2023). The capability of coupled fuzzy logic and adaptive neural network in the formability prediction of steel sheets. *Waves in Random and Complex Media*, 1-19. https://www.tandfonline.com/doi/abs/10.1080/17455030.2022.2162154?casa_token=bvKsCxSp4R4AAAAA:RMnh2ACo9lP7LEjhNx_vSCbvokigSOoQrbg3h8EREc6cAaXtbvFxO_jtsQy_x977El8p96-06D4wiXIA

Dhas, J. E. R., Lewise, K. A. S., & Laxmi, G. (2022). Submerged arc welding process parameter prediction using predictive modelling techniques. *Materials Today: Proceedings*, *64*, 402-409. https://www.sciencedirect.com/science/article/pii/S2214785322029285

Fatimah, S., Ragadhita, R., Al Husaeni, D. F., & Nandiyanto, A. B. D. (2022). How to calculate crystallite size from x-ray diffraction (XRD) using the Scherrer method. *ASEAN Journal of Science and Engineering*, *2*(1), 65-76. https://www.academia.edu/download/83983534/15948.pdf

Glushkov, E. I., Chiginev, A. V., Kuzmin, L. S., & Revin, L. S. (2022). A broadband detector based on series YBCO grain boundary Josephson junctions. *Beilstein Journal of Nanotechnology*, *13*(1), 325-333. https://www.beilstein-journals.org/bjnano/articles/13/27

Goktas, A., Modanlı, S., Tumbul, A., & Kilic, A. (2022). Facile synthesis and Characterisation of ZnO, ZnO: Co, and ZnO/ZnO: Co nano rod-like homojunction thin films: Role of crystallite/grain size and microstrain in photocatalytic performance. *Journal of Alloys and Compounds*, *893*, 162334. https://www.sciencedirect.com/science/article/pii/S0925838821037440

Jiang, F., Du, C., Zhao, N., Jiang, W., Yu, X., & Du, S. K. (2022). Preparation and Characterization of quinoa starch nanoparticles as quercetin carriers. *Food Chemistry*, *369*, 130895. https://www.sciencedirect.com/science/article/pii/S0308814621019014

Khalafe, W. H., Sheng, E. L., Bin Isa, M. R., Omran, A. B., & Shamsudin, S. B. (2022). The Effect of Friction Stir Welding Parameters on the Weldability of Aluminum Alloys with Similar and Dissimilar Metals. *Metals*, *12*(12), 2099. https://www.mdpi.com/article/10.3390/met12122099

Li, C., Hu, H., Yang, M. F., Pei, Z. Y., Zhou, Q., Ren, X., ... & Ouyang, Z. (2022). Characteristics of the lunar samples returned by the Chang'E-5 mission. *National science review*, *9*(2), nwab188. https://academic.oup.com/nsr/article-abstract/9/2/nwab188/6397035

Li, L., Li, X., Sun, Y., & Xie, Y. (2022). Rational design of electrocatalytic carbon dioxide reduction for a zero-carbon network. *Chemical Society Reviews*. https://pubs.rsc.org/en/content/articlehtml/2022/cs/d1cs00893e

Li, X., Heidari, A., Nourbakhsh, S. M., Mohammadi, R., & Semiromi, D. (2022). Design and fabrication of elastic two-component polymer-metal disks using a 3D printer under different loads for the lumbar spine. *Polymer Testing*, *112*, 107633. https://www.sciencedirect.com/science/article/pii/S014294182200157X

Mao, F., Li, J., Yang, J., Liu, Z., & Zhang, M. (2023). A Grain Boundary Defects Detection Algorithm with Improved Localisation Accuracy Based on EfficientDet. *Automatic Control and Computer Sciences*, *57*(1), 81-92. https://link.springer.com/article/10.3103/s0146411623010078

Panicker, S. (2023). Knowledge-based Modelling of Additive Manufacturing for Sustainability Performance Analysis and Decision Making. https://trepo.tuni.fi/handle/10024/145266

Peng, G., Sun, Y., Zhang, Q., Yang, Q., & Shen, W. (2022). A collaborative design platform for new alloy material development. *Advanced Engineering Informatics*, *51*, 101488. https://www.sciencedirect.com/science/article/pii/S1474034621002378

Sifan, L. (2022). *Two-Dimensional Materials-Based Synapic Devices for Neuromorphic/In-Memory Computing Applications* (Doctoral dissertation, National University of Singapore (Singapore)).

https://search.proquest.com/openview/79c6b9bf9edb6ae8ab1e70ebadac6ffa/1?pq-origsite=gscholar&cbl=2026366&diss=y

Song, R., Ponge, D., Kaspar, R., & Raabe, D. (2022). Grain boundary characterization and grain size measurement in an ultrafine-grained steel. *International Journal of Materials Research*, *95*(6), 513-517. https://www.degruyter.com/document/doi/10.1515/ijmr-2004-0101/html

Wang, R., Tang, Y., Lei, Z., Ai, Y., Tong, Z., Li, S., ... & Bai, S. (2022). Achieving high strength and ductility in nitrogen-doped refractory high-entropy alloys. *Materials & Design*, *213*, 110356. https://www.sciencedirect.com/science/article/pii/S0264127521009114

Yu, H., Marchek, J. E., Adair, N. L., & Harb, J. N. (2022). Characterization of minerals and coal/mineral associations in pulverized coal. In *The impact of ash deposition on coal fired plants* (pp. 361-372). Routledge. https://www.taylorfrancis.com/chapters/edit/10.1201/9780203736616-33/characterization-minerals-coal-mineral-associations-pulverized-coal-yu-marchek-adair-harb

Zhang, H., Zhang, X., Zhang, G., Dong, K., Deng, X., Gao, X., ... & Yang, M. (2022). Size, morphology, and composition of lunar samples returned by Chang'E-5 mission. *Science China Physics, Mechanics & Astronomy*, *65*, 1-8. https://link.springer.com/article/10.1007/s11433-021-1818-1

McRae, Michelle & Julien, Philippe & Salvo, Santino & Goroshin, Samuel & Frost, David & Bergthorson, Jeffrey. (2018). Stabilized, flat iron flames on a hot counterflow burner. Proceedings of the Combustion Institute. 37. 10.1016/j.proci.2018.06.134**.**

Brígido, Baldomero & García, Laura & Mugas, Growene & Penco, Fernando & Salazar, J.M & Criado, Antonio. (2019). Softening and Loss of Mechanical Strength of Carbon Steels for Extended Periods of Time at Room Temperature. Application to the Specific Case of the Santa María de la Asunción de Laredo Steel Chain Corresponding author.

Banerjee, K. & Rollett, A.D.. (2003). Microstructure and crystallization texture of a low carbon strip cast steel. Ironmaking and Steelmaking. 30. 62-68.

APPENDICES

Matlab Code

```
function varargout = edge_detectors(varargin)
% EDGE_DETECTORS MATLAB code for edge_detectors.fig

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @edge_detectors_OpeningFcn, ...
    'gui_OutputFcn',  @edge_detectors_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before edge_detectors is made visible.
function edge_detectors_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to edge_detectors (see VARARGIN)

% Choose default command line output for edge_detectors
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes edge_detectors wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = edge_detectors_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
% --- Executes on button press in load.
function load_Callback(hObject, eventdata, handles)
% hObject    handle to load (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% load the images
global im Igray Gx Gy Gmag Gdir Idouble
[filename, user_canceled] = imgetfile();
if user_canceled
    msgbox(sprintf('Error: no image is selected'),'Error','Error');
    return
end
im = imread(filename);
% input image is grayscale or not. If not then convert grayscale and double
% presision
if size(im,3)==3
    Igray = rgb2gray(im);
    Idouble = im2double(Igray);
else
    Igray = im; % for compatibility
    Idouble = im2double(im);
end

[Gx,Gy] = imgradientxy(Idouble,'Sobel'); % directional gradient of the image
% Calculate the gradient magnitude and direction using the directional gradients
[Gmag, Gdir] = imgradient(Gx, Gy);
axes(handles.imgshow);
imshow(im);

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
% hObject    handle to reset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% reset button
global im
axes(handles.imgshow);
imshow(im)

% --- Executes on button press in graylevel.
function graylevel_Callback(hObject, eventdata, handles)
% hObject    handle to graylevel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% converts into the gray scale
global Igray im_write
im_write = Igray; % for 'save to file' option
axes(handles.imgshow);
imshow(Igray);

% --- Executes on button press in histogram.
function histogram_Callback(hObject, eventdata, handles)
% hObject    handle to histogram (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% shows the histogram of the image
global Igray
axes(handles.imgshow);
imhist(Igray)

% --- Executes on button press in canny.
function canny_Callback(hObject, eventdata, handles)
% hObject    handle to canny (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% using canny edge detector
% Apply Canny Edge Detector to the image
global Igray im_write
BW_canny = edge(Igray,'canny');
im_write = BW_canny; % for 'save to file' option
axes(handles.imgshow);
imshow(BW_canny);

% --- Executes on button press in sobel.
function sobel_Callback(hObject, eventdata, handles)
% hObject    handle to sobel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Sobel operator
global Igray im_write
BW_sobel = edge(Igray,'Sobel');
im_write = BW_sobel;% for 'save to file' option
axes(handles.imgshow);
imshow(BW_sobel);

% --- Executes on button press in prewitt.
function prewitt_Callback(hObject, eventdata, handles)
% hObject    handle to prewitt (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Prewitt operator
global Igray im_write
BW_prewitt = edge(Igray,'Prewitt');
im_write = BW_prewitt ;% for 'save to file' option
axes(handles.imgshow);
imshow(BW_prewitt);

% --- Executes on button press in roberts.
function roberts_Callback(hObject, eventdata, handles)
% hObject    handle to roberts (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Igray im_write
BW_roberts = edge(Igray,'Roberts');
im_write = BW_roberts ; % for 'save to file' option
axes(handles.imgshow);
imshow(BW_roberts);

% --- Executes on slider movement.
```

```
function threshold_Callback(hObject, eventdata, handles)
% hObject    handle to threshold (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
global Igray im im_write
thres = get(hObject,'Value'); % gets threshold value
set(handles.text3,'String',num2str(thres));
level = graythresh(im) ; % using OTSU method
set(handles.text4,'String',num2str(level*255));
threshed = Igray >= thres ; % if the input value is greater than thres value sets as
1
im_write = threshed ; % for 'save to file' option
axes(handles.imgshow);
imshow(threshed);

% --- Executes during object creation, after setting all properties.
function threshold_CreateFcn(hObject, eventdata, handles)
% hObject    handle to threshold (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes during object creation, after setting all properties.
function text3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% --- Executes during object deletion, before destroying properties.
function text3_DeleteFcn(hObject, eventdata, handles)
% hObject    handle to text3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in log.
function log_Callback(hObject, eventdata, handles)
% hObject    handle to log (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Finds edges by looking for zero-crossings after filtering I with
% a Laplacian of Gaussian filter.
global Igray im_write
BW_log = edge(Igray,'Log');
im_write = BW_log;  % for 'save to file' option
axes(handles.imgshow);
imshow(BW_log);

% --- Executes on button press in inverse.
```

```
function inverse_Callback(hObject, eventdata, handles)
% hObject    handle to inverse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Igray im_write
inverse = 255-Igray;
im_write = inverse ; % for 'save to file' option
axes(handles.imgshow);
imshow(inverse);

% --- Executes on button press in Gx.
function Gx_Callback(hObject, eventdata, handles)
% hObject    handle to Gx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% directional gradient of image to locate breaks in uniform regions
global Gx im_write
axes(handles.imgshow);
im_write = Gx;  % for 'save to file' option
imshow (Gx,[]);

% --- Executes on button press in Gy.
function Gy_Callback(hObject, eventdata, handles)
% hObject    handle to Gy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Gy im_write
axes(handles.imgshow);
im_write = Gy;  % for 'save to file' option
imshow (Gy,[]); % [] means that the display range is min(Gy(:)) to max(Gy(:))


% --- Executes on button press in exit.
function exit_Callback(hObject, eventdata, handles)
% hObject    handle to exit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
clear all;
close all;

% --- Executes on button press in save.
function save_Callback(hObject, eventdata, handles)
% hObject    handle to save (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im_write
imwrite(im_write,'result.png');

% --- Executes on button press in kirsch.
function kirsch_Callback(hObject, eventdata, handles)
% hObject    handle to kirsch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% The Kirsch operator
global Igray im_write kirsch_im
% Igray is grayscale of the original image
im_double=double(Igray);
% Eight directions Kirsch edge masks. below masks are gettting from
% rotation of the one mask.
g1=[5,5,5; -3,0,-3; -3,-3,-3]; % South
g2=[5,5,-3; 5,0,-3; -3,-3,-3]; % Southeast
g3=[5,-3,-3; 5,0,-3; 5,-3,-3]; % East
g4=[-3,-3,-3; 5,0,-3; 5,5,-3]; %
g5=[-3,-3,-3; -3,0,-3; 5,5,5];
g6=[-3,-3,-3; -3,0,5;-3,5,5];
g7=[-3,-3,5; -3,0,5;-3,-3,5];
g8=[-3,5,5; -3,0,5;-3,-3,-3];
% filtering with Kirsch mask
% edges in all the direction
%     Each mask respondsmaximally to an edge oriented in a particular general
%     direction. The maximum value over all eight orientations
%     is the output value for the edge magnitude image.
x1=imfilter(im_double,g1,'replicate');
x2=imfilter(im_double,g2,'replicate');
x3=imfilter(im_double,g3,'replicate');
x4=imfilter(im_double,g4,'replicate');
x5=imfilter(im_double,g5,'replicate');
x6=imfilter(im_double,g6,'replicate');
x7=imfilter(im_double,g7,'replicate');
x8=imfilter(im_double,g8,'replicate');

y1=max(x1,x2);
y2=max(y1,x3);
y3=max(y2,x4);
y4=max(y3,x5);
y5=max(y4,x6);
y6=max(y5,x7);
kirsch_im=max(y6,x8); % result image
axes(handles.imgshow);
% image mapping in the interval of 0 to 255
kirsch_im = (255)*((kirsch_im-min(kirsch_im(:)))/(max(kirsch_im(:))-
min(kirsch_im(:))));
im_write = kirsch_im;  % for 'save to file' option
imshow (kirsch_im,[]);

% --- Executes on button press in mag_of_grad.
function mag_of_grad_Callback(hObject, eventdata, handles)
% hObject    handle to mag_of_grad (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im_write Gmag
axes(handles.imgshow);
im_write = Gmag;  % for 'save to file' option
imshow(Gmag,[])

% --- Executes on slider movement.
function sliderkirsch_Callback(hObject, eventdata, handles)
% hObject    handle to sliderkirsch (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
global im_write kirsch_im
thres = get(hObject,'Value'); % gets threshold value
set(handles.text3,'String',num2str(thres));
threshed = kirsch_im >= thres ; % if the input value is greater than thres value sets
as 1
im_write = threshed ; % for 'save to file' option
axes(handles.imgshow);
imshow(threshed);

% --- Executes during object creation, after setting all properties.
function sliderkirsch_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sliderkirsch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on button press in dir_of_grad.
function dir_of_grad_Callback(hObject, eventdata, handles)
% hObject    handle to dir_of_grad (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im_write Gdir
axes(handles.imgshow);
im_write = Gdir;  % for 'save to file' option
imshow(Gdir,[])

% --- Executes on button press in twobytwosliding.
function twobytwosliding_Callback(hObject, eventdata, handles)
% hObject    handle to twobytwosliding (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im_write Idouble Gmag_SlidingMethod

% Mamdani FIS will be used to make decision on set edge or background
fis = newfis('2x2window');
% 2x2 sliding window has 4 pixel elemets
% first input
fis = addvar(fis, 'input', 'p1', [0 255]); % for gray level 0 to 255
fis = addmf(fis, 'input', 1, 'black', 'trimf', [0 0 255]);
fis = addmf(fis, 'input', 1, 'white', 'trimf', [0 255 255]);
% second input
fis = addvar(fis, 'input', 'p2', [0 255]); % for gray level 0 to 255
fis = addmf(fis, 'input', 2, 'black', 'trimf', [0 0 255]);
fis = addmf(fis, 'input', 2, 'white', 'trimf', [0 255 255]);
% third input
fis = addvar(fis, 'input', 'p3', [0 255]); % for gray level 0 to 255
```

```
fis = addmf(fis, 'input', 3, 'black', 'trimf', [0 0 255]);
fis = addmf(fis, 'input', 3, 'white', 'trimf', [0 255 255]);
% fourth input
fis = addvar(fis, 'input', 'p4', [0 255]); % for gray level 0 to 255
fis = addmf(fis, 'input', 4, 'black', 'trimf', [0 0 255]);
fis = addmf(fis, 'input', 4, 'white', 'trimf', [0 255 255]);
% output
fis = addvar(fis, 'output', 'pout', [0 255]); % for gray level 0 to 1
fis = addmf(fis, 'output', 1, 'black', 'trimf', [0 0 70]);
fis = addmf(fis, 'output', 1, 'edge', 'trimf', [90 130 170]);
fis = addmf(fis, 'output', 1, 'white', 'trimf', [200 255 255]);

% if there is 3B and 1W in the window or vice versa, set the output as edge
% 3B(3W) and 1W(1B) set the output as E
% if there is 2W and 2B in the window, set the output as edge
rules = [1 1 1 1 1 1 1; % B B B B -> B
    1 1 1 2 2 1 1; % B B B W -> E
    1 1 2 1 2 1 1; % B B W B -> E
    1 1 2 2 2 1 1; % B B W W -> E
    1 2 1 1 2 1 1; % B W B B -> E
    1 2 1 2 2 1 1; % B W B W -> E
    1 2 2 1 2 1 1; % B W W B -> E
    1 2 2 2 3 1 1; % B W W W -> W
    2 1 1 1 1 1 1; % W B B B -> E
    2 1 1 2 2 1 1; % W B B W -> E
    2 1 2 1 2 1 1; % W B W B -> E
    2 1 2 2 2 1 1; % W B W W -> E
    2 2 1 1 2 1 1; % W W B B -> E
    2 2 1 2 2 1 1; % W W B W -> E
    2 2 2 1 2 1 1; % W W W B -> E
    2 2 2 2 3 1 1];% W W W W -> W
fis = addrule(fis, rules);
% % optional
% % rules of the system
% showrule(fis)
% figure
% plotfis(fis)
% %  Open Rule Viewer
% ruleview(fis)
% %  Open Surface Viewer
% surfview(fis)
[m, n] = size(Idouble);
Iout = zeros(m-1, n-1);
for i=1: m-1
    for j=1: n-1
        sub_window = Idouble(i:i+1,j:j+1);
        p1(i,j) = sub_window(1,1);
        p2(i,j) = sub_window(1,2);
        p3(i,j) = sub_window(2,1);
        p4(i,j) = sub_window(2,2);
    end
end

for i = 1:size(p1,1)
    Iout(i,:) = evalfis([p1(i,:); p2(i,:); p3(i,:); p4(i,:)],fis);
```

```
end

% First derivative of the image
[Gmag_SlidingMethod, ~] = imgradient(Iout);
Gmag_SlidingMethod = (255)*((Gmag_SlidingMethod-
min(Gmag_SlidingMethod(:)))/(max(Gmag_SlidingMethod(:))-min(Gmag_SlidingMethod(:))));
% map into the range of 0 to 255
% default threshold value is 30
thresh = 30; % threshold value for output of the 'evalfis'
% threshold value can be changed for the best reult
axes(handles.imgshow);
Ibin = Gmag_SlidingMethod<thresh; % binary
im_write = 1- Ibin;  % for 'save to file' option
imshow(1-Ibin,[])

% --- Executes on button press in gradientmethod.
function gradientmethod_Callback(hObject, eventdata, handles)
% hObject    handle to gradientmethod (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Idouble Igray im_write Ieval
[m, n] = size(Idouble);
% First step:
% Gradient of an image and mapping into 0 to 100

[Gx,Gy] = imgradientxy(Idouble,'Sobel'); % directional gradient of the image
% Calculate the gradient magnitude and direction using the directional gradients
[Gmag, Gdir] = imgradient(Gx, Gy);

% mapping Gmag into the 0 to 100
Gmag = (100)*((Gmag-min(Gmag(:)))/(max(Gmag(:))-min(Gmag(:)))); % map into the range
of 0 to 100

% Second step:
% SD of each pixel and mapping into 0 to 100
SD = stdfilt(Igray);
% Brighter pixels in the filtered image correspond to neighborhoods in the
% original image with larger standard deviations.

% mapping Isd into the 0 to 100
SD = (100)*((SD-min(SD(:)))/(max(SD(:))-min(SD(:)))); % map into the range of 0 to
100

% Third step
% Create a fuzzy inference system (FIS)using two inputs
% There will two inputs and each of them will have three fuzzy sets that
% describe the edginess

fis = newfis('EdgeDetection');
% interval mapping
% First Input
% Specify the image magnitude of the gradient
fis = addvar(fis,'input','Gmag',[0 100]);
fis = addmf(fis,'input',1,'low','trapmf',[0 0 20 40]);
fis = addmf(fis,'input',1,'medium','trapmf',[20 40 60 80]);
```

```
fis = addmf(fis,'input',1,'high','trapmf',[60 80 100 100]);

title('using gradient magnitude to create first input variable')
% Standart Deviation of each pixel
fis = addvar(fis,'input','SD',[0 100]);
fis = addmf(fis,'input',2,'low','trapmf',[0 0 20 40]);
fis = addmf(fis,'input',2,'medium','trapmf',[20 40 60 80]);
fis = addmf(fis,'input',2,'high','trapmf',[60 80 100 100]);
title('using Standart Deviatione to create second input variable')
% output of the FIS
fis = addvar(fis,'output','edginess',[0 1]);
fis = addmf(fis,'output',1,'black','trapmf',[0 0 .25 .5]);
fis = addmf(fis,'output',1,'gray','trimf',[.25 .5 .75]);
fis = addmf(fis,'output',1,'white','trapmf',[.5 .75 1 1]);

% Fourth step:
% Specify and apply rules to the system
rules = [1 1 1 1 1; % L L -> L
        1 2 1 1 1; % L M -> L
        1 3 2 1 1; % L H -> M
        2 1 1 1 1; % M L -> L
        2 2 3 1 1; % M M -> M
        2 3 3 1 1; % M H -> H
        3 1 2 1 1; % H L -> M
        3 2 3 1 1; % H M -> H
        3 3 3 1 1];% L H -> H

fis = addrule(fis, rules);
% Evaluate the output of the edge detector for each row of pixels in I using
% corresponding rows of Gmag and SD as inputs.
Ieval = zeros(m, n);
for i = 1:m
    Ieval(i,:) = evalfis([(Gmag(i,:)); (SD(i,:));]',fis);
end
Ieval = (255)*((Ieval-min(Ieval(:)))/(max(Ieval(:))-min(Ieval(:)))); % map into the
range of 0 to 255

axes(handles.imgshow);
Ithre = Ieval < 30; % make binary with threshold value (30)
im_write = 1 - Ithre;  % for 'save to file' option
imshow(1-Ithre,[])


% --- Executes on slider movement.
function slider15_Callback(hObject, eventdata, handles)
% hObject    handle to slider15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
global im_write Gmag_SlidingMethod
thres = get(hObject,'Value'); % gets threshold value
set(handles.text3,'String',num2str(thres));
```

```
threshed = Gmag_SlidingMethod <= thres ; % if the input value is greater than thres
value sets as 1
im_write = 1- threshed ; % for 'save to file' option
axes(handles.imgshow);
imshow(1-threshed,[]);

% --- Executes during object creation, after setting all properties.
function slider15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function sliderforgradient_Callback(hObject, eventdata, handles)
% hObject    handle to sliderforgradient (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
global im_write Ieval
thres = get(hObject,'Value'); % gets threshold value
set(handles.text3,'String',num2str(thres));
threshed = Ieval <= thres ; % if the input value is greater than thres value sets as
1
im_write = 1- threshed ; % for 'save to file' option
axes(handles.imgshow);
imshow(1-threshed,[]);

% --- Executes during object creation, after setting all properties.
function sliderforgradient_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sliderforgradient (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```