COS700 Research Proposal

Automated Design of GA for Image Segmentation

Student number: u15209190

Supervisor(s):
Dr Thambo Nyathi
t.nyathi@cs.up.ac.za

Abstract

Automation is a familiar process used to reduce user interaction to improve efficiency and productivity. The use of machine learning algorithms to automate such a process has been widely adopted. However, in some problem domains such as image segmentation, setting up a machine learning algorithm requires crucial user interaction to configure. In genetic algorithms, these configurations can be biased towards the effort given by the user. This paper aims to define an automation function for configurations of genetic algorithms applied to image segmentation. The paper proposes the use of genetic algorithms to automate the configurations, that is autoGA to automate genetic algorithm.

Keywords:

Automation, Genetic Algorithm, Image segmentation, Threshold, Parameter Turning

1 Introduction

The development of optical instruments such as cameras has allowed the capturing of information in variation to the traditional document-based information. Information captured in images is immense and therefore requires a series of analytical tools to extract. The information can vary from counting the number of people or objects present in an image to detect cancerous cells in x-ray images. The successful extraction of such information is very important to the future of research in various fields such as health, in medically diagnosing patients. Different Techniques such as image segmentation have been used to extract such information.

Image segmentation is an image processing technique in computer vision that aims to detect patterns within an image using the variation of color pixels. The success of the process is measured by evaluation functions that authenticate the validity of these patterns. Thresholding is one such technique that demonstrates image segmentation. However, thresholding can lag in computation when dealing with the extraction of multiple patterns within images. Therefore Genetic algorithms are applied to image segmentation problems. While Genetic algorithms are a better alternative for multilevel, i.e. multiple patterns, and thresholding they also have criticism. This is due to parameter turning that is required to get the best configuration of the genetic algorithm.

Thus this paper attempts to automate the process of parameter turning by applying an automation function. Genetic algorithms i.e. autoGA, are chosen as the automation function for the genetic algorithm on image segmentation. This process aims to reduce the complexity of the configuration of genetic algorithms.

The rest of the paper is divided into 4 sections. Section 2 highlights the problem statement addressed in this paper. Section 3 covers the literature that contributes to the problem statement and proposed solution. Lastly, 4 highlights the proposed solution and the project plan in 5.

2 Problem Statement

Genetic algorithms (GA) have been used to solve optimization problems. They can easily be modelled and configured in various ways to solve specific problems from various problem domains. Image segmentation is one such problem where GAs have been applied. Generally, the process of configuring GAs to solve a specific problem is manual. This is also the case in image segmentation. As it is widely accepted that the effectiveness of an EA is directly influenced by its configuration [14], it therefore implies that the effectiveness of GA in image segmentation is also influenced by the configuration. It has been shown that the manual design approach is not only laborious and tedious but it is also fraught with vulnerabilities such as human bias. A substantial amount of work has been done in using GAs to perform multilevel thresholding, however the approaches have all used manual design in the configuration of the GA. It therefore stands to reason that the approaches also remain susceptible to the weaknesses of manual design.

This research will explore the automated configurations of GA, for multilevel thresholding.

2.1 Aim and Objectives

2.2 Aim

The aim of this study is to investigate the feasibility of automating the design of genetic algorithm for multilevel thresholding.

2.3 Objectives

In order to achieve this aim the following objectives have to be met

1. To automate the design of GA using multi-point and single-point search.
2. To compare the effectiveness of the automated GA for multilevel thresholding to a manually configured genetic algorithm on specific image problem instances.
3. To evaluate the best approach to automate the design of GA.

3 Literature Study

3.1 Evolutionary algorithm

Evolutionary algorithms (EA) are a family of algorithms inspired by nature, specifically the evolution of living organisms, as theorised by Charles Darwin [4]. The algorithms follow processes that imitate the evolution of living organisms. The generalised algorithm cycle is highlighted in Algorithm 1. The algorithm first starts by generation a random population of individuals where each individual is a representation of a problem to be solved. After that the fitness of each individual in the population is evaluated. The fitness of an individual is a measure or function of how well that individual is able to solve the problem at hand. If the objective function (the purpose of the algorithm) is met then the algorithm terminates else individuals of the population are selected to act as parents to generate new offspring for the new population. Selection is performed using a selection method that usually biased towards high fitness. After selection, genetic operators are applied to the parents and offspring are generated. The population is updated after each iteration of the algorithm. Each iteration is known as an evolution.Again the fitness of the offspring is evaluated and if they meet the desired outcome the algorithm terminates or else it loops again until a stopping criterion is met.

Evolutionary algorithms perform multi-point search in the search space. The processes perform exploration and exploitation of the search space. A number of evolutionary algorithms can be found literature such as genetic algorithms, genetic programming [11], differential evolution [22][23] amongst others. Evolutionary algorithms differ in the way they represent individuals and the space

Algorithm 1 Generational evolutionary algorithm

```
Create the initial population
repeat
    evaluate elements of the population
    selection parents
    apply genetic operators to parents
until termination criterion is met
```

they search. However, they all draw inspiration from the principles of natural selection.

In the next section we review genetic algorithms as this is the EA of concern in this study.

3.2 Genetic algorithm

Genetic algorithms are the most widely used forms of EA. Traditionally, genetic algorithm individuals are encoded as fixed length binary notation chromosomes. Each individual represents a solution of the problem being considered. Algorithm 2 is an algorithm flow of GA and is illustrated as follows.

Algorithm 2 Generational genetic algorithm

```
Create an initial random population
repeat
    evaluate individuals of the population
    select parents
    apply crossover
    apply mutation
    update the population
until termination criterion is met
```

3.2.1 Initial population

The initial population is created by randomly initializing a population of chromosomes. The value of each gene is randomly obtained from a set of feasible values that form a solution for a problem. The number chromosomes created is determined by a user defined population *size* parameter. As stated the representation of encoded information within genes is usually binary. Other forms of encoding such as permutation encoding have been used [2]. In binary encoding, chromosomes are strings of 1s and 0s, i.e. a gene is represented by either 1 or 0. For permutation encoding each chromosome is a string of numbers. The type of encoding used is problem dependant.

3.2.2 Evaluation of Individuals

All chromosomes within the population have a fitness value associated with them. The fitness is first evaluated for every new member of the population after the population is initialized, line 3. These fitness values are maintained throughout the generations and are only evaluated for new individual offspring produced through mating, line 7. The fitness function is used to calculate the fitness values. The general approach to the fitness functions to determine how good the individual solves the problem, such that the fitness value is higher for individual chromosomes that produce a better solution to the problem.

3.2.3 Selection

Various selection methods have been applied for selecting parents [5]. An example of selection methods is tournament and fitness proportionate selection methods. In tournament selection, a number (tournament-size) of individuals are randomly selected from the population and the best out of these individuals are selected to become parents. Fitness proportionate selection method uses the fitness of each individual to act as a threshold probability of the individuals being selected as parents, high fitness individuals have a higher chance of being selected as parents.

3.2.4 Regeneration

The mating process involves crossover and mutation. In GA the selected individuals for regeneration are called parents. The selection methods stated in section 3.2.3 are biased towards fitness therefore the parents usually have high fitness. The selected parent undergoes regeneration operations and the first of those is the crossover. The crossover operation selects a set of genes from both parents to form a new individual, called the offspring, line 5. They are various techniques 5 applied to select the crossover point at which the genes are to be selected in the parents[10]. In a single-point crossover, a single point is selected at random within the chromosome length and all the genes that follow the crossover point in both parent chromosomes are swapped to produce 2 offspring. In a two-point crossover, two points are selected at random within the chromosome length and the genes that lie between the two points in both parents are swapped. Lastly, uniform crossover randomly generates several points that lie within the chromosome length at which corresponding genes in both parents are swapped. Crossover is followed by the mutation operation. It is applied to the offspring by randomly selecting one or more genes within the offspring's chromosomes. In binary encoding selected genes are flipped, i.e. 1 changes to 0 and visa versa, and for permutation encoding genes are swapped between selected points within the chromosome. The various techniques used to apply the genetic operators, are highlighted in [12].

3.2.5 Termination

The algorithm stops once the termination criterion has been met. The termination criteria can be the objective function of the algorithm being met or a set number of generations.

3.3 Image segmentation

Images are made up of small grid blocks known as pixels. Each pixel is represented by a numeric value that is associated with a specific color. In image segmentation pixels of digital images are partitioned into multiple segments, i.e. a set of pixels[21]. Segments are created by assigning labels to pixels in the image to which pixels with similar characteristics, i.e. color, are expected to share a label. The segments map out the entire image, i.e. they collectively cover the image. This process is used to locate objects and boundaries in images. There are several image segmentation techniques such as thresholding, edge-based, region-based, watershed, clustering-based, etc. The next section explores the thresholding technique, specifically multilevel thresholding for image segmentation.

3.4 Thresholding

3.4.1 Binary

Thresholding in image segmentation is the process by which greyscale images are translated into black and white images by classifying pixels into either of the two classes separated by the threshold value [17]. That is, given that pixels of a picture lie with in a range $\{1, 2, \ldots, L\}$. Let T be the threshold value that lies within the same range, i.e. $T = \{1, 2, \ldots, L\}$. Pixels below the threshold would translate to black (segment T_{black}) and those above would translate to white (segment T_{white}) or vice versa. Pixels are classified into a segment based on their intensity value. This thresholding is known as binary thresholding due to only having two segmentation classes. An evaluation function is applied to the threshold value to determine its accuracy. Two algorithms have been used to evaluate binary thresholds, that is, Otsu and Kapur.

3.4.1.1 Otsu uses a process of minimizing intra-class intensity variance, or equivalently, by maximizing inter-class variance[15]. That is . . .

let inter-class variance

$$V = (Weight_{black} x Var_{black}) + (Weight_{white} x Var_{white})$$

where *Var* is the variance of the respective classes and *Weight* is the relative number of pixels that belong to a class.

Threshold values are chosen based on the highest difference between the overall class average. That is where the distance between classes is the highest. This method ensures that the threshold captures the different attributes of the images through grouping by similarity.There are other variations of the Otsu algorithm namely the 2-dimensional Otsu method[6] and Minimum Error thresholding[9].

The 2-dimensional Otsu method is mostly applicable in image segmentation applications where noise is presented in the image data. This means that there is little variation in the pixels in the images that potentially differentiate objects in the image, due to poor image quality. To overcome the challenge 2-D Otsu considers the average intensity of the immediate neighboring pixels. Pixels in images are at least closer to other pixels with similar intensity. The minimum Error thresholding method is applicable in image segmentation applications where the unbalanced variation in the normal distributions of the image histogram describes the different segments.

3.4.1.2 Kapur uses a process of entropy-based thresholding[7]. Entropy is an attempt to measure the degree of randomness within a class attribute. Kapur attempts to minimize the disorganization among members of a segment by comparing elements of the segment and measuring how they correlate. That is trying to infer a pattern from a set of data points. Kapur's entropy function segments grayscale images by maximizing the entropy of an image histogram. The entropy function is highlighted in [7].

3.4.2 Multilevel

Traditional thresholding segments images into only two classes black and white. This is known as binary threshold. When images become more complex, that is a lot of information is presented in a single image, two classes are unable to correctly segment image attributes. Multilevel thresholding introduces the use of multiple threshold values. The challenge in multilevel thresholding is determining the ideal number of classes required to best represent the images. This introduces the second component apart from determining the threshold values. Thresholding is a linear single point search algorithm, it iteratively explores the entire search space to determine the threshold values and the ultimate number of classes. This makes multilevel thresholding computational expensive. To archive, the best performance for both computation and segmentation different algorithms is used to optimize multilevel thresholding into a multi-point search. We explore the application of GA to multilevel thresholding.

3.5 Application of GA to multilevel thresholding

Multilevel thresholding introduces the need for an algorithm that can optimize the number of classes required and the different threshold points. GAs have previously been applied to solve the multilevel thresholding problem. The subsequent sub-sections explore use of GA for multilevel thresholding.

3.5.1 Color image segmentation

In color images, pixes are presented as an RGB value i.e a 3-dimensional vector. The combination of the three values makes up a specific visual color based on their vector scale. HSI is an alternative representation of the RGB color model, which closely aligns with the way human vision perceives color-making attributes. In this multilevel thresholding application [1], image pixels are segmented into 3 predefined classes labeled hue, saturation, and intensity (HSI). This application withdraws the complexity of determining the number of segments. The GA is represented as a 12-character string, i.e chromosome, and its objective is to find the lower and upper boundaries of each of the 3 segments.

3.5.2 Multilevel thresholding for segmentation of medical brain images

HSI (hue, saturation, and intensity) is an alternative representation of the RGB color model. In grayscale images, pixels only carry the intensity (I) information, i.e the amount of light. The contrast ranges from black at the weakest intensity to white at the strongest. In this multilevel thresholding application for gray-scale images [13], Kapur is used to maximizing the entropy of the segmented histogram so that each segment has a more centralized distribution. The optimal multilevel thresholding problem is configured as an n-dimensional optimization problem, where n is the number of segments. Each segment covers a range of intensity values, i.e. image pixels. Real-coded GA was used to evolve the number of thresholds and evaluated using a uniformity measure. The value of the uniformity measure u lies between 0 to 1. The value of u closer to 1, indicates the better uniformity in the segmented image and quality of thresholding.

3.5.3 minimum cross entropy threshold selection

Entropy is an attempt to measure the degree of randomness within a class attribute. Cross entropy, a deviation of entropy, attempts to find the solution closest to the desired or expected result by minimizing the difference using prior and posterior probabilities. The minimum cross entropy method selects several thresholds by minimizing the cross-entropy between the original image and the resulting image. In this multilevel thresholding application for gray-scale image [24], a real-coded GA, based on the recursive programming technique, is applied for solving the optimal thresholds T for n segments by minimizing the cross-entropy. The GA is used to optimize the number of iterations for thresholding beyond 2 segments given the number of segments n.

3.5.4 Multilevel automatic thresholding method

An image histogram is a graphical representation of the color distribution in an image. In this multilevel thresholding, application [3], the length of the original histogram is reduced using the wavelet transform. The procedure produces a histogram that contains the overall characteristics of the original histogram. The reduced, lower resolution histogram, accelerate the thresholding process. The GAs string representation is of the same length as the histogram x-axis range, whereby a value of 0 in the string indicates a valley, i.e. a threshold. The number of threshold values is reflected by the number of zeros in the string. The evaluation of the GA is performed by projecting the thresholds onto the original image space.

3.5.5 Quantum-Inspired Multilevel Thresholding

Quantum Genetic Algorithm (QGA) introduces the concept and theory of quantum computing into GA. Individuals in QGA are represented as a quantum-inspired bit instead of the traditional representations. This method puts emphasis on computation cost. In this application of QGA on multilevel image segmentation [26], the Kapur method is applied to obtain the best thresholds T

with maximum information gain, for a given number of segments, by evaluating and comparing entropy values of different combinations of threshold values. Quantum Angle Encoding is applied to the representation, i.e. Quantum bits, to improve efficiency in computing a higher number of thresholds.

3.6 Automated Design

The rapid digitization of information introduced easy access to the processing of historical events. This is presented by applying statistical and machine learning models to the available data to generate intelligence of some sort. The models used to generate such intelligence are not always well-formed. They require critical thinking induced by a trial and error process to archive their desired outcome.

It is without a doubt that humans have limitations to the amount of work they can do and the precise accuracy they can archive in performing different work tasks. The issues of efficiency and accuracy in the task that requires human interaction introduce a problem in the production output. To counter the problem, automation has been applied in various instances. Automation is a procedure that seeks to reduce human interaction by introducing well-calculated mechanical control of repetitive tasks. Some machine learning (ML) algorithms have been used to automate such tasks and procedures to increase efficiency and accuracy. In the subsequent section, we explore the use of ML algorithms for the automation of the manual process.

3.6.1 Automated Design of Classification Algorithms

Classification problems aim to generate classification rules by studying know incidents to predict future events. Genetic programming (GA), a variation of EA, have been previously applied to classification problems. The process of applying GP demonstrate the need for human interaction to configure for optimal outcomes. This configurations include similar genetic parameters discussed in sections 3.2. Unlike GAs, GP are represented by a tree instance instate of a chromosome but have the same operators in the evolution process, i.e. initialization, selection, crossover, mutation and terminations criteria. To eliminate the bias of human error and save time it was demonstrate [16] that Genetic algorithm and grammatical evolution (GE) [19] [25] can be applied to automate the configurations of the GP. The results of the process is a less design time and high performance classifications in comparison to human configurations.

3.7 Optimization Problem

Search algorithms are applied to a data structure containing discrete or continuous data to extract information based on the search objective, i.e. problem domain. These applications of search algorithms include optimization problems. Optimization algorithms are search algorithms designed to find near-optimal solutions to hard problems, i.e. optimization problems. The evaluation of search algorithms mainly lies in their computational time and ability to find the optimal solution. Search algorithms can be classified into two categories, single-point search and multi-point search.

3.7.1 Single-point Search

A single point algorithm explores a single point in the search space at a time. The evolution of this point is carried iteratively through the search space to a better location, i.e near-optimal solution. An example of a single-point search is Simulated annealing (SA). SA algorithm is an optimization algorithm based on annealing processes in metallurgy, whereby material structures are controlled through heating and cooling to get the desired structure [20] [8]. The algorithm iterate by choosing the new potential solution based on a neighbourhood criteria and is evaluated by an acceptance probability function.

3.7.2 Multi-point search

Multi-point searches are population-based algorithms. This category of algorithms encompasses the EA algorithms highlighted in the earlier section 3.1 and 3.2[18]. Members of the population in the search algorithm serve as a potential solution to the problem. Therefore the algorithm searches the space from multiple points as it evolves its population.

3.8 Critical Analysis

On a review of the literature discussed in the previous sections, an area of interest is flagged and proposed for exploration. Section 3.3 highlighted a real-world problem statement to which algorithms can be applied as discussed in section 3.4. The solution provided in section 3.4.1 are problem dependent and are unable to efficiently handle problems with higher search space, i.e. multilevel thresholding section 3.4.2. The area of multilevel thresholding requires a multi-point search algorithm as highlighted in section 3.5. GAs as in 3.5 are deployed to handle the extended problem space of image segmentation, i.e. multilevel segmentation. While the solution to the problem seems probable it is without doubt, as discussed in section 3.2, that GAs are not immune to error mainly due to high dependence on user interaction in the configuration. These configurations determine the optimally of the GAs performance. Therefore in previous work, as highlighted in section 3.6.1, automation can be applied to the configuration process of GAs to reduce user interaction and increase performance.

This research will explore the application of single-point, simulated annealing search, and multi-point search, Genetic algorithm, to automate the configuration of GA on image segmentation. The following section discusses the area of exploration.

3.8.1 Automated Design of GA on image segmentation

Genetic algorithm parameter turning involves the designing and implementation of a Genetic algorithm to solve binary and multilevel image segmentation. The various parameters of the GA stand open for exploration. The attributes include evaluation, regeneration, and mating components.

- Evaluation: decision threshold between Otsu and Kapur fitness
- Regeneration: population size and the termination criterion

- Mating:
  - selection: method (elitism, tournament-size, Roulette wheel etc)
  - crossover: method (one-point, two-point, uniform etc) and probability of occurrence
  - mutation: method (bit-flip, swap) and probability of occurrence

The Automating algorithms will use the combination of PSMR (Peak signal-to-noise ratio) and SSIM (Structural Similarity Index) algorithms to evaluate the objective. The automating algorithms representation will be the different possible combinations of parameters for the image segmentation GA. The automation then searches for the best-performing GA configuration. The automating algorithm is evaluated on the number of GA parameters it can properly automate while archiving substantially good results.

3.8.2 Contributions

User interaction in parameter turning is a tedious cycle applied to explore the optimal combination of parameters. Despite the tedious manual process, only a subset of the search space is explored. Automation seeks to reduce the need for user interaction to archive substantially better results. This contribution eliminates the need for users to focus on GA parameter turning. In essence, GA parameters are exhaustively explored to get the best combination, limiting user errors or bias.

3.8.3 Potential Future Research

It is widely accepted that automation is a time-intensive process and multilevel thresholding with Ostu and Kapur is also known to be computationally costly. Consequently, the combination of the two issues has the potential of limiting the use of automation of GA on image segmentation. To counter this obstacle, testing the algorithm performance will measure computation cost, i.e. average run time, which is a potential subject of future research work. Therefore further exploring different automation algorithms to archive significantly better run time performance, to reduce computational cost.

4 Methodology

The research will follow the Proof by Demonstration methodology. The project will be divided into demonstrable components that make up the algorithm that will be applied. Ultimately we deduce an objective measure of the algorithm designed. The algorithm 3 demonstrate the automation design of GA for image segmentation.

The following sections demonstrate the components of the proposed algorithm that will be explored.

Algorithm 3 Generational evolutionary algorithm

```
Setting out image data according to image segmentation
Create the initial automation function
repeat
    Evaluate the automation function on image segmentation
    Improve the automation function based on evaluation
until termination criterion is met
Evaluate the final automation function
```

4.1 Setting Out Image Data

The steps in setting out image data include data collection and image conversion to a histogram. For data collection, images will be collected and cleaned for any discrepancy such as image type and format. A histogram will be created for each image by counting the frequency of each pixel value.

4.2 Automation Function

Exhaustively explore automated design of different combination of GA parameters to produce optimize configuration for images segmentation.

4.2.1 Initial Function

AutoGA function will be a population of chromosomes representing the different configuration parameters of GA. That is each gene in the chromosome represent a GA parameter such as population size, selection method, etc as discussed in section 3.8.1.

4.2.2 Improve Function

Apply genetic operators to the autoGA population. This includes selection, crossover and mutation.

4.3 Evaluating the Automation Function

GAs binary and multilevel thresholding for image segmentation will be evaluated using Otsu and Kapur methods. This is used to evaluate each parameter setting generated by autoGA.

4.4 Final Automation Objective

Since image segmentation is the objective of the GA, the automation of GA automatically inherits this objective. Therefore the automated designed GA must be able to accurately detect the different objects within images. This is applied by evaluating the image histogram to identify noticeable normal distributions. The final Automated function, i.e. GA produced, should identify all observable normal distributions by thresholding them into different segments.

5 Planning

The tasks and milestones for the research project. You can also add a figure such as a Gantt chart.

Task	July	August				September				October		
	Week 4	Week 1	Week 2	Week 3	Week 4	Week 1	Week 2	Week 3	Week 4	Week 1	Week 2	Week 3
Multilevel thresholding using GA												
GA to configure a GA												
Simulated Annealing to configure a GA												
Compare the effectiveness of the automation												
Final Report												

Table 1: Project Schedule

References

[1] M. Abbasgholipour, M. Omid, A. Keyhani, and S. S. Mohtasebi. Color image segmentation with genetic algorithm in a raisin sorting system based on machine vision in variable conditions. *Expert Systems with Applications*, 38(4):3671–3678, 2011.

[2] S. Bates, J. Sienz, and V. Toropov. Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. In *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, page 2011, 2004.

[3] K. Hammouche, M. Diaf, and P. Siarry. A multilevel automatic thresholding method based on a genetic algorithm for a fast image segmentation. *Computer Vision and Image Understanding*, 109(2):163–175, 2008.

[4] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[5] K. Jebari and M. Madiafi. Selection methods for genetic algorithms. *International Journal of Emerging Sciences*, 3(4):333–344, 2013.

[6] L. Jianzhuang, L. Wenqing, and T. Yupeng. Automatic thresholding of gray-level pictures using two-dimension otsu method. *China., 1991 International Conference on Circuits and Systems*, pages 325–327 vol.1, 1991.

[7] J. Kapur, P. Sahoo, and A. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, 29(3):273–285, 1985. ISSN 0734-189X. doi: https://doi.org/10.1016/0734-189X(85)90125-2.

[8] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science (New York, N.Y.)*, 220:671–80, 06 1983. doi: 10.1126/science.220.4598.671.

[9] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986. ISSN 0031-3203. doi: https://doi.org/10.1016/0031-3203(86)90030-0. URL `https://www.sciencedirect.com/science/article/pii/0031320386900300`.

[10] P. Kora and P. Yadlapalli. Crossover operators in genetic algorithms: A review. *International Journal of Computer Applications*, 162(10), 2017.

[11] J. R. Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4(2):87–112, 1994.

[12] S. M. Lim, A. B. M. Sultan, M. N. Sulaiman, A. Mustapha, and K. Y. Leong. Crossover and mutation operators of genetic algorithms. *International journal of machine learning and computing*, 7(1):9–12, 2017.

[13] S. Manikandan, K. Ramar, M. W. Iruthayarajan, and K. Srinivasagan. Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm. *Measurement*, 47:558–568, 2014.

[14] M. Mosayebi and M. Sodhi. *Tuning Genetic Algorithm Parameters Using Design of Experiments*, page 1937–1944. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450371278. doi: 10.1145/3377929.3398136. URL `https://doi.org/10.1145/3377929.3398136`.

[15] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. doi: 10.1109/TSMC.1979.4310076.

[16] N. Pillay and T. Nyathi. *Automated Design of Classification Algorithms*, pages 171–184. Springer International Publishing, Cham, 2021. ISBN 978-3-030-72069-8. doi: 10.1007/978-3-030-72069-8_10. URL `https://doi.org/10.1007/978-3-030-72069-8_10`.

[17] T. W. RIDLER and S. CALVARD. Picture thresholding using an iterative selection method. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(8):630–632, 1978. doi: 10.1109/TSMC.1978.4310039.

[18] O. Roeva, T. Slavov, and S. Fidanova. Population-based vs. single point search meta-heuristics for a pid controller tuning. 1:200–230, 01 2013. doi: 10.4018/978-1-4666-4450-2.ch007.

[19] C. Ryan, J. J. Collins, and M. O'Neill. Grammatical evolution: Evolving programs for an arbitrary language. In W. Banzhaf, R. Poli, M. Schoenauer, and T. C. Fogarty, editors, *Proceedings of the First European Workshop on Genetic Programming*, volume 1391 of *LNCS*, pages 83–96, Paris, 14-15 Apr. 1998. Springer-Verlag. ISBN 3-540-64360-5. doi: doi:10.1007/BFb0055930. URL `http://citeseer.ist.psu.edu/ryan98grammatical.html`.

[20] J. Shea. Smithells metal reference book, 8th ed. [book review]. *Electrical Insulation Magazine, IEEE*, 21:56– 56, 04 2005. doi: 10.1109/MEI.2005.1412231.

[21] K. K. Singh and A. Singh. A study of image segmentation algorithms for different types of images. *International Journal of Computer Science Issues (IJCSI)*, 7(5):414, 2010.

[22] R. Storn. On the usage of differential evolution for function optimization. In *Proceedings of north american fuzzy information processing*, pages 519–523. Ieee, 1996.

[23] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[24] K. Tang, X. Yuan, T. Sun, J. Yang, and S. Gao. An improved scheme for minimum cross entropy threshold selection based on genetic algorithm. *Knowledge-Based Systems*, 24(8):1131–1138, 2011.

[25] P. A. Whigham et al. Grammatically-based genetic programming. In *Proceedings of the workshop on genetic programming: from theory to real-world applications*, volume 16, pages 33–41. Citeseer, 1995.

[26] J. Zhang, H. Li, Z. Tang, Q. Lu, X. Zheng, and J. Zhou. An improved quantum-inspired genetic algorithm for image multilevel thresholding segmentation. *Mathematical Problems in Engineering*, 2014, 2014.