Raw coal ore classification using image segmentation methods

MW Theunissen



orcid.org 0000-0002-7456-7769

Dissertation submitted in fulfilment of the requirements for the degree *Master of Engineering in Computer and Electronic Engineering* at the North-West University

Supervisor: Prof WC Venter

Graduation May 2018

Student number: 22721339

ABSTRACT

The research done in order to complete this dissertation can be summarised as the investigation, implementation, analysis, and comparison of data analysis techniques with the intention to segment a digital image of a raw coal sample into its constituent materials. The goal is to obtain a per-pixel classification of the image with a sufficient level of accuracy, to be used to generate a viable washability curve.

An extensive literature survey was done in order to investigate the current state of the applicable fields of image segmentation, data classification, clustering, and machine learning. The identified techniques that are both pertinent and suitable to the problem defined above were then implemented in a common development environment. In order to investigate, rate, and compare the techniques they were analysed using internal analysis techniques as well as compared to ground truth classifications obtained by expert geologists.

The research is based on previous work [1], and the explicit goal is to improve an existing system of image classification. The existing system makes use of feature extraction and a clustering algorithm, called k-means, to identify similar groups of pixels and then assign them to model values that are selected by a user. The results of the research presented here makes use of a system using a similar high level topology with several alterations that were made to improve the accuracy of the image segmentation. The most prominent of these alterations is the use of a mean shift algorithm to “group” the pixels together and assign them to the models. This choice was, largely, made due to the higher level of spatial information about the image pixels that is incorporated in the mean shift algorithm. This is as opposed to the k-means algorithm that only makes use of range values.

It was found that image segmentation techniques can, indeed, be used to achieve a sufficient level of accuracy towards raw coal ore classification under certain conditions. The success of each technique, and the constraints under which they achieve it, are identified and thoroughly motivated in this dissertation.

The first half of the dissertation investigates the research problem, existing system, and possible alternatives in the literature. The second half presents the results of the research and discusses the knowledge obtained from it.

Keywords: Image segmentation; Cluster analysis; Statistical classification; Machine learning

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisors, Prof. Willie Venter and Prof. Pieter Van Vuuren, for their advice and guidance throughout the lifecycle of this research project.

Secondly, I would like to thank Herman Dorland and his colleagues for supplying the digital images of coal ore to be used as training data sets in my research.

Finally, I would like to thank Christine Knoetze, the light that keeps me going.

CONTENTS

ABSTRACT ..... **I**

ACKNOWLEDGEMENTS ..... **II**

CONTENTS ..... **III**

LIST OF FIGURES ..... **VI**

LIST OF TABLES ..... **VIII**

LIST OF ALGORITHMS ..... **IX**

LIST OF EQUATIONS ..... **IX**

1. INTRODUCTION **1**

1.1. RESEARCH BACKGROUND 1

1.1.1. Energy 1

1.1.2. Coal 1

1.1.3. Washability 3

1.1.4. Image processing 4

1.1.5. Coal ore images 7

1.2. RESEARCH DEFINITION 10

1.2.1. Problem statement 10

1.2.2. Scope 10

1.2.3. Verification and validation 11

1.3. RESEARCH LOGISTICS 11

1.3.1. Methodology of practice 11

1.3.2. Document outline 12

2. IMAGE SEGMENTATION **13**

2.1. THE FIELD OF IMAGE SEGMENTATION 13

2.1.1. Image 13

2.1.2. Segmentation problem 14

2.1.3. Approaches 15

2.2. IMAGE SEGMENTATION BY THRESHOLDING 18

2.2.1. Fundamentals of thresholding 18

2.2.2. Thresholding algorithm types 20

2.2.3. Issues with thresholding 25

2.2.4. Summary of thresholding 27

2.3. IMAGE SEGMENTATION BY EDGE DETECTION 28

2.3.1. Fundamentals of edge detection 28

2.3.2. Edge detection algorithm types 30

2.3.3. Issues with edge detection 39
2.3.4. Summary of edge detection 41
2.4. IMAGE SEGMENTATION BY REGION EXTRACTION 41
2.4.1. Fundamentals of region extraction 41
2.4.2. Region extraction algorithm types 42
2.4.3. Issues with region extraction 46
2.4.4. Summary of region extraction 47
2.5. IMAGE SEGMENTATION BY CLUSTERING 47
2.5.1. Fundamentals of clustering 47
2.5.2. Clustering algorithm types 50
2.5.3. Issues with clustering 53
2.5.4. Summary of clustering 54
3. THEORETICAL DESIGN 56
3.1. THEORETICAL DESIGN: THE SYSTEM 56
3.2. THEORETICAL DESIGN: FEATURE EXTRACTION 57
3.2.1. Feature engineering 57
3.2.2. Colour spaces 58
3.2.3. Texture analysis 61
3.2.4. Alternative components for feature extraction 64
3.3. THEORETICAL DESIGN: CLUSTERING 65
3.3.1. k-means 65
3.3.2. k-related algorithms 67
3.3.3. Particle swarm optimisation 69
3.3.4. Mean shift 73
3.3.5. Alternative components for clustering 76
3.4. THEORETICAL DESIGN: MODELS AND FINAL ASSIGNMENTS 77
3.4.1. Existing models 77
3.4.2. Multiple models 78
3.4.3. Model parameters 78
3.4.4. Model assignment topologies 78
3.4.5. Alternative components for models and final assignments 79
4. TESTING 80
4.1. GROUND TRUTHS 80
4.1.1. Input images 80
4.1.2. Classification data 86
4.2. TESTING THE FEATURE EXTRACTION 93
4.3. TESTING THE CLUSTERING ALGORITHMS 94
4.4. TESTING THE MODEL PARAMETERS 97
4.5. TESTING THE FINAL ASSIGNMENTS 97
4.6. TESTING THE SYSTEM 98

5. RESULTS 99

5.1. FEATURE EXTRACTION RESULTS 99
5.1.1. Feature distributions 99
5.1.2. Differentiating models 103
5.1.3. Ratings 107
5.1.4. t-tests 108
5.1.5. Alternative feature space 110
5.2. CLUSTERING RESULTS 118
5.2.1. Controlled variables 118
5.2.2. Evaluation of clustering 120
5.3. MODEL PARAMETERS RESULTS 122
5.4. FINAL ASSIGNMENTS RESULTS 125
5.4.1. Controlled variables 125
5.4.2. Evaluation of final assignments 125
5.5. THE SYSTEM RESULTS 129
5.5.1. The alternative system 129
5.5.2. Image segmentations 129
5.5.3. Evaluation of the system 142

6. DISCUSSION 145

6.1. FEATURE EXTRACTION DISCUSSION 145
6.2. CLUSTERING DISCUSSION 147
6.3. MODELS AND FINAL ASSIGNMENTS DISCUSSION 148
6.4. THE SYSTEM DISCUSSION 148

7. CONCLUSION 149

REFERENCES 151

APPENDIX 156

A. MODEL FEATURE VALUE DISTRIBUTIONS 156
L* feature 156
b* feature 157
v* feature 157
SVD feature 158
Entropy feature 158
Sobel + median feature 159
Sobel + mean feature 159
Sobel + gauss feature 160
Roberts + median feature 160
Roberts + mean feature 161
Roberts + gauss feature 161

Prewitt + median feature 162
Prewitt + mean feature 162
Prewitt + gauss feature 163
Standard deviation (SD) feature 163
B. FEATURE VALUE T-TESTS 164
Chromaticity 164
Regularity 164
Roughness 166
C. CLASSIFICATION TESTS 167
Sample 0 167
Sample 1 169
Sample 2 172
Sample 3 174
Sample 4 176
Sample 5 177

LIST OF FIGURES

Figure 1: Chart of total global primary energy supply for 2014 adapted from [4] 2
Figure 2: Chart of total global energy supply for 2014 adapted from [4] 2
Figure 3: Example of a washability curve 4
Figure 4: Main classification flow diagram 5
Figure 5: Example coal ore sample A 8
Figure 6: Example coal ore sample B 8
Figure 7: Example coal ore sample C 9
Figure 8: Example coal ore sample D 9
Figure 9: Diagram of simplified research methodology 12
Figure 10: Example of a simple image segmentation exercise 15
Figure 11: Diagram of image segmentation categories 16
Figure 12: Depiction of image segmentation terminologies 18
Figure 13: Example of a histogram 21
Figure 14: Image entropy illustration 22
Figure 15: Example of edge matching 23
Figure 16: Global- and local thresholding example 25
Figure 17: Single level threshold feature spaces 25
Figure 18: Multi-level threshold feature spaces 26
Figure 19: Adjusted multi-level threshold feature spaces 27
Figure 20: Examples of step-, ramp-, and roof edges 28
Figure 21: Ideal image segmentation with edge detection 29
Figure 22: Example of spatial frequency filtering 30
Figure 23: Example of kernel convolution 33
Figure 24: Example of differential operator edge detection 34
Figure 25: Example of Laplacian edge detection 35
Figure 26: Example of Laplacian-of-Gaussian edge detection 36
Figure 27: Edges in terms of differentials 37
Figure 28: Example of Marr-Hildreth edge detection 37
Figure 29: Example of Canny edge detection 39
Figure 30: Example of pixel aggregation 43
Figure 31: Example of splitting-and-merging 45
Figure 32: Example of using clustering to segment an image 49
Figure 33: Example of hierarchical clustering 52

Figure 34: Example of Partitional Clustering 52
Figure 35: Main flow diagram with highlighted areas for improvement 56
Figure 36: Process of feature engineering 58
Figure 37: Illustration of colour channels with key legend 61
Figure 38: PSO neighbourhood topologies 73
Figure 39: Model highlighting 77
Figure 40: Model centroids 77
Figure 41: Model-to-cluster assignment 79
Figure 42: Sample 0 annotated image 81
Figure 43: Sample 1 annotated image 82
Figure 44: Sample 2 annotated image 83
Figure 45: Sample 3 annotated image 84
Figure 46: Sample 4 annotated image 85
Figure 47: Sample 5 annotated image 86
Figure 48: Sample 0 hand classification 87
Figure 49: Sample 1 hand classification 88
Figure 50: Sample 2 hand classification 89
Figure 51: Sample 3 hand classification 90
Figure 52: Sample 4 hand classification 91
Figure 53: Sample 5 hand classification 92
Figure 54: Alternative cluster-to-model assignments 98
Figure 55: Sample 0 b* feature distribution 100
Figure 56: Sample 0 v* feature distribution 100
Figure 57: Sample 1 svd feature distribution 101
Figure 58: Sample 1 entropy feature distribution 101
Figure 59: Sample 3 Sobel + median feature distribution 102
Figure 60: Sample 3 Sobel + Gauss feature distribution 102
Figure 61: Parallel coordinates of baseline feature space (Sample 0) 111
Figure 62: Parallel coordinates of alternative feature space (Sample 0) 112
Figure 63: Parallel coordinates of baseline feature space (Sample 1) 112
Figure 64: Parallel coordinates of alternative feature space (Sample 1) 113
Figure 65: Parallel coordinates of baseline feature space (Sample 2) 113
Figure 66: Parallel coordinates of alternative feature space (Sample 2) 114
Figure 67: Parallel coordinates of baseline feature space (Sample 3) 114
Figure 68: Parallel coordinates of alternative feature space (Sample 3) 115
Figure 69: Parallel coordinates of baseline feature space (Sample 4) 115
Figure 70: Parallel coordinates of alternative feature space (Sample 4) 116
Figure 71: Parallel coordinates of baseline feature space (Sample 5) 116
Figure 72: Parallel coordinates of alternative feature space (Sample 5) 117
Figure 73: Sample 0 baseline segmentation A 130
Figure 74: Sample 0 baseline segmentation B 130
Figure 75: Sample 0 alternative segmentation 131
Figure 76: Sample 1 baseline segmentation A 132
Figure 77: Sample 1 baseline segmentation B 132
Figure 78: Sample 1 alternative segmentation 133
Figure 79: Sample 2 baseline segmentation A 134
Figure 80: Sample 2 baseline segmentation B 134
Figure 81: Sample 2 alternative segmentation 135
Figure 82: Sample 3 baseline segmentation A 136
Figure 83: Sample 3 baseline segmentation B 136
Figure 84: Sample 3 alternative segmentation 137
Figure 85: Sample 4 baseline segmentation 138
Figure 86: Sample 4 alternative segmentation 138
Figure 87: Sample 5 baseline segmentation A 139
Figure 88: Sample 5 baseline segmentation B 140
Figure 89: Sample 5 baseline segmentation C 140
Figure 90: Sample 5 baseline segmentation D 141

Figure 91: Sample 5 alternative segmentation 141
Figure 92: Baseline roughness compared to baseline regularity 145
Figure 93: Problematic minerals 146
Figure 94: Issues with photo quality 147

LIST OF TABLES

Table 1: Thresholding criterion 27
Table 2: Edge detection criterion 41
Table 3: Region extraction criterion 47
Table 4: Clustering criterion 54
Table 5: Alternative components for feature extraction 64
Table 6: k-means criterion 67
Table 7: PSO criterion 73
Table 8: Mean shift criterion 76
Table 9: Alternative components for clustering 76
Table 10: Alternative model parameters 79
Table 11: Summarised baseline system 98
Table 12: Model feature values A 104
Table 13: Model feature values B 105
Table 14: Model feature values C 106
Table 15: Total sum of differences between models A 107
Table 16: Total sum of differences between models B 107
Table 17: Total sum of differences between models C 107
Table 18: SVD/Entropy t-test p-values 109
Table 19: Sobel + Median/Sobel + Gauss t-test p-values 110
Table 20: Summarised results of feature extraction test 110
Table 21: The controlled variables during the clustering test 119
Table 22: Down sampled clustering test Davies-Bouldin index 120
Table 23: Down sampled clustering test Dunn index 120
Table 24: Down sampled clustering test average silhouette 121
Table 25: Down sampled clustering test mean intra-cluster distance 121
Table 26: Down sampled clustering test mean inter-cluster distance 121
Table 27: Clustering test Davies-Bouldin index 122
Table 28: Clustering test Dunn index 122
Table 29: Clustering test average silhouette 122
Table 30: Clustering test mean intra-cluster distance 122
Table 31: Clustering test mean inter-cluster distance 122
Table 32: Sample 0 model parameter test 123
Table 33: Sample 1 model parameter test 123
Table 34: Sample 2 model parameter test 123
Table 35: Sample 3 model parameter test 124
Table 36: Sample 4 model parameter test 124
Table 37: Sample 5 model parameter test 124
Table 38: Example confusion matrix 125
Table 39: Sample 0 highest accuracy classification 126
Table 40: Sample 1 highest accuracy classification 127
Table 41: Sample 2 highest accuracy classification 127
Table 42: Sample 3 highest accuracy classification 127
Table 43: Sample 4 highest accuracy classification 128
Table 44: Sample 5 highest accuracy classification 128
Table 45: General performance of classifiers 128
Table 46: Comparison of baseline- and final system 129
Table 47: Sample 0 system test results 142
Table 48: Sample 1 system test results 142

Table 49: Sample 2 system test results 143
Table 50: Sample 3 system test results 143
Table 51: Sample 4 system test results 143
Table 52: Sample 5 system test results 144

LIST OF ALGORITHMS

Algorithm 1: Image segmentation 14
Algorithm 2: Thresholding 19
Algorithm 3: Pixel aggregation 42
Algorithm 4: Splitting-and-merging 44
Algorithm 5: Clustering 49
Algorithm 6: Singular value decomposition 63
Algorithm 7: k-means 66
Algorithm 8: Mean shift 74

LIST OF EQUATIONS

Equation 1: Shannon entropy 22
Equation 2: Niblack thresholding 24
Equation 3: Discrete Fourier transform 31
Equation 4: Inverse discrete Fourier transform 31
Equation 5: Simple spatial differentials 31
Equation 6: Image gradient 31
Equation 7: Image gradient magnitude 32
Equation 8: Image gradient angle 32
Equation 9: Roberts cross operator 32
Equation 10: Sobel operator 32
Equation 11: Prewitt operator 32
Equation 12: Image Laplacian 34
Equation 13: Laplacian operator 34
Equation 14: Gaussian filter 35
Equation 15: Laplacian of Gaussian 35
Equation 16: Square error of a cluster configuration 53
Equation 17: L*a*b* conversion 60
Equation 18: L*u*v* conversion 60
Equation 19: YU'V' conversion 60
Equation 20: XYZ conversion 60
Equation 21: Standard deviation 63
Equation 22: Mean square error 64
Equation 23: Euclidean distance 66
Equation 24: k-means objective function 66
Equation 25: Manhattan distance 68
Equation 26: PSO fitness function 70
Equation 27: Alternative PSO fitness function 71
Equation 28: PSO velocity update function 71
Equation 29: PSO velocity clipping function 72
Equation 30: PSO guaranteed convergence criteria 72
Equation 31: Flat kernel 74
Equation 32: Gaussian kernel function 74
Equation 33: Davies-Bouldin index 94
Equation 34: Dunn index 94
Equation 35: Silhouette 95
Equation 36: Mean intra-cluster distance 96

Equation 37: Mean inter-cluster distance 96

1. INTRODUCTION

1.1. Research background

The objective of this section is to provide background information regarding the research presented in this dissertation. The following are some of the key points of discussion:

- The applicability of the research in the industry
- Attempts at similar research
- The material from which the presented research is derived

1.1.1. Energy

The current state of the world's energy generation, consumption, and distribution is one of great importance. An expanding global population (resulting in rising demand) [2] coupled with pressure from many governments (e.g. the Paris agreement of 2016) to transition to cleaner energy sources create an interesting outlook to global energy usages. The main driving force behind these energy changes is the increasing figures of greenhouse gasses, in the form of anthropogenic CO_2 emissions, that accelerate the greenhouse effect to dangerous levels [3].

According to a 2016 report by the International Energy Agency (IEA), the burning of coal accounts for 28.6 % of the world's total primary energy supply and 40.8 % of the world's electricity generation [4]. Coal is also responsible for 45.9 % (a majority) of CO_2 emissions, and consequently to global climate change. Regardless of the prudence or ethicality of continuing to use coal as one of the main sources of energy generation, it is in the best interest of humanity to increase the efficiency with which the resource is converted to energy. This is because an increase in efficiency results in less coal having to be burned for equivalent energy, which results in reduced pollution and increased economic value.

1.1.2. Coal

Coal is often seen as an outdated source of energy. This, however, seems to be a misconception, when considering the fact that the burning of coal accounts for the second highest contribution to the world's total energy supply, and the highest contribution to the world's electricity supply (almost double that of its closest competitor, natural gas) [4]. This data is depicted in figures 1 and 2. The misconception is possibly brought on by the introduction of several alternative energy sources.

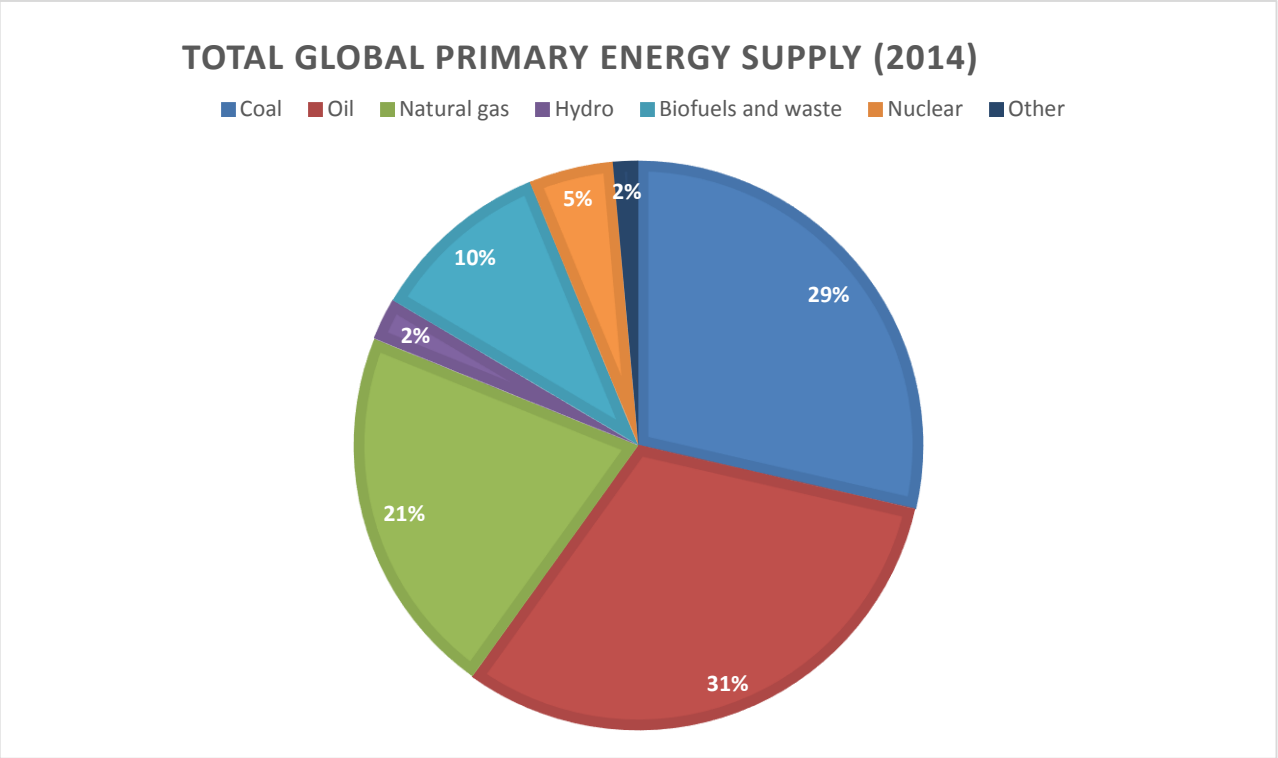


Figure 1: Chart of total global primary energy supply for 2014 adapted from [4]

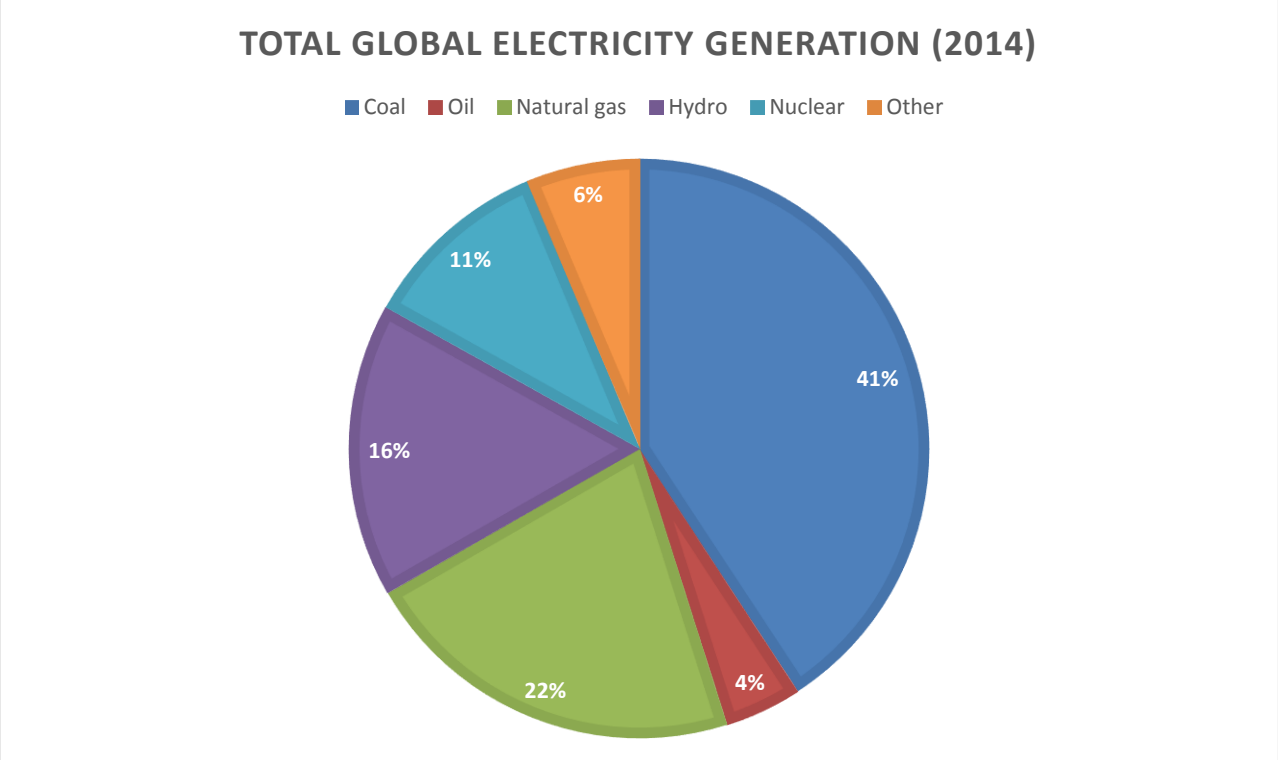


Figure 2: Chart of total global energy supply for 2014 adapted from [4]

It can be deduced that a very large portion of the coal supply of the world is dedicated to the generation of electricity. This is indicated by the fact that some of the world's largest coal consumers devote a majority of their coal to electricity, namely, China, India, USA, and Australia. The exact portions are 52%, 66%, 92 %, and 90%, respectively [5].

The process of generating power through the combustion of coal can be summarised as follows:

- Coal ore is extracted from the ground
- The ore is pulverised
- The resulting product is burned in a furnace to boil the water in a boiler
- The steam from the boiler turns a turbine
- The turbine is coupled to a generator to generate electricity

There have been many improvements made to the process, such as gasifying the coal and using a gas turbine, but the fundamental principles remain the same. Improving the efficiency with which coal fired power plants convert coal to electricity has been in continuous development for the last century [6]. The research done in this dissertation concerns itself with improving the quality of the coal provided to the power plant and not the power plant itself (e.g. the thermal efficiency).

Before coal ore is transported to its designated market (usually to be converted to energy) it passes through a coal preparation plant. The goal of this plant is to remove as much of the undesirable material from the raw ore as possible, resize the samples, and sort it into groups based on the quality of the ore. The efficacy of these plants, therefore, has a direct impact on the cost of transport, the market value, and environmental impact of the ore. Washability information about the raw ore samples is used to design and calibrate the operations of a coal preparation plant.

1.1.3. Washability

In practice, the necessary washability information is obtained by generating a washability curve. The washability curve reflects the density and density distribution of the materials in the ore. Information obtained from this curve is used to estimate the complexity of separating usable coal from non-usable mineral matter. An example of such a curve is presented in figure 3. The importance of the density information comes from the fact that usable coal mineral components (e.g. vitrinite), typically, has much lower density than the unusable components.

The data for the washability curve is conventionally obtained by repeated float-sink analyses. During a float-sink test crushed coal is added to liquids of different densities. The liquids available for such a test include organic liquids, aqueous solutions, and suspensions. [7] Each time the coal is mixed with the liquid and then allowed to sink or float based on the coal density relative to the liquid density. Then, the floating ore particles and sinking ore particles are separated and subjected to a different float-sink test to divide them again. This is repeated until enough data has been gathered regarding the density composition of the entire sample.

These tests make use of toxic organic liquids that are difficult to handle safely, and need to be replaced after a time in order to insure that the liquids have the required densities. The act of disposing of these hazardous materials also requires special care and additional costs. Therefore, a need exists in the field of coal preparation for a washability curve generating method that is time- and cost effective, and does not require the use of hazardous materials [8].

Attempts at such a method include the hyperspectral analysis of the 3D structure of the ore samples [9], water only fractioning [7], and various image processing methods used to identify particle sizes and distributions after the ore samples have been crushed [10] [11] [12]. These methods meet one or two of the needs expressed in the previous paragraph, but none of them address all three. They are also still in early developmental stages.

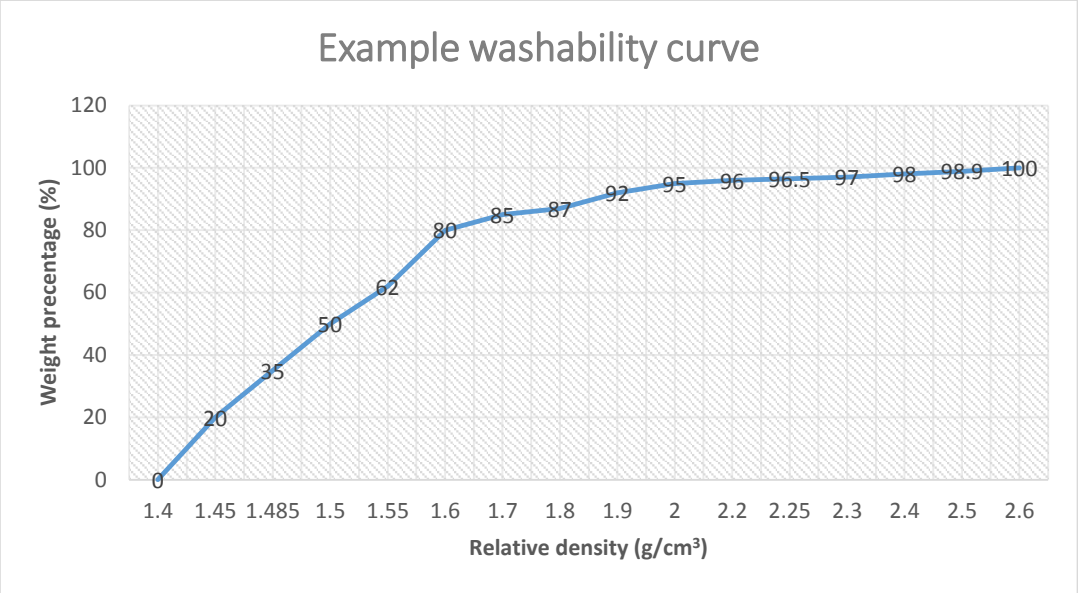


Figure 3: Example of a washability curve

1.1.4. Image processing

The research presented in this dissertation is, however, based on a novel method proposed in [13] and [1] that employs image processing techniques on photographs of ore samples before they are crushed. The idea behind this method is to classify photographs of several ore samples into the materials that they are comprised of. Theoretical calculations are then done to approximate the percentage of each material contained in the samples. Density estimates can then be used to generate an approximated washability curve for each sample. Several of these approximations are then taken into account to generate the final washability curve.

The exact procedure is summarized in [13]:

1. *Photograph the core or run-of-mine coal at close range*
2. *Map out the components of the coal to scale on the photographs*
3. *Create a base grid over the map of the coal at a grid cell size of 0.5 mm × 0.5 mm. The grid cell size must be small enough that most of the grid cells only consist of a single component*
4. *Estimate the areas of components in the base grid cells*
5. *Assign estimated densities and ash to the different mapped components*
6. *Calculate density and ash for each drawn block*
7. *Import grid, density and ash data into geological resource estimation software*
8. *Create a model based on the imported data in the geological resource estimation software*
9. *Estimate density and ash for different block sizes using geological resource estimation software*
10. *Calculate the washability tables per block size from the different block sizes*

Traditional float-sink analyses have an added disadvantage of being restricted to one crush size per test. If the power plant using the coal, from which the washability information is gathered, crushes it at a different crush size there could be large discrepancies with regards to the yield of the coal. The presented method of "virtual crushing" has the capability of simulating washability curves at any possible crush size on several (or the same) ore samples.

A method by which the second item in the above listed procedure (mapping out the components of the coal in the photograph) can be accomplished is proposed in [1]. This step in the procedure is of vital importance. Without an accurate mapping of coal components all other calculations done in order to generate the washability curve would be erroneous. The proposed method classifies the photograph according to a set of user defined areas in the image (to be used as model values for the classes).

The classification process is done in two main parts. First, four descriptive feature values are generated for each pixel in the digital image. Then the resulting multivariate feature space is clustered using cluster analysis techniques. Thereafter, the final identified clusters are assigned to the most fitting model values. Figure 4 presents a diagram of the method by which the classification in achieved. The circular components represent the passive input/output entities processed by the active components (rectangular components).

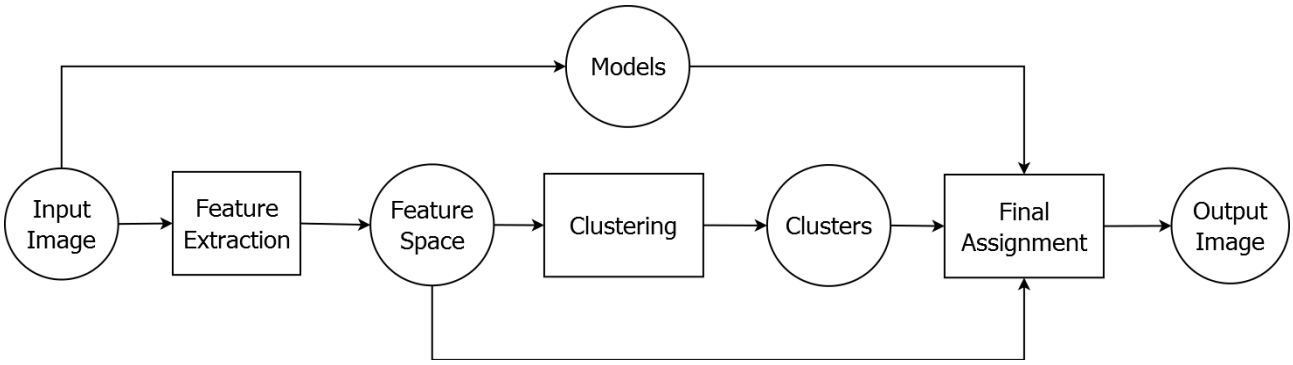


Figure 4: Main classification flow diagram

The components in figure 4 are explained as follows:

- Input image

  This is the input data of the system. It takes the form of a digital image of the coal ore sample to be classified. A digital image can be seen as a two dimensional lattice of three dimensional data values. These three values correspond to the red, green, and blue values (of the ubiquitous RGB colour space) of the pixel in question. Each dimension has a minimum value of 0 and a maximum value of 255.

- Models

  The data that is generated from the selection areas in the image to be used as model values for each material in the coal ore.

- Feature Extraction

  This component of the system receives the unaltered digital image as input. The purpose of this component is to extract feature values from the image that are better suited to differentiate between the coal component materials in the image. The four features that were decided upon are luminance, chromaticity, roughness, and regularity. The last two of these four features are specifically developed for the application in this dissertation and will be explained in section 3.2.3.

- Feature Space

  The feature space is the output data from the feature extraction component. The data takes the form of a two dimensional lattice of four dimensional data values. Each data value corresponds to a pixel in the input image.

- Clustering

  This component of the system receives the feature space as input. The purpose of this component is to group the data points in the feature space into groups that are more similar to each other than data points in other groups. The algorithm used to achieve this is the commonly used k-means clustering algorithm.

- Clusters

  This component is the output data from the clustering component. The data takes the form of a two dimensional lattice of values identifying the class to which each data point is assigned (e.g. an integer value ranging from 0 to 6, if there are 7 classes).

- Final Assignment

  This component of the system receives the clusters, model values and feature space as input. The purpose of this component is to assign the data points of each cluster to the model value that is the most similar to the data points in the cluster. The method used is a simple nearest neighbour assignment.

- Output Image

  This is the output data of the system. The data takes the form of a two dimensional lattice of values identifying the material model to which each data point is assigned (e.g. an integer value ranging from 0 to 2, if there are 3 classes). This effectively identifies which coal component each pixel in the image is a member of.

The method of image classification presented above has been shown to achieve levels of accuracy that are promising and it seems very feasible that image classification can be used to generate washability information. There is, however, plenty of room for further research and improvement. [1]

1.1.5. Coal ore images

Before defining the research problem, in the next section, it would be prudent to investigate a few example images of coal ore. These are the type of images that the system is expected to segment.

Figure 5 depicts a digital image of a coal sample containing siderite, vitrinite, mudstone, and inertinite. The brownish mineral component is the siderite, the smooth grey regions are made up of mudstone, and the rough, reflective regions contain vitrinite and inertinite. From this image it can be seen that coal ore contains very little colour information that can be used to distinguish between the mineral components. The highest contrast in colour is present between the siderite (tan/brown colour) and vitrinite (blueish tinge in reflected light).

The sample in figure 6 contains two mineral components: vitrinite and mudstone. The vitrinite is, again, represented by the reflective regions, and the mudstone by the smooth, grey regions. From this figure it can be seen that, in addition to a lack of colour information, coal ore samples have very little homogeneity in their constituent regions. No assumptions can be made about the underlying structure and positioning of the mineral components, and the task of segmenting such an image is non-trivial.

Figures 7 and 8 depict the high level of variation in roughness of typical coal textures. Some form of texture analysis is, therefore, expected to be used to distinguish between these variations. The presence and effect of shadows are also well depicted in figure 8. Certain areas of the sample are obscured by dark patches as a result of natural shadows. These regions can result in a misclassification of the affected minerals.

To summarise, digital images of raw coal ore samples, typically, contain the following:

- Very little colour information (the most prominent colours are yellow and blue)
- Very little homogeneity
- Large amounts of texture information
- No orientation or structural information
- Inconsistent lighting

It is unlikely that a single descriptor, such as pixel brightness, will be sufficient to differentiate between the various amounts of coal minerals. It is, therefore, expected to make use of multivariate data values to group the pixels together according to the mineral components they represent.



Figure 5: Example coal ore sample A

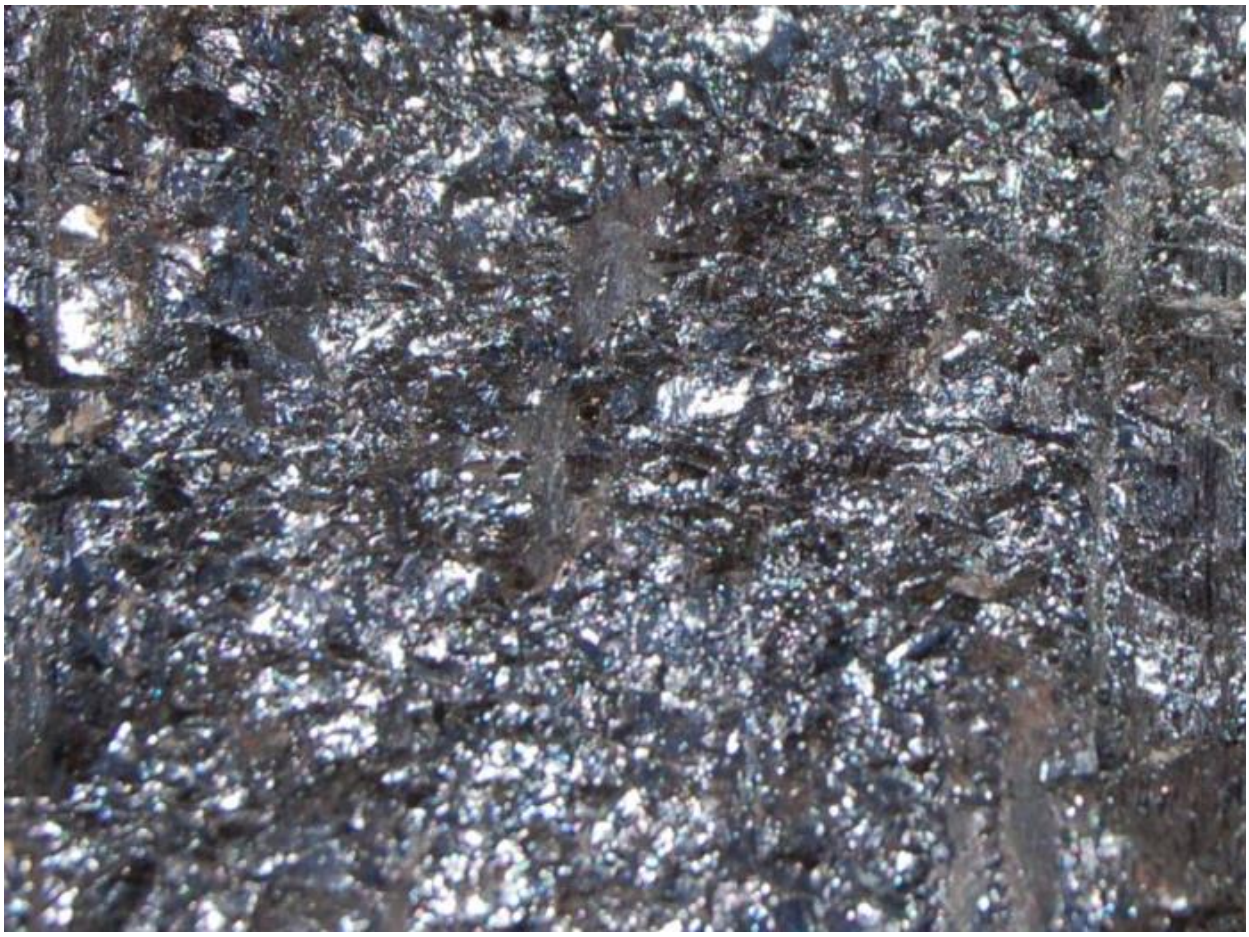


Figure 6: Example coal ore sample B

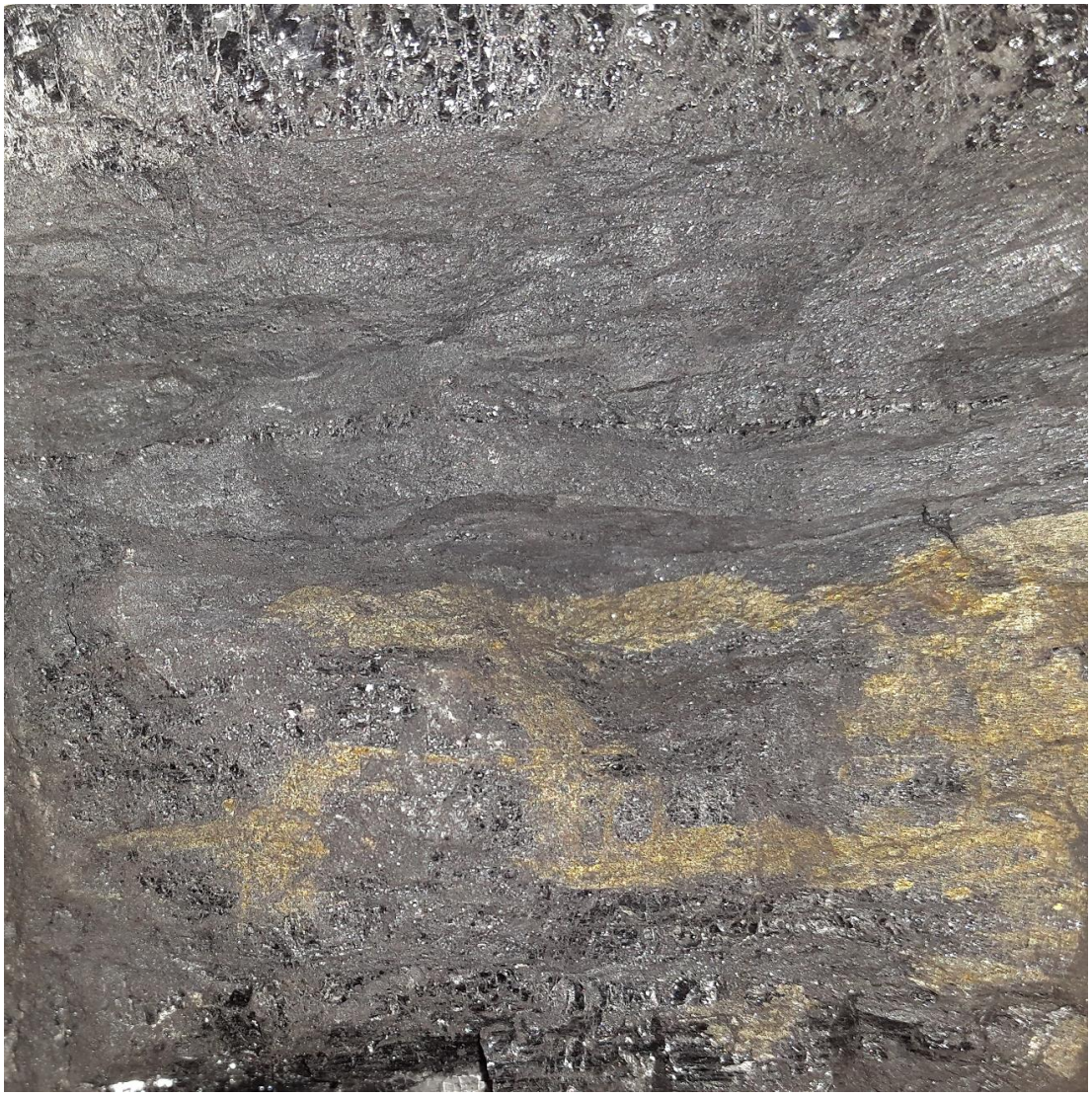


Figure 7: Example coal ore sample C



Figure 8: Example coal ore sample D

1.2. Research definition

The goal of this section is to define and clarify the research presented in this dissertation. The following are some of the key points of discussion:

- A definition of the research problem
- The scope and constraints of the research
- The methodology and motivation behind the verification and validation techniques

1.2.1. Problem statement

The goal of the research presented in this dissertation is to develop a system that can reliably classify a digital image of a raw (unprocessed) coal ore sample according to the materials of which it is comprised. The research done in [1] serves as inspiration. The following sub-goals are identified:

- Reproduce the results found in [1] in a different development environment, while using the same algorithms
- Analyse the system and identify areas of possible improvement
- Identify and implement alternative (or additional) techniques in said areas
- Compare the performance of the modified system with the performance of the original system using valid testing methodologies and controlled datasets
- Investigate the constraints and applicability of the improvements to ensure that the developed systems is at least as robust as the original

The research problem, therefore, boils down to an image segmentation problem. There is an existing system to base the image segmentation methodologies on. Several sub-components in the existing system have been identified (see figure 4). These components are investigated thoroughly to identify possible options for improvement.

1.2.2. Scope

Factors that are within the scope of the research are listed below:

- The investigation and implementation of image segmentation algorithms including sub-fields such as:
    - Feature extraction
    - Clustering
    - Classifying
- Testing and comparison of these algorithms
- Identifying and implementing methods to verify and validate the algorithms' level of accuracy in segmenting a photograph of a raw coal ore sample into its comprising materials

Factors that are outside the scope of the research are:

- Generating a washability curve from the classified image data
- Investigating image segmentation methods that would require large amounts of training data to implement (e.g. artificial neural networks or support vector machines)

1.2.3. Verification and validation

In order to verify the accuracy of the techniques used to segment the image, internal analyses are conducted with the resulting classification data. Metrics such as the Davies-Bouldin index, the Dunn index and silhouettes are used to compare the algorithms. Note that these techniques only verify that the algorithms outperform each other (and the original system) in terms of the metrics. It is, therefore, used purely to compare the algorithms through a common analysis method.

Validation is done by comparing the final classified image to external data. This will test the image segmentation methods' information retrieval capabilities and not just its classification capabilities. The external data takes the form of hand classifications (made by observing the photograph with the naked eye or under a microscope) obtained from expert geologists. These external classifications are accepted as ground truths. Comparing the classification made by the system and the ground truth classification, the overall classification accuracy can then be quantified as well as the accuracy in identifying each specific mineral in the ore sample (through the use of confusion matrices).

1.3. Research logistics

This section provides information regarding the logistical methodologies used to do the research. The following are some of the key points of discussion:

- A practical explanation of the research process
- The structure of the rest of the dissertation

1.3.1. Methodology of practice

The presented research can be classified as mixed methods research and is based on existing research. Consequently, the research methodology reflects that. A qualitative approach was used to explore possible solutions to the research problem and a quantitative approach was used to test hypotheses about the possible solutions. Using the paper [1] and consulting with its main author, the existing system was thoroughly investigated. The motivations behind its design and choice of lower level components were critically analysed. This was done to obtain an in depth understanding of the system in terms of, both, the higher- and lower level components. The mechanisms by which the modules are integrated and any components that were omitted in the paper were also identified.

Once a satisfactory level of understanding was obtained the system was reproduced in the Microsoft Visual studio development environment using the C# programming language. This environment has been specifically chosen instead of Matlab (with which the original research was conducted) because of the researcher’s familiarity with the environment as well as to independently verify the results. During the implementation of the system, special care was taken to keep the software architecture as modular as possible. This was done in order to easily accommodate any alterations that was made in the next phase of the research.

The developed system was then tested to ensure that it performs at least as well as the original with identical data sets. After this was achieved, an extensive literature review was done to identify applicable techniques and algorithms that can be incorporated into the system as alternatives (or additions) with possible advantageous characteristics. The advantages and disadvantages of these alternatives were then carefully considered before they were introduced into the system. Once the alternative component was successfully implemented it was tested using a black box approach.

Alternative components that yielded results that are more favourable to their original counterpart, were then tested as part of the entire modified system and any changes in performance were interpreted. If the resulting system's advantages outweigh the disadvantages (relative to the original) it was taken note of as a viable improvement of the original system. This process of repeated implementation and testing is commonly referred to as rapid prototyping (in terms of software development). Due to the fact that the research is based on existing material and is software based this method of development was more appealing than its structured counterparts (such as using the waterfall life cycle). The approach proved to be well suited to the research problem. A simplified flow diagram of the process described above can be seen in figure 9.

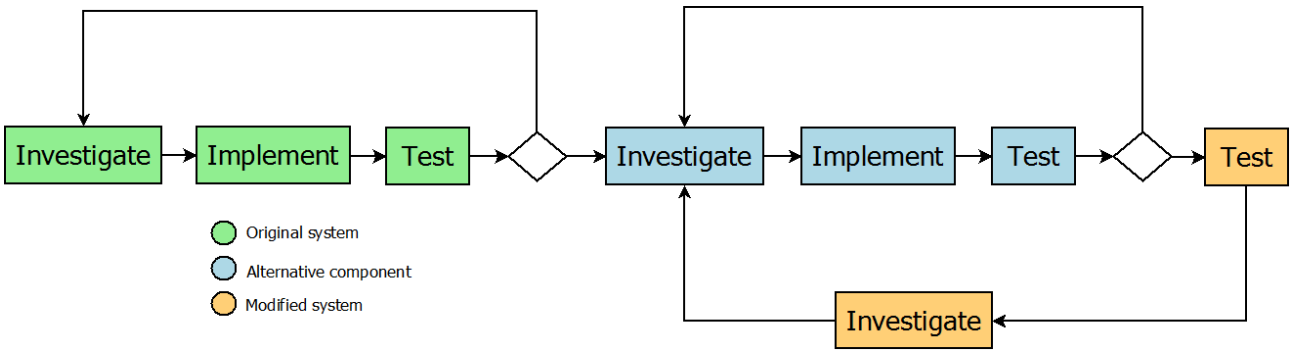


Figure 9: Diagram of simplified research methodology

1.3.2. Document outline

The rest of the dissertation is structured in such a way as to provide the reader with all the information regarding the research process and the motivations and conclusions drawn from it. There will be recurring references back to figure 4 (the main classification flow diagram). This is because of the fact that the research is done in such a modular way, with frequent shifts of focus to different sub-sections of the whole system.

The following chapter (**Image segmentation**) will provide an in depth presentation of the literature review done to investigate different approaches to image segmentation that could be considered to make improvements to the system. Then, a chapter (**Theoretical design**) will identify the methods and techniques that were selected to be incorporated into the system. The fourth chapter (**Testing**) will explain the techniques used to test the system.

This will be followed by a chapter (**Results**) that will provide the results of the previously mentioned tests. The bulk of the dissertation's findings can be found in the results chapter. The penultimate chapter (**Discussion**) will present an investigation, interpretation, and discussion of the results. Lastly, the seventh chapter (**Conclusion**) will close off the dissertation with final remarks regarding the results, a summary of the findings, and some points of interest for further research.

2. IMAGE SEGMENTATION

2.1. The field of image segmentation

The purpose of this section is to provide an overview of the process called image segmentation. It forms part of an interdisciplinary field, called computer vision, and the methods and techniques involved in it are often used in other fields of study as well. The following are some of the key points of discussion:

- A clear definition of an image
- A definition of the image segmentation problem
- A discussion on the different families of image segmentation algorithms

2.1.1. Image

Before defining image segmentation it would be prudent to, first, define an image. An image can be seen as a two dimensional function of feature values: $f'(x, y)$, where the feature value at the spatial coordinates (x, y) is $f'(x, y)$. The type of image is dependent on the type of feature values found in the function. Examples of image types include [14]:

- Light (visual) intensity images have variation of light intensity as feature values
- Range (depth) images have depth information as feature values
- Nuclear magnetic resonance images (MRI) have intensity variation of radio waves, generated by biological systems when exposed to radio frequency pulses, as feature values
- Thermal images have infrared energy as feature values

A digital image is a discrete version ($f(x, y)$) of the image function ($f'(x, y)$), with a digitised coordinate system and feature values. A digital image can, therefore, be thought of as a two dimensional matrix of feature values. Its row- and column indexes identify the feature value in that specific position. This can be defined as:

$$F_{PxQ} = [f(x, y)]_{WxH}$$

where WxH is the size of the image and $f(x, y) \in G_L\{0,1, \dots, L-1\}$ the set of discrete levels of feature values [14]. A photograph would be categorised as a light intensity image, because it depicts a three dimensional scene on a two dimensional space using the light intensity and frequency gathered at each spatial position in the light sensing device.

A digital photograph will be a two dimensional matrix of pixel values and can be defined as:

$$P_{WxH} = [p(x,y)]_{WxH}$$

where W is the image width, H is the image height, and the pixel value $p(x,y)$ is

$$p(x,y) = (R,G,B)$$

where $R,G,B \in \{0,1,\dots,255\}$ assuming the image data is in the red-green-blue colour space.

2.1.2. Segmentation problem

Image segmentation is best defined in terms of set theory. Keep in mind that image segmentation is implementable on any type of digital image, regardless of the type of feature values it contains. A formal description of image segmentation follows [14] [15]:

Algorithm 1: Image segmentation

$If\ I\ is\ a\ set\ of\ all\ pixel\ values\ in\ an\ image$:

$I = \{p_1, p_2, \dots, p_N\}$

$where\ N = image\ width\ x\ image\ height$

$Then\ the\ segmentation\ of\ I\ is\ a\ set\ S, such\ that$:

$$S = \bigcup_{i=1}^{n} R_i$$

$where$

- $R_i \subseteq I, i = 1,2,\dots,n$
- $R_i\ is\ a\ connected\ set,\ i = 1,2,\dots,n$
- $P(R_i) = true, i = 1,2,\dots,n$
- $R_i \cap R_j = \emptyset, \forall\ i \neq j$
- $P(R_i \cup R_j) = false\ if\ R_i\ is\ adjacent\ to\ R_j$
- $n < N$

$Where$

$P(R_k)\ is\ a\ logical\ predicate.$

In other words, image segmentation divides an image into a collection of connected sub-regions of pixels that:

- Do not overlap
- Satisfies some predicate
- Does not satisfy the predicate if it is merged with any adjacent regions

Figure 10 shows an example of a very simple image segmentation exercise. The first panel provides the original digital image, the second panel shows a single threshold algorithm's result (an explanation of this algorithm will be provided on page 18), and the third panel highlights the borders between the segmentation regions in red. The predicate in this case is that all the pixel values in the region must be either above or below some threshold value.



Figure 10: Example of a simple image segmentation exercise

Depending on the predicate used to divide or merge regions within the image, image segmentation can be implemented to achieve a vast variety of results and an equally vast variety of applications. The following paragraphs present and explain various approaches to image segmentation, highlighting their advantages and disadvantages, as well as providing examples of each.

2.1.3. Approaches

Categorising approaches to image segmentation is a deceptively complicated task. Many approaches are derived from each other, or incorporate each other as part of their fundamental internal operations. Some essentially performing the same operations from a different perspective. Typically, surveys elect to categorise the algorithms according to how the algorithms perceive and process the image data [16] [17] [14], while others categorise them according to specific families of segmentation algorithms [18] [19] [20] [21] [15]. However, there is a broadly accepted idea that some algorithms mainly operate in the range-space domain of the pixel data, and some algorithms mainly operate in the spatial domain of the image data. Hence forth, let "range-space" refer to the former, and let "domain-space" refer to the latter.

Range-space algorithms group the pixels into regions based purely on their feature values, usually, with no regard to their position in the image coordinate matrix. The general structure behind these algorithms is:

- Translate the pixel data in the image to some feature space (i.e. a histogram or multivariate cloud of data points)
- Keep track of the coordinate index of each data point in the original image
- Perform the necessary operations on the new data structure and find the desired regions
- Replace the segmented pixel data values back in their original position in the digital image

These algorithms will ensure that the groups of pixels identified within the feature space are very homogenous according to the chosen predicate, but they in no way take into account where in the image matrix these pixels are located. They, therefore, assume that pixels that are coherent in the feature space are coherent in the image space [16].

Algorithms that fall under the domain-space category apply much more emphasis on where the pixels are located in the image matrix. This class of algorithm seeks out patterns and structures in the arrangement of pixel data within the two dimensional image matrix. These algorithms ensure that the regions are well connected with regards to the spatial domain, but not necessarily in terms of all the feature values [16].

The two kinds of segmentation algorithms explained above are, often, further divided into two families of algorithms: region (continuity) focused, and boundary (discontinuity) focused. The former attempts to identify groups of homogenous pixels, and the latter attempts to, first, identify the borders between these groups, thereby identifying the groups indirectly. Figure 11 provides a simple hierarchy of the categories of image segmentation approaches explained above.

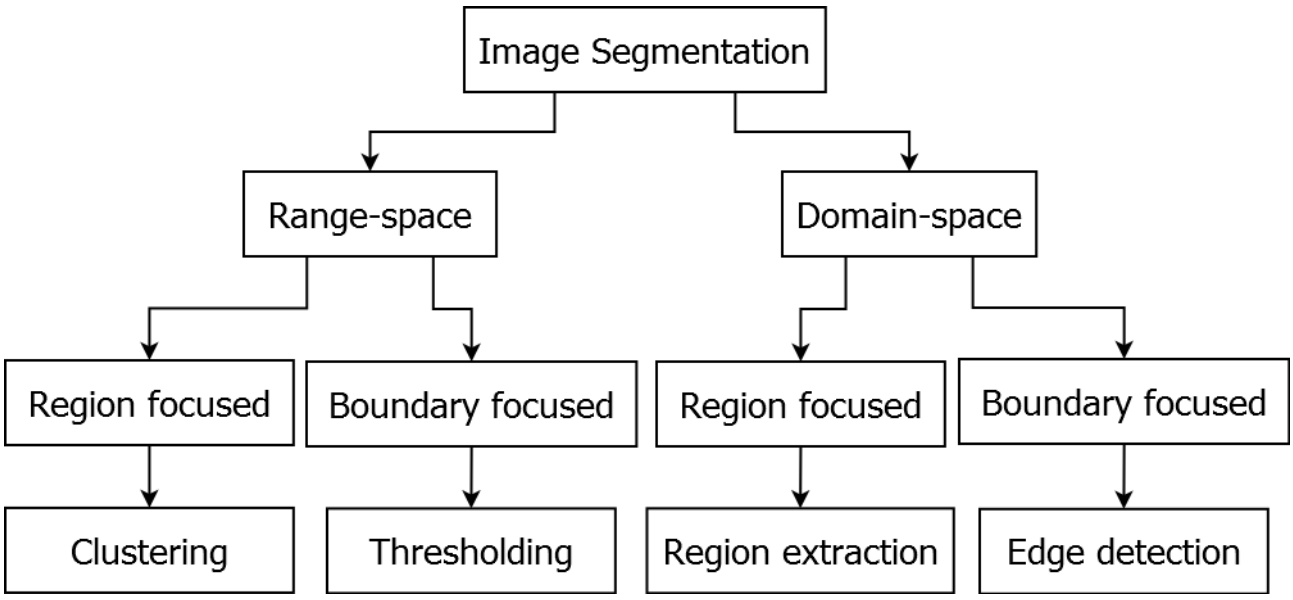


Figure 11: Diagram of image segmentation categories

The lowest level of the hierarchy in figure 11 lists some of the most used methods of image segmentation according to the category to which they belong. Take note that these are by no means the only methods available in the literature. They have been chosen, for consideration, because they are found readily in the literature, meaning that they are applied in the most practical implementations and have the lowest chance of being esoteric and constrained with regards to their applicability to the research problem. Furthermore, most of these methods have several variations and subcategories that will be explained when needed.

Before moving on to the discussions on the methods of image segmentation some terminologies need to be explained. These terms will be used, frequently, for the remainder of this chapter:

- Image

  From this point on, when referring to an "image", the concept that is being referred to is a two dimensional digital image with unspecified width and height, with an unspecified type of feature value at each point in the two dimensional matrix.

- Matrix

  A rectangular two dimensional grid/array of equidistant points. The width of the matrix corresponds to the number of indexes arranged from the leftmost index to the rightmost index in a horizontal line. The height corresponds to the number of indexes arranged from the topmost index to the bottommost index in a vertical line.

- Coordinates

  The location of the pixel in the image matrix. This is usually given by a set of (x, y) coordinates. The x-coordinate dictates where the pixel is located along the horizontal axis of the image, and the y-coordinate dictates where the pixel is located along the vertical axis.

- Feature value

  This is the value of a pixel given its coordinates. It can be a single value or it can be a multidimensional value. The discussion on image segmentation approaches will be kept as generalised as possible, in order to accommodate both cases.

- Regions

  Regions refer to groups of pixels that are connected in the image matrix. These regions are not necessarily segmented into a group unless it is specified.

- Classes

  Many image segmentation algorithms, especially the range-space algorithms, segment the image into regions by identifying classes. The pixels belonging to a class, typically, have some characteristic in common with regards to their feature values. Classes are not to be confused with segmented regions. Classes are groupings of pixels based on their feature values, and regions are groupings of pixels based on their location in the image matrix.

- Feature space

  The abstract space the pixels occupy when their location in the image matrix is disregarded is called the feature space. Its dimensionality matches that of the feature values it contains.

For clarification, figure 12 provides a visual presentation of the above listed terminologies. The presentation depicts a digital image that has been segmented into eight regions according to five classes. Take note that the feature space has three dimensions, inferring that the image also has three dimensional feature values.

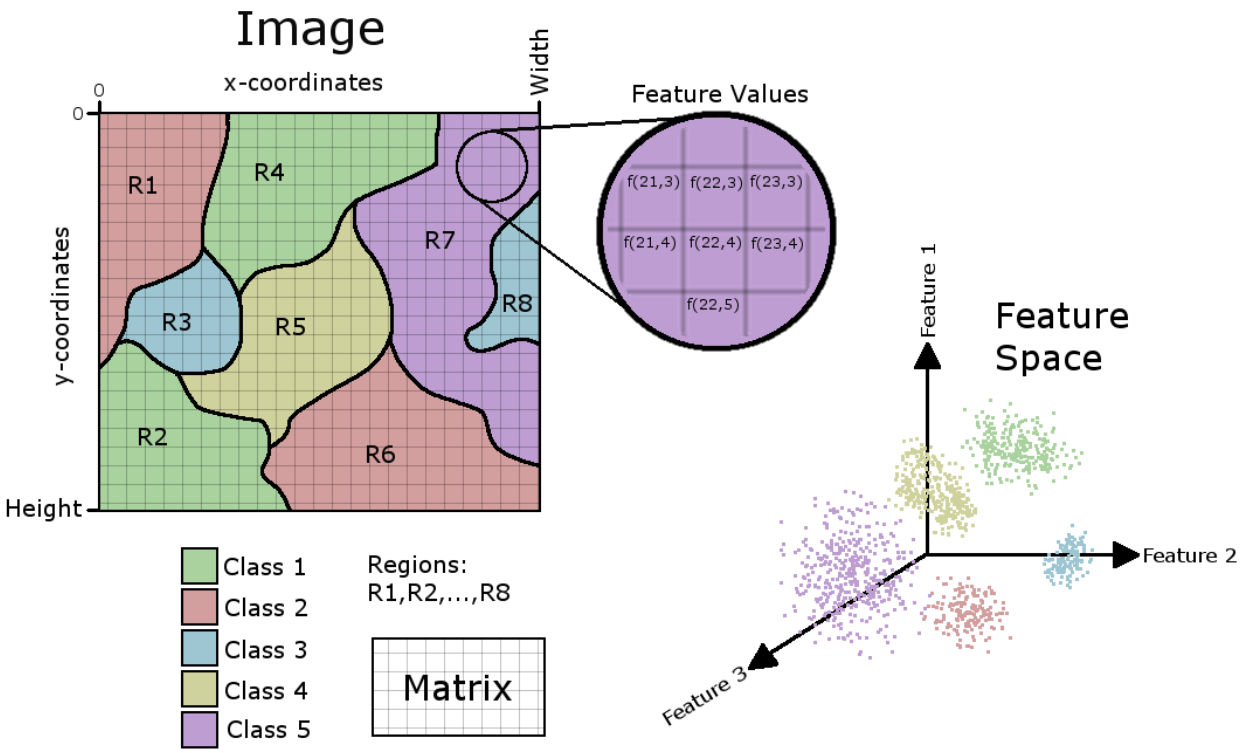


Figure 12: Depiction of image segmentation terminologies

2.2. Image segmentation by thresholding

The purpose of this section is to investigate the thresholding approach to image segmentation. This is the approach that was used to implement the segmentation in figure 10. The following are some of the key points of discussion:

- The fundamental ideas behind thresholding
- Various methods of implementing thresholding
- Problems with thresholding
- A summation of the advantages and disadvantages of thresholding

2.2.1. Fundamentals of thresholding

The first method of image segmentation to be discussed is usually the thresholding method. This is because it is widely regarded as the simplest to understand and implement. Notice that in figure 11, thresholding falls under the range-space category, meaning that it groups pixels based on their feature value instead of their location in the image matrix. It is also categorised into the boundary focused type of algorithm. The algorithm, therefore, focusses on finding the borders between similar feature values in the feature space. This algorithm works with the postulate that regions of one class fall above some threshold feature value and regions of another class fall below said threshold [15]. It can, however, be expanded to include multiple thresholds to generate more classes. A formal description of the general procedure of thresholding follows:

Algorithm 2: Thresholding

$Given\ some\ image\ I\ with\ feature\ values\ f(x,y).$

$Also\ given\ a\ set\ of\ N\ threshold\ feature\ values:$

$T = (t_1, t_2, \dots, t_N)$

$where\ t_1 < t_2 < \cdots < t_N$

$and\ t_1 > minimum\ f(x,y)\ in\ I$

$and\ t_N < maximum\ f(x,y)\ in\ I$

$Then\ each\ feature\ value\ in\ the\ segmented\ image\ I':$

$$f'(x,y) = \begin{cases} C_0\ if\ f(x,y) < t_1 \\ C_{i,j}\ if\ t_i \leq f(x,y) < t_j\ , i+1 = j \\ C_k\ if\ f(x,y) \geq t_N \end{cases}$$

$where\ C_0\ is\ a\ constant\ assigned\ to\ features\ below\ t_1$

$and\ C_{i,j}\ is\ a\ constant\ assigned\ to\ features\ between\ the\ ith\ and\ the\ jth\ threshold$

$and\ C_k\ is\ a\ constant\ assigned\ to\ features\ above\ t_N.$

Notice that the number of classes that will be identified in the segmented image is one more than the number of thresholds used. For an example of the thresholding algorithm in its simplest form refer back to figure 10. In this example only one threshold is used. Take note that all the pixels that fall below the threshold are presented as black pixels and all the pixels that fall above the threshold are presented as white pixels in the middle panel. This, essentially, creates two classes: black and white.

In practice, thresholding with a single threshold value is usually done in an implementation of background removal (or object segmentation). This is the task of differentiating between an object and the background in the image. This implementation assumes that there is a very clear difference, in whatever feature value the image contains, between the object and the rest of the image [22]. Additional image processing might be necessary if the object- and background regions aren't entirely homogenous, but single thresholding with a well selected threshold value is a fundamental part in most approaches to background removal.

In a scenario where multiple objects, or regions of importance, need to be differentiated, it stands to reason that multiple thresholds need to be used. This is commonly referred to as multi-level thresholding. If thresholding is to be used for the research problem it would have to contain multiple levels.

The most difficult aspect of this technique is to determine the thresholds. The thresholds can be defined based on global parameters or local parameters. Many of the methods of obtaining the thresholds involve implementing techniques used in other approaches to image segmentation, such as histogram based methods or cluster analysis. Thresholds can be determined using the following methods [22]:

1. analysing the peaks, valleys, and curvatures of histograms of the image
2. using simple clustering techniques
3. analysing the entropy contained in different regions of the image
4. comparing the grey-level and binarized images (object attribute based)
5. using spatial methods that analyse high-order probability distribution and correlation between pixels
6. considering the local characteristics of each pixel's immediate vicinity (local methods)

2.2.2. Thresholding algorithm types

The first method of finding thresholds is by means of investigating a **histogram** generated from the image data. A histogram is a way of presenting data as an estimated probability distribution of the data. Mathematically, a histogram is a function with the value determined by the number of observations belonging to the index value of the function. Therefore, in order to generate a histogram from an image, the feature values would be grouped into "bins" in an ascending order. Each bin would represent a possible feature value. Figure 13 provides an illustration of a histogram generated from the given grey scale image.

The horizontal axis of a histogram contains the possible values an observation can have. The vertical axis represents the number of observations, within the set, having the value at each point on the horizontal axis. Take note that in figure 13, the feature values range from 0 to 255. This is the possible values a pixel in a grey scale digital image can have. It can, therefore, be deduced that the sum of the area of the histogram would be equal to the number of pixels within the image.

The image histogram in figure 13 is slightly skewed to the left and multimodal. This means that the majority of the pixels in the image have higher than average values with several peaks and valleys present in the bulk of the distribution. The five histogram-image pairs following the first pair have areas within the histogram, and the corresponding areas in the image, highlighted in red. This is provided to illustrate the translation of pixel data from the image to the histogram. Notice that:

- Lower feature values in the histogram correspond to darker pixels
- Higher feature values in the histogram correspond to brighter pixels
- The larger the highlighted area in the histogram, the larger the corresponding image area
- Certain peaking sections of the distribution that are divided by valleys, often, correspond to adjacent homogenous regions in the image

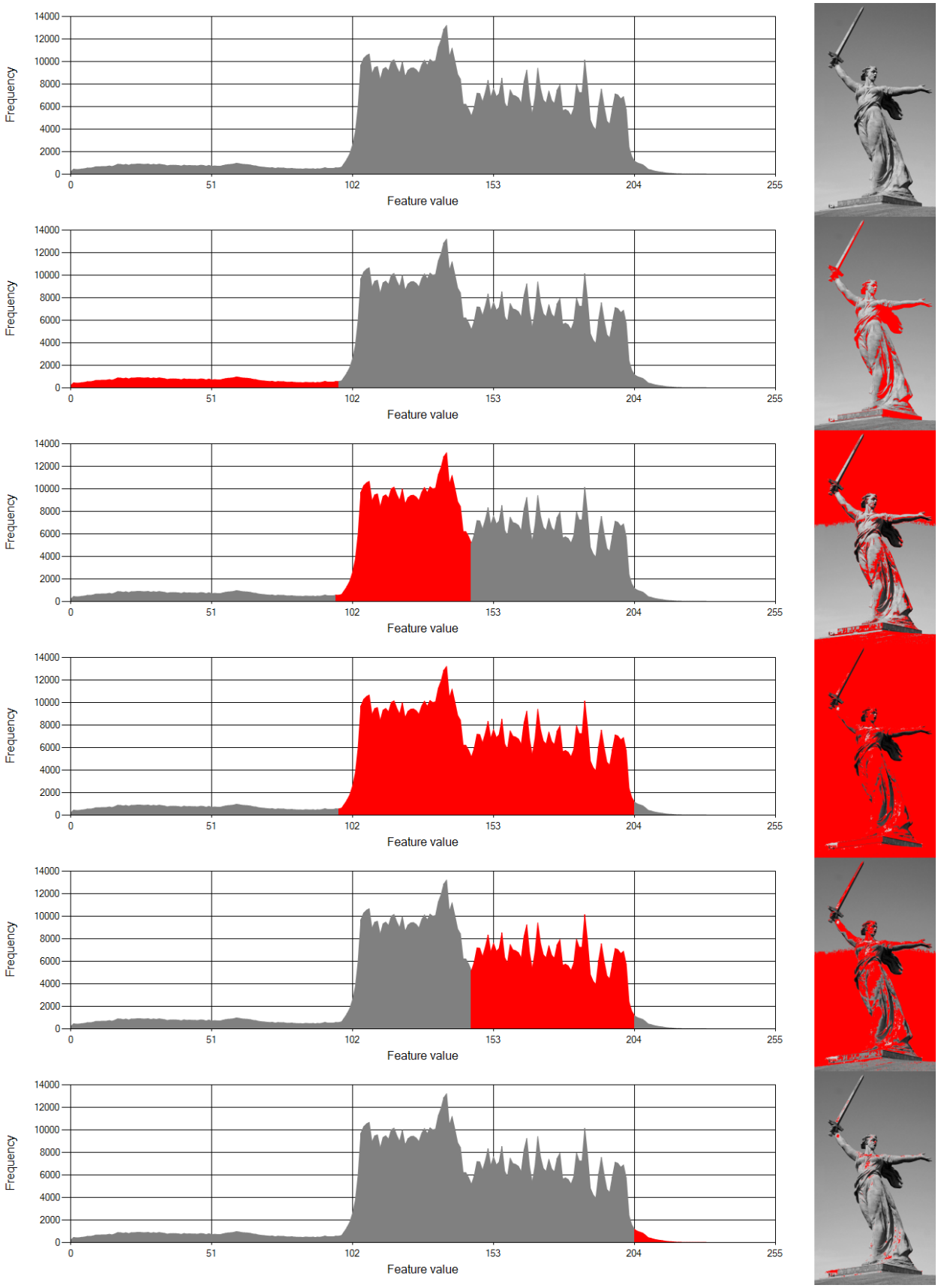


Figure 13: Example of a histogram

Thresholds are usually placed at the deepest part of the valleys, separating the peaks, in a histogram because these correspond to the boundaries between two adjacent homogenous regions of the image. Algorithms that detect the peaks and valleys in the histogram, such as the one presented in [23], are used to determine the appropriate location for thresholds in order to classify the pixel data into groups of similar feature values. These classes are then used to segment the image into regions.

The second way of identifying appropriate threshold values is by using simple **clustering** techniques. Using clustering techniques will be extensively discussed during the clustering approach to image segmentation. For now, it suffices to say that when satisfactory clusters have been identified they are used to place the thresholds in the feature space.

The third method of identifying threshold values is through the use of image **entropy** information. With regards to image processing, entropy refers to the "busyness" or "randomness" of the pixel values in the image. A typical metric for this is the Shannon entropy, which is defined in equation (1) [24].

$$S = -\sum_{i=1}^{L} p_i \log_2(p_i) \qquad (1)$$

where S is the entropy value of the image, L is the total possible feature values a pixel in the image can have, and p_i is the probability (usually generated with a histogram) associated with the i_{th} feature value. This equation originates in information theory and is an estimation of the uncertainty of the outcomes of an experiment [25]. The equation also provides an estimate for the minimum amount of data needed to convey information in signal compression [26]. Figure 14 shows the Shannon entropy values of five grayscale images. Notice that the entropy value increases with more complex arrangements and wider ranges of feature values. Images containing more entropy are less predictable, requiring more bits to encode (because more information needs to be stored) and is, therefore, less compressible.

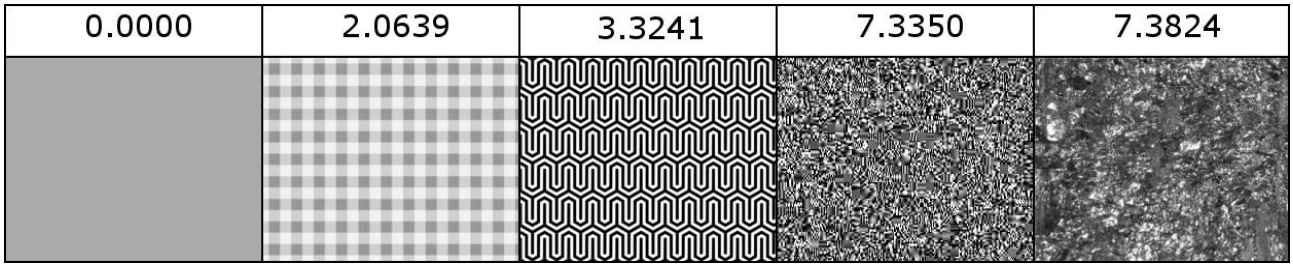


Figure 14: Image entropy illustration

The optimal thresholds can be found by investigating the entropy values of the regions that have been classified according to the thresholds. Approaches to this include [22]:

- Maximising the total entropy (sum of the entropy values of the classes) in the thresholded image, thereby maximising information transfer
- Minimizing the cross-entropy between the original image and the thresholded image, thereby maximising information preservation

The fourth approach to identifying the appropriate thresholds is based on **object attributes**. This means that the thresholds are chosen in such a way that a high similarity remains between the original image and the thresholded image in terms of some similarity measure. Examples of such a similarity measure are grey-level moments, edge matching, and shape compactness [22]. The researchers in [27] used edge-matching to choose the threshold value so that the edges of the original image matches the edges of the thresholded image as closely as possible. This ensures that the segmented image retains the shape of the object in the original image.

Figure 15 provides an illustration of the idea behind object attribute based threshold selection. The first panel is the original grey scale image, the second panel presents the edges of the original image, the third panel is a thresholded version of the original image, the fourth panel shows the edges of the thresholded image, and the final panel presents the edges of panel four (in green) superimposed onto the edges in panel two (in red). By choosing the threshold value for panel three such that the edges in panel five remain as similar as possible, it is possible to more accurately segment the object from the background. Note that in panels two and four the edge values have been made black and the background made white for better visual illustration. In practice the edges are depicted by white pixels on a black background. This is because of the fact that higher pixel values are represented by lighter components and lower pixel values are represented by darker components.



Figure 15: Example of edge matching

The fifth type of algorithm used to choose threshold values involves using spatial information about each pixel in the image matrix. These "**spatial methods**" differ from the four previously discussed methods in that they attempt to incorporate information about each pixel's immediate vicinity into the global threshold values. A very simple example is presented in [28] where an image histogram is adjusted by applying more emphasis on pixel values that are very similar to the other pixel values in their immediate vicinity. This results in a histogram with deepened valleys, easing the processing of placing an appropriate threshold. Another method of incorporating local information is to use a two dimensional histogram (also called a probability mass function) of the image (generated from the pixel values and the mean pixel values in the local neighbourhood of each pixel) to generate two dimensional entropy values for entropy based thresholding [29].

The final class of threshold finding algorithms is distinct from the others, because these algorithms define a threshold for each pixel in the image. **Local methods** have the advantage of inherently taking spatial information into account. This is because the threshold value is different for pixels in different location in the image matrix. Using global threshold values can sometimes result in the inclusion of pixels into the class that are similar in terms of the feature values, but clearly different in terms of their location in the image. Pixels can also be excluded from a class because they have dissimilar feature values to the class, even if they are encapsulated by the pixels of the class. These errors are reduced by defining different thresholds for different regions of an image. Another advantage of local methods is that they alleviate some of the issues that arise when there is a variation in illumination in the image. Regions of the image containing shadows are often regarded as a different class to what they actually are and the same happens for regions with excessive lighting. A robust local thresholding model counters these issues to a certain extent [30].

Figure 16 presents a scenario where local thresholding is advantageous. The first panel in the figure is the original grey scale image. The goal is to segment the image into two classes: black for the text and white for the background. The second panel is the global thresholding of the image by maximising the Shannon entropy in the two classes. Take note that the shaded background resulted in a suboptimal segmentation. The third panel is a segmentation by using a local thresholding method. The exact local thresholding method used is called Niblack thresholding [31]. This method assigns thresholds to each pixel with the formula presented in equation (2).

$$T(x,y) = m(x,y) + k\sigma(x,y) \tag{2}$$

where $T(x,y)$ is the threshold value assigned to the pixel at (x,y), $m(x,y)$ is the mean feature value of the neighbourhood of the pixel, $\sigma(x,y)$ is the standard deviation of feature values in the neighbourhood of the pixel, and k is a biasing constant. The value for k is calibrated according to the size of the selected neighbourhood of the pixel. In this specific instance the neighbourhood was a 35 by 35 pixel grid centred around the pixel at (x,y), and the value for k was -0.8. For reference, the entire image consists of 300 by 300 pixels.



Figure 16: Global- and local thresholding example

Notice that the local thresholding method did not misclassify the darker portion of the shaded background. This is because the global thresholding method considers the entire image histogram when selecting the threshold, and the local thresholding method only considers the feature values in the immediate vicinity of each pixel (thereby disregarding the large difference between the lower and the upper portion of the background). It is for this reason that local thresholding methods are commonly used to segment text from backgrounds that are not uniform.

2.2.3. Issues with thresholding

The discussion on thresholding up to this point has neglected to mention the possibility of the image having multidimensional feature values. Thresholding is an algorithm that has been developed for grey scale images. This means that it is designed to be implemented on images with one dimensional feature values. As a result, it is very limited with regards to multidimensional feature values [21]. Nonetheless, attempts have been made to incorporate multiple dimensions into the algorithm. This kind of thresholding is called multi-band thresholding. The problem with multi-band thresholding can be seen in figure 17. The figure depicts the thresholding of a one dimensional-, two dimensional-, and three dimensional feature space. These dimensions have been chosen because they are the most naturally comprehended by humans.

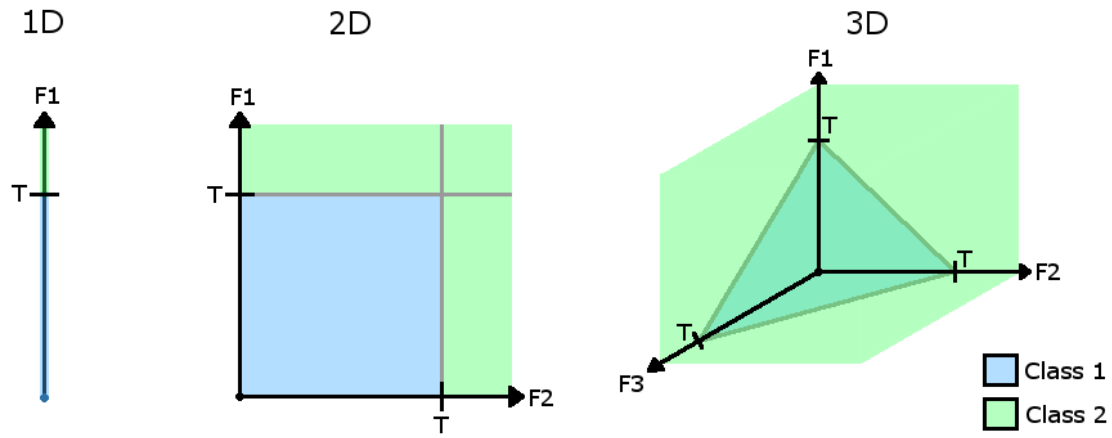


Figure 17: Single level threshold feature spaces

By definition, thresholding partitions the one dimensional feature space into N+1 classes. With N referring to the number of thresholds. This type of feature space partitioning is perfect for one dimensional data, because there is very little concept of “shape” in one dimension. The problem of implementing an algorithm intended for one dimensional data on higher dimensional data can be observed in the corresponding partitioning of the two- and three dimensional feature spaces in figure 17. The shapes of the areas allotted to each class is very restrictive. The two dimensional data is grouped into a rectangular portion, and an “L” shaped portion. The three dimensional data is grouped into one of two shapes, a triangular area and the rest of the space. The desired groupings of data seldom fit into these shapes [32]. The problem becomes more prominent with multi-level thresholding. Figure 18 presents a multi-levelled depiction of figure 17.

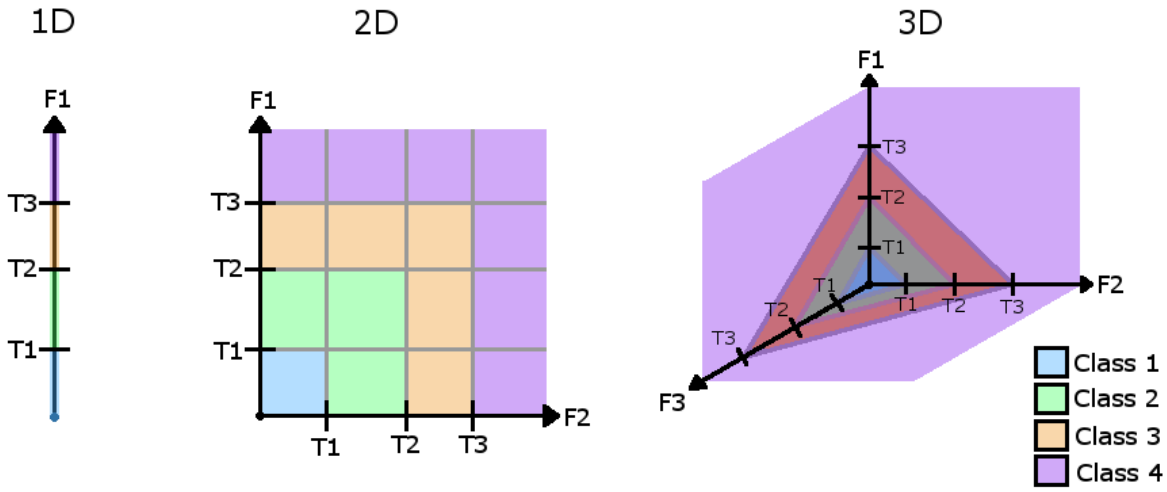


Figure 18: Multi-level threshold feature spaces

From figure 18 it can be seen that multi-levelled thresholding of multi-dimensional data constrains the shapes of data classes even further. Two dimensional data is grouped into “L” shapes, ascending in size, and three dimensional data is grouped into ascending slivers of a pyramid. In order to reduce the constraint on area shapes, one can attempt to define different threshold configurations for each dimension separately. An example of this is shown in figure 19.

Using separate sets of thresholds for each dimension improves the freedom with which areas can be defined within the feature space, however it does not prevent the rectangular and triangular nature of the defined shapes. It also drastically complicates the algorithm. Notice that the amount of defined classes in this case far exceeds the N+1 value according to the definition of thresholding. In [33] colour images were segmented using separate sets of thresholds for each colour dimension. However, in order to obtain appropriate threshold values, particle swarm optimisation with a fitness function based on fuzzy entropy information was used.

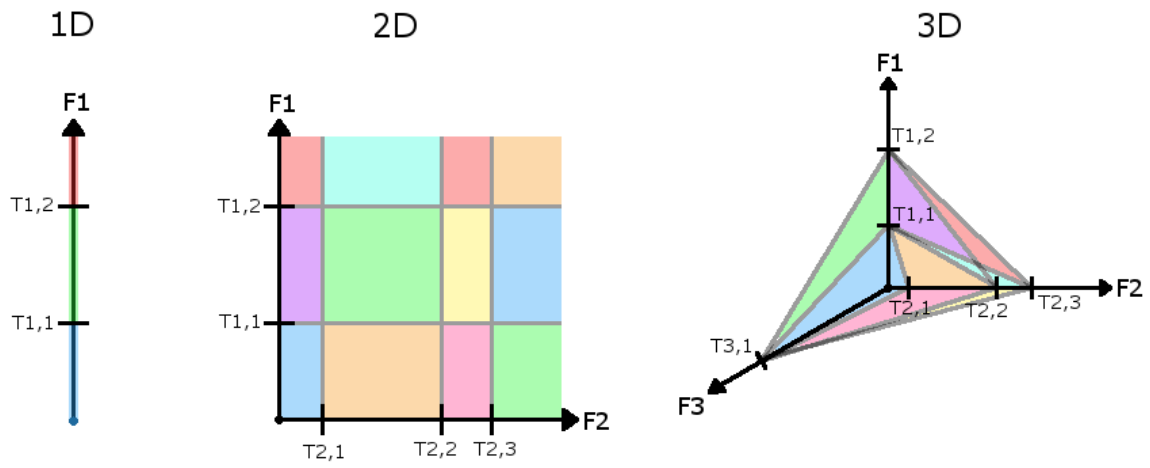


Figure 19: Adjusted multi-level threshold feature spaces

It is for the reasons explained above that simple thresholding algorithms should be avoided with higher dimensional data. Thresholding in higher dimensional feature spaces should be approached with a distance calculation (e.g. using a static Euclidean distance from some exemplar feature value, as threshold, to determine which class the pixel is to be assigned to) [34]. However, if the data complexity reaches a point where multi-level and multi-band thresholding is needed, alternative algorithms should be sought, such as cluster analysis techniques. Cluster analysis is much more suited to distance-based classifications.

2.2.4. Summary of thresholding

Table 1 provides a summary of the advantages and disadvantages of using thresholding as a means to segment an image.

Table 1: Thresholding criterion

Advantages	Disadvantages
• The fundamental operations aren't computationally complex • Excellent for background removal or object segmentation • Usually does not require any prior information about the image data • Good for real-time applications	• Very limited with data having more than one dimension • Finding the right threshold values can be non-trivial • Requires images with large homogenous regions corresponding to histogram peaks • Little (if any) spatial information is considered • Noise sensitivity

Thresholding is most effective in images where there are large (but few) regions of high similarity and it is, therefore, commonly used to separate objects in an image from the background. The image segmentation method used to address the current research problem will have to classify complicated data sets with little in the way of homogeneity. This approach is, therefore, not considered a fitting method of image segmentation.

2.3. Image segmentation by edge detection

The purpose of this section is to investigate the edge detection approach to image segmentation. The following are some of the key points of discussion:

- The fundamental ideas behind edge detection
- Various methods of implementing edge detection
- Problems with edge detection
- A summation of the advantages and disadvantages of edge detection

2.3.1. Fundamentals of edge detection

The second method of image segmentation to be discussed is edge detection. Where thresholding focusses on finding boundaries between homogenous groups of pixels in the feature space, edge detection attempts to find the boundaries in the domain space. Consequently, in figure 11, edge detection falls under the domain-space and boundary focussed categories. As such, this type of algorithm does not define classes according to which the image is segmented. Instead, it identifies the segmented regions directly and then assigns classes if necessary.

An edge is a sharp local change in the feature values between adjacent pixels in the image [34]. These edges usually occur on the boundary between two relatively constant regions of feature values [35]. Figure 20 provides illustrations of three kinds of edges. The first edge is a step edge, the second is a ramp edge, and the third is a roof edge [34]. The graph underneath each edge is an intensity profile corresponding to the edge depicted above it.

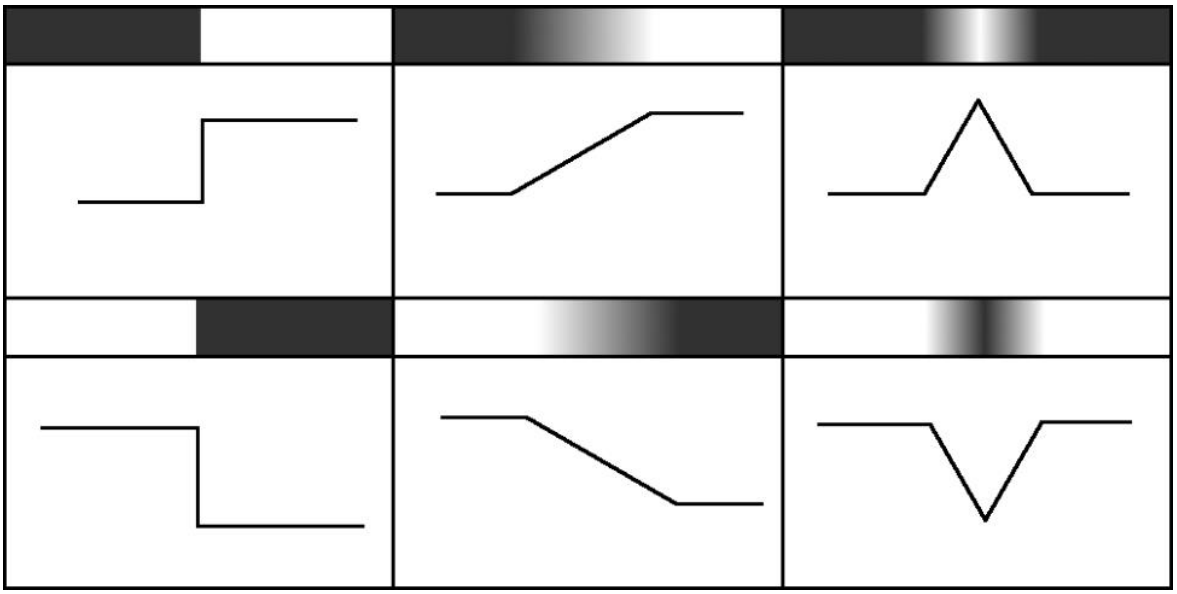


Figure 20: Examples of step-, ramp-, and roof edges

Image segmentation through edge detection works with the assumption that the desired segmented regions of pixels are separated by discontinuities in the feature values within the image matrix. These discontinuities are referred to as edges. Edge detection is a well-researched field in image processing, with many methods of identifying edges already developed. The ideal case for this kind of image segmentation can be seen in figure 21. The first panel is the input image. The second panel shows the edges (in white) contained in the input image. The third panel shows the six regions that have been segmented by filling in the areas defined by the extracted edges. Matching the identified regions with each other according to their similarity in feature values is a trivial task. This is provided in the final panel. This approach to image segmentation is very similar to how humans perceive objects [21].

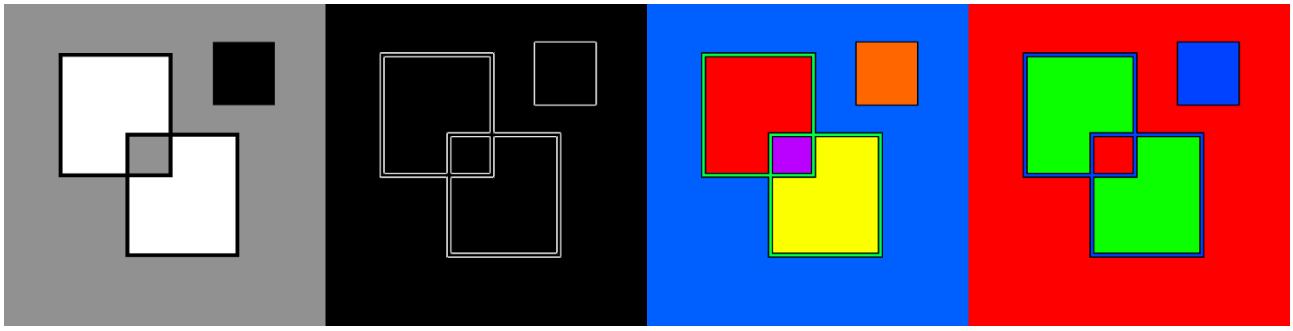


Figure 21: Ideal image segmentation with edge detection

The example presented above is, of course, an unrealistic and oversimplified scenario. The edges contained in real digital images are degraded by factors such as [36]:

- Photon noise
- Blurring/defocusing
- Irregularities of the objects' surface

These factors result in false edges being generated, edges being omitted, and/or multiple edges being assigned to a single true edge. These issues rarely allow for a region in the image to be encapsulated by a solid edge boundary, as is the case in figure 21. For this reason many edge detection algorithms have been developed to compensate for these factors. The general procedure that is followed by an edge detection image segmentation algorithm is as follows:

1. Find the edges contained in the image
2. Identify the regions bordered by the edges
3. Assign classes to these regions if necessary

The most important operation is finding the correct edges and consequently, the most research has been done towards this end.

2.3.2. Edge detection algorithm types

In the literature, there are two types of edge detection algorithms: parallel and sequential. Parallel edge detectors decide whether a pixel is part of an edge based on its feature value and the feature values of the other pixels in its neighbourhood. This decision is never dependant on whether other pixels have been designated as edge pixels. This means that the algorithm can be implemented at several (or all) pixels at the same time. Sequential edge detectors assign pixels to edges by taking the results of the previously processed pixels into account [21] [36].

Sequential edge detectors are inherently slower to implement due to the dependence of their decision making structure. Each result affects two factors: which pixel will be investigated next, as well as the result of the next pixel. These edge detectors are also very dependent on the choice of initial parameters. Assumptions are made to decide on the starting position of the algorithms. [36] Considering the complex nature, and lack of prior knowledge, of the digital images that will be segmented in this dissertation, it was decided to focus on parallel edge detectors.

Parallel edge detection algorithms make up most of the literature. These algorithms include spatial frequency filtering and directional differentiation. **Spatial frequency filtering** is done by implementing a two dimensional Fourier transform on the image data. This translates the image (a two dimensional spatial function) into the frequency domain. The frequency data is then filtered using a high-pass filter. This is done because high frequency components correspond to abrupt changes in feature values in the spatial domain. An inverse Fourier transform is then implemented on the filtered data to reconstruct the image with the lower frequency components reduced or removed.

Figure 22 provides an example of using spatial frequency filtering to detect edges. The first panel represents the input image; the second panel is the magnitude plot of the complex frequency data with the low frequency components removed (low frequency components are situated near the centre of the plot); the final panel shows the filtered and reconstructed image. In the last panel the edges are highlighted and the smooth areas are reduced to featureless regions.

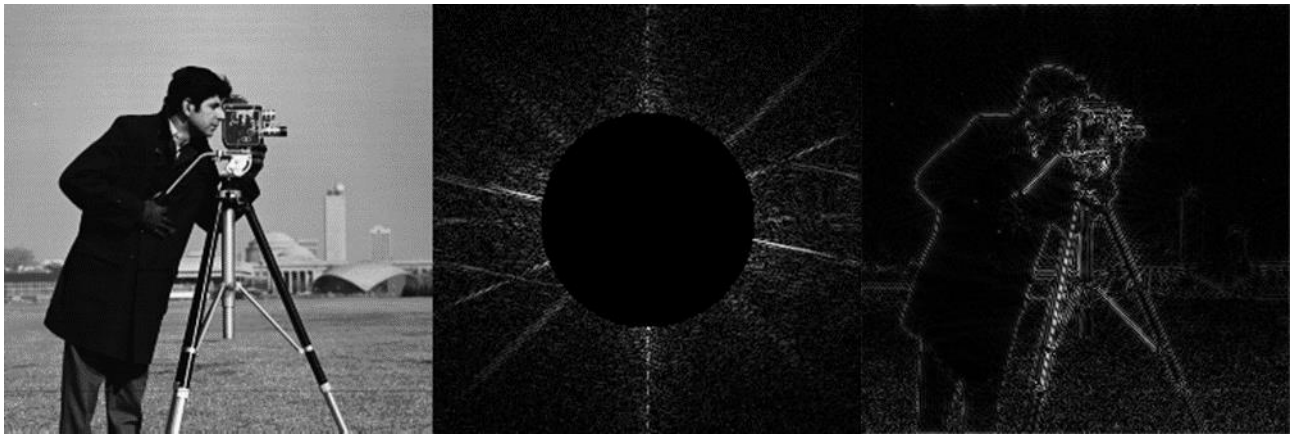


Figure 22: Example of spatial frequency filtering

The Fourier transform is implemented by using a Discrete Fourier Transform, presented by equation (3) [34].

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{3}$$

where $F(u,v)$ is the frequency component of the pixel $f(x,y)$ in an M by N size digital image. The inverse Fourier transform is obtained with the Inverse Discrete Fourier Transform, presented by equation (4) [34].

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\, e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{4}$$

The other type of parallel edge detectors are the directional differentiation algorithms. In contrast to spatial frequency filtering, which finds the edges by processing frequency data, directional differentiation finds the edges in the spatial domain directly. These algorithms investigate the difference between adjacent pixels in a specific direction. Given a pixel at (x,y), the absolute difference between the pixel's feature value and the feature value of the pixel to the right of it can be used as indication of whether the pixel is part of an edge [36]. Using this concept, the following values can indicate the presence of an edge in the specified direction:

$$\begin{gathered} horizontal\ edge\ to\ the\ right\colon |f(x,y) - f(x+1,y)| \\ horizontal\ edge\ to\ the\ left\colon |f(x,y) - f(x-1,y)| \\ vertical\ edge\ to\ the\ bottom\colon |f(x,y) - f(x,y+1)| \\ vertical\ edge\ to\ the\ top\colon |f(x,y) - f(x,y-1)| \end{gathered} \tag{5}$$

A high absolute difference in feature values in the indicated direction means that there is, likely, an edge there. The formulas, listed above, assume that the coordinate system of the image matrix is labelled from the left of the image to the right for horizontal coordinates, and from the top to the bottom for the vertical coordinates. Using the difference between adjacent pixel values, several differential operators have been developed to approximate the gradient in an image. The gradient of an image is a vector property that defines the strength and direction of the differences between pixels. The gradient is defined with equation (6) [34].

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{6}$$

It is, therefore, a vector (composed of the discreet differential, $\frac{\partial f}{\partial x}$, between a pixel and its horizontal neighbours, and another discreet differential, $\frac{\partial f}{\partial y}$, between it and its vertical neighbours) that points in the direction of the greatest rate of change in feature values for a given pixel. The strength of the gradient can be calculated by obtaining the magnitude of the vector:

$$M(x,y) = \sqrt{{g_x}^2 + {g_y}^2} \tag{7}$$

The direction of the gradient can be calculated by obtaining the angle of the vector:

$$\alpha(x,y) = \tan^{-1}\left(\frac{g_y}{g_x}\right) \tag{8}$$

Three of the most famous differential operators are the **Roberts cross** [37], **Sobel operator** [34], and **Prewitt operator** [38]. These operators use two kernels that are convolved with the image matrix to produce the horizontal differential-, and vertical differential in the image gradient. These kernels are presented with equations (9) to (11).

Roberts cross:

$$g_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}; \qquad g_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}; \tag{9}$$

Sobel operator:

$$g_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}; \quad g_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}; \tag{10}$$

Prewitt operator:

$$g_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}; \quad g_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}; \tag{11}$$

Figure 23 provides an illustrative example of how a 3x3 kernel is convolved with a single pixel in an image to produce a resulting horizontal- or vertical differential value for that pixel. The resulting value is the summation of the products of the kernel values and feature values (in reverse order) in the pixel's neighbourhood. The Roberts cross, which uses a 2x2 kernel, would have the specific pixel centred on the first value in the kernel, instead of the centre value.

This operation can be repeated for every pixel in the image, with both the g_x and the g_y kernels, to produce a horizontal and vertical gradient for the image. These values can then be used to calculate the magnitude of the gradient. This magnitude value is an estimation of the edge strength contained by each pixel. Figure 24 shows the results of convolving the kernels in equations (9) to (11) with the entire image in figure 23, as well as the magnitude value (obtained with equation (7)) resulting from each. Take note of the following in figure 24:

1. Both the Roberts kernels emphasise horizontal and vertical edges due to the diagonal nature of the kernels, while the Sobel and Prewitt operators emphasise the horizontal and vertical edges separately with their kernels.
2. The Sobel and Prewitt operators produce extremely similar results due to the similarity of their kernels.
3. The magnitude of these gradients provide very good estimations of the edges in the image.



Figure 23: Example of kernel convolution

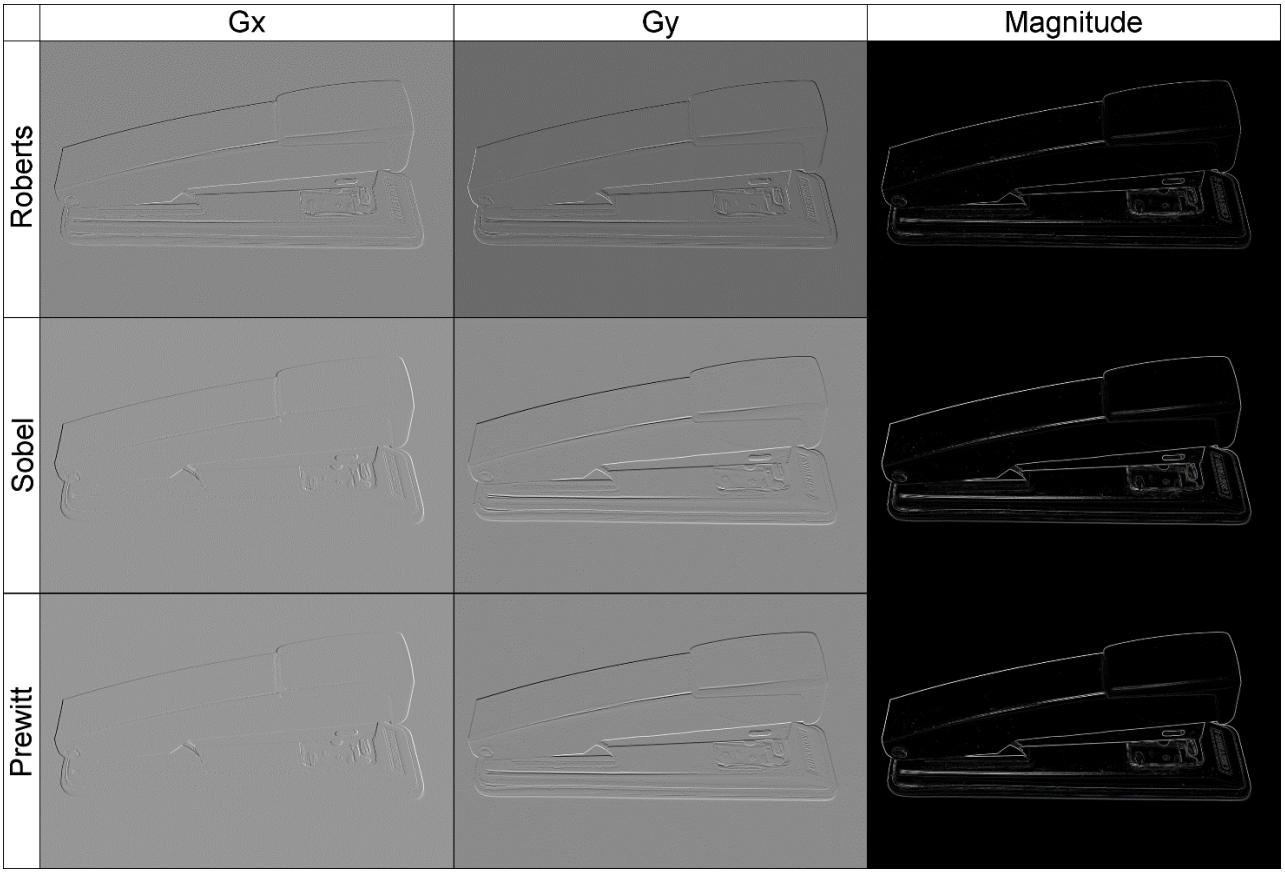


Figure 24: Example of differential operator edge detection

The three differential operators listed above approximate the first differential of the image data. They make use of small static image kernels. It has been argued that using small kernels is insufficient for optimal edge detection because discontinuities in feature values are dependent on image scale. Therefore, an alternative, and easily scalable, approach was sought. One option is the second derivative in the form of a two dimensional Laplacian of the image data:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{12}$$

This two dimensional **Laplacian** can be approximated with the kernel in equation (13).

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{13}$$

Convolving this kernel with the image in figure 23 produces the results in figure 25. The main shortcoming of using the second derivative can be observed in figure 25. This operator is very susceptible to noise. This is due to the fact that it approximates the second derivative and not the first. The second issue is that the Laplacian has no concept of the edge direction.

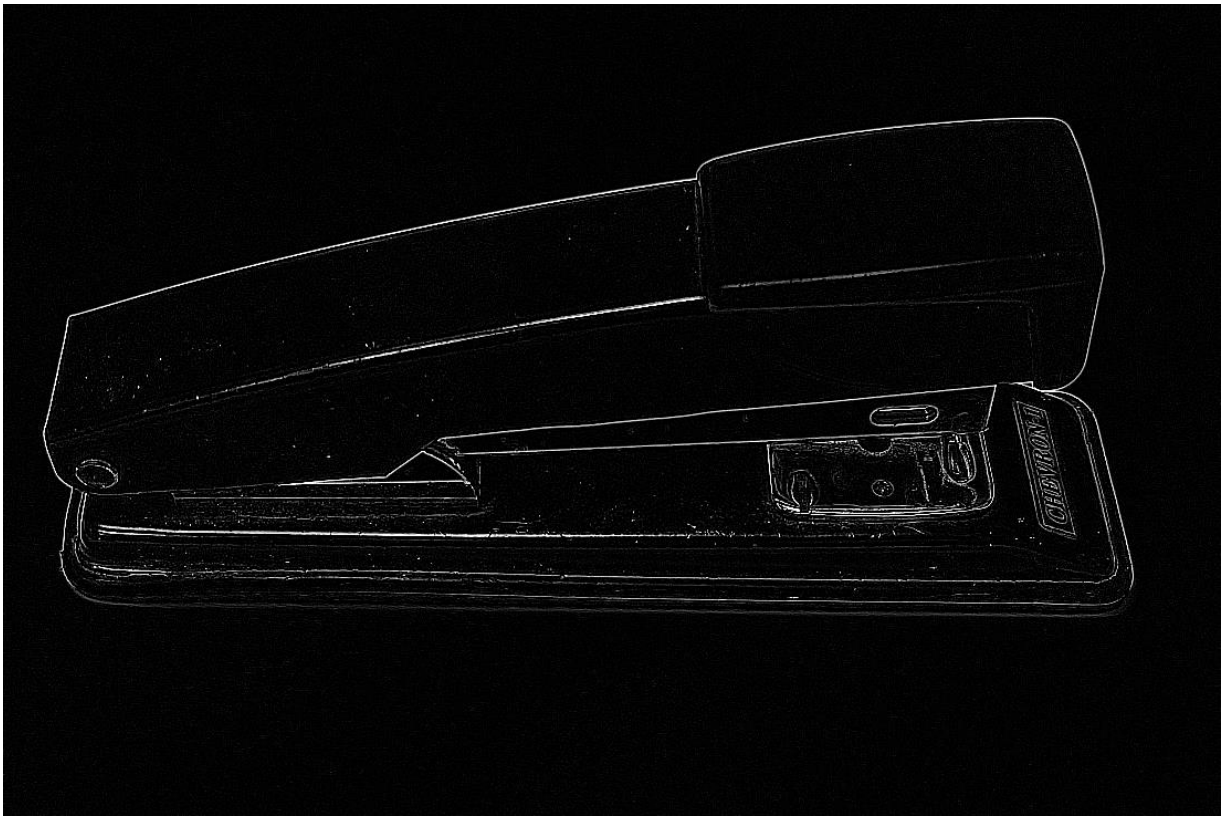


Figure 25: Example of Laplacian edge detection

In order to alleviate the problem a second order derivative has with noise, a Gaussian filter can be applied to the image to reduce noise. A two dimensional Gaussian filter is defined in equation (14).

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{14}$$

where σ is the standard deviation, which specifies the steepness of the Gaussian distribution. Convolving a kernel, sampled from equation (14), with the image would reduce the noise in the image. However, due to the linearity of the Gaussian and Laplacian functions, they can be combined to create a **Laplacian-of-Gaussian** (LoG) kernel by convolving the Laplacian kernel with the Gaussian. This means that only one pass would have to be made through the data to do the edge detection.

The LoG can be defined with the following equation [34]:

$$\nabla^2 G(x,y) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{15}$$

Convolving a 13 by 13 LoG kernel with a standard deviation of 2 with the image in figure 23 produces the results in figure 26. Note that much of the noise-induced false edges in figure 25 are not present in figure 26. However, using the second derivative still has the disadvantage of having no knowledge of edge direction.

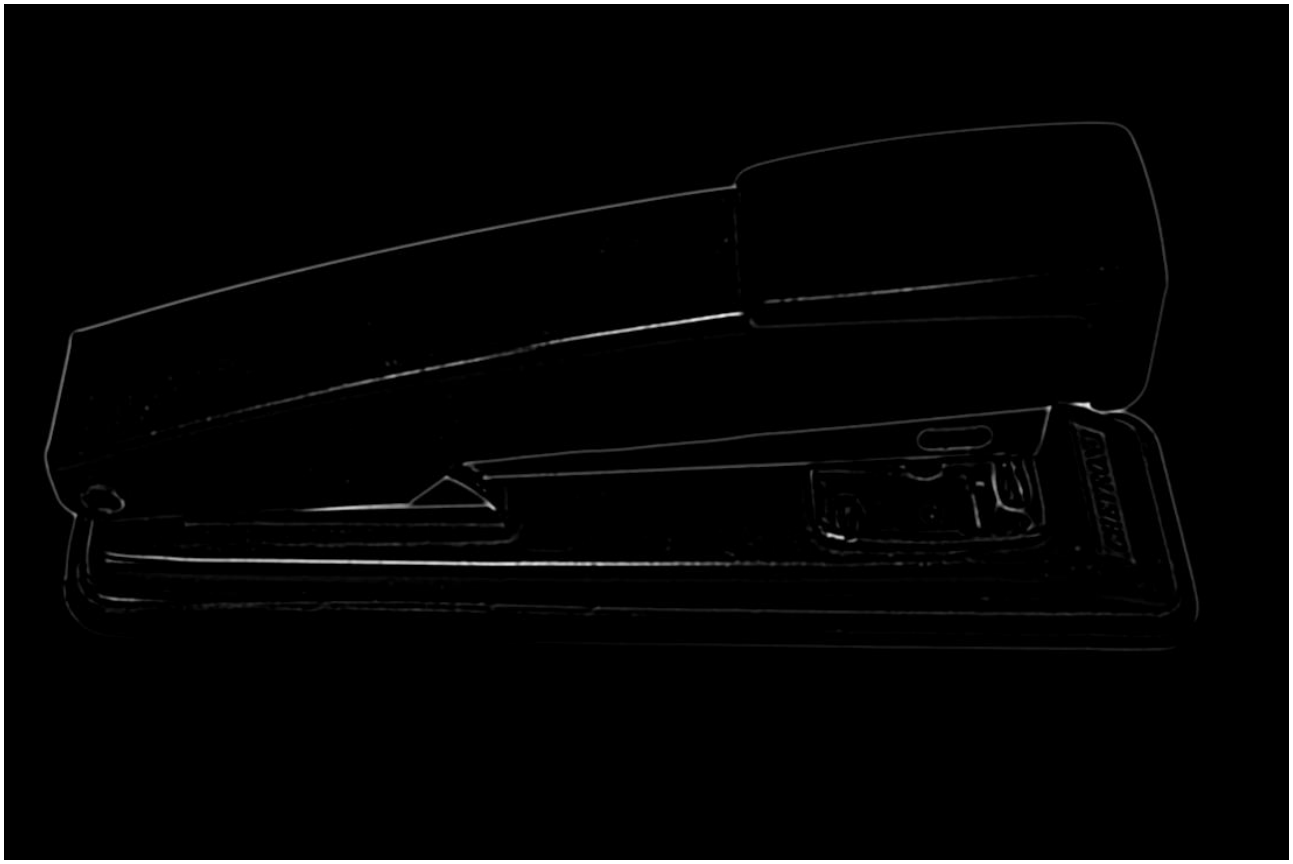


Figure 26: Example of Laplacian-of-Gaussian edge detection

Notice that the methods described, above, only produce a measure that indicates edge strength at each point in the image matrix. They do not quantitatively "detect" edges and separate them from the background regions. More advanced algorithms are used to accomplish this. An example of such an algorithm is the **Marr-Hildreth algorithm** [39]. This algorithm, first, filters the image with the previously defined Laplacian-of-Gaussian (LoG) operator, then finds the zero crossings of the resulting image data.

The Marr-Hildreth algorithm works with the assumption that a relatively fast change in feature values will induce a peak or a valley in the first derivative and a zero crossing (point at which the value changes signs) in the second derivative, hence the use of the Laplacian operator. In order to visualise this idea, figure 27 shows the first and second derivative of the feature values of twenty adjacent pixels in some direction. This approach has the advantage of being scalable (a LoG operator of any size can be generated) and isotropic. Having no directional information, the operator is rotation invariant and equally responsive to discontinuities in the horizontal and vertical direction.

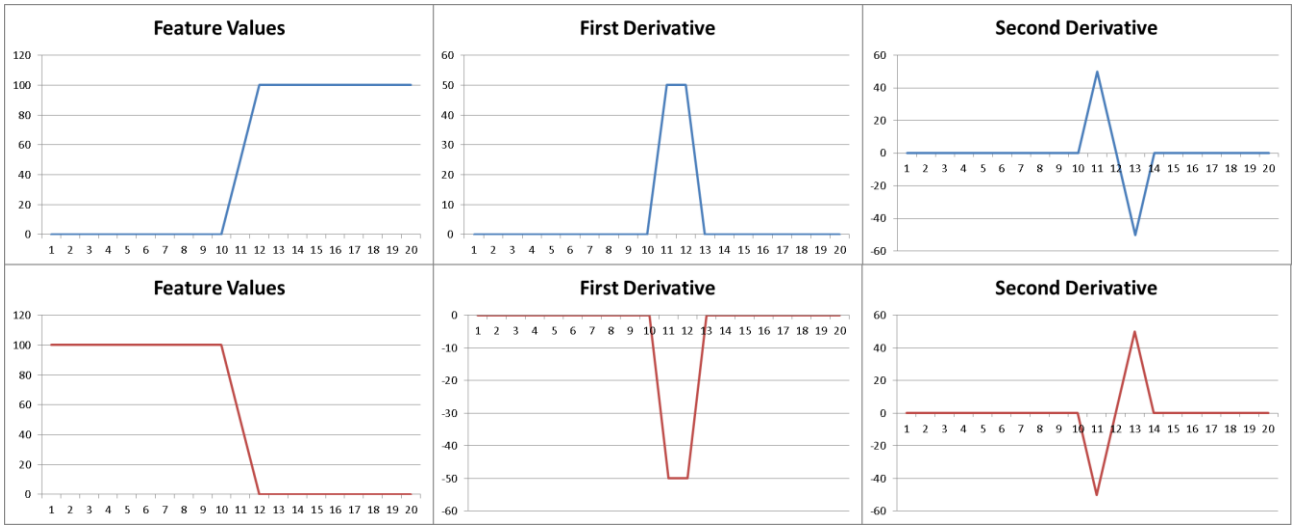


Figure 27: Edges in terms of differentials

A simple method of finding the zero crossings in the image data that has been filtered by the LoG operator is to investigate each pixel location and compare its eight neighbouring pixels. If any pairs of opposing neighbouring pixels do not have the same sign, then the pixel being investigated can be identified as a zero crossing. All zero crossing pixels are then labelled as edges. Figure 28 shows the results of a Marr-Hildreth edge detector implemented on the image in figure 23. Note that all the edges form closed loops. This is a disadvantage of the algorithm. It can be prevented by thresholding the LoG data. [34]

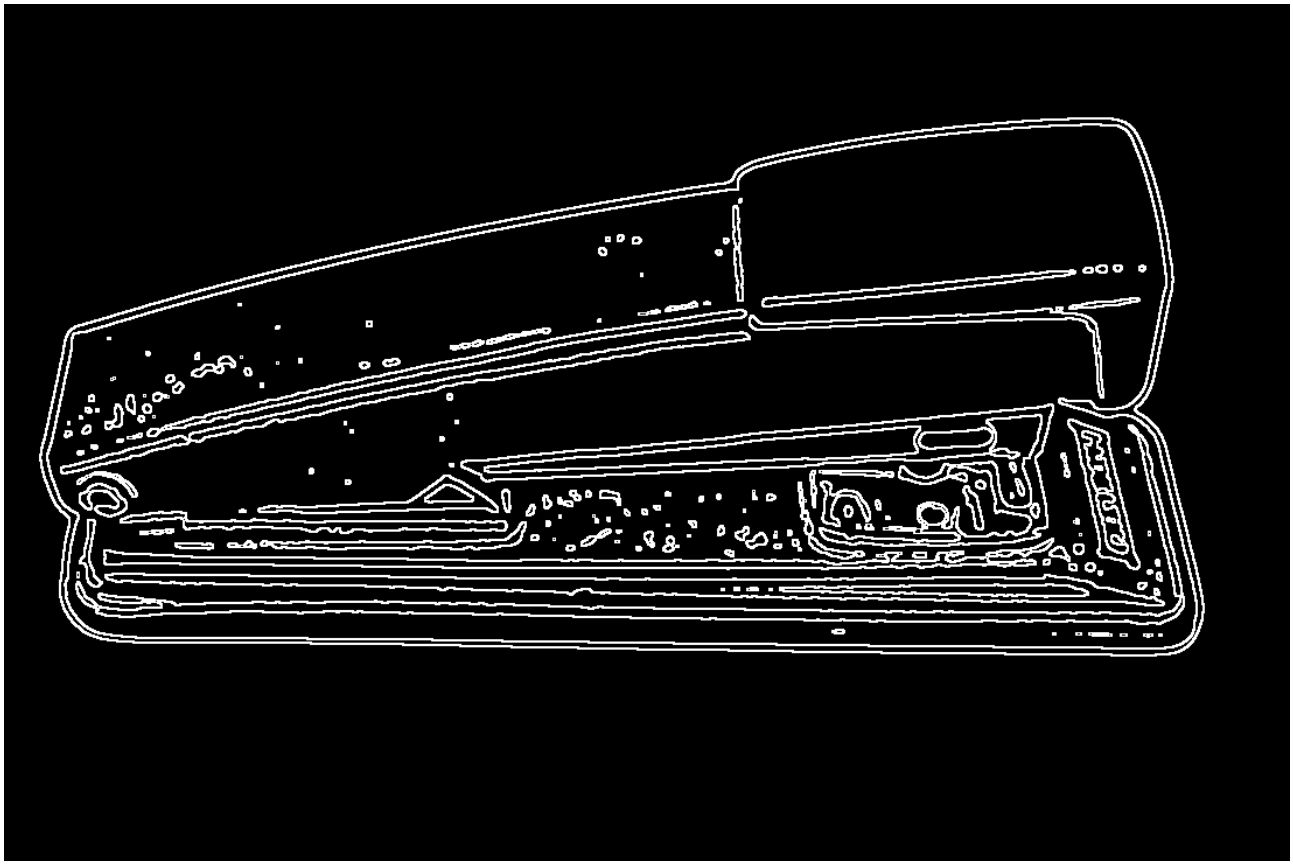


Figure 28: Example of Marr-Hildreth edge detection

Another advanced edge detection algorithm is the **Canny edge detector** [40]. This is a multi-stage algorithm and is one of the most popular edge detection algorithms. The algorithm has 5 steps [41]:

1. Filter the image to reduce noise (Gaussian)
2. Generate intensity gradients (Sobel)
3. Non-maximum suppression to thin the edges out
4. Double thresholding to identify candidate edges
5. Track by hysteresis to identify the final edges

The first step is a simple image smoothing operation done to reduce the amount of noise content in the image. It is accomplished by convolving a Gaussian kernel with the original image. The second step is to obtain the gradient of the filtered image. This gradient can be obtained by means of any of the first order differential operators (e.g. Roberts, Sobel, or Prewitt). The third step is implemented to remove excessive edge thickness from the gradient plots. In order to do this the direction of the gradients are needed, hence the use of first order differential operators.

The angles of the gradient values are rounded to one of four possible angles: 0°, 45°, 90° or 135°. Then each pixel's gradient strength is compared to the two adjacent pixels' gradient strength in the direction defined by its angle. If the pixel does not have the largest gradient in that direction it is suppressed (reduced to zero). This operation only preserves the highest edge strengths on an edge in a specific direction.

The goal of the fourth step is to separate strong candidate edges from the remaining spurious edges. A high threshold- and a low threshold gradient strength value are identified empirically. The gradients above the high threshold are labelled as strong edges, the gradients below the low threshold are suppressed, and the intermediate gradients are labelled as weak edges. The final step is done to finalise the edge detection process by further differentiating between strong and weak edges. The local neighbourhood of every weak edge pixel is investigated. If the neighbourhood contains a strong edge, then the weak edge pixel is redefined as a strong edge. The resulting image only contains strong edges and non-edges. Figure 29 provides the results if implementing Canny edge detection on the image in figure 23. The advantages of the Canny edge detector is that it has a low error rate, the resulting edges are well localised (they are thin and close to the actual edges in the scene), and only one edge is identified for each actual edge in the scene, as opposed to the Marr-Hildreth algorithm that frequently identifies two.

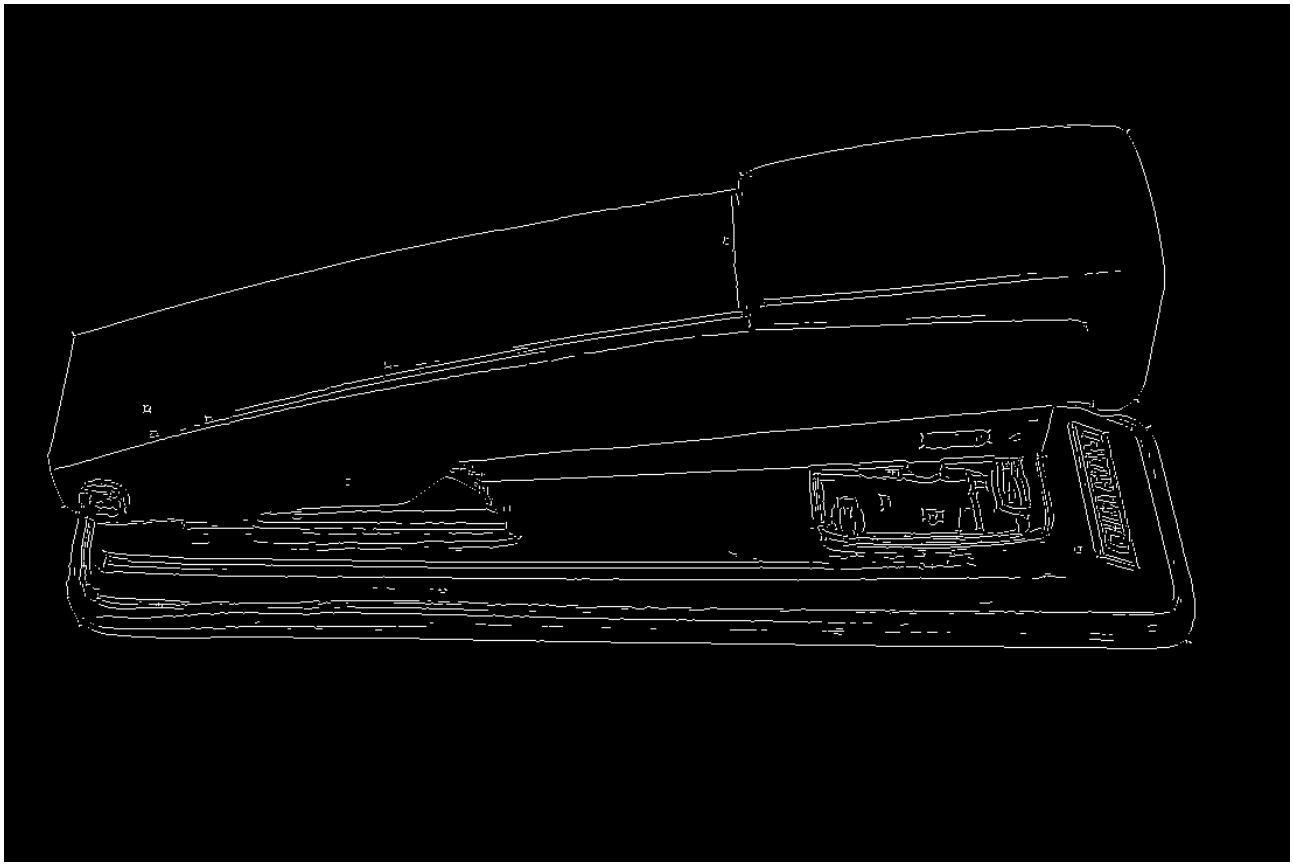


Figure 29: Example of Canny edge detection

2.3.3. Issues with edge detection

In the ideal case, the edges obtained from edge detectors are thin, solid, and unbroken lines of connected pixels that define the exact boundary between two regions of relative feature value homogeneity. In this case it is a trivial task to compare the regions within the boundaries and assign informative classes to each. However, due to factors such as noise and inconsistent illumination in the scene, the edges usually take the form of disconnected lines of pixels and isolated groups of edge sections. For this reason, edge detection is conventionally followed by algorithms that attempt to connect the broken edge lines and remove the uninformative groups of false edges. These algorithms are referred to as edge linking algorithms [34].

One method of linking edges is by investigating the local neighbourhood of each pixel that is designated as an edge pixel. If two edge pixels, within a neighbourhood, have similar values (in terms of similarity measures such as edge strength and edge direction) then they are linked. By this is meant that the non-edge pixels separating these two pixels will also be designated as edge pixels.

Another edge linking method requires knowledge about the boundaries of the regions within the image. If the location of the boundaries are known, then approximations of these boundaries can be made using the existing edge pixels. This approach requires that the regions within the image is known before completing the edge detection. Therefore, this method would be impossible to implement in an image segmentation operation where the edge detection is implemented with the intention of obtaining these regions.

If no knowledge of the edge properties or image regions exists, then a global approach is necessary. This is done by determining whether edges fall on specific kinds of curves and shapes. If edge pixels do satisfy these curves partially then the curves are completed and used as region boundaries. This shape extrapolation operation can be done by using a Hough transform. This transform looks for accumulation in the parametric space of the shape [42].

In addition to the issues edge detection has with noise, there is also a problem with regards to multivariate image data. As it has already been stated, an edge in an image with one dimensional feature values can be defined as a discontinuity in the feature values across the image matrix. The term discontinuity, in this definition, refers to a sudden change (difference between the feature values of adjacent pixels) in the feature values. Therefore, the question has to be asked of how a discontinuity is defined in a multi-dimensional feature space.

In [43] three methods of detecting a discontinuity in multi-dimensional feature spaces are provided:

- A metric distance (e.g. Manhattan distance or Euclidean distance) in the feature space can be used to compare feature values with each other. If the distance (within the feature space) between two adjacent pixels is too large, then a discontinuity is detected. This method reduces the operation to a computation of a one dimensional value and can therefore be no more accurate than the edge detection of an equivalent one dimensional feature space.
- One dimensional edge detection can be implemented on each feature dimension separately, followed by merging the results of each. This method may fail to identify certain edges that are equally strong in two feature dimensions, but facing the opposite direction.
- One can find the edges in each dimension separately, but constrain them with some measure of uniformity. This will allow each feature dimension to affect the edges obtained in other dimensions without reducing the operation to a one dimensional computation. An example of such a measure of uniformity would be the edge angle.

The experiments in [43] used the third method of detecting multi-dimensional discontinuity in natural colour images. However, the results showed that most of the edges in the chromatic feature values (red, green, and blue) were contained in the edges in the luminance (grey) feature values. This implies that most of the edge information in an image with multi-dimensional feature values is contained in the edge information of a one-dimensional feature valued equivalent image. Therefore, it is possible to detect edges in an image with a multi-dimensional feature space, however, some of the multi-dimensional information will be lost.

Taking these issues into account, the segmentation of images of raw coal ore (which would contain multi-dimensional data and plenty of inconsistent edge shapes) using edge detection is feasible, but not optimal. Referring back to figure 6, it is clear that the boundaries (edges) between the visually perceptible regions are not readily identifiable. It is also worth noting that there is little variation in feature value with regards to one dimension, such as pixel intensity. This is just one example to show how edge detection would be a sub-optimal method of image segmentation for photographs of raw coal ore.

2.3.4. Summary of edge detection

Table 2 provides a summary of the advantages and disadvantages of using edge detection as a means to segment an image.

Table 2: Edge detection criterion

Advantages	Disadvantages
• Great for estimating region boundaries for other image segmentation algorithms • The fundamentals require no prior knowledge about the image • Multi-dimensional edge detection is plausible	• Extreme sensitivity to noise • Requires non-trivial post processing to identify the closed boundary curves • Struggles to separate regions without clear boundaries between them

Like thresholding, edge detection is most effective in images containing large homogenous regions. Counter intuitively, image segmentation through edge detection is most effective when there are very few edges. An excessive number of edges complicates the process of determining which edges are the desired boundary regions and which are noise or texture properties. The texture of raw coal ore is very rough (high amount of change in local gradient values), and texture is an important feature value that separates one type of material from another. This approach is, therefore, not considered a fitting method of image segmentation.

2.4. Image segmentation by region extraction

The purpose of this section is to investigate the region extraction approach to image segmentation. The following are some of the key points of discussion:

- The fundamental ideas behind region extraction
- Various methods of implementing region extraction
- Problems with region extraction
- A summation of the advantages and disadvantages of region extraction

2.4.1. Fundamentals of region extraction

The third method of image segmentation to be discussed is region extraction. Both edge detection and region extraction algorithms focus largely on the arrangement of pixels in the image matrix to segment the images, however region extraction is based on finding continuity in the image instead of discontinuity [17]. Therefore, figure 11 categorises region extraction under domain-space and region focused algorithms. Similar to edge detection, these algorithms do not define classes before segmentation, instead they find the segmentation regions before assigning classes.

The fundamental idea behind region extraction algorithms is that visually distinct groups of pixels have similar feature values and are grouped together in the image matrix. In other words: pixels in a digital image that are close to each other are expected to be inherently similar. The method groups subsections in the two dimensional lattice of pixel values into larger sections based on their similarity.

The core part of these algorithms is finding regions of pixels that are similar to each other, and dissimilar to pixels of other regions, according to some measure of homogeneity. There are two approaches to finding these regions: Region growing, and Splitting-and-merging. The former is the process of grouping the pixels into larger and larger regions according to a homogeneity criteria. The latter is done by segmenting the image into an initial set of estimated regions and merging or dividing these regions based on their rating in terms of the homogeneity criteria [34]. Both of these approaches need some metric of similarity to be used when deciding if a pixel should be included or excluded from a region. This metric is dependent on the problem and the image data that is available.

2.4.2. Region extraction algorithm types

The first of the two methods of region extraction is **region growing** [44]. As the name indicates, this method starts off with a set of initial seed points within the image matrix and "grows" the regions by adding the neighbouring pixels to the region if they are similar enough to the pixels already added to the region. This is called pixel aggregation. A flood fill algorithm can be used to grow the region. A simplified description of such an algorithm follows:

Algorithm 3: Pixel aggregation

$Given\ some\ image\ I\ with\ feature\ values\ f(x,y).$

$Also\ given\ a\ set\ of\ seed\ points\ S\ containing\ one\ seed\ point.$

$Repeat\ while\ new\ seed\ points\ are\ added:$

$For\ each\ seed\ point\ (i,j)\ in\ S:$

$If\ P\big(f(i+1,j)\big) = true\ add\ (i+1,j)\ to\ S$

$If\ P(f(i,j-1)) = true\ add\ (i,j-1)\ to\ S$

$If\ P(f(i-1,j)) = true\ add\ (i-1,j)\ to\ S$

$If\ P(f(i,j+1)) = true\ add\ (i,j+1)\ to\ S$

$Where\ P\ is\ some\ similarity\ predicate$

The definition provided, above, uses one initial seed point to grow a single region within the image by repeatedly investigating the pixels to the right, above, to the left, and below each seed point that has already been aggregated. This process is repeated until no more pixels, connected to the region, satisfy the similarity predicate. Figure 30 illustrates this operation with an example. The left panel is the original image. The right panel shows the result of implementing a flood fill algorithm with a single initial seed point (the red dot) and the predicate: "Is the point the same colour as the initial seed point?". In this example the pixels that are horizontally and vertically connected are investigated, but the definition can easily be modified to include the remaining diagonally connected pixels.

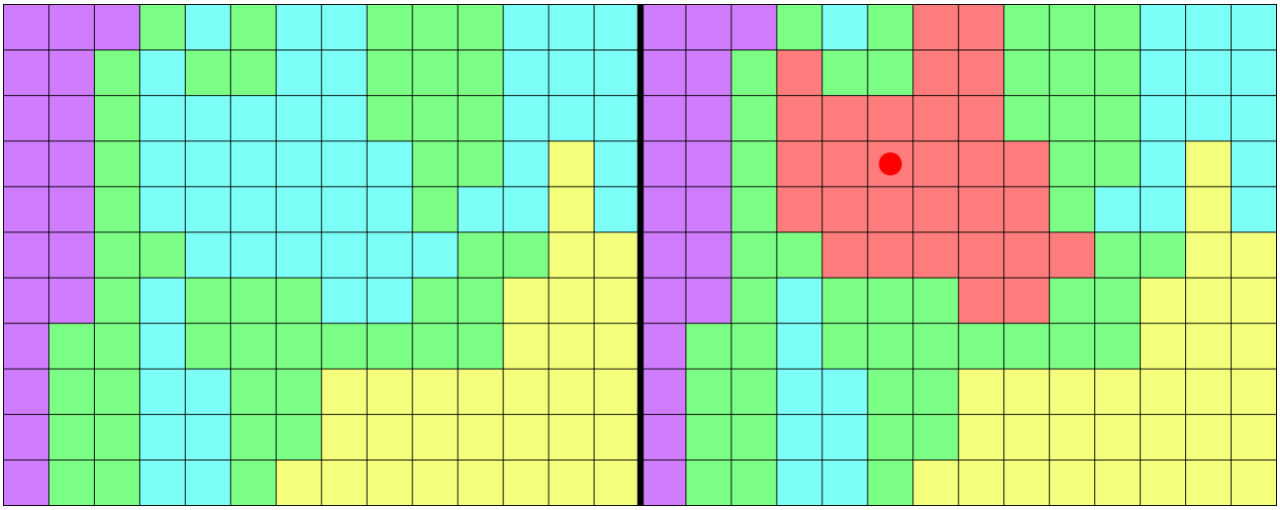


Figure 30: Example of pixel aggregation

The results of the region growing algorithm is dependent on [15] [21]:

- The number of initial seed points
- The position of the initial seed points
- The similarity predicate
- The pixel connectivity used

The similarity predicate for region growing can be based on the feature value of the initial seed point, a value that is derived from all the feature values currently included in the region, such as the mean feature value, or the feature value of the connected boundary pixel. Using the initial seed point will constrain the growth of the region heavily, using the mean value will provide an intermediate degree of freedom to grow, and using the boundary pixel will result in a lot of freedom to grow. The decision is based on the problem being addressed by the algorithm.

In practice, even with the appropriate selection of seed points, the results usually contain unrealistically small regions and separate regions that are adjacent and fairly similar. This is called oversegmentation and is a result of noise. In order to prevent this, region growing algorithms are commonly followed by a region merging algorithm. This involves the merging of excessively small regions with a neighbouring region that is the most similar to it, and the merging of regions that satisfy the predicate when they are merged [45].

The second method of region extraction is **splitting-and-merging**. The algorithm segments the image into an initial estimated collection of regions, and then merges and divides these regions according to the homogeneity criteria until a convergence state has been achieved. Structurally, the algorithm consists of two parts: A top down part where the entire image is successively divided into smaller and smaller regions; and a bottom up part where the smaller regions are merged together to form larger regions. The basic procedure for splitting-and-merging follows:

Algorithm 4: Splitting-and-merging

$Given\ some\ image\ I.$

$Define\ the\ initial\ region\ R_0$ as the entirety of I.

$Repeat\ until\ no\ more\ splitting\ is\ possible$:

$Split\ any\ region\ R_i\ for\ which\ P(R_i) = false\ into\ four\ quadrants$

$Repeat\ until\ no\ more\ merging\ is\ possible$:

$Merge\ any\ adjacent\ regions\ R_j\ and\ R_k\ for\ which\ P(R_j \cup R_k) = true$

$Where\ P\ is\ some\ similarity\ predicate$

The data structure that is used to divide the regions sequentially is called a quadtree representation. This means that each parent region is divided into four child regions. [16] The predicate used in the splitting-and-merging algorithm can be based on a variety of descriptors such as the mean feature value in the region, the gradient content, the standard deviation, or entropy. Just like the region growing algorithm, it depends on the image segmentation problem.

Figure 31 provides the results of implementing this algorithm on the image in figure 30. The first four panels shows the repeated subdivision of the regions in the image depending on the predicate: “Are all the pixels in the region the same colour?”. The shaded regions are the regions that satisfy the predicate, and therefore do not undergo any more splitting. The last two panels show the results of the merging of regions that satisfy the predicate. The result is eleven homogenous disjoint regions that can easily be matched to generate the four classes (purple, green, blue, and yellow). This method has the advantage of not being dependant on the location of initial seed points, as is the case in region growing.

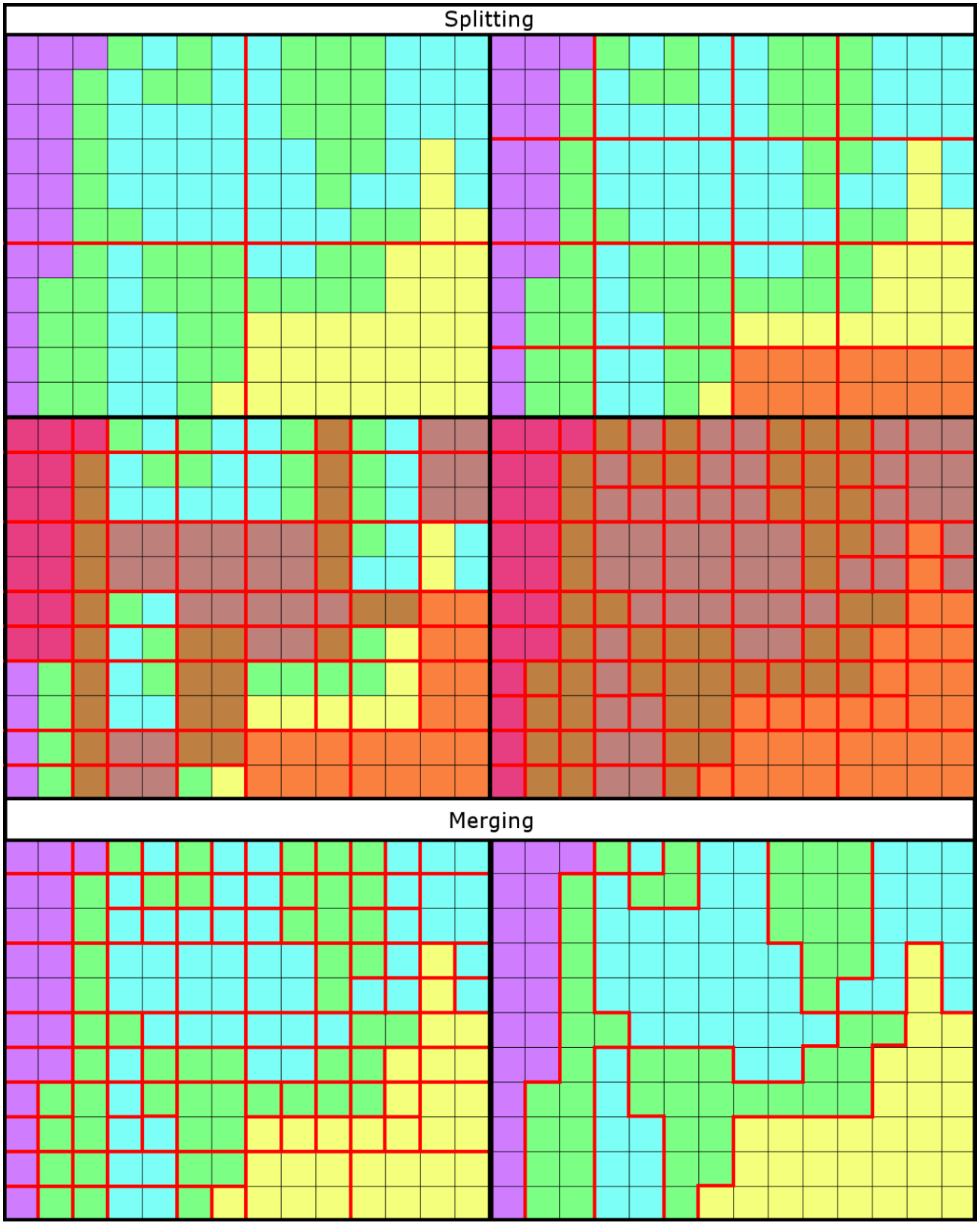


Figure 31: Example of splitting-and-merging

2.4.3. **Issues with region extraction**

The main problem with **region extraction** is the inherently sequential nature of the algorithms. This means that the segmentation regions are slow to compute, and suffer from order dependency. The order in which the region growing algorithm processes the seed point, and the order in which the splitting-and-merging algorithm merges/splits the regions has a large effect on the resulting segmentation. This is especially true for the smaller regions that are produced towards the end of the algorithms. [20] [16]

With specific regard to the **region growing** algorithm, there is a problem with defining the initial seed points. They have to be chosen to be a very good approximation of the general characteristics of the pixel values contained in the desired segmented region. They, also, have to be located in such a way so that the region growth can have easy access to the entire region, without being hindered by obstacles (pixels that do not satisfy the homogeneity predicate).

Many implementations require manual intervention to select the initial seed points correctly [15] [46]. This is an unappealing fact when an automated image segmentation algorithm is necessary. Even with manual interaction this method is prone to error because the seeding location can be selected on a pixel that contains a noise component, resulting in a cascade of erroneous decision making. If no prior knowledge is available, the conventional way of obtaining initial seed locations is to find the pixel with the most centrally located feature value within the local or global feature space. This can be done by using the peaks in histograms or centroids in cluster analysis. [34]

Another problem with region growing is the way a stopping rule is defined [15] [34]. The basic idea is to stop growing the region when no more connected pixels satisfy the homogeneity predicate. In some applications, simply using the feature values is insufficient. A better method of deciding whether to include a pixel into a region would take the mean feature value-, the size-, and shape of the region into account. However, these values would require some information about the expected results of the segmentation exercise.

An issue that is a problem for the **splitting-and-merging** algorithm alone is that the results of the segmentation tend to mimic the specific data structure that was used during the splitting phase of the algorithm. This means that, if a quadtree structure is used, that the resulting segmentation regions would appear very square. [21] Additionally, the splitting-and-merging algorithm relies extensively on local information. There is almost no way to incorporate global information about the image to refine the segmentation process. [20]

Fortunately, region extraction has no problem incorporating multidimensional feature values into its fundamental operations. The homogeneity criteria will have to be modified to take all the feature dimensions into account, but this can be done with metric distances in a similar fashion to how a multidimensional discontinuity can be defined for edge detection.

2.4.4. Summary of region extraction

Table 3 provides a summary of the advantages and disadvantages of using region extraction as a means to segment an image.

Table 3: Region extraction criterion

Advantages	Disadvantages
• Resulting regions are always, spatially, well connected • Reasonable noise resistance due to the emphasis on continuity	• Computationally expensive • High order dependency • Results can mimic the data structure used (square regions) • Difficult to incorporate global prior information without manual intervention

Region extraction is very effective in images where the homogeneity criteria can be easily defined. Digital images of raw coal ore samples contain regions where there is a large variation in one feature dimension but a high consistency in another feature dimension. This will make it very difficult to develop a similarity predicate that will successfully segment all regions within the image with a region extraction algorithm. One of the major flaws of region extraction is the computational complexity. This is exceptionally undesirable, because one of the goals of the research is to develop a relatively fast method of image segmentation. For the reasons, above, this algorithm is deemed unsuitable for the research project.

2.5. Image segmentation by clustering

The purpose of this section is to investigate the clustering approach to image segmentation. The following are some of the key points of discussion:

- The fundamental ideas behind clustering
- Various methods of implementing clustering
- Problems with clustering
- A summation of the advantages and disadvantages of clustering

2.5.1. Fundamentals of clustering

In the previous section it was explained that region extraction seeks out continuity in the domain space of the image data. In contrast to this, clustering attempts to find continuity in the feature space of the image data. Figure 11 categorises clustering under range space and region focussed algorithms. It, therefore, classifies pixels based on their feature values instead of their location in the image matrix. Also take note that clustering identifies pixel classes before the actual segmentation regions.

Clustering is not a technique that is dedicated to solving an image segmentation problem. It is used in various applications of pattern analysis, data mining, and document retrieval. It is especially useful in problems where there is little prior information. This is because clustering is an unsupervised method and can be used to investigate the underlying structure in a vast array of data types [47].

The link between image segmentation and clustering is in the fact that both try to find structure in a data set. Image segmentation tries to find regions of pixels that are similar to each other, and dissimilar to pixels of other regions, in terms of feature values. Clustering attempts to find groups (referred to as clusters) of data points that are similar to each other, and dissimilar to data points in other clusters. An image segmentation problem can, therefore, be translated to a clustering problem by:

- Translating the feature values in the image to a feature space
- Applying cluster analysis techniques to identify the classes contained in the data set
- Replacing the data points (now labelled with the appropriate class) back into their original location in the image matrix

Figure 32 provides a visual representation of how clustering can be used to segment a digital image. The photo in the top left panel is the input image. The scatter plot in the bottom left panel represents the feature space of the original image. Notice that the three dimensions of the feature space correspond to the red-, green-, and blue colour channels of the RGB colour space. The scatter plot in the bottom right panel show the feature space after it underwent some clustering operation to group the data points into five classes. The exact clustering algorithm used is a k-means clustering algorithm that aims to minimise the within-class Euclidian distances. This algorithm will be explained in section 3.3.1. The image in the top right panel is the result of replacing the clustered data points from the feature space back into their original position in the image matrix. Take note that the results are, by definition, a segmentation of the original image.

Just like thresholding algorithms, and any algorithm that segments images based on their feature values, clustering algorithms work with the assumptions that feature values that are coherent in the feature space are coherent in the domain space as well. In other words, data points in the feature space that are densely packed together will be densely grouped together in the image matrix as well. In the example, presented above, this seems to be the case. In fact, this is the case for most digital images, provided that there is not an excessive amount of noise in the image.

Because of the similarities between clustering and image segmentation, this technique is commonly employed as a pre-processing step in other image segmentation algorithms to provide an estimation of the final results or as a method of obtaining global information about the image. Examples of this include: using clustering to find the feature values with the highest density to select a threshold value; or using clustering to select a seed point in a region growing exercise.

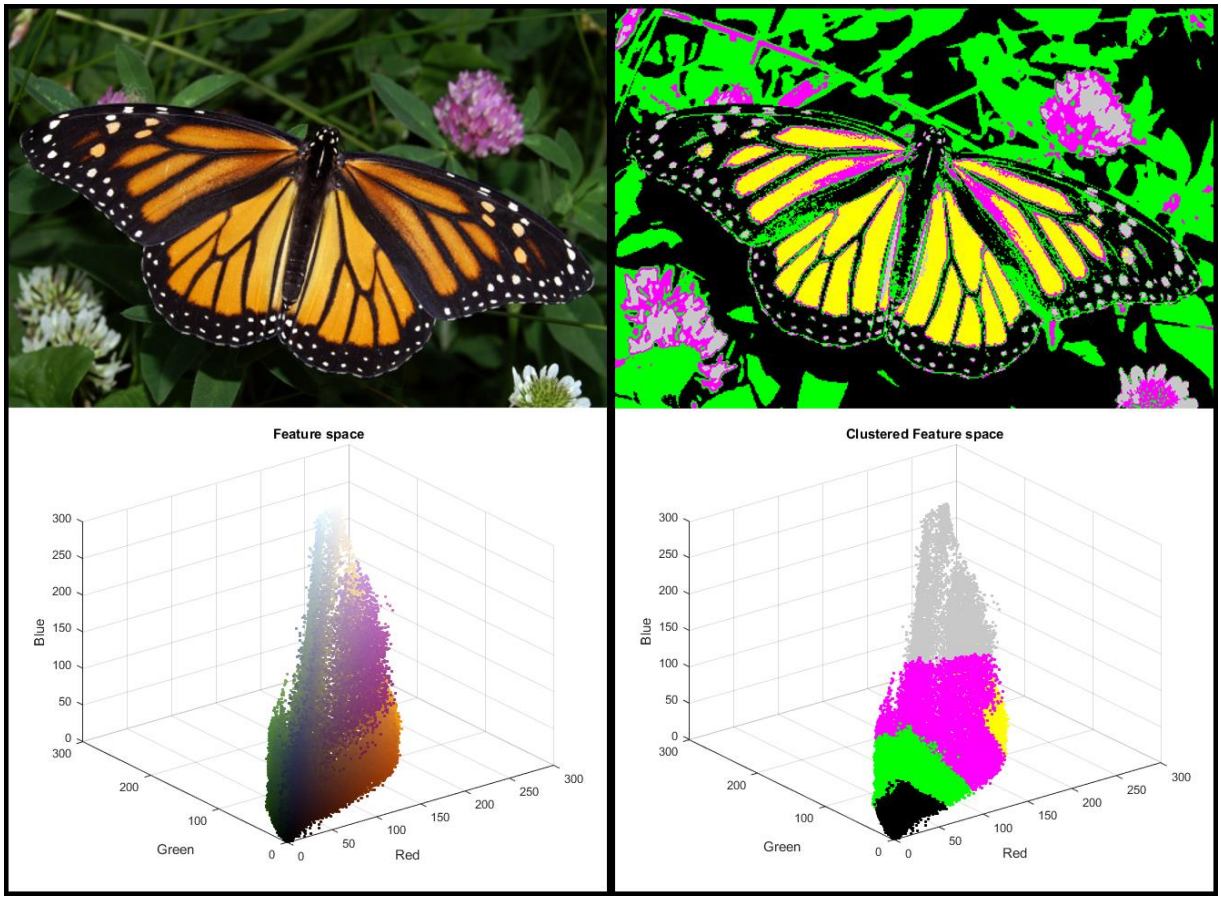


Figure 32: Example of using clustering to segment an image

The similarity between image segmentation and clustering can also be seen when comparing the definitions of the two. A formal description of clustering follows [48]:

Algorithm 5: Clustering

$If\ X\ is\ a\ set\ of\ N\ data\ points$:

$X = \{x_1, x_2, \dots, x_N\}$

$where\ x_i$ is a feature vector in a p − dimensional feature space.

$Then\ the\ clustering\ of\ X\ is\ a\ set\ of\ M\ clusters$:

$C = \{c_1, c_2, \dots, c_M\}$

$where$

- $c_i \neq \emptyset, i = 1,2, \dots, M$
- $\bigcup_{i=1}^{M} c_i = X$
- $c_i \cap c_j = \emptyset, \forall\ i \neq j$

However, it should be added:

$$P(x_i, c_i) > P(x_i, c_j)\ \forall\ x_i \in c_i, c_i \neq c_j$$

$where\ P\ is\ a\ measure\ of\ similarity$

Informally, clustering can be defined as the grouping of n-dimensional data points (e.g. feature vectors or scalars) into classes, called clusters, in such a way that the data points within a cluster are more similar to the other data points in that cluster than data points in any other cluster.

One of the most desirable characteristics of clustering is the high level of applicability the algorithms have on higher dimensional data [49]. Clustering algorithms are almost always applicable to feature values of any dimensionality. For this reason, image segmentation implementations of multispectral or colour images typically use this approach.

2.5.2. Clustering algorithm types

Clustering algorithms are conventionally dichotomised as follows: Hierarchical clustering (HC) and Partitional clustering (PC). In short, an HC algorithm produces a nested result of cluster configurations, and a PC algorithm produces only one. There are further ways to compartmentalize the field of cluster analysis such as hard classification/fuzzy classification, deterministic/stochastic, and agglomerative/divisive. However, these separations are found in both HC and PC. A brief explanation of these divisions follow [47]:

- Hard and Fuzzy clustering

  A hard clustering algorithm assigns each data point to only one cluster at any given time. A fuzzy clustering algorithm assigns data points to several clusters by making use of degrees of membership to all possible clusters. This means that the algorithms can consider more scenarios simultaneously when deciding to which cluster a data point should be assigned. Any fuzzy clustering configuration can be converted to a hard clustering configuration by assigning the data points to the cluster with which they have the largest membership value. An image segmentation would, naturally, require a hard clustering configuration due to the definition of image segmentation requiring disjoint regions.

- Deterministic and Stochastic clustering

  Deterministic algorithms cluster the data points according to strict sequential procedures that make use of cause-and-effect decision making. An alternative approach is the stochastic algorithmic approach. Algorithms belonging to this category make use of uncertainty to implement random searches through a problem space to consider all possible solutions and find the optimal solution as a result. Stochastic algorithms tend to take longer to converge but they have the opportunity to consider solutions that a deterministic algorithm is not able to. Stochastic methods are usually used by PC algorithms.

- Agglomerative and Divisive clustering

  This distinction relates to the structure of the algorithm. An agglomerative algorithm is a bottom-up algorithm that starts off by having all the data points in a cluster of their own, and then sequentially merges clusters until a specific criterion is satisfied. The other type of algorithm (divisive) is a top-down algorithm in which all data points are initially in a single cluster, and the cluster is then divided until a criterion is met. This distinction is more often made in HC than PC. Take note that these two approaches are reminiscent of the splitting-and-merging region extraction algorithm.

- Monothetic and Polythetic clustering

  A monothetic algorithm considers data points sequentially when decisions are made with little regard to data points that are not immediately related to the computations. A polythetic algorithm considers information about all the data points in the set during the clustering process. Most clustering algorithms tend to be polythetic. Again, a similarity is drawn between monothetic- and polythetic clustering and local- and global information in image segmentation.

- Incremental and Non-incremental clustering

  The distinction, here, is self-evident. The former uses repeated increments to converge to a solution, and the latter only uses one pass through the data set. These types of characteristics only need to be considered when the data set becomes very large.

As it has been stated, the main two classes of clustering algorithms are HC and PC. **Hierarchical clustering** uses a dendrogram to determine the class to which each data point should be assigned. The dendrogram is generated by using heuristic techniques to merge and/or split (depending on if it is an agglomerative or divisive algorithm) potential clusters until either all the data points are in one cluster, or each data point is in its own personal cluster.

Figure 33 illustrates how HC is used to segment an image. Note that the clusters that are similar to each other are merged lower on the dendogram. When complete, the dendogram can then be cut, anywhere along the horizontal axis to produce a number of cluster configurations, ranging from one to as many data points that are present in the data set. The result of segmenting the image according to three of these cluster configurations is shown next to the original image.

This method of clustering is, naturally, very simple to understand and implement. In addition to the two structural variations of the HC algorithm (agglomerative and divisive) there are also variations in the measure of similarity used to merge or divide the clusters. These variations are: single-link; complete-link; and minimum variance. Single-link algorithms base the decision on the minimum distance between any two data points from each cluster (one from each), and complete-link algorithms base it on the maximum distance between any two data points from either cluster [18]. Algorithms that minimise variance investigate the Euclidian distance between the cluster centres. Computationally, generating the necessary dendogram is very expensive, because it is implemented using a distance matrix. This an important disadvantage of HC algorithms.

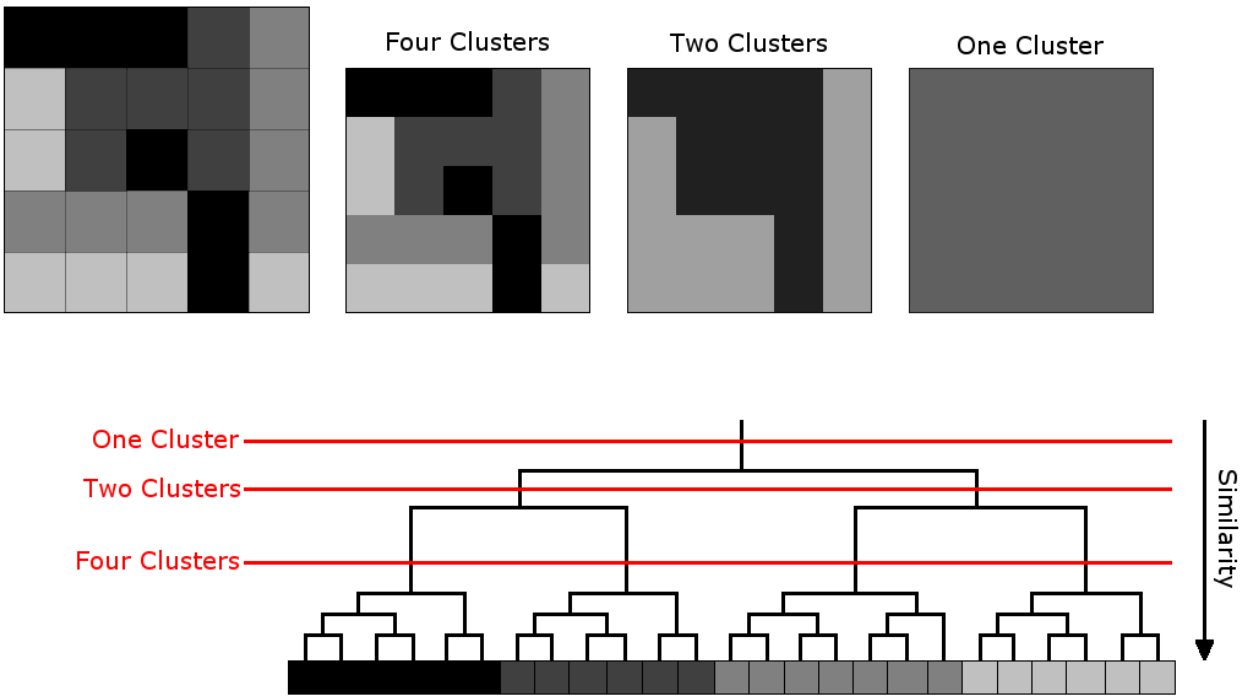


Figure 33: Example of hierarchical clustering

Partitional clustering, on the other hand, does not usually require a distance matrix and is computationally much less expensive. The defining difference between HC and PC is the output. PC algorithms produce a single cluster configuration instead of a collection of possible configurations. These algorithms generally find the cluster configuration by optimizing a criterion function through incremental adjustments and repetitions [47].

Figure 34 provides an example of how a PC algorithm works. The feature space in panel one has two dimensions. In panel two, the data set is, first, clustered using an estimated configuration. Then, in panel three the cluster configuration is adjusted to move the centres of the clusters to more densely packed regions in the feature space. The last panel is the result of repeating the adjustment until no further adjustments result in better clustering configurations.

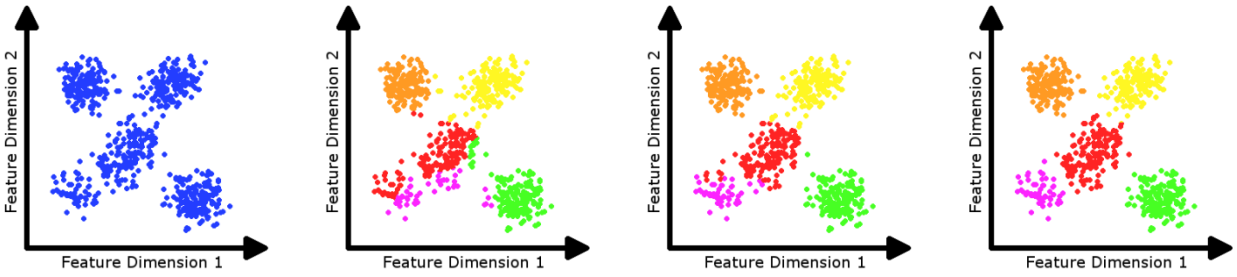


Figure 34: Example of Partitional Clustering

The results of a PC algorithm depend heavily on the initial positions of the clusters and also the criterion that has been chosen to be optimised. This is the main drawback of the PC algorithm. However, it has the advantage of being computationally inexpensive (usually $O(N_p)$) and, therefore, applicable to very large data sets.

PC methods are categorised according to the criterion they seek to optimise. This includes: square error minimisation; graph theoretic; mixture resolving; and mode seeking algorithms. The square error of a cluster configuration is defined as:

$$e^2 = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \left\| x_i^{(j)} - c_j \right\|^2 \quad (16)$$

where K is the number of clusters, n_j is the number of data points in the j^{th} cluster, $x_i^{(j)}$ is the i^{th} data points in the j^{th} cluster, and c_j is the centroid of the j^{th} cluster. The centroid of a cluster is the most centrally located point within the cluster in the feature space. Optimising this criterion produces densely packed clusters. This is the approach represented by the example in figure 34, and is the most frequently used type of algorithm because it is simple to implementation and intuitive to understand. [47]

Graph theoretic algorithms generate a minimum spanning tree (MST) of the data points and successively removes the longest MST edges to produce the clusters [50]. Mixture resolving algorithms are predicated on the assumption that the data points are members of a number of distributions (e.g. Gaussian). The goal is to obtain the parameters of these distributions and then use them to construct the clusters [51]. Mode seeking algorithms finds the local maximum density of each data-point to construct the clusters [52].

2.5.3. Issues with clustering

The main advantages of using clustering as a means to segment an image, instead of the alternative approaches is the ease with which clustering can transition from one-dimensional image data (e.g. grey scale images) to multi-dimensional image data (e.g. colour or multispectral images). However, this improvement comes with three serious disadvantages, one of which pertains to HC alone, one pertains to PC alone, and one pertains to both variations of the algorithm.

Recall that the HC method uses a dendogram to cluster the data set. In practice, the generation of a dendogram requires a distance matrix. That is a matrix containing the distances (within the feature space) between all possible pairs of data points in the data set. This means that for a data set of n data points, $\frac{1}{2}n(n-1)$ distance measures need to be stored in memory. For the example in figure 33 that would require 300 distances. Assuming an 8-byte memory unit for each distance, the memory space required would be approximately 2.3 kilobytes. However, the distance matrix for a commonly used imaging resolution, 1920x1080 pixels would require approximately 15.6 terabytes. HC is, computationally, very expensive, with time and space complexities around $O({N_p}^2 log N_p)$ and $O({N_p}^2)$, respectively. For this reason it is generally not considered a well suited clustering method for large data sets. [19]

The problem with PC is the fact that the performance of the algorithm is extremely dependant on the initial conditions. The initial conditions refer to the number of-, and position of initial clusters. This is due to the iterative nature of most of these types of algorithms. Higher level knowledge of the image can be used to select the initial conditions appropriately. For instance, manual interventions might be required to select the position of the initial clusters with an estimation. However, this is not a very attractive approach when trying to develop an automatic image segmentation algorithm. Fortunately, there are methods of initialising the clustering algorithms in such a way as to allow the best chance of converging to an optimal solution. An example of this is placing the initial cluster centroids as far apart as possible within the feature space. This approach is called k-means++ [53]. However, the initial conditions remains a source of serious consideration in most PC implementations.

The issue that both variations of clustering suffers from is the lack of spatial information used to classify the data points, and ultimately segment the image. While it is true that assuming coherence between feature values and spatial location in an image is a reasonable postulate, it would obviously improve an algorithm to consider the spatial relations between pixels as well. In order to introduce some spatial information into the clustering operation, one can generate an additional feature dimension by calculating some metric of local characteristics in the neighbourhood of every pixel. However, this spatial information would still be of a local nature. Another way of incorporating spatial information is to add two feature dimensions to the feature space that represent an appropriately normalised rendition of the horizontal and vertical spatial axis of the image [54].

In addition to these three problems, there is also an issue that needs to be considered when deciding whether to use a HC or PC method. HC algorithms do not incorporate global information about the shape of and size of clusters into their computation of the cluster configuration, and as a result they struggle to differentiate between overlapping clusters. [55] PC algorithms are a lot better suited to handle overlapping clusters. However, due to the nature of their algorithmic structure, they struggle with non-isotropic clusters. HC algorithms are better suited to handle elongated or chain-like clusters [47].

2.5.4. Summary of clustering

Table 4 provides a summary of the advantages and disadvantages of using clustering as a means to segment an image.

Table 4: Clustering criterion

Advantages	Disadvantages
• Easily scalable to higher dimensions • Simple to implement • Computationally inexpensive (PC only)	• Neglects spatial information • Computationally expensive (HC only) • Dependant on initial conditions (PC only) • Struggles with overlapping clusters (HC only) • Assumes isometric clusters (PC only)

Clustering is a robust method of image segmentation with the attractive advantages of being scalable to higher dimensions. The segmentation of digital images of raw coal ore will have to be accompanied by several feature dimensions (this will be explained in section 3.2). Therefore this approach to image segmentation is the logical choice. However, one of the goals of the research is to develop a reasonably fast and inexpensive method of image segmentation, and for that reason HC is not a feasible option. The images to be segmented will have to be of a very high resolution (containing approximately 1000 000 data points) and the HC algorithm will require excessive computer volatile memory and is computationally expensive at that resolution. The PC algorithm is better suited to the research. It is not as demanding in terms of time and memory, and handles overlapping clusters very well.

3. THEORETICAL DESIGN

3.1. Theoretical design: The system

This section will briefly recapitulate the existing system of coal ore classification. In section 1.1.4 it was explained that the method [13] of generating a washability curve requires that the coal materials be mapped out onto a photograph of coal ore. This constitutes an image segmentation problem. The proposed method [1] uses a two-step approach, making use of a feature extraction operation, followed by a classification (clustering followed by model assignment) operation. The exact procedure was depicted with a flow diagram in figure 4. To summarise:

- A digital image of raw coal ore is received
- Features are extracted and models are generated
- The feature space is clustered
- The clusters are matched to the most appropriate model value
- The results are mapped out onto an output image

The same flow diagram as in figure 4 is presented in figure 35, with four modules highlighted in blue. The highlighted modules are those that have been identified as areas with the possibility of improvement. The modules that are not highlighted are either too trivial a concept or a static constraint of the system (e.g. the input and output). The input and output images are raster graphics digital images with feature values in the RGB format. The input is the original photograph and the output is the same image, labelled per pixel, by a specific colour representing the coal material that is present at that pixel. The feature space is a matrix of multi-dimensional feature vectors to be classified, and the clusters is a one-dimensional feature valued version of the same matrix where each feature value represents the cluster to which that pixel belongs.

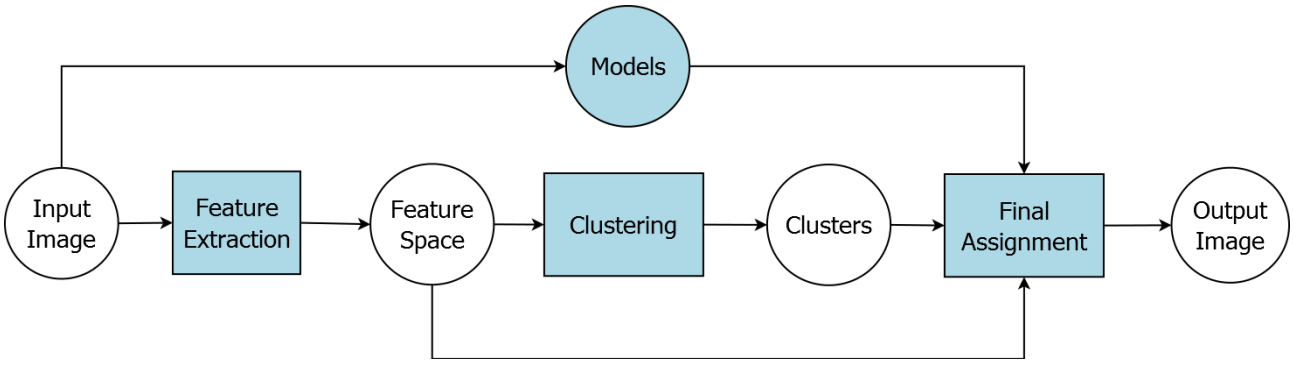


Figure 35: Main flow diagram with highlighted areas for improvement

3.2. Theoretical design: Feature extraction

The purpose of this section is to investigate the possibility of improving upon the existing method of feature extraction. The following are some of the key points of discussion:

- An explanation of the motivations behind the existing method of feature extraction
- Considerations with regards to the colour space
- Considerations with regards to the texture analysis
- A summation of the alternative components to be implemented

3.2.1. Feature engineering

Within the current context, feature extraction is the task of taking an initial set of data points and deriving new data points that are more informative, more applicable, and/or less redundant. In digital image processing the initial data points are usually in the form of a set of red, green, and blue (RGB) colour values for each pixel. If the image is in grayscale, each data point has one value that is determined by its intensity.

In machine learning there are methods of extracting features automatically (usually by means of deep learning algorithms), but these methods require vast amounts of labelled or unlabeled inputs. This is called feature learning. If the system is to be used on coal ore, this would require images of minerals that are already classified and can be used to train the algorithm. The amount of required training images is further increased when considering the wide variation in camera resolution, lighting, and photographing techniques that can be used to obtain the input images. Classified images of coal ore samples are very difficult or costly to obtain, therefore [1] has opted to use features that are derived from the way geologists visually determine what materials coal ore samples are made of. This approach is referred as feature engineering.

Geologists and mineralogists can identify minerals based on their physical properties. Only two of these properties are strictly visual (and, therefore, applicable for image processing). These properties are lustre and streak [56]. Lustre is defined as the way light reflects off of the mineral. Terms that are commonly used to define lustre are: metallic, silky, waxy, or greasy. These terms are much too abstract to be readily used as features in pattern recognition software, and are usually learned by mineralogists with practice. Streak is the term used to define the colour of the powder left by a mineral when it is dragged across a smooth surface. This kind of test can, obviously, not be included in an image processing analysis. However, colour can easily be used as a feature in various types of colour spaces [56].

Lower level features that can be used to estimate lustre and streak in an image are: luminance, chromaticity, and texture. Luminance can be used in conjunction with texture to approximate lustre and chromaticity can be an estimate of the streak. Luminance is the amount of light reflected off a surface in a specific direction. A value for this can be obtained for each pixel in the input image by using the pixel intensity (in other words, the brightness regardless of colour). Chromaticity is the quality of the colour of the pixel (in other words, colour regardless of brightness). These two values can be obtained with several kinds of colour spaces. Texture analysis is usually used to generate a texture value for pixels.

The process of feature engineering explained, above, is depicted in the hierarchy in figure 36. The top of the hierarchy represents the highest level of abstraction and the bottom level represents features that are suitable to be used by a clustering algorithm. Specific texture analysis techniques and colour spaces will be discussed in the sequel.

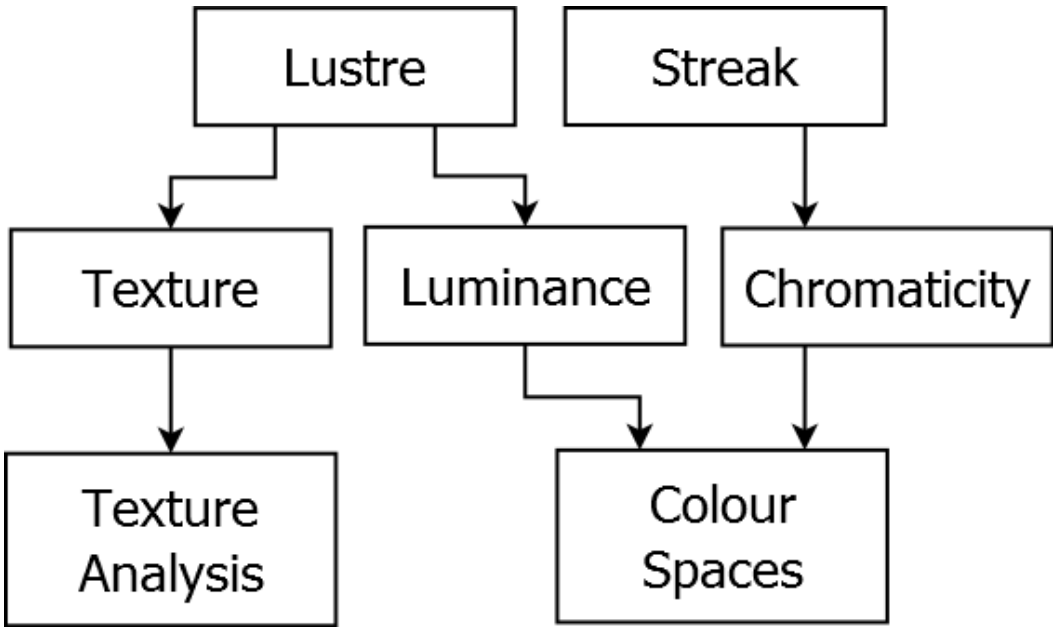


Figure 36: Process of feature engineering

3.2.2. Colour spaces

A colour space is a model that standardises the specification of colours. It is used to enable systems to agree on a specific colour value in terms of a set of defining variables [34]. Most digital images use the RGB colour space but there are several other colour spaces in the literature that have certain advantages and disadvantages to be considered. These colour spaces include [57]:

- Red, Green, Blue (RGB)

  This is, by far, the most commonly used colour space. It is additive, which means that colours are made up of a linear addition of the base channels: red, green, or blue. The main drawback of this colour space is that it is non-linear with visual perception. The simplest way to obtain an indication of the luminance of a pixel is to calculate the average of the red, green, and blue colour channels, effectively obtaining a grey scale representation of the pixel. The chromaticity will then comprise of the red, green, and blue colour channels separately.

- Cyan, Magenta, Yellow, Black (CMY(K))

  An alternative to the RGB colour space is the cyan, magenta, yellow, and black (CMY (K)) colour space. This space is subtractive, non-linear with visual perception, and used in printing. The black channel is added to prevent printers from having to use the other three channels to produce a black colour. Converting from the RGB colour space to CMY(K), accurately is often difficult. Therefore, converting from the native RGB colour space would require effort with almost no reward with regards to approximating a geologist's perception of colour.

- Hue, Saturation, and Lightness (HSL)

  The hue, saturation, and lightness (HSL) colour space can refer to a family of similar colour spaces, including: Hue, Saturation, Intensity (HSI); Hue, Saturation, Value (HSV); Hue, Colourfulness, Intensity (HCI) etc. These colour spaces have the advantage of separating the luminance component from the colour component, thereby allowing easier operations on the intensity value of the image. It is also extremely intuitive to select a specific colour from these colour spaces. However, they are still device dependant and non-linear with human visual perception.

- Television Transmission

  Television transmission colour spaces (e.g. YIQ, YUV, YCbCr, YCC) convert RGB colour information into luminance and chrominance information in such a way that is useful to compression operations. These types of colour spaces are device dependent and meant to be used in systems with stringent operational parameters to which the colour conversion is calibrated.

- CIE

  CIE colour spaces are near linear with visual perception, because they are based on a system that is, in turn, based on human vision. This means that a change in colour value in a CIE colour space results in a similar change in human perception of the colours. They are device independent. There are two variations of this colour space: CIELuv and CIELab.

Keeping these colour spaces in mind, the two most important factors to consider would be the accuracy with which the colour space approximates the human visual perceptions, and the device independence. The choice is then obvious. The CIE colour space is the only one that is close to linear with visual perception, and the only one that is device independent. It is also for this reason that [1] decided to use this colour space. The specific CIE colour space that was used is the CIEL*a*b* (or L*a*b* for brevity) colour space. This colour space has the following channels:

- The L* channel contains the “lightness” information of the colour. This is a good indication of the luminosity of the pixel because it correlates very well with the human perception of lightness.
- The a* channel contains the colour information, contrasting red and green. A very high value indicates a colour with a large red component relative to the green.
- The b* channel contains the colour information, contrasting yellow and blue. A very high value indicates a colour with a large yellow component relative to the blue.

It was decided to use the L* channel as an indication of luminosity, because it provides a good approximation for how humans perceive lightness. The b* channel was used to incorporate the chromaticity information into the system because it provides a good contrast between yellow and blue colour components. It was found that, the little, colour information that can be obtained from coal minerals is, often, polarized between these two colours [1]. A conversion from the native RGB colour space to the L*a*b* colour space can be achieved with the following equations [57]:

$$L^* = \begin{cases} 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 \;\; if \;\; \frac{Y}{Y_n} > 0.008856 \\ 903.3\left(\frac{Y}{Y_n}\right) \;\; if \;\; \frac{Y}{Y_n} \leq 0.008856 \end{cases}$$

$$a^* = 500(f\left(\frac{X}{X_n}\right) - \; f\left(\frac{Y}{Y_n}\right)) \qquad (17)$$

$$b^* = 200(f\left(\frac{Y}{Y_n}\right) - \; f\left(\frac{Z}{Z_n}\right))$$

$$where \; f(t) = \begin{cases} t^{\frac{1}{3}} \;\; if \;\; t > 0.008856 \\ 7.787t + \frac{16}{116} \;\; if \;\; t \leq 0.008856 \end{cases}$$

where XYZ refers to the fundamental measurements in the CIE colour systems, and the subscript n refers to the corresponding value at the white point, which in the case of an 8-bit digital image is (255,255,255).

A very similar alternative to this colour space is the CIEL*u*v* (L*u*v* for brevity) colour space. Converting from RGB to L*u*v* can be done with the following equations:

$$u^* = 13(L^*)(u' - \; u_n') \qquad (18)$$
$$v^* = 13(L^*)(v' - \; v_n')$$

where u' and v' is from the CIE YU'V' colour space:

$$u' = \frac{2x}{6y - x + 1.5} \qquad (19)$$
$$v' = \frac{4.5x}{6y - x + 1.5}$$

where x and y, in turn, is from the original CIE XYZ colour space:

$$x = \frac{X}{X + Y + Z} \qquad (20)$$
$$y = \frac{Y}{X + Y + Z}$$

Note that the L* channel is omitted. This is because the L*u*v* and L*a*b* colour spaces have the same L* value. The results of the conversion functions, provided above, can be seen in figure 37. It can be observed that the a^* and u^* channels, and b^* and v^* channels are very similar. This is due to the fact that both are derived from the same original colour space, the CIE XYZ 1931 colour space. This being said, there is a slight difference between the a^* and u^* colour channels. The u^* channel seems to depict the blue regions with much darker pixel values than the a^* channel. This is because the a^* channel does not contrast red and blue as extensively as the u^* channel.

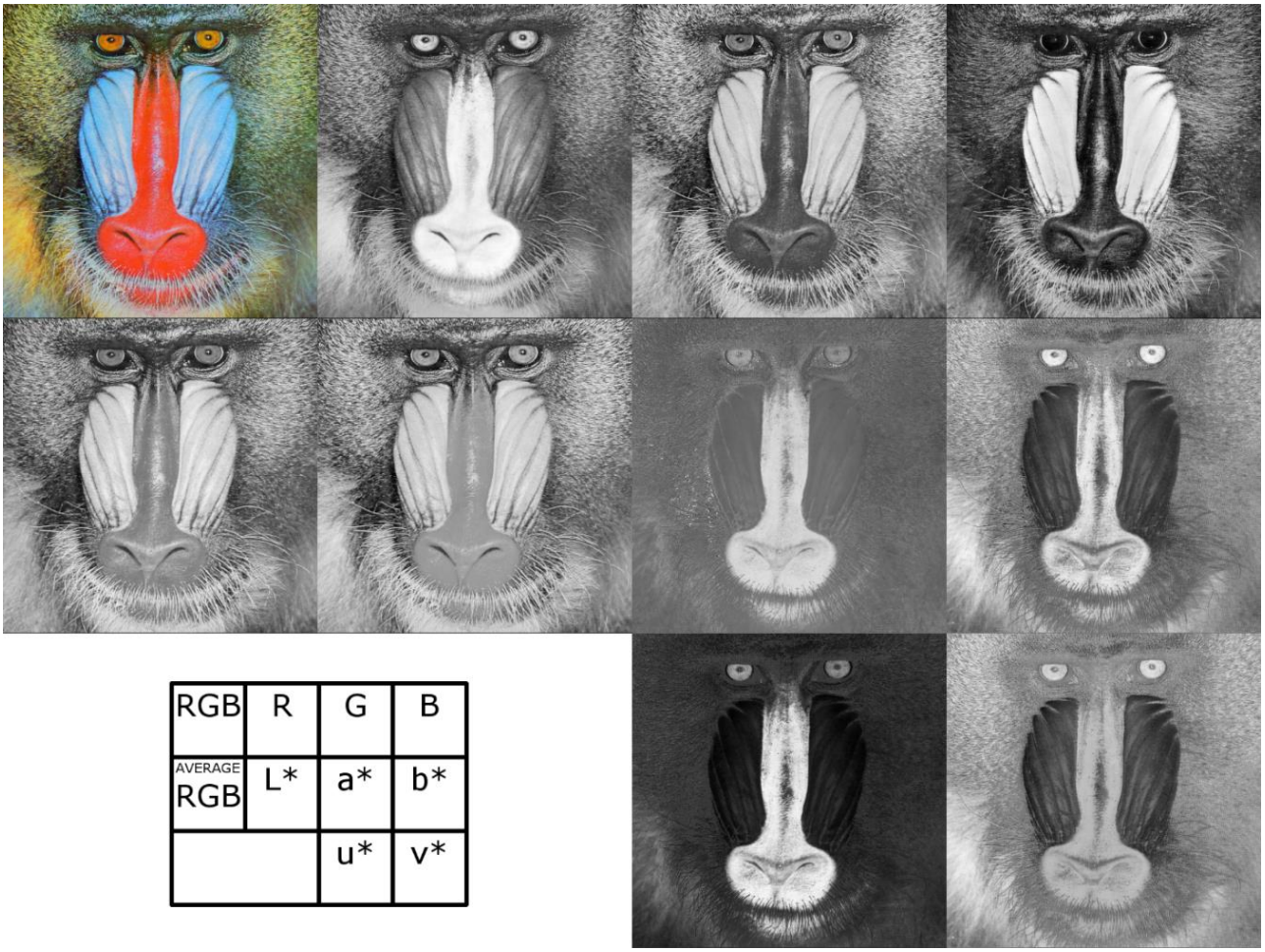


Figure 37: Illustration of colour channels with key legend

To conclude the investigation of colour spaces, the existing system uses the L*a*b* colour space because it is derived from the CIE XYZ colour space that is based on human visual perception and it is device independent. The L* channel is used to estimate the luminosity in the image because it correlates well with human perception of lightness, and the b* channel is used to estimate chromaticity because it contrasts yellow and blue components very well. The L*u*v* colour space is very similar to L*a*b* and it is worth investigating as an alternative feature value.

3.2.3. Texture analysis

Texture can be easily described with semantic meaning, however, there is no mathematically rigorous definition for texture. Definitions of visual texture are commonly based on the specific application where the textures are being identified and used [58]. In the field of image processing, in generalised terms, texture can be defined as a property of an image region [59]. Image textures are quantified using techniques that fall under three main approaches. These approaches are [60] [61]:

- syntactic (commonly referred to as structural)
- statistical
- spectral

Structural approaches are based on the idea that textures are made up of primitive components called texels. Identifying and analysing the arrangement of these components are then used to define texture models [58]. This approach is effective when used on synthetic textures or textures containing repetitive patterns [58] and, therefore, not a good approach for the research pertaining to raw coal ore textures (which will only contain natural textures) [61].

The **statistical** approach models textures based on statistical calculations done on the arrangement of pixel intensities in regions of the image. This approach can characterise textures based on how smooth, course, or grainy they are. They are, often, computationally easy, but seldom do they take the relative spatial information of the pixel intensities into account. [58] This means that two textures that are perceptually different might yield the same texture model [61].

As the name suggests the **spectral** approach makes use of the spatial frequency information of the image by applying the Fourier transform. Global texture patterns that are hard to identify using spatial approaches, can easily be identified with the spectral approach by analysing the high energy bursts in the spectrum. The texture model generated by this approach needs a very high dimensionality in order to effectively describe the texture information [1].

The method proposed in [1] makes use of a more intuitive technique to quantify image textures. Based on the research done in [62], three features are identified that are sufficient to accurately describe texture information. These features are roughness, regularity, and orientation.

- Roughness is the rate of change of pixel intensity in an area of the image
- Regularity refers to the predictability of the texture
- Orientation refers to the level of directionality of the texture along one dimension

These features correlate well with an unrelated study [63] where it was found that a majority of human impressions of visual complexity perception can be explained by perceived levels of: regularity, understandability, directionality, and roughness. Take note that these features were identified using the idea that the human visual system uses a small amount of broadly tuned filters as opposed to a large amount of narrowly tuned filters. What is meant by this, is that the system employed by humans for the interpretation of visual information uses a relatively small amount of sensors, each processing a large amount of information and not a large amount of sensors that are meant to process a small amount of information each. Due to the fact that textures found in coal are largely without any discernible orientation, the only features that are deemed useful are roughness and regularity.

The **roughness** of a surface is the deviations of the normal vector to its ideal direction relative to the surface face of an image region. With regards to image textures, this translates to the rate of change in pixel intensities in the region being analysed. The method used in [1] assigns the median of the gradients of pixel intensities in the immediate region to each pixel. This technique to model roughness can be referred to as a spatial-statistical [58] approach to quantify texture. The spatial information comes from the fact that local information is used to generate the statistical information. It was shown in section 2.3.2 (see equations (9) to (11), and (13)), that there are several methods of obtaining the gradient of an image such as:

- Sobel operator
- Roberts cross
- Prewitt operator
- Laplacian operator

Using the median of the gradients provides a good indication of the tendency that the pixels in any given region have to deviate from the norm, ergo providing a measure of roughness. There are also alternatives to consider with regards to the median filter. This filter provides an indication of the central tendency of the gradients in the image regions. However, there are several filters that can be used to indicate this tendency, such as:

- Mean filter
- Median filter
- Gaussian filter

The "gradient-then-filter" process of generating roughness, explained above, uses the gradient to estimate the degree of variation each pixel provides with regards to its immediate neighbours, and the filter provides an indication of variation in the local neighbourhood of that pixel. An alternative to this process is to use the standard deviation of the pixel intensities in the local neighbourhood of each pixel. The standard deviation is a measure of the variation of data values in a data set. This is very analogous to the definition of roughness, provided previously. The standard deviation has the advantage of being inherently scalable. Meaning that the window used to sample the local neighbourhood can be of variable sizes, as opposed to the gradient operators which uses a fixed 2x2 or 3x3 kernel. Simple standard deviation is defined as:

$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}} \tag{21}$$

where N is the number of data values, x_i is the i^{th} data values, and $\bar{x}$ is the mean data value in the set.

The term "**regularity**", used previously, refers to how predictable a texture is. Textures with repeated patterns are very predictable, and textures with no discernible pattern are difficult to predict. In order to model the predictability of the texture, [1] has used the error produced by creating a low-fidelity model of the texture. The low-fidelity model used is a principle component analysis technique that uses singular value decomposition (SVD) to imperfectly reconstruct the texture and analyse the error incurred after reconstruction. A formal description of SVD follows [64].

Algorithm 6: Singular value decomposition

$Given\ an\ m\ x\ n\ matrix\ M$

$The\ SVD\ of\ M\ is\ given\ by$:

$U, \quad S, \quad V$

$such\ that$:

$U\ is\ an\ m\ x\ m\ matrix$

$S\ is\ an\ m\ x\ n\ diagonal\ matrix$

$V\ is\ an\ n\ x\ n\ matrix$

and

$M = USV^T$

By truncating the singular values contained within the S matrix and reconstructing M, an imperfect model M' can be obtained. By comparing M and M', the error in reconstruction can be used as a measure of the irregularity of M. The error is referred to as the mean squared error, and is given by:

$$e_{ms} = \sum_{j=1}^{N} (x_j - x_j{}')^2 \quad (22)$$

where N is the number of data values, x_j is the j^{th} data value in the set, and $x_j{}'$ is the j^{th} data value in the reconstructed set. An alternative way of approximating regularity is to calculate the entropy contained in the local neighbourhood of each pixel. Recall from section 2.2.2, that image entropy is a measure of the "busyness" or "randomness" of an image. It is related to image compression in that it estimates the minimum amount of data needed to store an image. In this way, irregular patterns in images will also affects local entropy.

To summarise the investigation of texture analysis, the existing system uses measurements of roughness and regularity to model the texture information. In order to obtain roughness values, the median value of local gradient values around each pixel is used. Regularity is obtained by taking the mean square error incurred from a low fidelity model of each pixel's local neighbourhood. Alternatives to these methods include different gradient metrics, smoothing filters, and statistical calculations.

3.2.4. Alternative components for feature extraction

In the previous sections it was explained that the existing system makes use of four features to segment the image, one of which is for luminosity, one for chromaticity, and two are used to model the texture. Table 5 provides a summation of the existing components and the alternatives that were identified in this section. These are the feature extraction techniques that will be incorporated into the system and tested in chapters 4 and 5.

Table 5: Alternative components for feature extraction

Channel	Baseline	Alternatives
Luminosity	• L^*	
Chromaticity	• b^*	• v^*
Roughness	• Sobel operator • Median filter	• Roberts cross; Prewitt operator; Laplace operator • Mean filter; Gaussian filter • Local standard deviation
Regularity	• Singular value decomposition	• Local entropy value

3.3. Theoretical design: Clustering

The purpose of this section is to investigate the possibility of improving upon the existing method of clustering the data points in the feature space. The following are some of the key points of discussion:

- The k-means clustering algorithm
- Algorithms related to k-means
- Particle swarm optimisation
- An applicable mean shift algorithm
- A summation of the alternative approaches to be implemented

3.3.1. k-means

In chapter 2, it was motivated that the most suitable method of image segmentation, for the current field of research, is the partitional clustering algorithm. The most popular partitional clustering algorithm, and the algorithm used in the existing system is k-means clustering. The method can be seen as an optimization problem with the solution being a classification of the data points, into clusters/classes, with minimum intra-class sum of squares. In other words, it seeks to find a partitioning of the data points where the metric distances between data points and the other data points in their class are as small as possible [65].

The two main drawbacks of the k-means algorithm is that it converges to local optimums and will, therefore, not always converge to the global optimum solution, and that it is susceptible to errors due to outliers [66]. This clustering method works by the repeated updating of virtual data points, called centroids, according to the mean feature values of the data points assigned to them. All data points are then reassigned to the new set of centroids until no new assignments are made. The algorithm is heavily influenced by the initial position (and amount) of the centroids. It is necessary to define the number of centroids to be used for convergence before the convergence process is initiated. The k-means clustering method was used by [1] because of its robustness and the fact that it runs in linear time. The general procedure for k-means clustering is as follows:

Given a set of data points in some feature space

1. Choose K initial centroids
2. Assign each data point to its closest centroid to form one cluster per centroid
3. Redefine the centroids as the mean data point value in each cluster
4. Repeat steps 2 and 3 until no further reassignments are possible

The algorithm is fairly simple to comprehend and implement. However, some clarification is needed with regards to step 1 and 2. In step 1, the location of the initial centroids is decided using prior information (such as model values), or chosen at random. In step 2, the word "closest" is synonymous with "most similar in terms of feature values". This measurement is usually made by computing the Euclidean distance between the data point and the cluster centroid. Euclidean distance is defined as:

$$d(a,b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{23}$$

where a and b are n-dimensional data point values, and the subscript i denotes the i^{th} dimension. In its simplest form, the algorithm can be formally described as [19]:

Algorithm 7: k-means

$Given\ a\ set\ of\ N\ data\ points$:

$X = \{x_1, x_2, \dots, x_N\}$

1. $Choose\ a\ set\ of\ K\ initial\ cluster\ centroids\ at\ random$:

   $C(1) = \{c_1(1), c_2(1), \dots, c_K(1)\}$

   $and\ define\ a\ corresponding\ set\ of\ empty\ clusters$:

   $Y(1) = \{y_1(1), y_2(1), \dots, y_K(1)\}$

2. $At\ the\ k^{th}\ iteration, distribute\ X\ among\ Y(K)\ with$:

   $x_i \in y_j(k)\ if\ d(x_i, c_j) < d(x_i, c_k)\ \forall\ i = 1,2, \dots, N, j \neq k$

3. $Compute\ C(k+1)\ with$:

   $c_i(k+1) = \frac{1}{N_i} \sum_{x \in c_i(k)} x, \qquad i = 1,2, \dots, K$

   $where\ N_i\ is\ the\ number\ of\ data\ points\ assigned\ to\ c_i(k)$

4. $Stop\ if\ Y(k+1) = Y(k)$

Therefore, the objective of the k-means algorithm is to minimise the following function [18]:

$$F = \sum_{k=1}^{K} \sum_{\forall x \in y_k} d(x, c_k) \tag{24}$$

The advantages and disadvantages of the k-means clustering algorithm is summarised in table 6.

Table 6: k-means criterion

Advantages	Disadvantages
• Easy to implement • Low time complexity $O(N_p)$	• K needs to be defined in advance • The dependence on initial conditions can result in suboptimal solutions • The use of the mean value to calculate centroids makes it susceptible to outliers

3.3.2. k-related algorithms

In addition to being one of the most popular clustering algorithms, k-means is also one of the most studied algorithms. As a result, the literature contains a large amount of variations of the k-means algorithm. These variations, typically, attempt to eliminate some of the disadvantages of the k-means algorithm.

One variation that is not as heavily influenced by outliers is the **k-medoids** algorithm. This algorithm converges with a similar mechanism to the k-means algorithm, however, instead of using virtual data points (centroids) that are continuously updated using the mean values of their cluster, this algorithm only uses data points that are a member of the dataset (called medoids). What this means is that the effect of outliers will be decreased significantly (because no average values are ever calculated) at the cost of computational complexity. As a result, it is more resistant to noise than k-means. Additionally, this algorithm is also applicable to data sets that cannot be expressed in metric space. [67] The most commonly used method of implementing k-medoids clustering is the partitioning around medoids (PAM) algorithm. This algorithm has the following procedure:

Given a set of data points in some feature space

1. Choose K initial medoids from the dataset
2. Assign each data point to its closest medoid to form one cluster per medoid
3. Swap each non-medoid with each medoid and calculate the global cost of the new configuration
    a. If the cost decreased, keep the configuration
    b. If the cost increased, revert to the previous configuration
4. Repeat step 3 while the global cost decreases

Instead of minimising the intra-class distances, the k-medoids algorithm finds a cluster configuration where the centres are the most centrally located data point in that cluster [68]. The “cost” of a configuration is the sum of distances from all data points to their respective medoids. The distance between data points can be the same as the Euclidean distance used by k-means, although the original realisation of the algorithm [69] used the Manhattan distance. This distance is defined as:

$$d(a,b) = \sum_{i=1}^{n} |a_i - b_i| \tag{25}$$

With each iteration this algorithm replaces every medoid with every other non-medoid data point in the set and calculates the distance of every data point to its currently assigned medoid to determine if the swap resulted in a better solution. This results in an algorithm that runs in quadratic time as opposed to the linear time of the k-means algorithm. It is for this reason that it is generally not favoured for large datasets [67]. In order to reduce this time complexity alternatives to the PAM algorithm have been proposed. One such alternative is presented in [70]. This algorithm is defined as follows.

Given a set of data points in some feature space

1. Choose K initial medoids from the dataset
2. Assign each data point to its closest medoid to form one cluster per medoid
3. Swap each medoid with each non-medoid, from the same cluster, and calculate the cost relating to that cluster
    a. If the cost decreased, keep the configuration
    b. If the cost increased, revert to the previous configuration
4. Repeat steps 2 and 3 while the cost decreases

These types of approaches are faster but, usually, make use of a distance matrix to efficiently compare distances between data points. In section 2.5.3 it was motivated that generating a distance matrix for the current field of research is unrealistic.

Another variation of k-means clustering is **k-medians** clustering. As the name suggests this algorithm calculates the centroid values by using the median feature value within the cluster. Its procedure is identical to that of k-means in every other way. Using the median feature value means calculating the median value of each dimension separately to construct the centroid feature value. Consequently, the within cluster absolute distances are minimised instead of the square distances. [71] It is for this reason that it is commonly used in location theory problems [65].

Considering the separation of dimensions during distance calculations in k-medians clustering, the Manhattan distance seems appropriate. However, as with k-medoids, other distance metrics (such as Euclidean distance) are not incompatible. Unlike the k-medoids algorithm, k-medians are not constrained to select centroids that are part of the dataset. Because of its inherent use of median values this algorithm is more resistant to outliers, and retains the time efficiency of k-means. However, the resulting centroids could possibly be a poor estimation of the central tendencies of the data clusters [72].

The final algorithm that is related to k-means is not a clustering algorithm in its own right. Instead, it is a method of choosing the initial position of the centroids used in the k-means algorithm. And, theoretically it can be used to improve the initial conditions of any iterative clustering algorithm that

requires the placement of centroids to be implemented. This algorithm is referred to as **k-means++** [73].

The idea behind k-means++ is to place the centroids as far apart as possible in the feature space. It has been empirically proven that this will consistently produce more accurate results than a classical randomly seeded k-means algorithm. While the initial procedure of seeding is, obviously, more involved with regards to time complexity, the overall convergence time is improved because of the careful seeding. The algorithm has the following procedure [53]:

1. Choose a data point from the data set X at random to be used as the first centroid
2. Choose the next centroid from the data set according to the probability:

$$\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$$

   where $D(X)$ denotes the shortest distance from a data point to the closest centroid already chosen
3. Repeat step 2 until K centroids have been found

3.3.3. Particle swarm optimisation

Particle swarm optimisation (PSO) is a stochastic optimisation algorithm that can be used for data clustering. It uses a collection (called a swarm) of objects (called particles) to iteratively explore a search-space to find a solution. The algorithm is analogues to how a flock of birds flies through the sky, searching for food. Each bird takes its own best location and the best location found by the whole group into account to adjust its course of flight. This exchange of knowledge enables the flock as a whole to find food sources more efficiently, than a single bird would be able to obtain alone.

Each particle consists of:

- Its current position
- Its personal best position
- Its personal best fitness value
- Its current velocity

During each iteration of the PSO algorithm every particle is updated according to its velocity. Its velocity is determined by:

- Its current position
- Its personal best position
- Its neighbourhood best position

The fitness value of a particle is determined by sum fitness function that is chosen to solve the problem being optimised. After several iterations, or after a stopping criteria is met, the global best position is accepted as the optimal solution to the problem.

The PSO algorithm can be used to implement data clustering by defining the following:

- A “position” is a possible cluster configuration of the dataset
- A “fitness” is some objective function, valued at a specific position, relating to data clustering

The general procedure for a PSO algorithm used for clustering is as follows:

Given a set of data points in some feature space

1. Initialize a swarm of particles, each containing K random centroids as position
2. Update the global and local best fitness values
3. Update the particle velocities
4. Update the particle positions
5. Repeat steps 2, 3, and 4 until convergence is reached

To compare positions (cluster configurations) and determine the best position, the positions are rated according to a fitness function. This function is dependent on the solution that needs to be found. An appropriate fitness function of a clustering exercise would be [18]:

$$F = w_{intra} d_{max} + w_{inter}(z_{max} - d_{min}) \quad (26)$$

where

- w_{intra} is a constant weight representing the importance placed on minimising the intra-class distances
- w_{inter} is a constant weight representing the importance placed on maximising the inter-class distances
- d_{max} is the maximum mean distance of data points to their assigned cluster centroid
- d_{min} is the minimum distance between any pairs of cluster centroids
- z_{max} is the maximum possible pairwise distance in the data set

Minimising this function will minimise the within-cluster distances and maximise the between-cluster distances. This is referred to as a multi-objective function because it optimises two criteria simultaneously.

An alternative fitness function that would only optimise the within-cluster distances is [74]:

$$F = \frac{1}{K}\sum_{j=1}^{K}\left(\frac{1}{N_j}\sum_{\forall x \in C_j} d(x, C_j)\right) \quad (27)$$

where

- K is the total number of clusters
- N_j is the total number of data points assigned to the j^{th} cluster
- $d(x, C_j)$ is the distance between data point x and its cluster centroid C_j

Take note that this fitness function is a single objective function and optimises the same criteria as k-means, k-medians, and k-mediods clustering. Particle positions are updated by adding a velocity vector to the position. The velocity of a particle is updated with the following equation:

$$V(t) = W_i V_p + W_c r_1(P_{pb} - P_c) + W_s r_2(P_{nb} - P_c) \quad (28)$$

where

- W_i is a constant representing the inertia weight of the update. This value determines how strongly a particle will retain its current trajectory before accelerating in another direction.
- V_p is the particle's velocity during the previous update iteration
- W_c is a constant representing the cognitive component of the algorithm. This value determines the importance a particle will place on its own best position.
- W_s is a constant representing the social component of the algorithm. This value determines the importance a particle will place on the best position obtained by the swarm as a whole
- r_1 and r_2 are random variables ranging from 0 to 1. These values introduce uncertainty into the algorithm to prevent premature convergence.
- P_c is the current position of the particle
- P_{pb} is the personal best position of the particle
- P_{nb} is the best position in the neighbourhood of the particle

The relation between the cognitive component, social component, and inertia weight determine the nature of the algorithm's convergence. A larger cognitive component promotes an exploratory search for the solution, where particles are more likely to search farther away from each other in the search-space. A larger social component results in an exploitative search through the search-space where particles quickly converge to the same position. The inertia weight also decreases exploration because it impedes the particle's ability to quickly change direction. The exploitative algorithm converges much faster than the exploratory algorithm, but is also more likely to prematurely converge to local optimum solutions.

Note, from the velocity equation, that several of the constants need to be carefully calibrated to prevent premature convergence. The convergence of the PSO algorithm has been studied extensively. A PSO algorithm that is poorly calibrated tends to either converge to a local optimum prematurely, take excessively long to converge, or become unstable (leave the search-space). Convergence refers to a state where all the particles have moved to the same location in the search space. This state can be identified by measuring the velocities of the particles in the swarm. When all the velocities have fallen below some threshold value, convergence is achieved.

In order to prevent the particles from exploding, a constraint can be placed on the achievable velocity of a particle. This is a simple clamping procedure, and can be implemented with the following equation [75]:

$$V = \begin{cases} V_T \; if \; V > V_T \\ -V_T \; if \; V < -V_T \end{cases} \tag{29}$$

where V_T is the threshold velocity. Other constraints have been proven to guarantee convergence such as [76]:

$$W_i > \frac{1}{2}(W_c + W_s) - 1 \tag{30}$$

In addition to such constraints, the concept of neighbourhood topologies can be introduced into the algorithm. In the most basic rendition of PSO a global topology is used. This means that all particles can exchange information. In this topology, the entire swarm will be affected by the global best position, and the swarm might converge to that position prematurely. Neighbourhood topologies limit the amount of particles that can exchange information about their best positions. This enables the algorithm to explore more locations in the search-space, and prevents premature convergence. Several studies have been done to inspect the effect these topologies have on the propagation of information in the swarm [77] [78].

Figure 38 provides some commonly used neighbourhood topologies. From left to right, they are listed below:

- The global topology. In this topology all particles are connected to each other.
- The star/wheel/hub topology. One particle acts as a hub that is connected to all other particles, but none of the other particles are connected.
- The circle/ring topology. This configuration connects all particles to only two other particles in a ring formation.
- The grid (Von Neumann) topology. The particles are connected in a two dimensional grid formation with a maximum of four connection per particle.

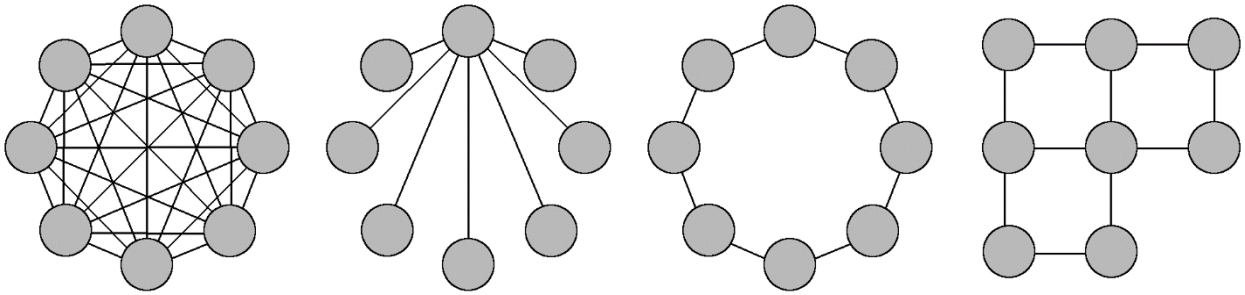


Figure 38: PSO neighbourhood topologies

The advantages and disadvantages of using the PSO algorithm to cluster the image data is summarised in table 7.

Table 7: PSO criterion

Advantages	Disadvantages
• Can optimise the intra- and inter cluster distances simultaneously • Can consider multiple solutions simultaneously	• Runs the risk of premature convergence • Large amount of parameters to be calibrated • Higher time complexity than classical clustering methods

3.3.4. Mean shift

The mean shift is a mode seeking algorithm that was first developed as a nonparametric method of finding the gradients in a density function [79]. What this means for data clustering is that, when the feature space is viewed as an empirical probability density function, the local maxima can be found. These local maxima will then correspond to the most densely packed regions in the feature space, which is a good indication of a cluster centre location [80].

The algorithm works by, iteratively, shifting each data point in the feature space to the average data point value in its neighbourhood until convergence. The shifting is done by positioning a kernel (with a dimensionality equal to that of the feature space) centred on the data point and calculating the mean value of all data points that fall within the kernel. The kernel is then centred on the mean value, and the calculation is repeated. The proof of this process converging can be found in [54].

The mean shift algorithm can be formally described as:

Algorithm 8: Mean shift

$Given\ a\ set\ of\ N\ data\ points$:

$X = \{x_1, x_2, \dots, x_N\}$

$Also\ given\ a\ kernel\ function\ K.$

$A\ mean\ shifted\ X\ is\ a\ set$:

$Y = \{y_1, y_2, \dots, y_N\}$

$Obtained\ by\ the\ following\ procedure$:

$for\ i = 1,2, \dots, N$

$1. x_c(0) = x_i$

$2. x_c(j+1) = \frac{1}{N_K} \sum_{x \in X} xK(x, x_c(j))$

$3. y_i = x_c(j)\ if\ x_c(j) = x_c(j+1)$

$where$

- $j\ denotes\ the\ iteration\ count$
- $N_k\ is\ the\ kernel\ count\ of\ K$

The kernel function determines which data points are included in the calculation to determine the shift vector in the next iteration. Examples of such a kernel are:

- A flat spherical kernel

$$K(x_1, x_2) = \begin{cases} 1\ if\ d(x_1, x_2) \leq d_K \\ 0\ if\ d(x_1, x_2) > d_K \end{cases} \quad (31)$$

where d_K is the radius of the kernel, and d is some distance metric.

- A Gaussian kernel

$$K(x_1, x_2) = e^{-\frac{d(x_1, x_2)^2}{2\sigma^2}} \quad (32)$$

where σ is the standard deviation of the kernel.

Applying the mean shift algorithm to a feature space, as defined above, will move the data points closer together according to the density distribution of the feature space. In other words, densely packed groups of data points will be packed even more densely. Obviously, in a clustering exercise, the actual data will not be manipulated, rather their points of convergence and shift vectors will be taken note of to indicate the cluster they belong to. With regards to image segmentation, that would result in a smoothing of relatively homogenous regions. Thereby, easing the process of clustering. If the algorithm is implemented to completion, meaning that all data points are allowed to converge regardless of the iteration count, the result would be a collection of feature values that correspond to the true local maxima. These local maxima are a good indication of the "true" cluster centres contained in the feature space. However, this is an ideal outcome, and assumes the kernel parameters are chosen perfectly.

Even in the original research where the algorithm was proposed [79] it was noted that such an algorithm is costly in terms of, both, time and space complexity. In [52] it was noted that this method of clustering is superior to classical methods of clustering (e.g. k-means) in the way that it does not make use of questionable initial clustering states that could be erroneous from the start. Mean shift algorithms are, in this way, more deterministic than stochastic by design. It was also stated that the shifting vectors can be used as centroid trajectories that merge at some point in the iterative process to produce a range of possible clusters. This is reminiscent of the merging and splitting mechanisms of the hierarchical clustering approach. However, it was again made clear that the mean shift algorithm suffers from high computational complexity ($O(n^2)$ for one iteration) and might not be suited to problems with prohibitive sizes and dimensionalities.

The research done in [81] aimed to reduce the computation time by making use of locality-sensitive hashing as a means of dimension reduction. This approach is unnecessary with the current field of study, because of the fact that a relatively low amount of feature dimensions are used. A colour image segmentation method is proposed in [82] that uses the mean shift algorithm as a pre-processing filter to enhance the computation time of a normalised cut algorithm. The exact mean shift algorithm that was used was proposed in [80], which is not entirely classified as a clustering algorithm, but has much lower computation time than the ideal algorithm defined above. And this, by no means, reduces its viability, because of the fact that the intention is to use clustering to segment an image. In [54] and [80] a mean shifting algorithm was proposed that is specifically developed to implement image segmentation. This algorithm, first, constructs a d-dimensional feature space out of the image data. This feature space contains the image feature values, appended by two feature dimensions that represent appropriately normalised spatial coordinates (the horizontal and vertical axes).

The joint spatial-range feature space is then filtered with a mean shift algorithm using a kernel that is calibrated using constants relating to the range and spatial information. This filter does not converge to the true local maxima due to the incorporation of spatial information. The resulting collection of filtered data points are analysed and those that are closer together than some threshold in the feature space are joined to create the final cluster configuration. This procedure has much lower time complexity than a mean shift algorithm applied to the original feature space alone, and the process can be accelerated further by limiting the kernel width in the spatial domain as well.

The algorithm described, above, appears to be well suited to the image segmentation problem being addressed in this dissertation, because it combines the attractive benefits of a clustering operation based on a mean shift algorithm with the spatial information of the image domain. Additionally, it reduces the time complexity which is typical of a classical mean shift algorithm. The advantages and disadvantages of using the mean shift algorithm to cluster the image data is summarised in table 8.

Table 8: Mean shift criterion

Advantages	Disadvantages
• Does not assume any prior knowledge about the data set • Requires no initial conditions to be calibrated • Clusters aren't constrained to being isometric as is the case with k-means	• Mean shift clustering in its purest form is computationally complex • Has some issues with higher dimensional feature spaces

3.3.5. Alternative components for clustering

In the previous sections it was explained that the existing system makes use of the k-means clustering algorithm to identify the grouping of pixels in the feature space. Several alternative clustering methods have been discussed. Table 9 provides a summation of the existing method of clustering and the alternatives that were identified in this section. These are the clustering techniques that will be incorporated into the system and tested in chapters 4 and 5.

Table 9: Alternative components for clustering

Baseline	Alternatives
• k-means	• k-medians • k-medoids • PSO • Mean shift

3.4. Theoretical design: Models and final assignments

The purpose of this section is to investigate the possibility of improving upon the existing method of modelling the coal materials in the sample. The following are some of the key points of discussion:

- The models used in the existing system
- Using multiple training areas for models
- The parameters used to represent models
- Cluster-to-model assignments
- A summation of the alternative approaches to be implemented

3.4.1. Existing models

The existing method of modelling the coal ore materials is to have the user highlight specific regions in the image as exemplars of that material class. The feature extraction is then done on the image and the median feature values contained in the highlighted areas are used as model centroids, according to which the final centroids will be assigned during the final assignments operation. Figures 39 and 40 provides a visual presentation of this process.

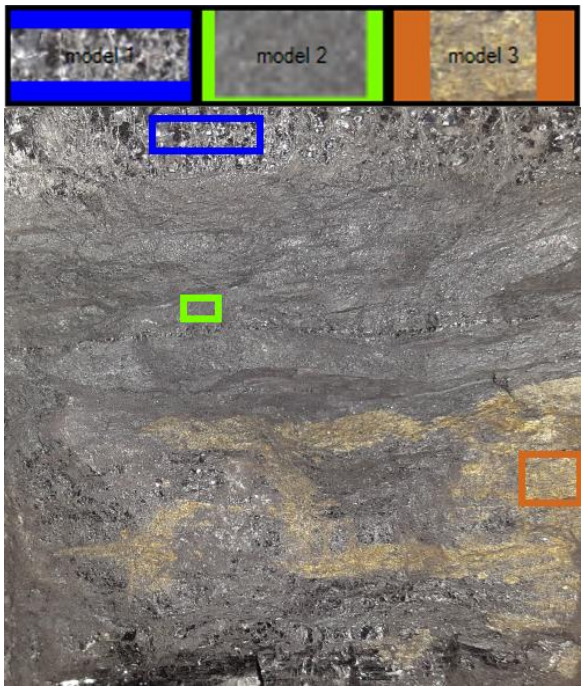


Figure 39: Model highlighting



Figure 40: Model centroids

Notice, from the radar charts in figure 40, that the four model centroids are four dimensional feature values corresponding to the feature space of the existing system. The feature values have been normalised to a range of 0 to 100. Also take note that the rougher texture in model 1 has an appropriately high roughness value compared to the other two models, and that the yellowish pigment of model 3 is well represented by its high value in terms of the b* colour channel.

3.4.2. Multiple models

In an ideal scenario, the clusters obtained from the clustering component of the system would correspond well to one, and only one, of these model centroids. However, in real world images the feature space rarely contains the exact amount of clusters as the model values. The amount of clusters in the feature space is, typically, higher than the model. This results in having to match several clusters to each of the model values. Edge cases can occur where a cluster would correlate equally well with two or more models.

Suppose that several areas are selected per model and that the clusters are then assigned to these “sub-models” before assigning them to the true model materials. This would significantly reduce the uncertainty of assigning edge cases to model values. Granted, that it is not in the spirit of automating the image segmentation process. However, the lack of training material has had an effect in deciding to use the manual selection process in the existing system. It is not unreasonable to exploit manual inputs a little further and make use of multiple area selection. This will increase the accuracy of the image segmentation, and in no way decrease the time efficiency of the system as a whole.

3.4.3. Model parameters

It was stated in the previous sections that the median feature value in the highlighted areas are used as model centroids. This decision was, presumably, made to reduce the effects of outlier pixels in the models. The obvious alternative is the mean feature value. This parameter might be more susceptible to outliers, but considering the extensive use that is made of mean feature values and Euclidean distances in the previously explain clustering approaches, it is not an unrealistic expectation that the mean might be a superior method of approximating the model area parameters.

3.4.4. Model assignment topologies

In the previous section it was noted that there are usually more clusters in the feature space than there are models to segment the image into. The logical process of assignment would be to assign each cluster to the model centroid with which it is the most similar. This is illustrated in the left panel of figure 41. The green spheres denote the models and the yellow spheres denote the clusters. Alternatively, each model can be matched to only one cluster. These clusters will then be used as the new “model” clusters, according to which the rest of the clusters are assigned. The intuition behind this is to identify one cluster per model that is physically a part of the clustered data set. This should be a more accurate model value.

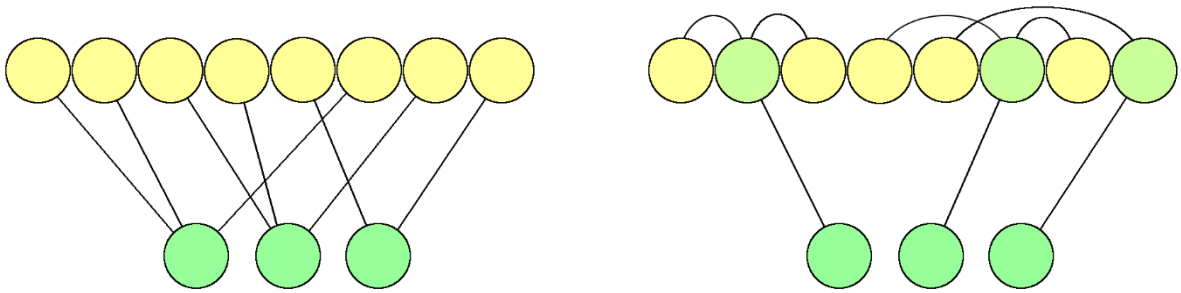


Figure 41: Model-to-cluster assignment

A second alternative is to, first, segment the image according to the clusters, and then assign the resulting regions to the model values. This would be achieved by calculating the mean/median feature value of every segmented region to compare it to the model centroids. Recall from section 2.1.3 that the regions and corresponding region classes (clusters in this case) do not share the exact same feature values. The results would, therefore, be different to the assignments in figure 41, and introduce an additional degree of spatial information into the entire system.

3.4.5. Alternative components for models and final assignments

Table 10 provides a summation of the existing method of coal material modelling and the identified alternatives. These are the alternatives that will be incorporated into the system and tested in chapters 4 and 5.

Table 10: Alternative model parameters

Baseline	Alternatives
• Single area per model	• Multiple areas per model
• Median feature value	• Mean feature value
• Assign clusters to the model with the highest similarity	• Assign clusters to a cluster acting as a model by proxy • Assign segmentation regions to the model values

4. TESTING

4.1. Ground truths

The purpose of this section is to discuss the image data that is used to test and rate the alternative techniques presented in chapter 3. The following are some of the key points of discussion:

- The original input images of raw coal ore samples
- The classification data

4.1.1. Input images

Six, partially labelled, digital images of raw coal ore samples are available. The coal photographed in these images is from various coal mines in Southern Africa. Annotated versions of these images (as they have been received from the geologists) are provided in figures 42 to 47. Henceforth, these input images will be referred to as sample 0 to sample 5, in the order that they are presented, below. Note the annotated areas that have been identified. These areas depict the mineral materials of which each sample consists.

- Sample 0

  The coal in this image is from Moatize in Mozambique. It is a microscopic photograph and contains vitrinite- and mineral matter components. The dark areas are cracks in the sample and will need to be segmented along with the other material classes.

- Sample 1

  This is a macro (but not microscopic) photograph of coal from a coalfield in Limpopo, South Africa. The main components consist of vitrinite and mineral matter, but there are minor regions of mudstone and calcite.

- Sample 2

  This photograph is of another coal sample from Moatize in Mozambique. However, this image is not microscopic. The photographed sample contains several mineral components such as vitrinite, pyrite, mudstone, and calcite.

- Sample 3

  The coal photographed in this sample is from a Witbank coalfield in South Africa. It contains pyrite, vitrinite, inertinite, and calcite components.

- Sample 4

  The fifth sample is another photograph of coal originating in Witbank. The coal consists of gradational mixtures of inertinite and macerals-plus-mineral matter.

- Sample 5

  The last sample also comes from Witbank and contains vitrinite and calcite in addition to the inertinite and macerals-plus-mineral matter in the previous sample.

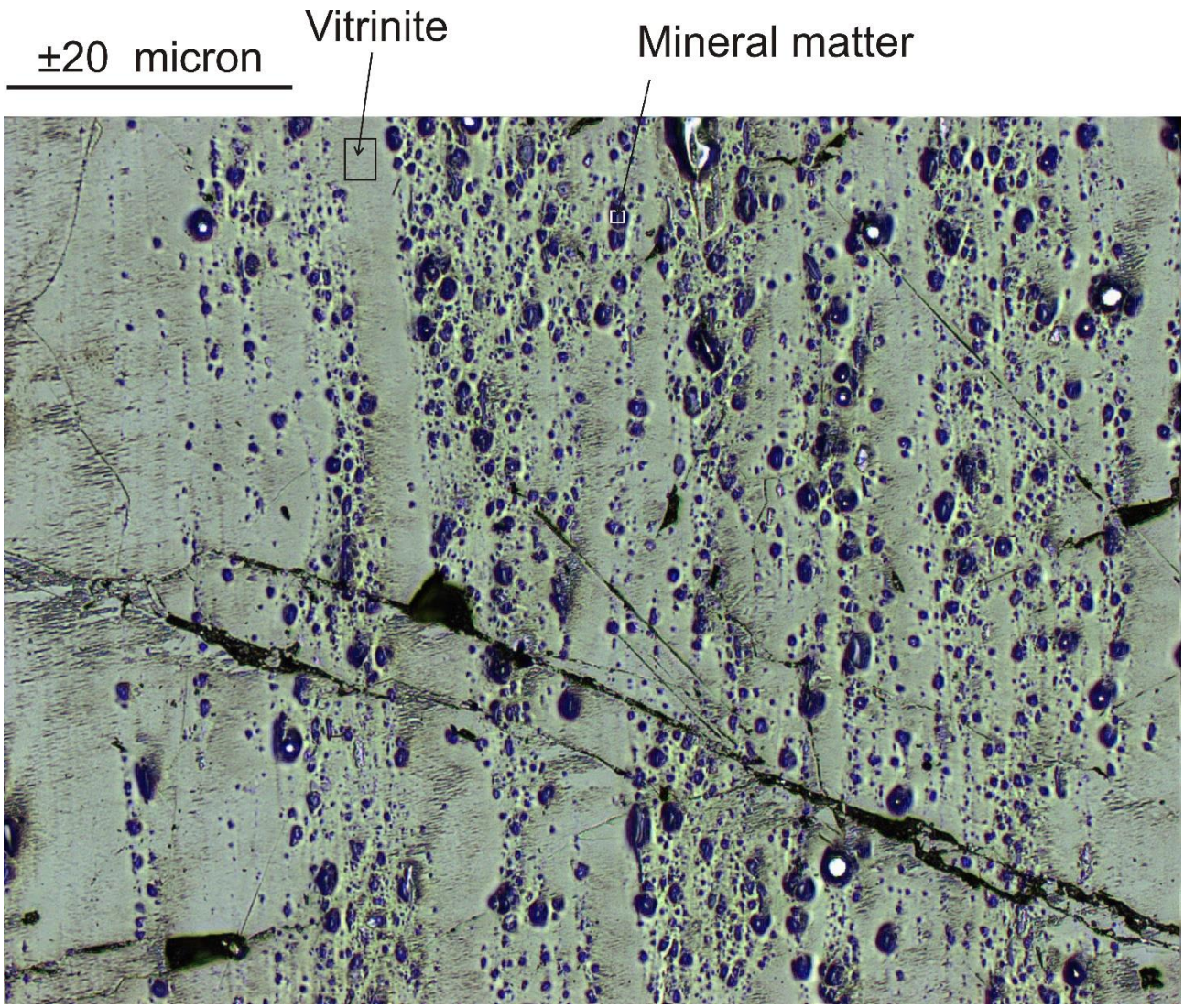


Figure 42: Sample 0 annotated image



Figure 43: Sample 1 annotated image

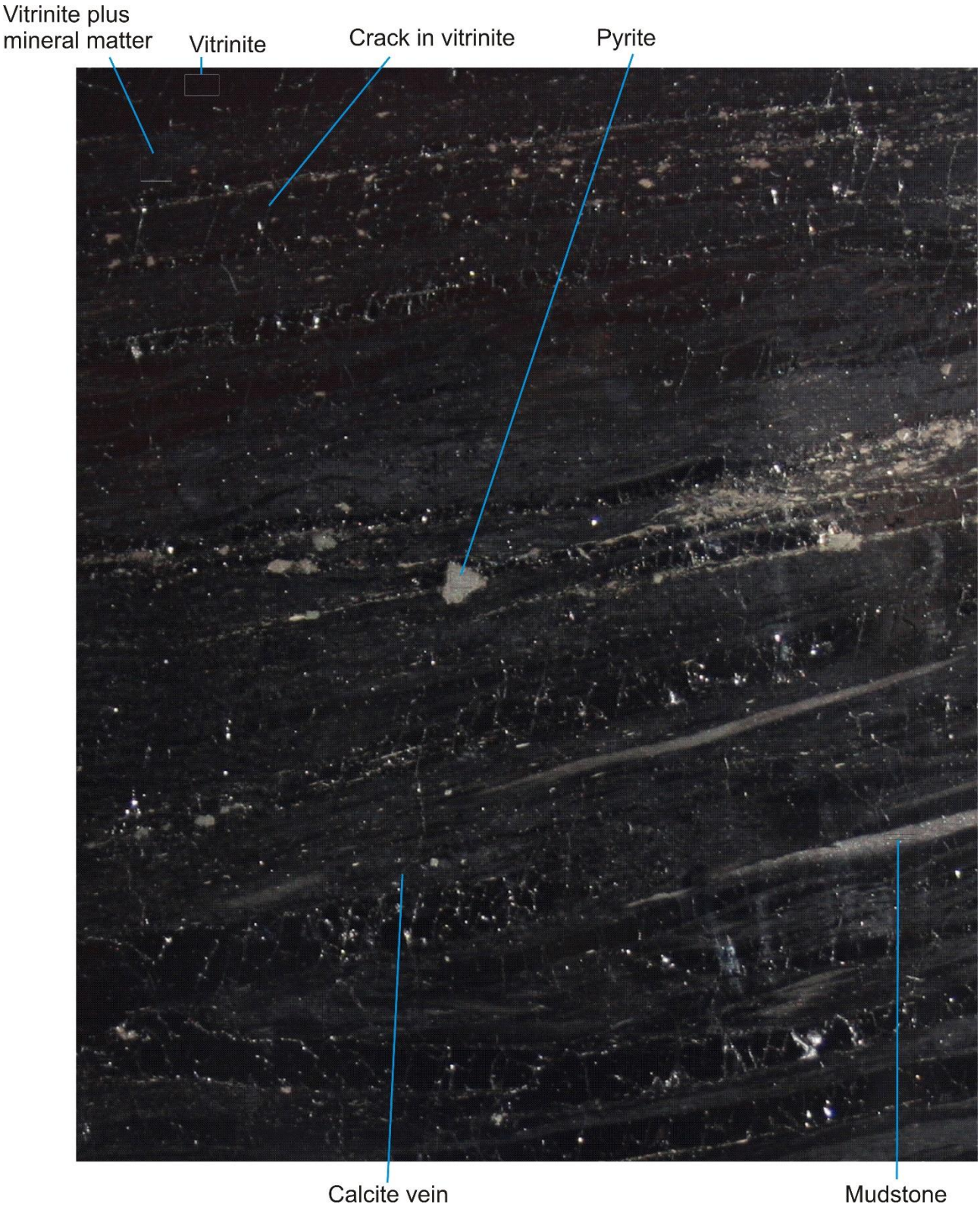


Figure 44: Sample 2 annotated image



Figure 45: Sample 3 annotated image



Figure 46: Sample 4 annotated image

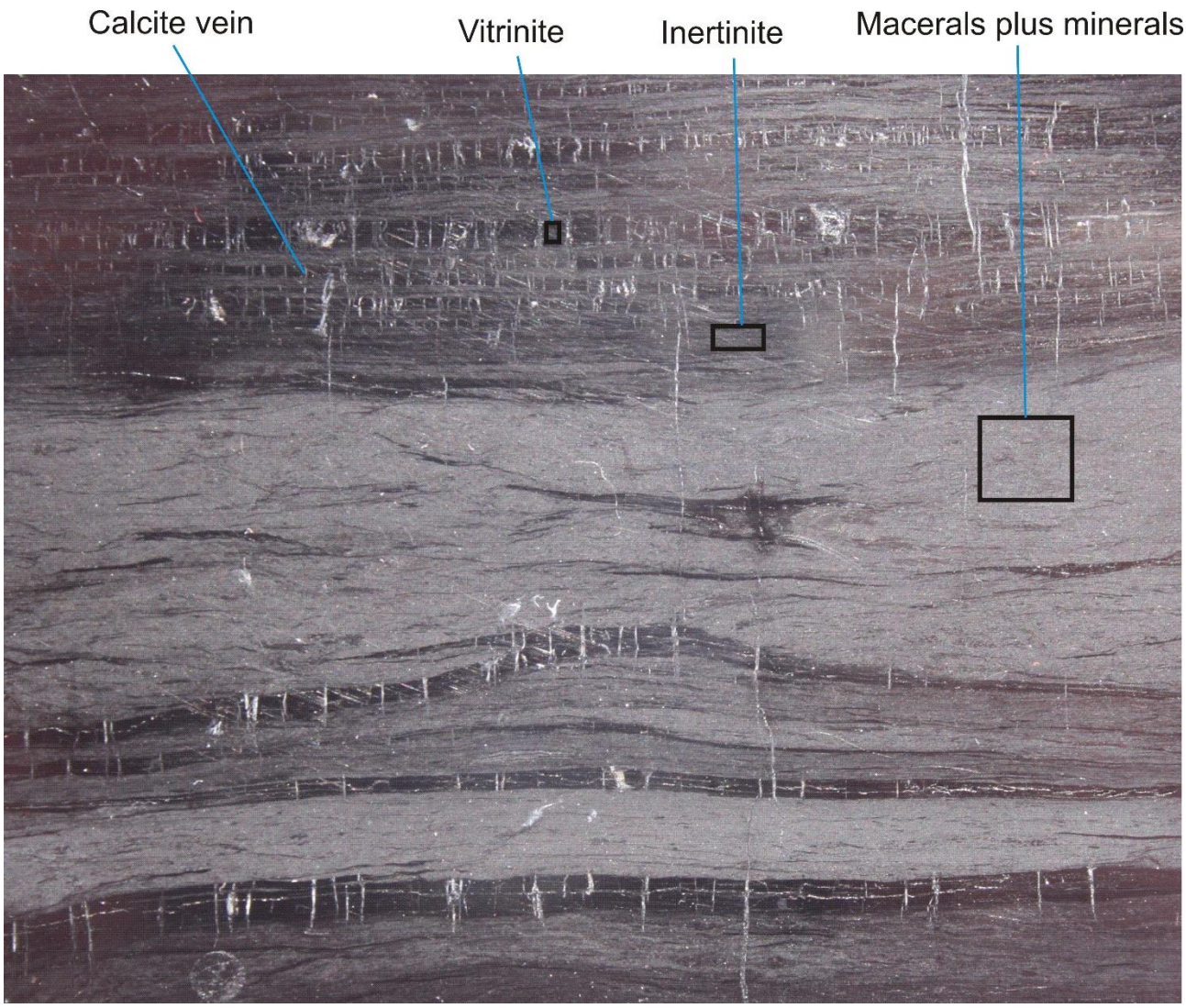


Figure 47: Sample 5 annotated image

4.1.2. Classification data

Without quantifiable data about the mineral classes within the image (according to the image coordinate matrix), rating the quality of segmentations of the sample images is not, at all, possible. The ideal ground truth data would be versions of the sample images where they have already been segmented according to the mineral contents. However, no such images are available and it is unrealistic to expect that a geologist can painstakingly classify each pixel in these images by hand (the smallest of the samples has more than 1.3 million pixels).

As a compromise, a software application was developed that allowed geologists to classify a collection of equidistant pixels in the images instead. These pixels are orders of magnitude fewer than the entirety of the image and are uniformly placed at static positions within the image matrix to provide a good representation of the overall image content. Figures 48 to 53 provide the results of these approximated hand-classifications, received from the geologists.

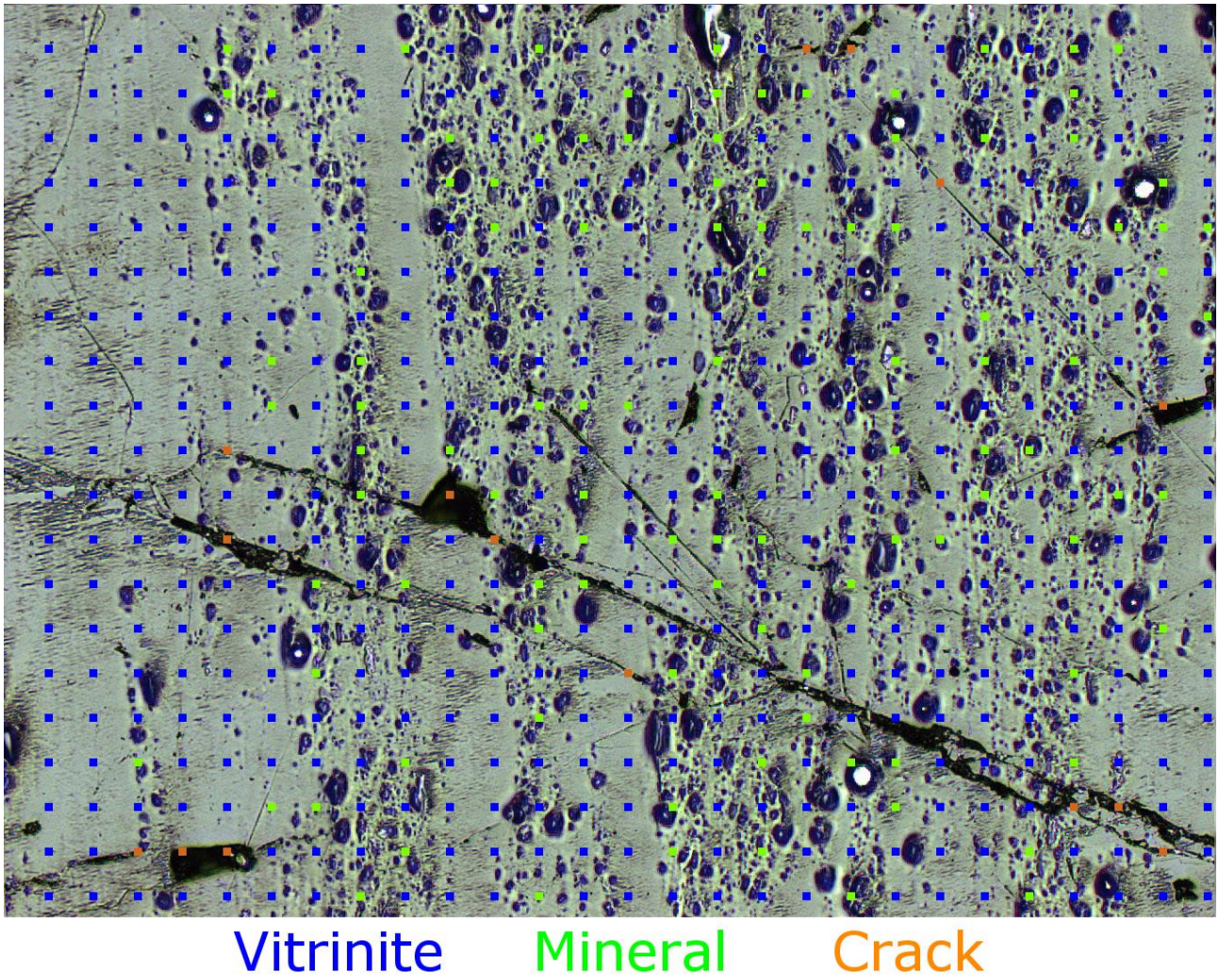


Figure 48: Sample 0 hand classification

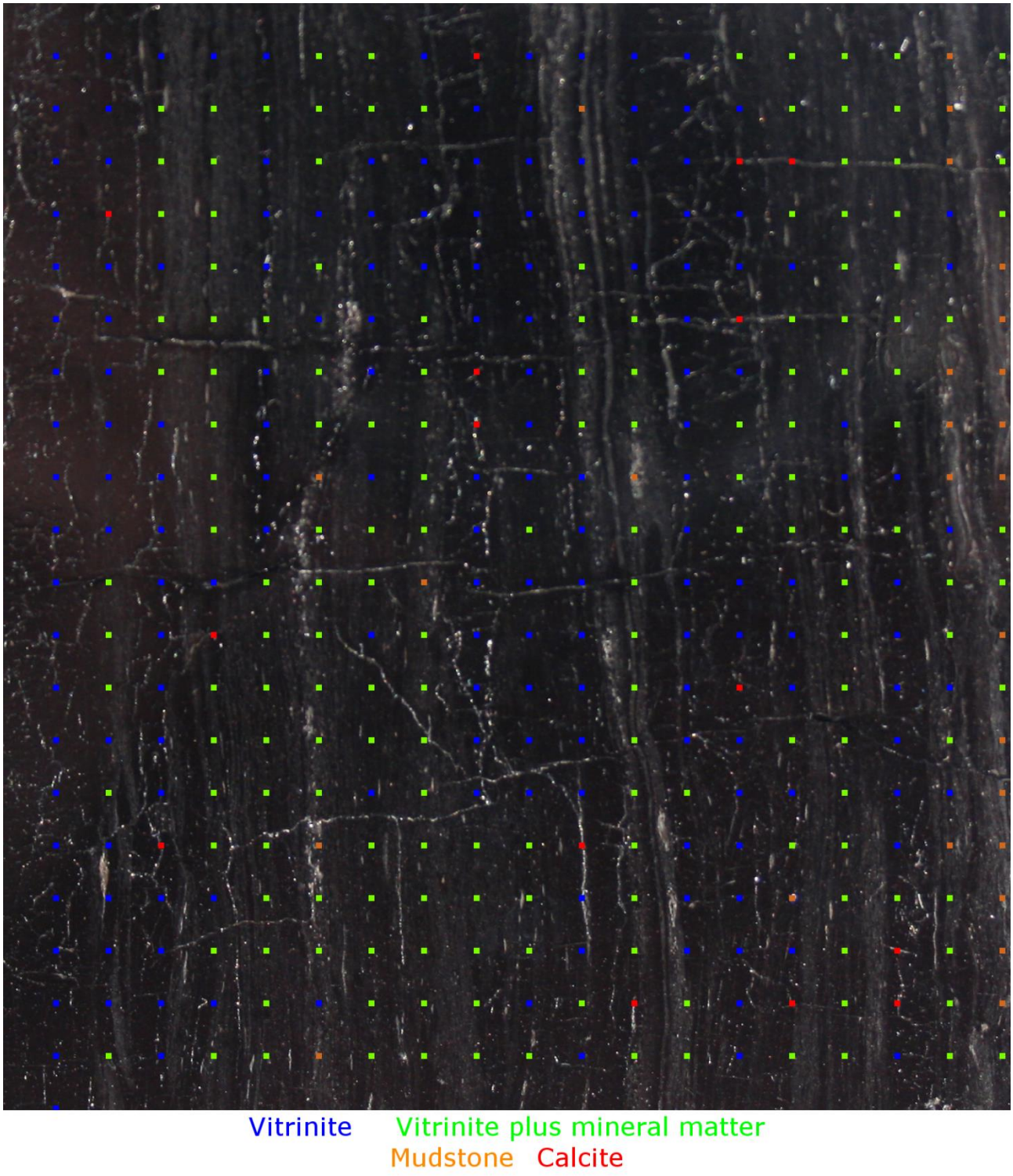


Figure 49: Sample 1 hand classification

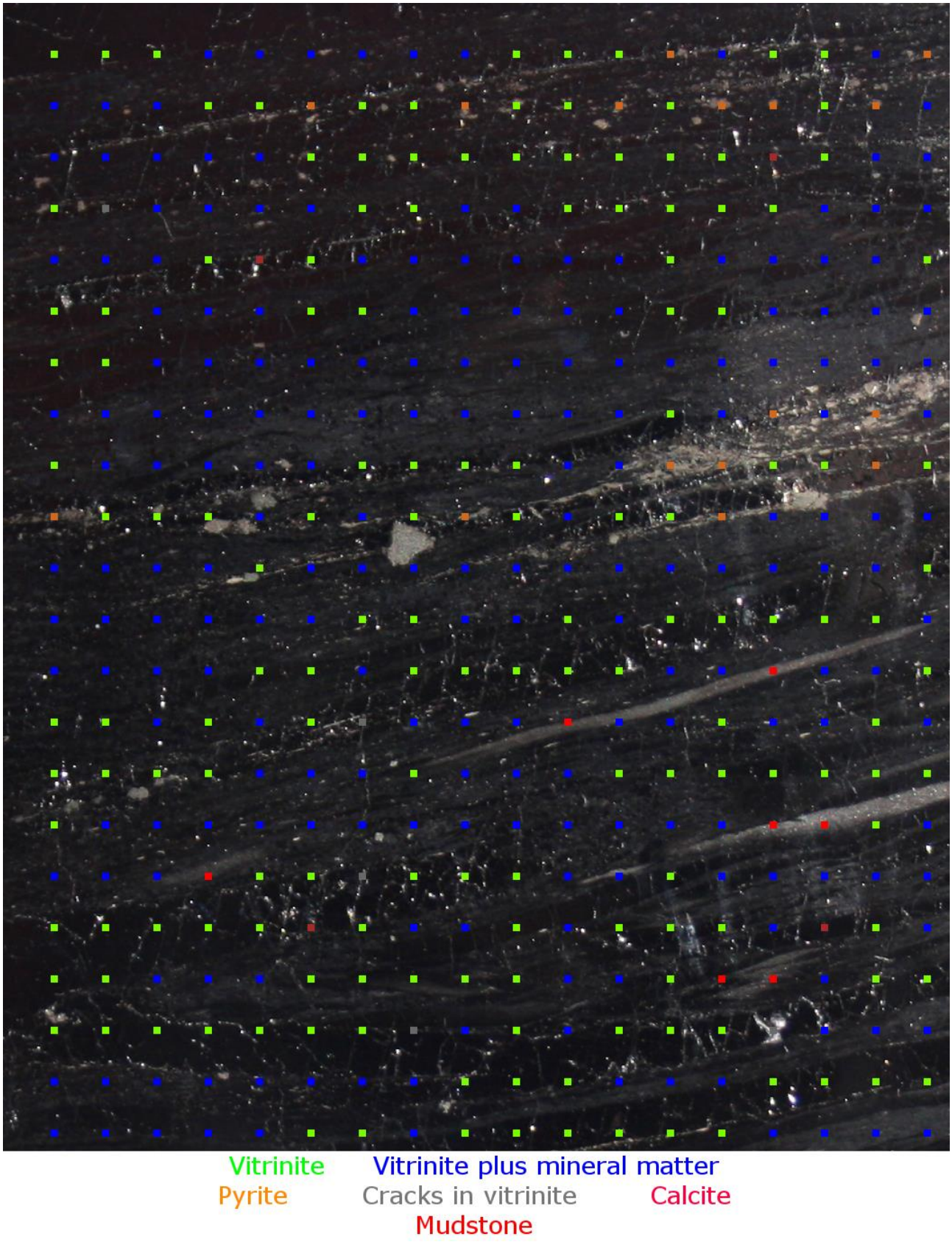


Figure 50: Sample 2 hand classification

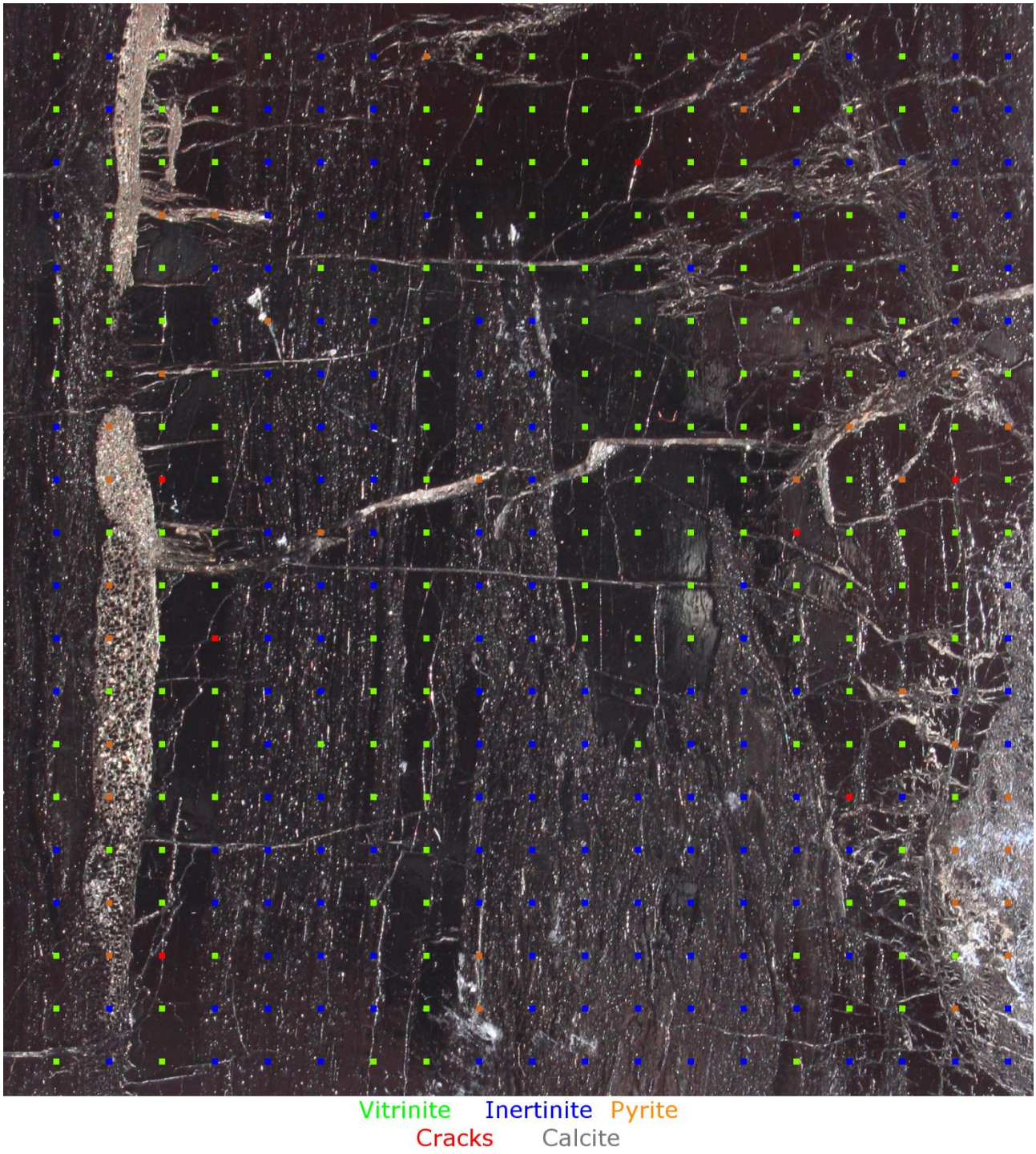


Figure 51: Sample 3 hand classification

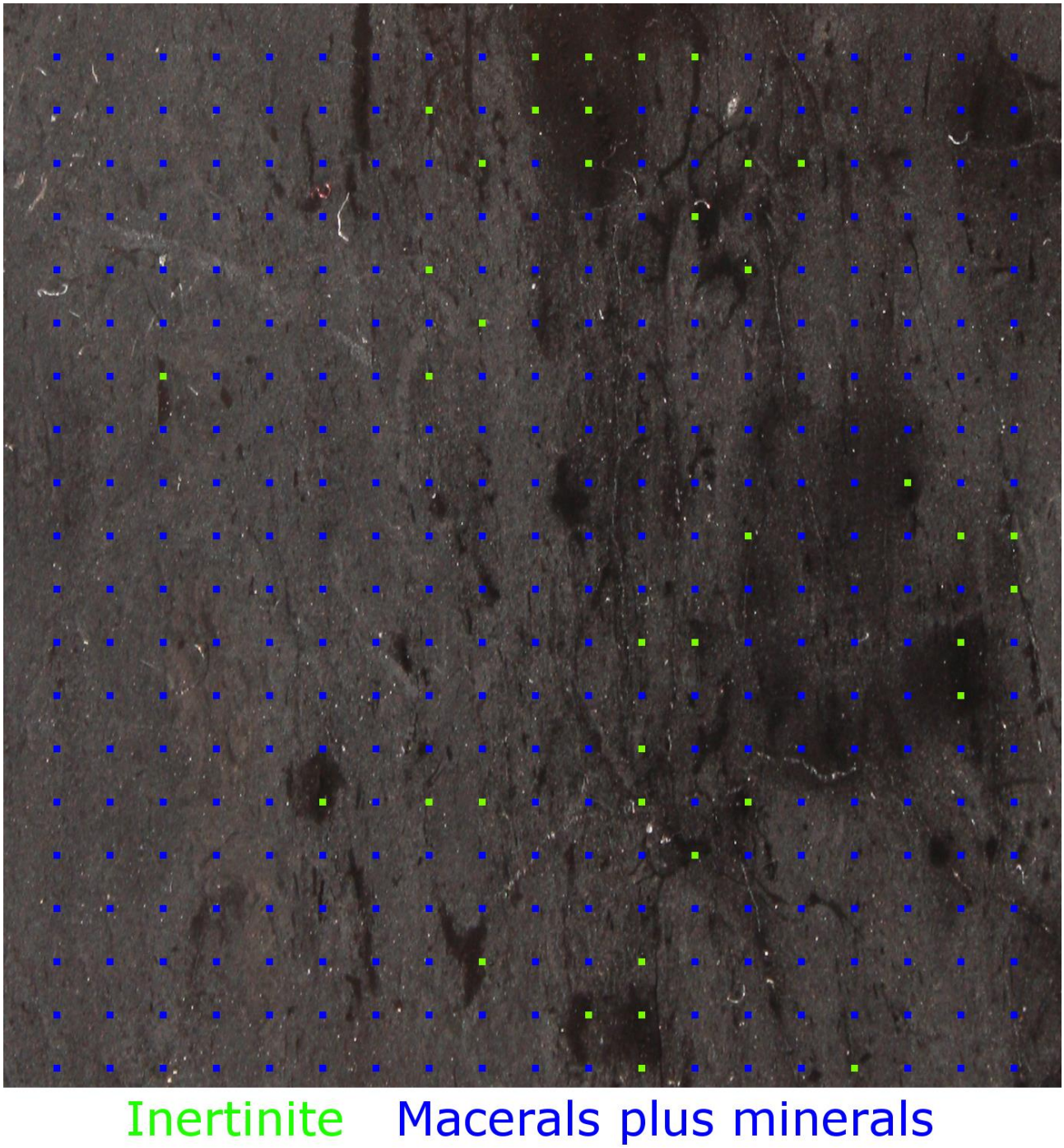


Figure 52: Sample 4 hand classification

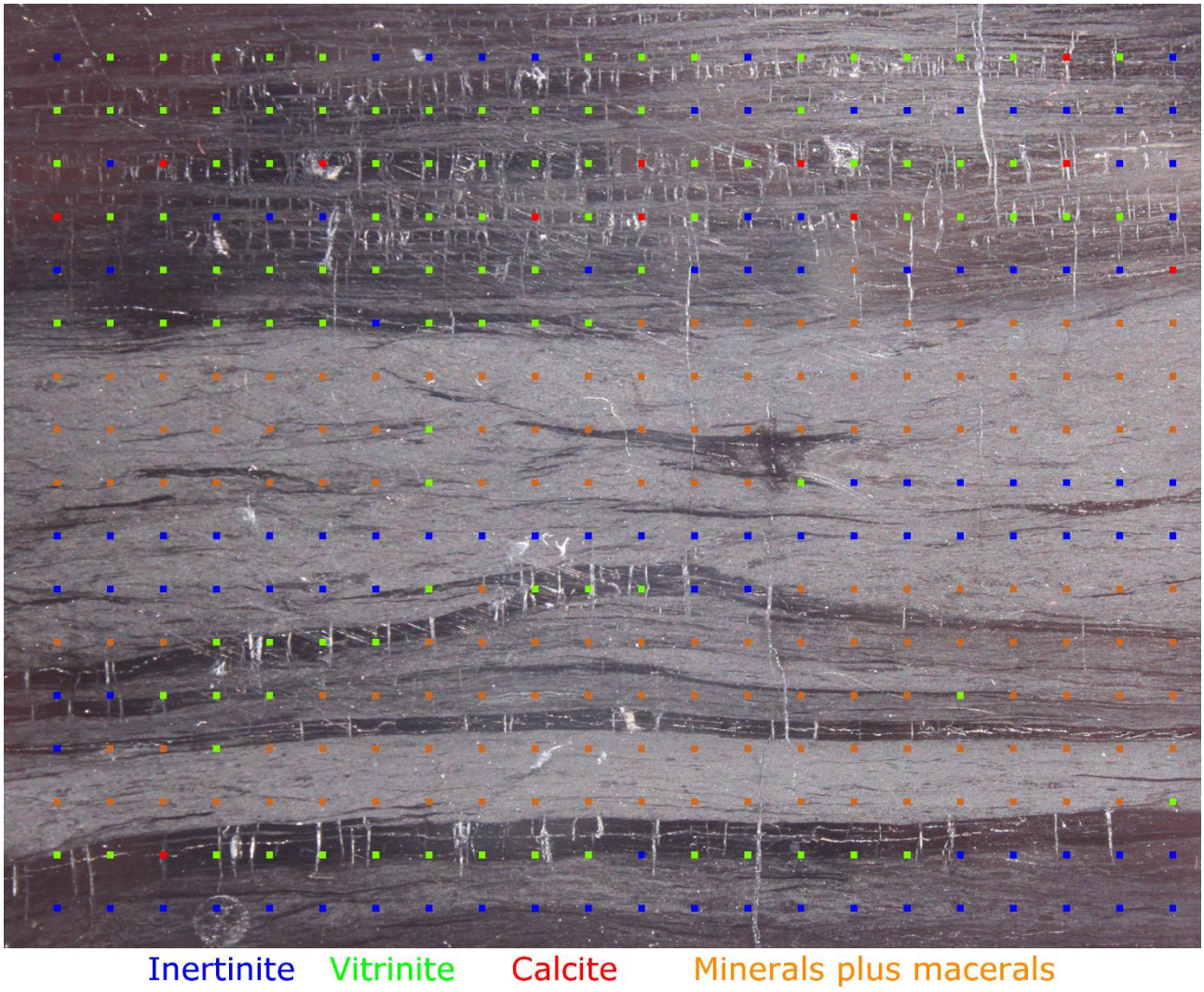


Figure 53: Sample 5 hand classification

4.2. Testing the feature extraction

The purpose of this section is to explain the methods that are used to rate the feature extraction components. The goal of feature extraction is to produce more descriptive feature values that can be used to differentiate between the regions of interest (the mineral components) in the image. Therefore, a good measurement that can be used to rate the alternative feature values is the extent to which they differentiate the mineral matter in the samples. The proposed method of doing so is to find model values for the collections of pixels that have been classified into each mineral class and compare them to determine the level of dissimilarity between models in terms of the feature value being rated. This test can be summarised as follows:

- For each feature value to be rated:
  1. For each sample image:
     a. For each mineral class in the sample:
        i. Gather the feature values assigned to the class
        ii. Find the mean value to be used as ground truth model value
     b. Compare these class models in a pair-wise manner and obtain the following:
        i. The minimum model value difference
        ii. The maximum model value difference
  2. Calculate the sum of these two differences (separately) across all image samples
- Rate the feature values according to the sum of differences calculated in step 2

The final ratings can then be compared to identify which of the feature values differentiate the model values the most (over all the sample images). A high minimum difference in model values indicates that the feature value differentiates well between all mineral models in the sample. A high maximum difference indicates that the feature value differentiates well between at least two mineral models in the sample.

This method of comparison has limited statistical value, due to the use of a single value (the mean) to represent the model distributions. In order to increase the validity, a two sample t-test will be done to ensure that the difference in mean values is due to the underlying data distributions (the actual model values) and not just due to the sample of values identified by the geologists.

The classical t-test assumes that the distributions, being compared, are normally distributed and have equal variance. As it will be shown in section 5.1.1, this is rarely the case. For this reason an alternative t-test will be used, namely Welch's t-test. This test does not assume equal variance but still assumes that the underlying population, from which the data is sampled, is normally distributed. This is a reasonable assumption, considering the population sizes that are present in the sample images.

The alternative feature space will then be visualised, and compared to the baseline, using parallel coordinates. This is a method of visualising multi-dimensional data. It plots the data points as lines crossing vertical axes, representing the feature dimensions, at the appropriate value within that dimension.

4.3. Testing the clustering algorithms

The purpose of this section is to explain the methods that are used to rate the clustering components. In order to rate the performance of the clustering algorithms of the system several clustering evaluation metrics can be used. These are general measurements of how well a clustering algorithm groups the data points together. The ideal clustering configuration has clusters that are compact and well separated. This test is designed to test the density and/or the separation of clusters. The metrics that were used are:

- The Davies-Bouldin index [83]

  This is a measure of the similarity of clusters and is defined by the following equation:

$$DBI = \frac{1}{N}\sum_{i=1}^{N} \underset{i \neq j}{max}\left(\frac{S_i + S_j}{M_{ij}}\right) \quad (33)$$

  where N is the number of clusters, M_{ij} is the distance between the i^{th} and j^{th} cluster's centroids, and S_i and S_j is the average distance between all data points and their corresponding centroid in the i^{th} and j^{th} cluster, respectively. A low DBI value indicates that the cluster configuration contains clusters that are densely packed and, consequently, that the data points within these clusters are very similar to their within-cluster neighbours.

- The Dunn Index [84]

  This index is a measure of how well the clusters in a cluster configuration are separated. It is defined with the following equation:

$$DI = \frac{\min distance(C_i, C_j)}{\max diameter(C_k)}, 1 \leq i, j, k \leq N, i \neq j \quad (34)$$

  where N is the number of clusters, C_i, C_j, and C_k are the i^{th}, j^{th}, and k^{th} clusters, respectively, and the distance and diameter of a cluster(s), as defined in the work that originated the index [84], are defined as:

$$distance(A, B) = \underset{\substack{x \in A \\ y \in B}}{min} d(x, y)$$

$$diameter(A) = \underset{x, y \in A}{max} d(x, y)$$

  and d is an appropriate distance metric.

Calculating the distance and diameter, as defined above, is computationally very complex. This is because of the fact that the distance between every possible pair of data points (selected from one or both of the clusters) has to be calculated. This results in a $O(n^2)$ complexity for the diameter and a $O(n_1 * n_2)$ complexity for the distance. As a compromise the following equations can be used:

$$distance(A, B) = d(C_A, C_B)$$

$$diameter(A) = \frac{2}{n}\sum_{x=1}^{n} d(x, C_A)$$

where n is the number of data points in cluster A, and C_A and C_B are the centroids of clusters A and B, respectively. These equations still provide an indication of the inter-class and intra-class distances but disregards the shapes of the cluster. Unlike, the Davies-Bouldin index, a high Dunn Index is more desirable and indicates a cluster configuration that has low intra-class distance and high inter-class distance.

- Silhouettes [85]

  In contrast to the previous two evaluation metrics, silhouettes are calculated for each data point individually. It provides an estimate of how well a data point fits into its assigned cluster. Each data point receives a value between -1 (indicating bad clustering) and 1 (indicating good clustering). The equation to calculate the silhouette of a clustered data point, i, is as follows:

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \quad (35)$$

  where

$$a(i) = \frac{1}{n}\sum_{x=1}^{n} d(x, i)\,, i \neq x$$

  where n is the total number of data points in the cluster of which i and x are members. And,

$$b(i) = \min\{d(i, C_j)\}, 1 \leq j \leq N\,, i \notin j$$

  where N is the total number clusters in the entire configuration and C_j denotes the centroid of the j^{th} cluster. Therefore, $a(i)$ represents how well the data point fits into its own cluster, and $b(i)$ represents how similar it is to the second closest cluster to it. The closest cluster would be the one it is assigned to.

Again the intra-class measure ($a(i)$) requires a high time complexity calculation. This calculation will, then have to be done, for every data point to generate a corresponding silhouette value. In order to prevent this, the distance between the data point and its assigned centroids will be used instead of the average distance between it and all the data points in its cluster. The average $s(i)$ over the entire data set can be used as an indication of how appropriately the data set has been clustered.

- Mean intra-cluster distance

  This is not a formal index, such as the previous three metrics, but a measure of clustering aptitude. It is a measure of how well the clusters in the configuration are packed together. The necessary equation is as follows:

$$D_{intra} = \frac{1}{N}\sum_{Y=0}^{N}\left(\frac{1}{n}\sum_{x \in Y} d(x, C_Y)\right) \quad (36)$$

  where N is the number of clusters in the configuration, n is the number of data points assigned to the Y^{th} cluster, C_y is the centroid of the Y^{th} cluster and d is an appropriate distance metric. A smaller mean intra-cluster distance is desirable, because this indicates that the data points in the average cluster in the configuration are very similar.

- Mean inter-cluster distance

  Where the previous metric indicates the average compactness of clusters in the configuration, this one indicates the average separation between clusters in the configuration. The following equation is used:

$$D_{inter} = \frac{2}{N(N-1)}\sum_{i=0}^{N}\sum_{j=0}^{N} d(C_i, C_j), i \neq j \quad (37)$$

  where N is the number of clusters in the configuration, C_i and C_j are the centroids of the i^{th} and j^{th} clusters, respectively, and d is an appropriate distance metric. A larger mean inter-cluster distance is desirable, because this indicates that the clusters in the configuration are located far apart.

Take note that these five evaluations are internal, by nature, and can, therefore, not be used to rate the information retrieval of the clustering methods. They are unsupervised and only rate the appropriateness of the clustering according to generally accepted measures of proficiency. They will be used to investigate the performance and underlying tendencies of the clustering algorithms.

4.4. Testing the model parameters

The purpose of this section is to explain the methods that are used to rate the approaches to modelling the mineral classes. In chapter 3 (table 10), it was stated that the existing system makes use of the median feature values in the model selection regions. It was also explained that the mean value might be a more fitting measurement. Therefore, the use of the median or mean values for model parameters are rated, separately. The method of doing so is as follows:

1. For each sample image:
    a. Generate the appropriate feature space (as identified by the feature extraction tests)
    b. Highlight the model areas for the minerals as defined by the corresponding image in figures 42 to 47
    c. For each highlighted area calculate the median and mean feature value
    d. Compare these mean and median values with those of the ground truth model values
2. Rate the mean and median values by determining which value correlates the highlighted and ground truth model values better over all the samples.

This procedure will determine which parameter metric (median or mean) relates the user defined model values and the actual mineral content in the images better.

4.5. Testing the final assignments

The purpose of this section is to explain the testing procedure that is used to rate the methods by which the system assigns the clustered data points to the appropriate model values. The clusters obtained from each of the clustering algorithms will be assigned to the model values, according to the three methods presented in table 10. These methods are:

- Assign the clusters to multiple model regions instead of one per model
- Assign clusters to a cluster that acts as a model-by-proxy instead of the nearest true model
- Assign the segmentation regions of the clusters to the model values

Notice that the alternatives are not mutually exclusive and some of them can be combined to produce different results. As such, the dendogram, provided in figure 54, presents a list of possible methods of assigning the clusters to model values as combinations of the above listed alternatives. For each of these alternatives (and for each sample image), the resulting image segmentation will be compared to the ground truths to produce a confusion matrix. This is a visual aid used, in machine learning, to interpret the accuracy with which a system classifies data into several classes.

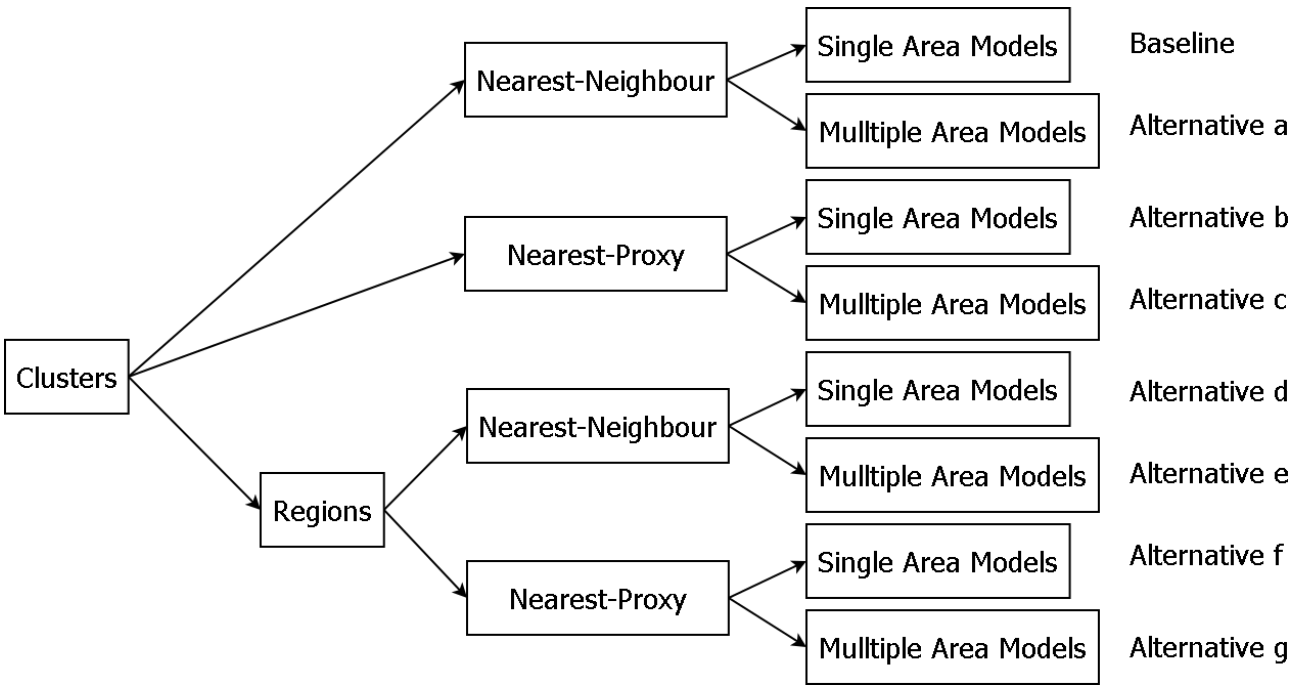


Figure 54: Alternative cluster-to-model assignments

4.6. Testing the system

The purpose of this section is to explain the manner by which the entire system will be tested against the existing system. In order to test the system as a whole, the baseline system will be compared to a system that incorporates the top rated alternative components of all the preceding tests. The exact baseline system is summarised in table 11.

Table 11: Summarised baseline system

Component	Technique
Luminosity	L*
Chromaticity	b*
Roughness	• Sobel operator • Median filter
Regularity	Singular value decomposition
Clustering	k-means
Model parameters	Median feature value
Final Assignments	• Cluster to nearest model value • Single region selection per model

For each of the six samples the baseline system will be used to segment the image, a corresponding confusion matrix will then be constructed, with regards to the ground truths. This will be repeated several times to take into account, the stochastic nature of the k-means clustering algorithm. The same process will then be repeated for the new system. These confusion matrices will be used to compare the two systems and note any improvements.

5. RESULTS

5.1. Feature extraction results

The purpose of this section is to present the results of the feature extraction test defined in section 4.2. The following are some of the key points of discussion:

- The distributions of model values in terms of the feature values
- The extent to which each feature value differentiates the model values
- The ratings obtained by the feature values
- The results of the t-tests
- A visual investigation on the alternative feature space that has been identified

5.1.1. Feature distributions

A histogram is generated for each sample image in terms of every considered feature value. These represent the distributions of the model values that were sampled by the geologists (see section 4.1.2). With fifteen feature values (see table 5) and six sample images, this culminates to ninety distributions. All of these are provided in appendix A. An informative few are presented and discussed here. For each distribution the feature values are normalised to a common range, namely from 0 to 100. This normalisation is done according to the minimum and maximum value in the feature dimension being normalised.

- Chromaticity

  Figures 55 and 56 show the b* and v* distributions of sample 0. From these figures it can be seen that, relative to b*, the v* feature spreads out the distribution of vitrinite slightly and localises the mineral and crack distributions. This could prove beneficial, provided that the means of these distributions are significantly farther apart than they are in the b* feature. It is difficult to discern the mean feature values in the models, from these histograms, and this will be determined in section 5.1.2.

- Regularity

  Figures 57 and 58 show the distributions of the regularity estimates through singular value decomposition (svd) and local entropy in sample 1. It is immediately apparent that the entropy distributions are spread out across a much wider range of values than the baseline (svd). This could be a poor indication if the means are also too close together. However, this seems not to be the case in figure 58. The entropy means appear to be separated much better than the svd means.

- Roughness

  All ten of the roughness feature distributions appear very similar, however the features using the mean or the Gaussian filters appear to spread out the model values slightly better than the ones using the median filter. This can be seen by comparing figures 59 and 60. These figures contain two roughness distributions for sample 3, namely, the Sobel operator combined with the median filter and the Sobel operator combined with the Gaussian filter.

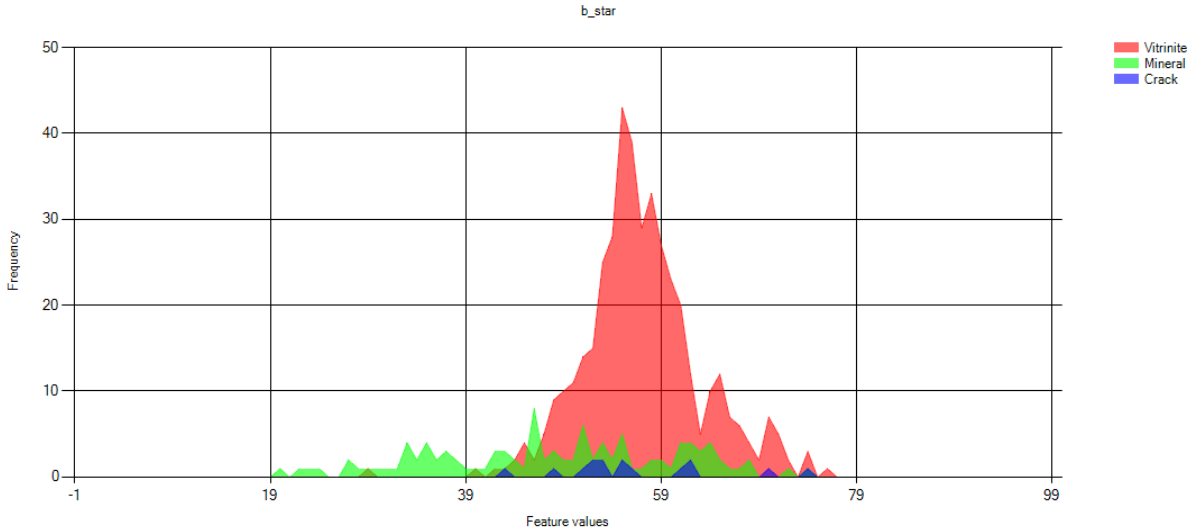


Figure 55: Sample 0 b feature distribution*

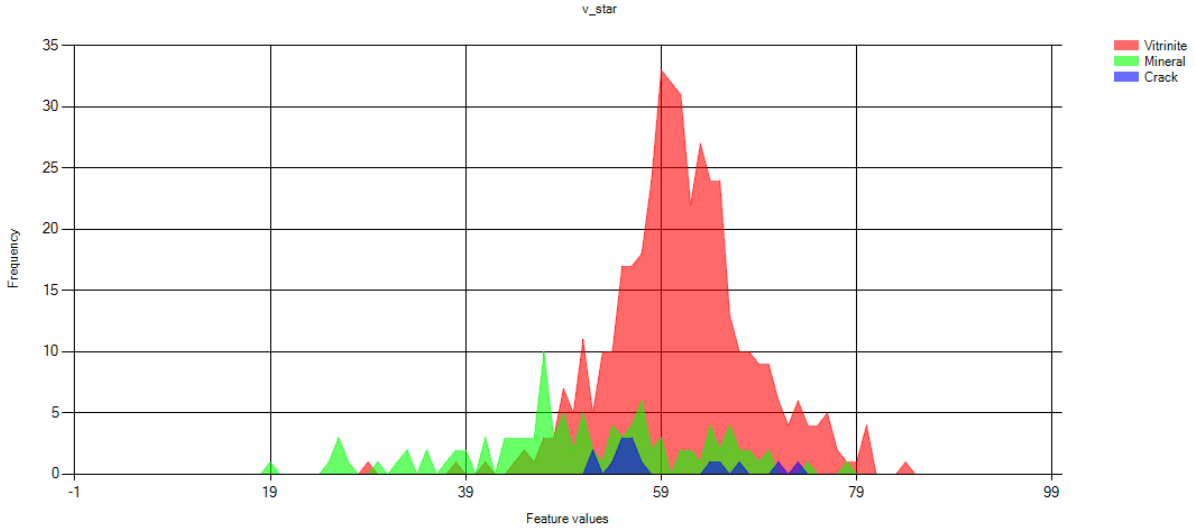


Figure 56: Sample 0 v feature distribution*

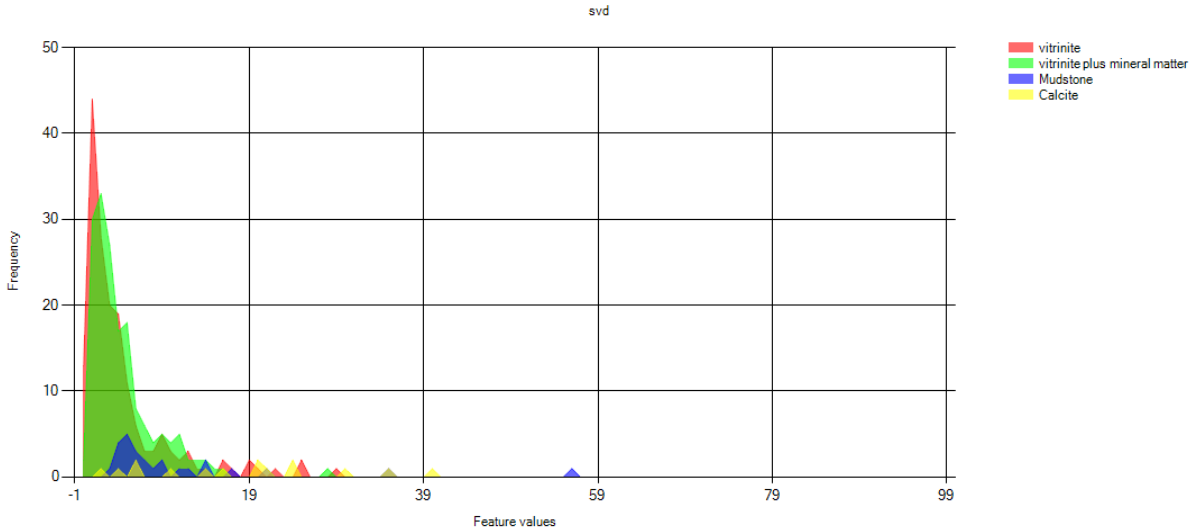


Figure 57: Sample 1 svd feature distribution

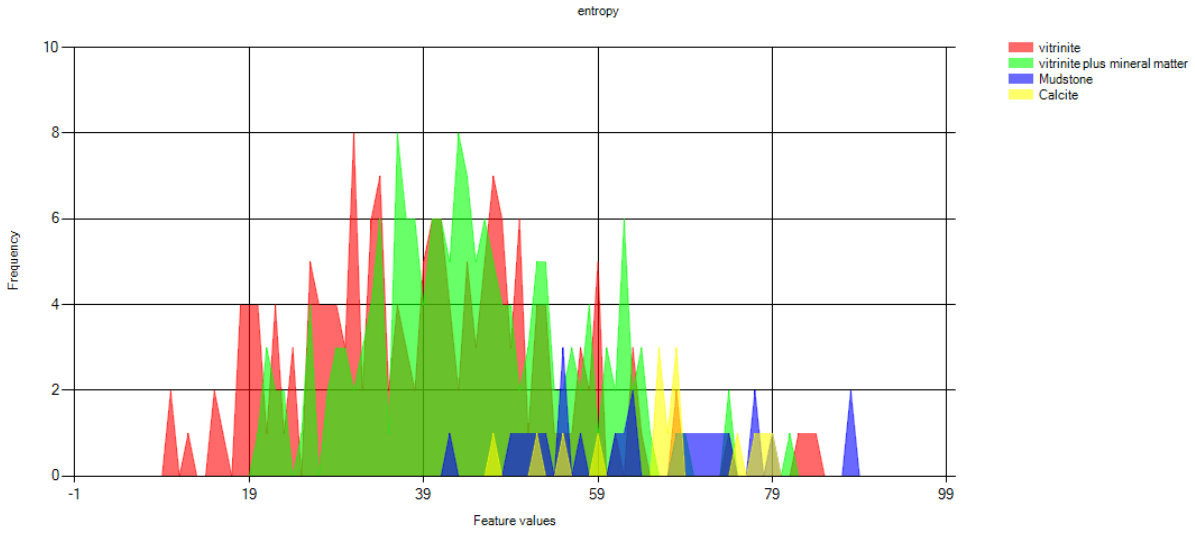


Figure 58: Sample 1 entropy feature distribution

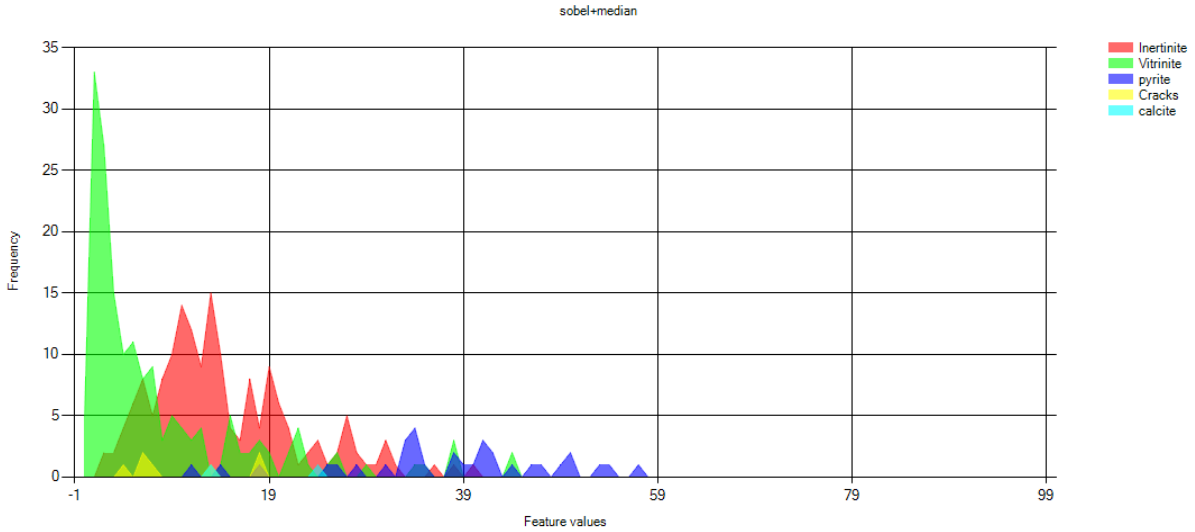


Figure 59: Sample 3 Sobel + median feature distribution

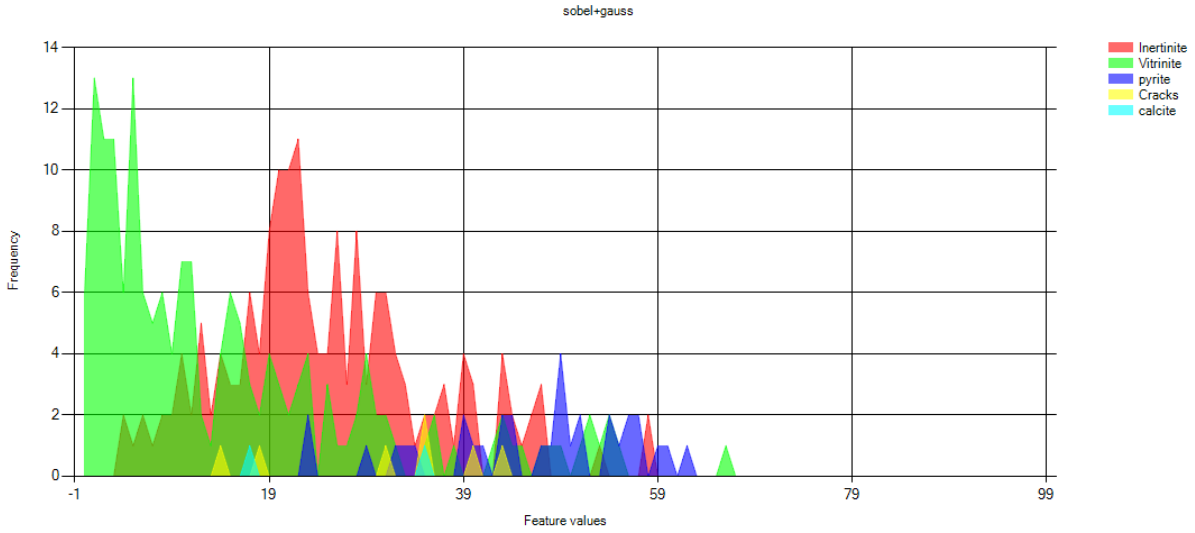


Figure 60: Sample 3 Sobel + Gauss feature distribution

5.1.2. Differentiating models

Model values were generated by calculating the mean feature values of the distributions presented in the previous section. A pair-wise comparison was then made between each model value to determine the absolute differences between model values. Finally, the maximum- and minimum differences were recorded. The results of this exercise are presented in tables 12 to 14. The columns separate the feature values, the rows separate the model values and/or the resulting differences. The cells, shaded in yellow, indicate the difference values that will be used to rate the feature values. To clarify the values in these tables the following example is presented with regards to the distribution in figure 55. This figure represents the distribution of the b* feature in sample 0.

- The mean feature value of all the data points belonging to the “Vitrinite” model is 57.56531939.
- The mean feature value of all the data points belonging to the “Mineral” model is 48.54611724.
- The mean feature value of all the data points belonging to the “Crack” model is 57.07097263.
- The maximum difference between these three values is:
    - 57.56531939 – 48.54611724 = 9.019202145
- The minimum difference between these three values is:
    - 57.56531939 - 57.07097263 = 0.494346759
- Therefore, the values in the b* column in table 12, under sample 0 are:
    - 57.56531939
    - 48.54611724
    - 57.07097263
    - 9.019202145
    - 0.494346759

From these results it can be seen that the “Mineral” class has a much lower value than the other two models, in terms of the b* colour channel. This is consistent with the visual representation of sample 0 (see figure 42), because the “Mineral” class appears to have a blue pigment that is not present in the other classes, and the b* contrasts yellow pigments with blue pigments. This procedure was repeated for all the features, and all the sample images to produce the measures of differentiating models presented in tables 12 to 14.

Table 12: Model feature values A

Model	L*	b*	v*	svd	entropy
Sample 0					
Vitrinite	66.91694145	57.56531939	61.56697339	22.74426597	74.43492734
Mineral	33.70441854	48.54611724	51.35685617	35.64417832	87.40869838
Crack	18.51184812	57.07097263	59.61932776	31.45004556	76.7855405
Max differences	48.40509332	9.019202145	10.21011722	12.89991235	12.97377104
Min differences	15.19257042	0.494346759	1.947645632	4.194132761	2.350613162
Sample 1					
Vitrinite	12.41639329	57.20160883	57.98176622	4.807269124	39.97458823
Vitrinite plus mineral matter	14.86462416	56.83991196	57.47740181	4.886819224	44.70283442
Mudstone	28.82691455	58.3207719	58.37306099	11.33753005	64.91814693
Calcite	27.72611779	60.0810098	59.73590956	18.5886577	66.52345111
Max differences	16.41052126	3.241097849	2.258507748	13.78138858	26.54886289
Min differences	1.100796763	0.361696871	0.391294771	0.0795501	1.605304181
Sample 2					
Vitrinite plus mineral matter	14.10853072	62.0836437	65.56641441	5.979683687	49.67985899
Vitrinite	11.53762906	62.05459184	65.5998348	7.596870656	49.65710187
Pyrite	38.41705459	67.90154851	69.88916023	20.71221243	80.56285814
Mudstone	30.02452811	64.34955874	67.00248662	8.112553115	70.89531461
Cracks in vitrinite	24.09435428	63.14373327	66.04472678	22.77054389	73.09267823
Calcite	18.75649776	64.12091108	66.84411657	19.60961744	68.89505757
Max differences	26.87942553	5.846956672	4.322745821	16.7908602	30.90575628
Min differences	2.570901658	0.029051866	0.03342039	0.515682459	0.022757121
Sample 3					
Inertinite	23.8693807	65.25254917	69.13558387	24.34355953	64.17851187
Vitrinite	17.39501007	64.80584253	68.88700743	13.94995392	46.54469761
pyrite	55.12112998	68.35925343	71.94873678	42.4553683	85.76522069
Cracks	41.57107103	63.53114933	67.82684269	34.9659239	63.19389834
calcite	28.49678121	66.94112633	70.543691	16.79119348	68.84009386
Max differences	37.72611991	4.828104093	4.121894093	28.50541438	39.22052308
Min differences	4.627400506	0.446706637	0.248576447	2.841239564	0.984613534
Sample 4					
Macerals plus minerals	24.99925098	48.73365427	54.71885172	6.727531205	52.60281446
Inertinite	14.03952979	49.06122539	53.75128876	5.097197235	47.60089867
Max differences	10.95972119	0.327571119	0.967562964	1.63033397	5.001915788
Min differences	10.95972119	0.327571119	0.967562964	1.63033397	5.001915788
Sample 5					
Inertinite	45.93393594	58.7854742	63.99070352	15.8768961	59.63015547
Vitrinite	33.22040205	56.74132907	62.46466442	16.24824372	60.62633245
Minerals plus macerals	55.97003109	59.11665139	64.39086003	15.24026741	59.45182044
Calcite	55.60530928	60.41396948	65.32459771	30.09006182	76.05021901
Max differences	22.74962903	3.672640401	2.859933296	14.84979441	16.59839856
Min differences	0.364721805	0.331177192	0.400156505	0.371347617	0.17833503

Table 13: Model feature values B

Model	sobel+median	sobel+mean	sobel+gauss	roberts+median	roberts+mean
Sample 0					
Vitrinite	22.61497359	28.47566971	28.41006001	23.93322382	29.81214994
Mineral	38.61548981	45.38755398	45.49290046	38.19851617	46.07984292
Crack	28.2174018	40.0705985	40.03819852	30.96963661	41.53356012
Max differences	16.00051622	16.91188426	17.08284045	14.26529235	16.26769298
Min differences	5.602428205	5.316955481	5.454701937	7.036412787	4.5462828
Sample 1					
Vitrinite	5.173303837	7.528272241	7.527383976	5.162459113	7.389569408
Vitrinite plus mineral matter	5.963003326	7.399671715	7.386325066	5.925580227	7.171074165
Mudstone	15.51165649	17.69522208	17.67081339	15.29983242	17.03542785
Calcite	23.16188238	27.71694001	28.07086799	20.63873493	27.37507694
Max differences	17.98857855	20.3172683	20.68454292	15.47627582	20.20400277
Min differences	0.789699489	0.128600526	0.14105891	0.763121114	0.218495243
Sample 2					
Vitrinite plus mineral matter	5.394955013	7.643403417	7.625765652	5.585617401	7.920749046
Vitrinite	5.573456925	9.617513419	9.586616435	5.602713972	9.934259743
Pyrite	21.55277623	29.0160284	29.08331672	22.86885722	29.55225623
Mudstone	8.971103427	12.4223267	12.37836723	9.074629066	12.58082635
Cracks in vitrinite	21.50508981	29.64101363	29.5507311	22.85710954	31.65955826
Calcite	16.48242911	20.97568577	21.07893785	16.31801873	22.1506933
Max differences	16.15782121	21.99761021	21.92496545	17.28323982	23.73880921
Min differences	0.047686412	0.624985228	0.467414381	0.011747674	2.013510698
Sample 3					
Inertinite	14.80337484	25.30431128	25.35940813	14.84907576	25.42172739
Vitrinite	8.168921498	15.46995677	15.47677943	7.948065329	15.1271486
pyrite	37.87823764	46.31042862	46.54141146	37.72679856	45.58883711
Cracks	10.75158377	30.45941606	31.1974945	10.84727171	30.62255768
calcite	19.06944084	26.38227684	26.47555804	18.83699239	25.75932689
Max differences	29.70931614	30.84047186	31.06463204	29.77873323	30.46168852
Min differences	2.582662272	1.077965567	1.116149916	2.89920638	0.337599503
Sample 4					
Macerals plus minerals	10.83741917	12.23259019	12.14754731	12.46273672	13.25731699
Inertinite	8.940521096	10.49058327	10.381743	9.541876064	10.75344448
Max differences	1.89689807	1.74200692	1.765804313	2.92086066	2.503872514
Min differences	1.89689807	1.74200692	1.765804313	2.92086066	2.503872514
Sample 5					
Inertinite	16.05451918	19.41675372	19.20118971	18.15856071	19.21142161
Vitrinite	16.15326716	21.15839719	20.91757963	17.34220635	19.85334536
Minerals plus macerals	16.33741252	19.50904835	19.28183307	19.21397127	19.83454543
Calcite	31.95974629	37.70241773	37.47959908	33.71909237	35.25788209
Max differences	15.90522711	18.28566401	18.27840937	16.37688601	16.04646048
Min differences	0.098747979	0.092294634	0.080643366	0.81635436	0.018799933

Table 14: Model feature values C

Model	roberts+gauss	prewitt+median	prewitt+mean	prewitt+gauss	sd
Sample 0					
Vitrinite	29.85772102	21.98616133	27.99848966	27.9662148	25.61775596
Mineral	46.34629974	37.73511117	44.78496071	44.94518989	44.88757121
Crack	41.64822904	27.80471976	39.41022541	39.4280815	43.3812065
Max differences	16.48857872	15.74894984	16.78647105	16.97897509	19.26981525
Min differences	4.6980707	5.818558423	5.374735306	5.517108394	1.506364709
Sample 1					
Vitrinite	7.408913831	5.191551967	7.533623508	7.532345317	9.076835897
Vitrinite plus mineral matter	7.179723388	5.978994079	7.409581727	7.395697542	8.641774289
Mudstone	17.02540122	15.65655382	17.72870575	17.70223818	18.53705974
Calcite	27.7498559	23.14043423	27.80864543	28.15410051	27.68103772
Max differences	20.57013251	17.94888226	20.3990637	20.75840297	19.03926343
Min differences	0.229190443	0.787442112	0.124041781	0.136647776	0.435061608
Sample 2					
Vitrinite plus mineral matter	7.904455385	5.430497529	7.533547024	7.5168126	7.775821163
Vitrinite	9.908751002	5.620935059	9.489557743	9.460041804	11.41624671
Pyrite	29.61917192	21.83768793	28.64843884	28.71854465	28.34391167
Mudstone	12.51245386	9.244730379	12.31884593	12.2797932	16.42205814
Cracks in vitrinite	31.53202301	20.77674722	29.12649375	29.03802253	26.99698489
Calcite	22.16188499	16.46374892	20.62050587	20.71908733	19.23131332
Max differences	23.62756763	16.4071904	21.59294673	21.52120993	20.56809051
Min differences	1.912851089	0.19043753	0.478054907	0.319477876	1.346926783
Sample 3					
Inertinite	25.47619852	14.8511853	24.92849001	24.98111376	27.51438167
Vitrinite	15.13128664	8.25746686	15.31664041	15.32297591	19.95641103
pyrite	45.79021392	37.70038153	45.56480178	45.78788053	47.97892324
Cracks	31.27725268	10.97280192	30.15062152	30.87630847	37.28429643
calcite	25.86868357	19.33328852	26.08731236	26.18687638	27.88538558
Max differences	30.65892728	29.44291467	30.24816137	30.46490462	28.02251221
Min differences	0.392485049	2.715335058	1.158822354	1.205762618	0.371003915
Sample 4					
Macerals plus minerals	13.17923048	10.72723994	12.24013188	12.16096954	11.19162797
Inertinite	10.64140473	9.039576407	10.56207229	10.45908146	11.26243007
Max differences	2.537825749	1.687663528	1.678059591	1.701888081	0.070802102
Min differences	2.537825749	1.687663528	1.678059591	1.701888081	0.070802102
Sample 5					
Inertinite	18.98880726	16.28370421	19.4877761	19.27144862	14.69979393
Vitrinite	19.61419156	16.55579662	21.3446293	21.1026538	20.14590787
Minerals plus macerals	19.59979838	16.52757897	19.51651522	19.2884291	13.92373251
Calcite	35.04518925	32.8757333	38.06589216	37.84168573	29.41504397
Max differences	16.05638199	16.59202909	18.57811606	18.57023711	15.49131146
Min differences	0.014393186	0.028217653	0.028739128	0.016980481	0.776061414

5.1.3. Ratings

Tables 15 to 17 provides the total sum of the differences, presented in tables 12 to 14, across all samples. It is clear from these tables that the b* feature differentiates between the various model values better than the v* feature. However, this is only by a small margin, and considering the fact that sample 0 is the only sample with substantial colour information, the v* feature should be further investigated. If the differences in sample 0 is used to rate the chromaticity feature values v* is rated higher by a slight margin. A t-test will be done in the next section to investigate these two features further.

The regularity based on entropy seems to be a better choice than the original regularity based on singular value decomposition (svd). It consistently has a larger maximum difference between model values, however, it narrowly rates better than svd in terms of the minimum difference. This means that, relative to the svd feature, it differentiates between specific minerals exceptionally well and differentiates between minerals in general reasonably well.

Table 15: Total sum of differences between models A

Totals	L*	b*	v*	svd	entropy
Total max differences	163.1305102	26.93557228	24.74076115	88.45770389	131.2492276
Total min differences	34.81611234	1.990550443	3.988656709	9.632286471	10.14353882

Table 16: Total sum of differences between models B

Totals	sobel+median	sobel+mean	sobel+gauss	roberts+median	roberts+mean
Total max differences	97.6583573	110.0949056	110.8011945	96.10128789	109.2225265
Total min differences	11.01812243	8.982808356	9.025772822	14.44770298	9.63856069

Table 17: Total sum of differences between models C

Totals	roberts+gauss	prewitt+median	prewitt+mean	prewitt+gauss	sd
Total max differences	109.9394139	97.8276298	109.2828185	109.9956178	102.461795
Total min differences	9.784816216	11.2276543	8.842453067	8.897865225	4.506220531

In terms of roughness the Sobel operator combined with the Gaussian filter performs better than the other nine alternatives. It has the best performance with regards to the maximum model value difference, and the sixth best in terms of the minimum model value difference. The performance of all the roughness feature values are very similar but that of the above mentioned version is the most consistent. Take note that the original Sobel operator and median filter combination is rated as the eighth best overall. The use of the Gaussian filter as opposed to the median filter (as well as the mean filter) reduces the effect of rectangular artefacts that occur when implementing the square sampling window. This could account for the improved roughness estimate that is made with the new filter.

5.1.4. t-tests

Several t-tests were done to determine if the differences in the baseline and alternative distributions are due to the actual underlying data distribution or due to outliers and/or the sampling done to obtain the distributions. All of these t-test are provided in appendix B, but a summary of the results is presented here. In all the tests, an alpha value of 0.05 is used to determine if the null hypothesis (the mean of the statistical populations are equal) can be rejected. The alpha value is chosen so as to achieve a 95% confidence interval.

- Chromaticity (b*/v*)

  The t-tests were done on the vitrinite, mineral, and crack model distributions in sample 0 to determine if the difference in model values is valid for the sampled distributions. The results are as follows:

  - Vitrinite attained a p-value of $4.98183\ x\ 10^{-18}$. This is much smaller than the alpha value and the null hypothesis is, therefore, rejected. The difference in the distribution means is due to the actual population data and not the sampling.
  - The mineral and crack models attained p-values of 0.091313148 and 0.36737564, respectively. These values are both greater than the alpha value and the null hypothesis stands. While the crack p-value can be attributed to the relatively low amount of sampled data points (there are only 15) in the distribution, the mineral sample size is 106.

  Taking these results into account, and considering the small difference in the mean values between the compared distributions, it is decided to keep the b* feature as the chromaticity component in the system. It performed better than the v* feature overall and where it is outperformed by v* (Sample 0) the difference is questionable, as indicated by the t-tests.

- Regularity (svd/entropy)

  The corresponding distributions, for all sample images, from the svd feature and the entropy feature were used to perform the t-tests. The resulting p-values are shown in table 18. The p-values that resulted in the null hypothesis being rejected are shaded in green. It clear from these tests that the difference in entropy and svd model values is not due to the sampling method, and the improvements resulting from using the local entropy to estimate regularity are statistically valid. As a results, it is decided to use the entropy feature instead of the svd feature.

Table 18: SVD/Entropy t-test p-values

Sample 0					
Vitrinite		Mineral		Crack	
0		0		0	
Sample 1					
Vitrinite	Vitrinite plus mineral matter			Mudstone	Calcite
0	0			0	0
Sample 2					
Vitrinite plus mineral matter	Vitrinite	Pyrite	Mudstone	Cracks in vitrinite	Calcite
0	0	0	0	0.001	0.006
Sample 3					
Inertinite	Vitrinite	Pyrite	Cracks	Calcite	
0	0	0	0.016	0.026	
Sample 4					
Macerals and mineral matter			Inertinite		
0			0		
Sample 4					
Inertinite	Vitrinite	Macerals and minerals			Calcite
0	0	0			0

- Roughness (Sobel operator and median filter/Sobel operator and Gaussian filter)

The same testing process was used to test the validity of using the Sobel operator and Gaussian filter instead of the original Sobel operator and Median filter. Table 19 presents the results of this test. The green cells represent the tests that resulted in the null hypothesis being rejected and the red cells represent the case where it remained. Note that not all of the t-tests, in this case, resulted in the null hypothesis being rejected. However, as with the chromaticity t-test, these tests correspond to the models with low amounts of sampled data points. Also take note that twice as many of the t-tests did, in fact, result in the null hypothesis being rejected. For these reasons it is decided to use the alternative roughness estimate instead of the original.

Table 19: Sobel + Median/Sobel + Gauss t-test p-values

Sample 0					
Vitrinite		Mineral		Crack	
0		0		0.059	
Sample 1					
Vitrinite	Vitrinite plus mineral matter			Mudstone	Calcite
0.004	0.005			0.617	0.365
Sample 2					
Vitrinite plus mineral matter	Vitrinite	Pyrite	Mudstone	Cracks in vitrinite	Calcite
0	0	0.026	0.191	0.516	0.665
Sample 3					
Inertinite	Vitrinite	Pyrite	Cracks	Calcite	
0	0	0.002	0.002	0.556	
Sample 4					
Macerals and mineral matter			Inertinite		
0			0.357		
Sample 4					
Inertinite	Vitrinite	Macerals and minerals			Calcite
0.001	0.002	0			0.355

5.1.5. Alternative feature space

To summarise the findings of the feature extraction test, table 20 presents the baseline- and the top ranked feature space found in this section.

Table 20: Summarised results of feature extraction test

Feature values	Baseline feature space	Alternative feature space
Luminosity	L*	L*
Chromaticity	b*	b*
Roughness	Sobel operator + median filter	Sobel operator + Gaussian filter
Regularity	Singular value decomposition	Entropy

As discussed in section 4.2, the parallel coordinates in figures 61 to 72 visually depict the feature spaces as a whole. There are twelve parallel coordinates in total; six for the baseline feature space and six for the alternative feature space. In other words, two for each sample image.

Each of the coloured lines correspond to a data point in the ground truth data set. The colour represents the model that it is classified to. The colour for each model is indicated by the colour coded model names to bottom-left of the figures. For the baseline feature space, the feature axes, from left to right, are L*, b*, svd (singular value decomposition), and “sobel+median”. For the alternative feature space the axes are L*, b*, entropy, and “sobel+gauss”. The point at which the lines cross each axis is determined by the value of the corresponding data point in terms of the feature represented by that axis. The lower part of the axis represents lower values, and the higher part represents higher values. This graphical aid is a good way to visualise the covariance in feature values in a feature space.

Notice that the two alternative feature dimensions tend to spread the data points out. This is a good indication because a feature dimension that assigns all data points to similar values is of no use in differentiating between model values. This can be seen with the b* feature in the sample images containing little colour information.

In addition to spreading out the data points, the alternative feature space also improves the difference in model values. Clear examples of this can be observed in the parallel coordinates of sample 0 and sample 3. The seemingly random model distributions and lack of covariance between the roughness and regularity features in the baseline feature space is replaced by well separated distributions with a high level of covariance.

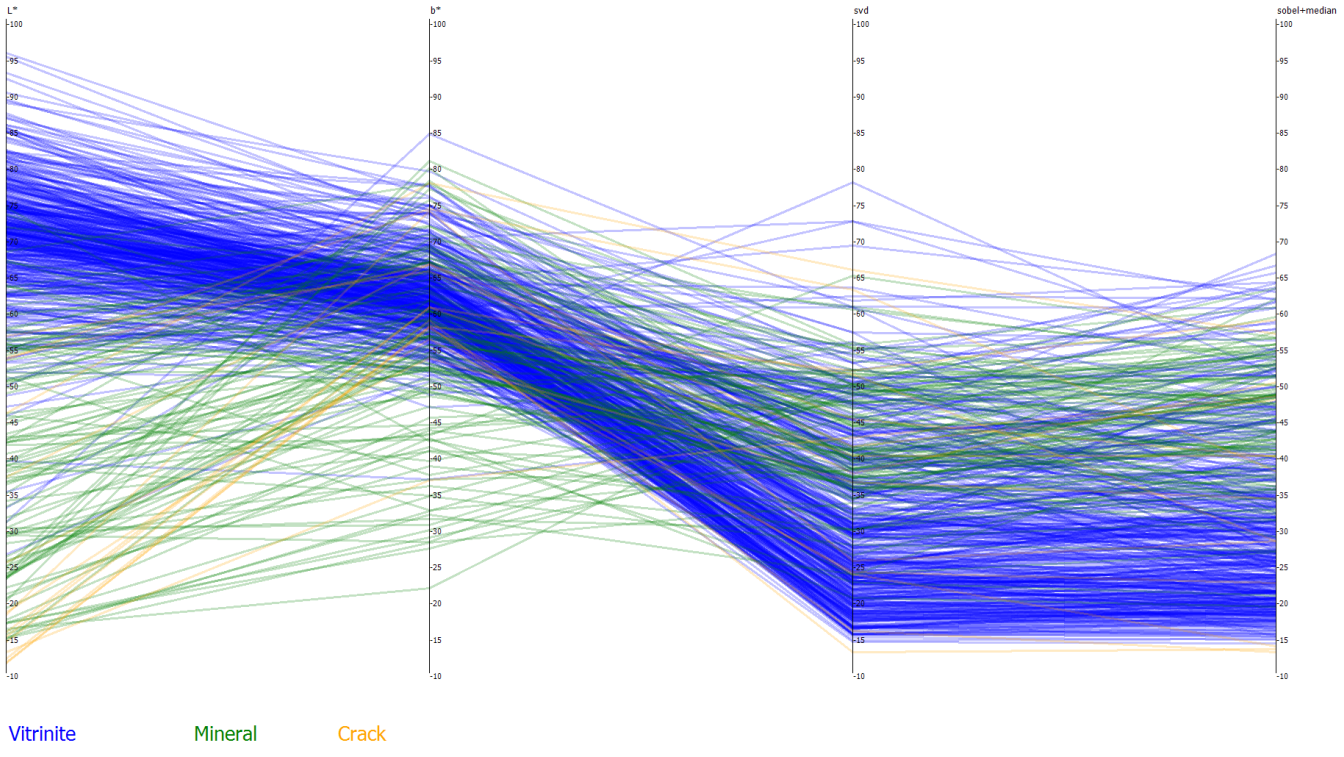


Figure 61: Parallel coordinates of baseline feature space (Sample 0)

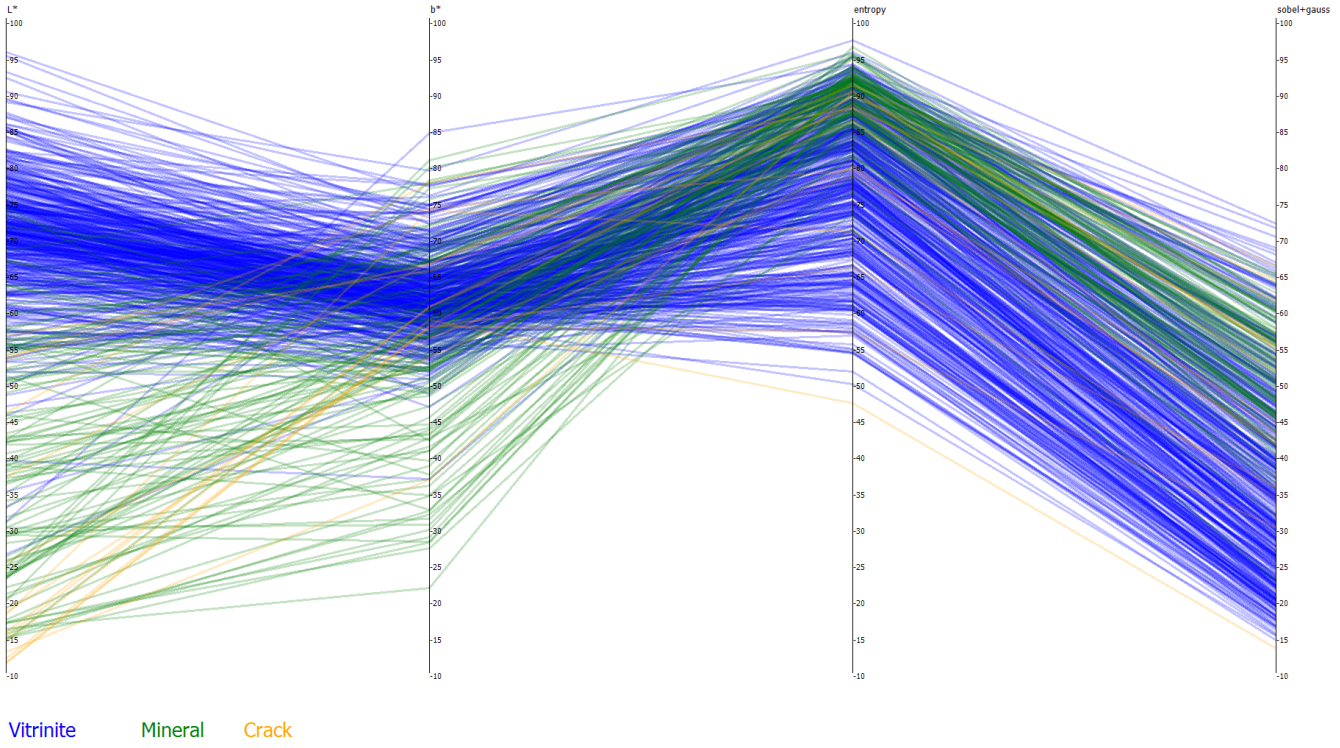


Figure 62: Parallel coordinates of alternative feature space (Sample 0)

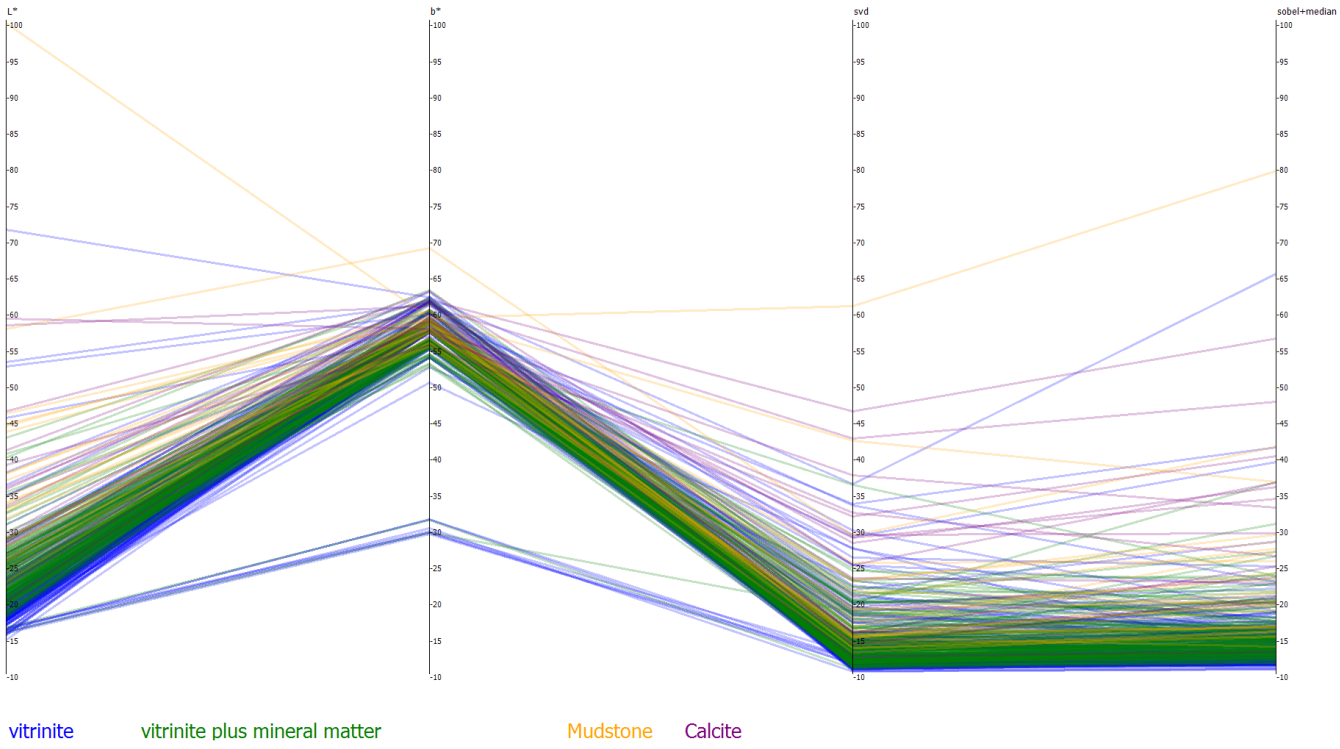


Figure 63: Parallel coordinates of baseline feature space (Sample 1)

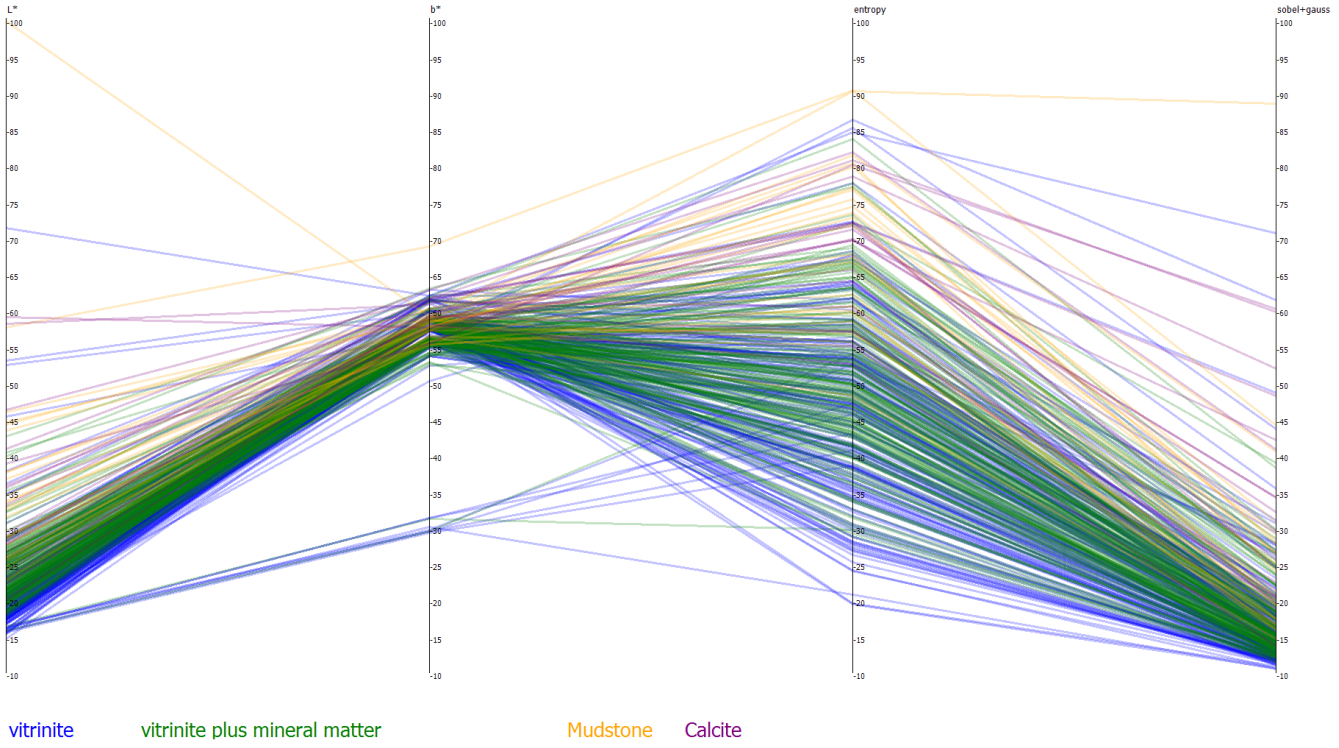


Figure 64: Parallel coordinates of alternative feature space (Sample 1)

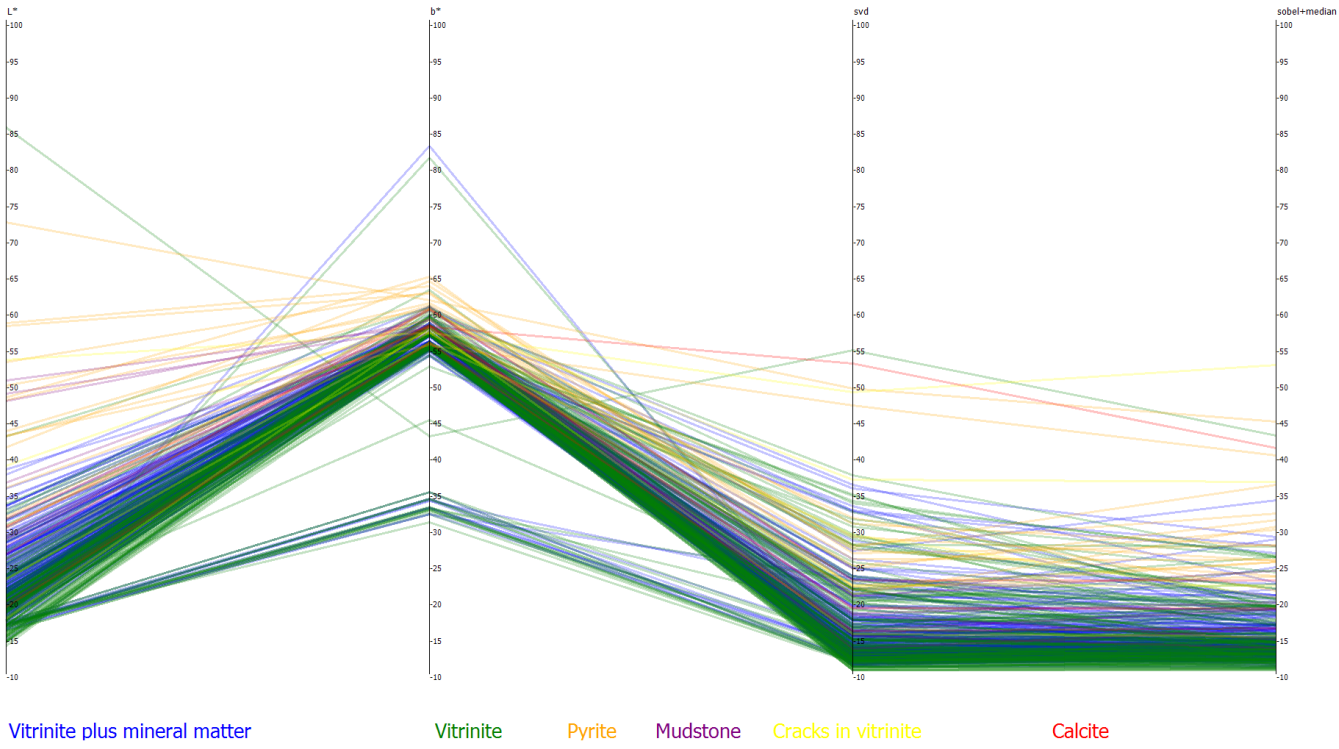


Figure 65: Parallel coordinates of baseline feature space (Sample 2)

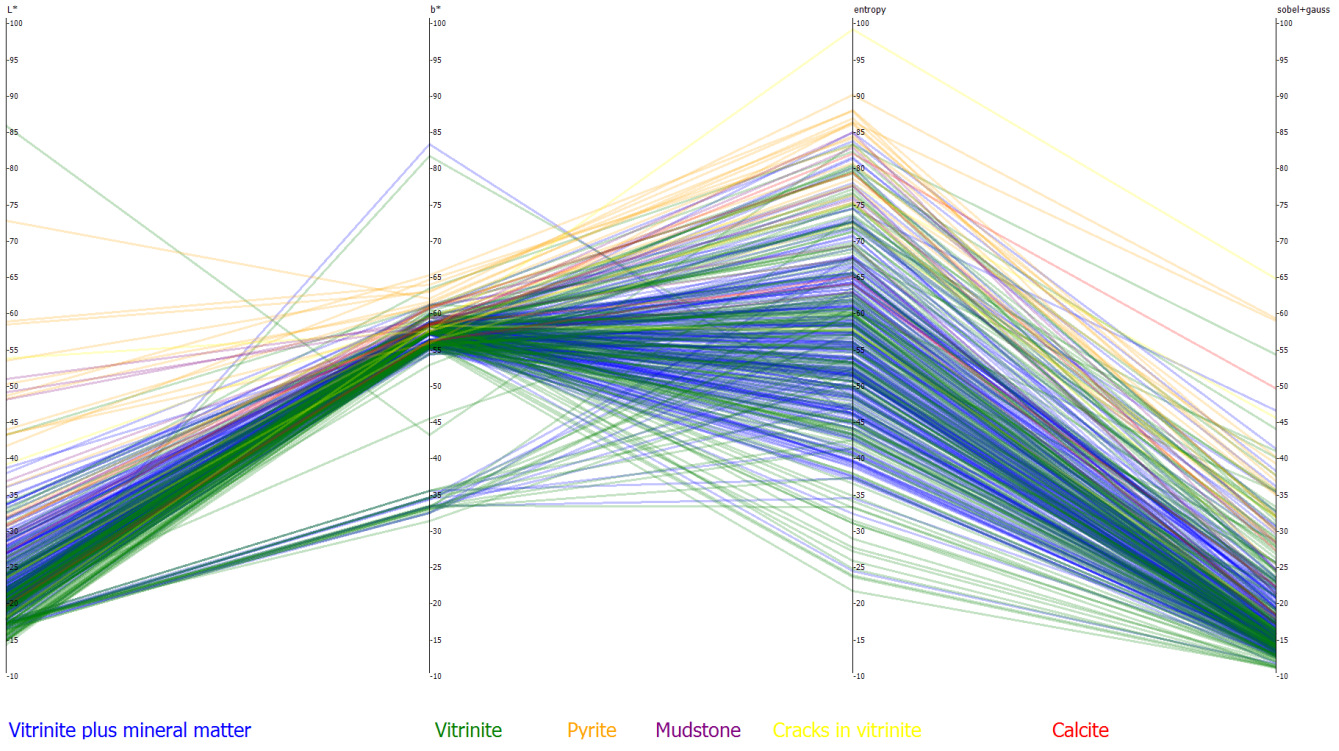


Figure 66: Parallel coordinates of alternative feature space (Sample 2)

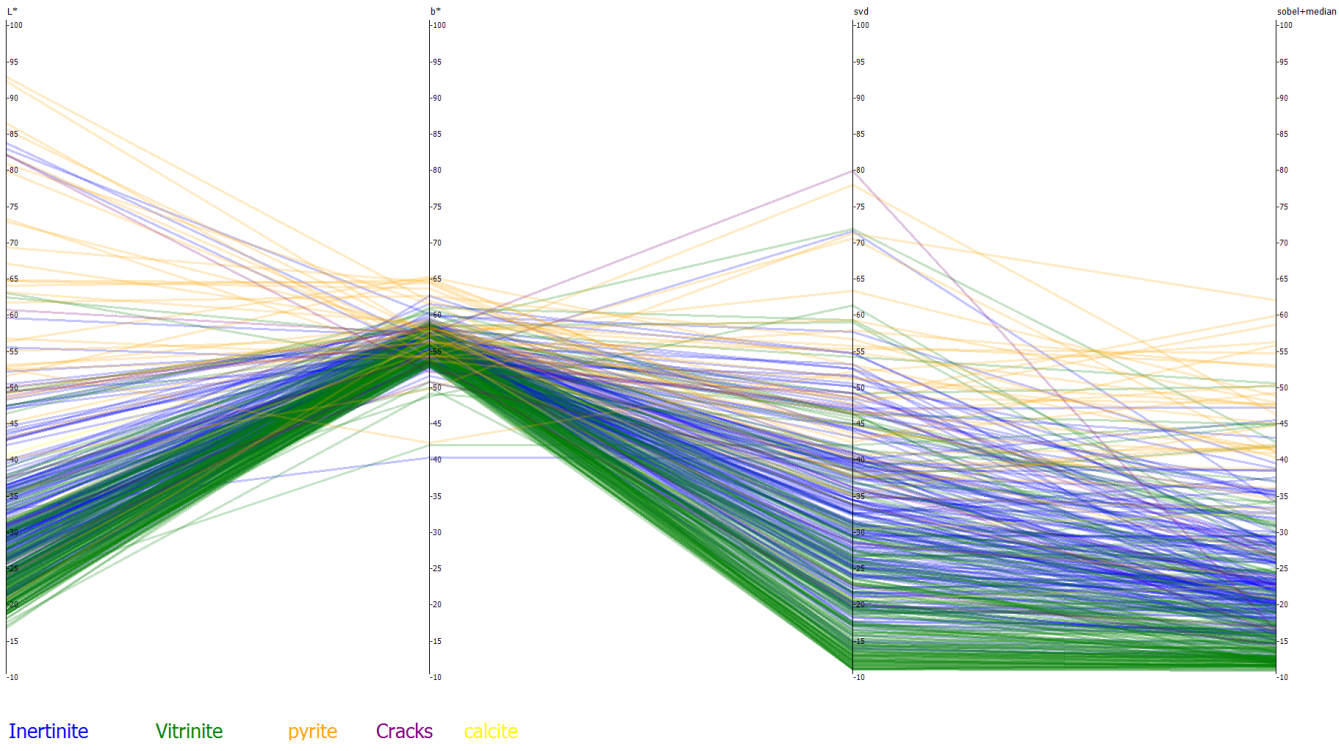


Figure 67: Parallel coordinates of baseline feature space (Sample 3)

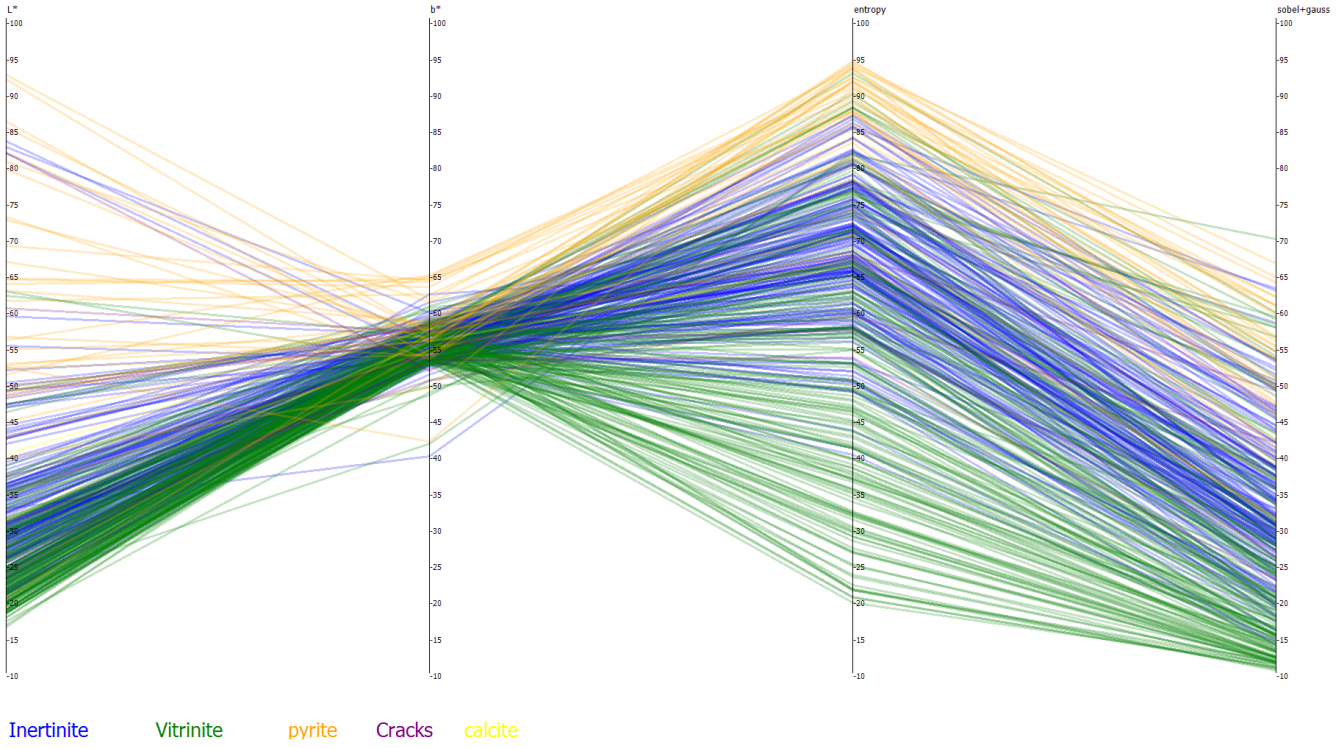


Figure 68: Parallel coordinates of alternative feature space (Sample 3)

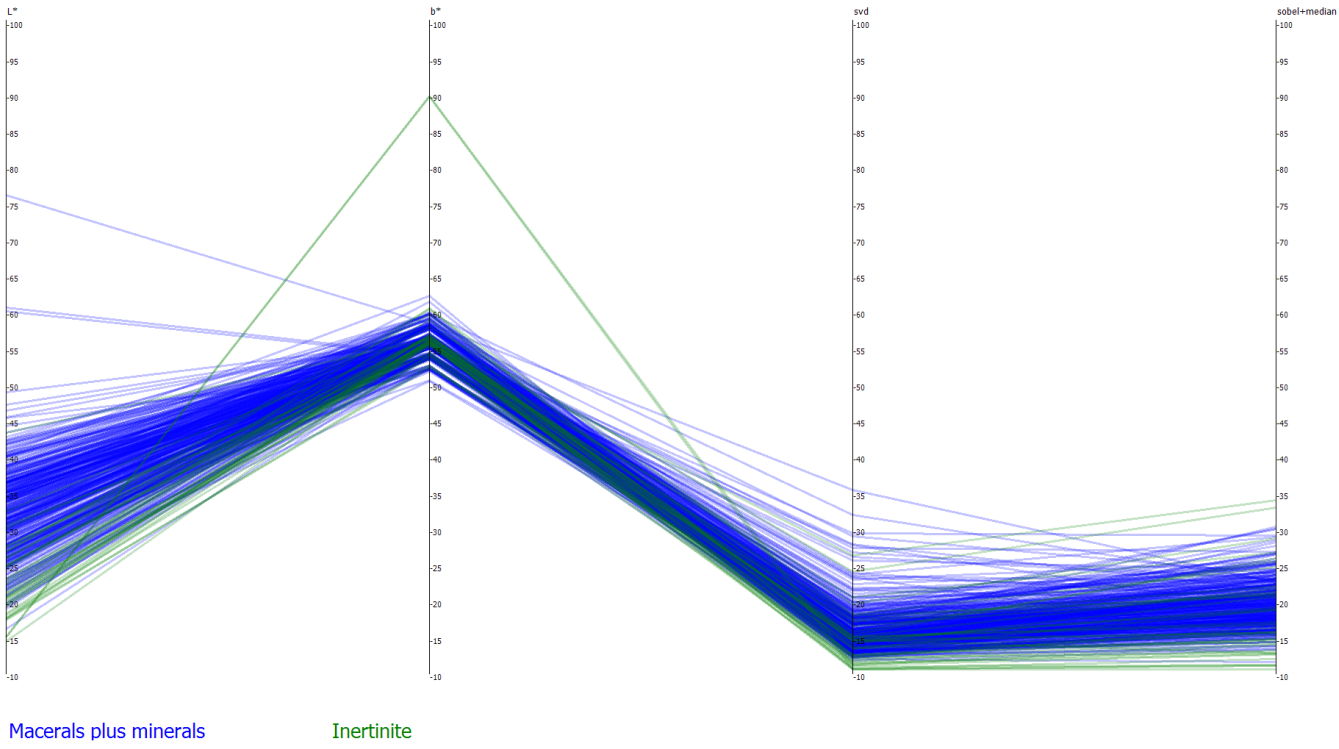


Figure 69: Parallel coordinates of baseline feature space (Sample 4)

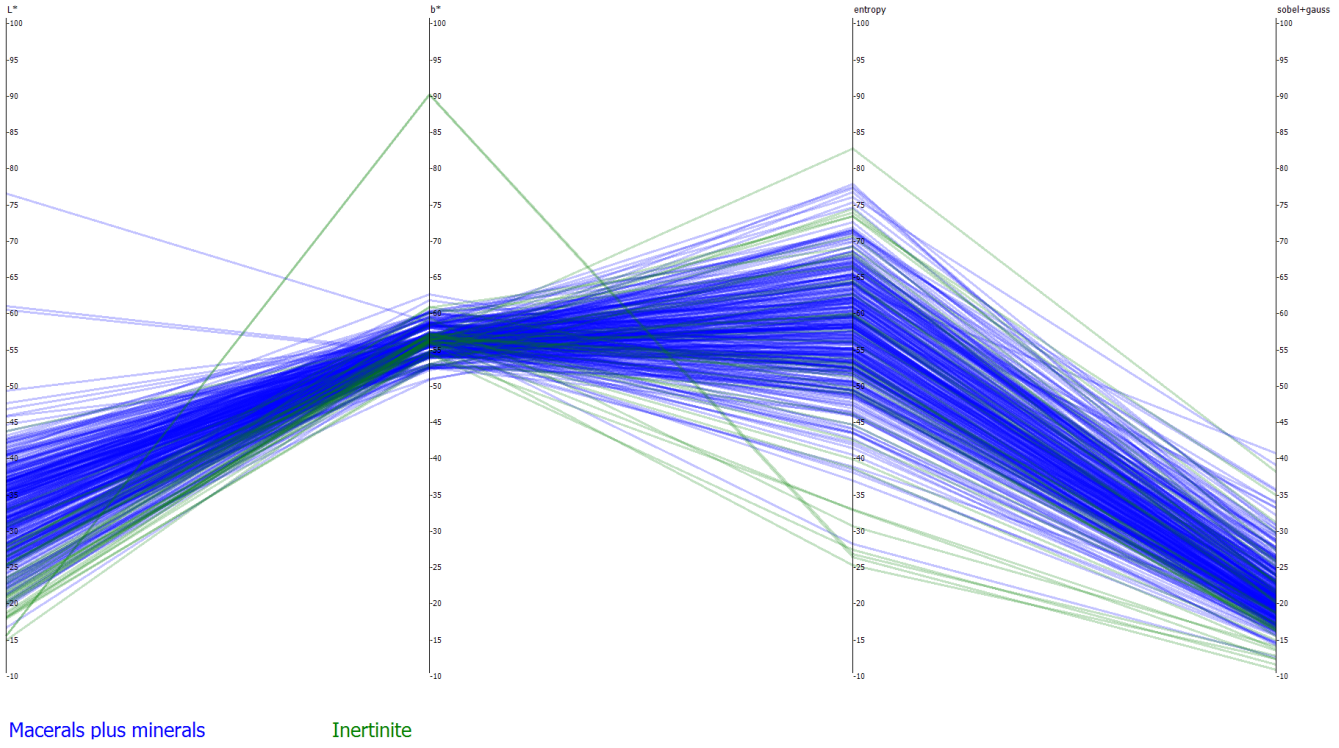


Figure 70: Parallel coordinates of alternative feature space (Sample 4)

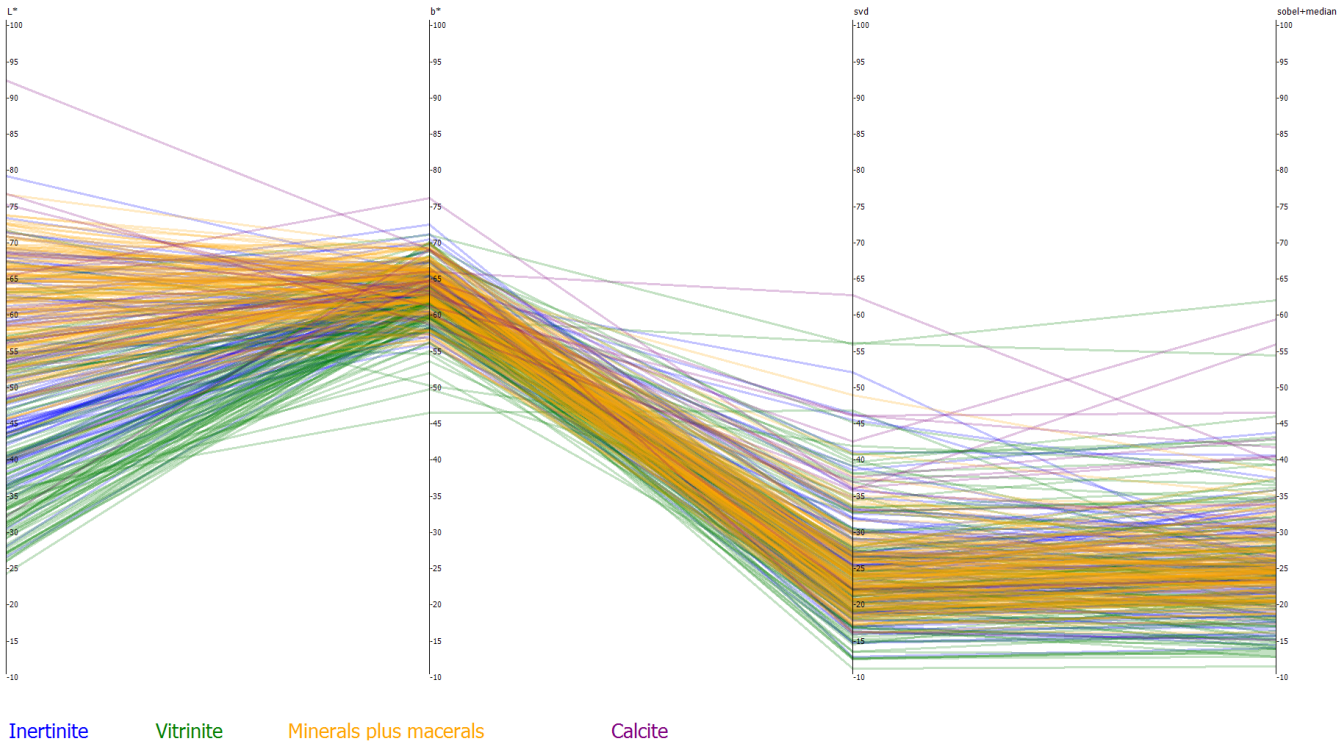


Figure 71: Parallel coordinates of baseline feature space (Sample 5)

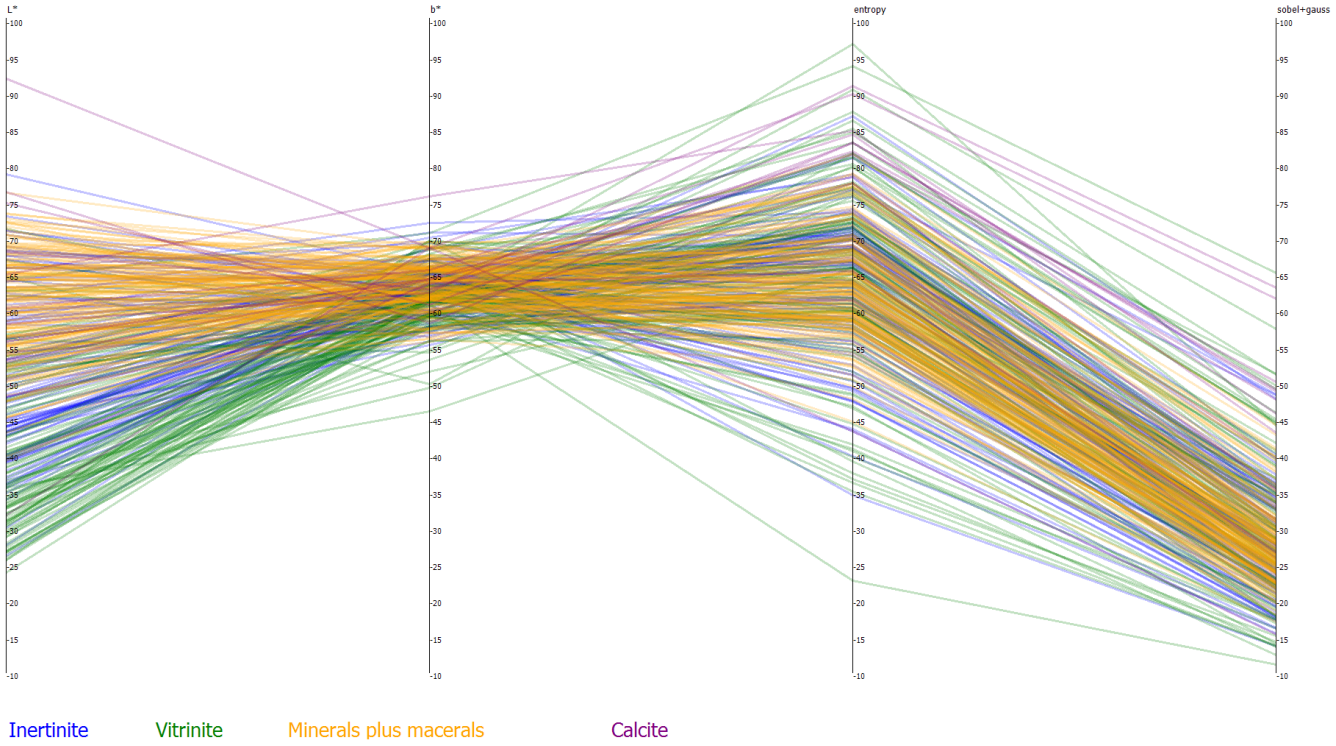


Figure 72: Parallel coordinates of alternative feature space (Sample 5)

5.2. Clustering results

The purpose of this section is to present the results of the clustering test defined in section 4.3. The following are some of the key points of discussion:

- The controlled variables in the test
- The results of the test
- The ratings achieved by the tested components

5.2.1. Controlled variables

In order to rate the clustering algorithms fairly, some parameters will need to be kept constant throughout the test. These parameters are:

- The feature space
- The distance metric
- The initial conditions (where applicable)

The top rated **feature space** that was identified in the feature extraction test will be used throughout this test. For each sample image the feature space will be extracted, once, and stored in memory. Each clustering algorithm will be implemented on said feature space and the resulting cluster configurations will be rated with the five cluster evaluation metrics, defined in section 4.3. The Euclidian **distance metric** will be used by all the clustering algorithms. This metric is chosen to correspond to the way the evaluation metrics rate the algorithms. Using this distance optimises the squared within- and between cluster distances.

Note that the mean-shifting algorithm will not be tested here. The mean shift algorithm is not a clustering algorithm and does not have a “fitness function” which it optimises. It can therefore not be rated according to classical clustering evaluation metrics. It will, instead, be tested as part of the system as a whole in sections 5.4 and 5.5.

In order to reduce the stochastic elements in the algorithms (and thereby increase the validity of comparing the evaluation metrics) a specific set of **initial conditions** will be used where possible. This will obviate the need to use time consuming repeated iterations and statistical comparisons. For the k-means, k-medians, and k-medoids algorithms an adjusted k-means++ algorithm will be used to place the initial centroids. Recall from section 3.3.2, that the k-means++ algorithm places the initial centroid positions as far apart as possible within the feature space, so as to increase the opportunity for the algorithm to converge to a good optimum. However, the first centroid is still chosen uniformly at random. To compensate for this, the first n centroid positions will correspond to the model parameters selected by the user, with n being the number of models. The rest of the centroids are placed in the same way the classical k-means++ would. This “seeding” approach will guarantee that each of the k-related algorithms have the exact same initial conditions.

The remaining clustering algorithm, namely, particle swarm optimization (PSO) cannot make use of this initialisation. PSO has an inherent stochastic structure, and makes use of multiple objects that iterate, semi-independently, to optimise the final solution. If all of these objects are initialised in the same way, little (if any) of the solution space will be explored. Therefore the particles in the PSO algorithm will be initialised with the classical k-means++ algorithm. This will retain the stochastic element in the PSO iterations and initialise it in such a way that is still competitive.

In section 3.3.3 it was made clear that the PSO algorithm uses several parameters that dictate the way the algorithm converges to an optimal solution. The parameters used in the PSO algorithm implemented for this test are as follows:

- The **swarm size** is calculated by multiplying the number of feature dimensions with the number of clusters that are being identified by the particles. This value is then incremented until a prefect square number is reached. The adjustment to a perfect square is done in order to incorporate the grid topology (see the next bullet) in the swarm.
- The **swarm topology** that was used is the grid (von Neumann) topology. The velocity parameters that are used promote quick convergence but this risks premature convergence. To reduce this effect, the grid topology will reduce the speed of convergence by limiting the field of view of each particle. More of the solution space can then be explored.
- The particle velocities are governed by the following parameters:
    - The **cognitive component** is set to 1
    - The **social component** is set to 1.49
    - The **inertia weight** is set to 0.5
    - The **velocity cap** is not used

  These velocity parameters are selected to promote an exploitative convergence, and will therefore converge relatively fast. Notice that they still satisfy the criteria for guaranteed convergence as stated in [76].
- The parameters that affect the fitness function (**intra-class weight** and **inter-class weight**) are dynamically selected by mimicking the ratio between the distance between all pairs of user selected model values and the distances between each user selected pixel and the corresponding model value. This will calibrate the weights according to the expected inter- and intra-class distances, as indicated by the user's selection.

Table 21 provides a summary of the controlled variables during the clustering test. Notice that the number of clusters are selected to be twice as many as the number of models in the image. This is a reasonable choice, given that "edge cases" are expected to exist, as explained in section 3.4.2. Furthermore, this variable is of little consequence as long as all the algorithms being evaluated adhere to a common number of clusters and as long as the number of clusters exceed the number of models.

Table 21: The controlled variables during the clustering test

Clustering algorithm	k-means	k-medians	k-medoids	PSO
Feature space	L*; b*; Sobel + Gauss; Entropy			
Distance metric	Euclidean distance			
Initial conditions	Modified k-means++			k-means++
Number of clusters	2 x #models			

5.2.2. Evaluation of clustering

It was found that the k-medoids algorithm is not a feasible alternative with the large data sets that are produced by the image samples. Even with the version, presented in section 3.3.2, that is less computationally complex, the algorithm failed to converge in a reasonable time. This is due to the fact that the algorithm runs in quadratic time. In other words, the relation between the amount of data points in the set, and the time complexity is exponential. Further optimisation would require the construction of a distance matrix before the convergence is initiated. And, as it has been stated, a distance matrix requires excessively large amounts of memory for the data sets used in this dissertation.

In order to determine if the k-medoids algorithm merits any further investigation, the clustering test was done with down sampled image data sets. The exact down sampling factors are as follows:

- Sample 0 = 6
- Sample 1 = 7
- Sample 2 = 6
- Sample 3 = 10
- Sample 4 = 7
- Sample 5 = 11

The results of this test are provided in tables 22 to 26. In this test “DBI” refers to the Davies-Bouldin index, “DI” refers to the Dunn index, “AS” refers to the average silhouette value, “D_intra” refers to the mean intra-cluster distance, and “D_inter” refers to the mean inter-cluster distance. The columns correspond to the sample images and the rows correspond to the clustering algorithms. The shaded cells provide the total sum of the indexes/distances across all samples. The shade colour ranks the results from best (green) to worst (red).

The k-medoids algorithm is ranked fairly unremarkably in this test. It was outranked by the baseline algorithm in all cases except with D_inter. Additionally, it did not perform exceptionally well in terms of any of the evaluations. For this reason the k-medoids algorithm will be disregarded as a viable alternative clustering method.

Table 22: Down sampled clustering test Davies-Bouldin index

	DBI							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.582953013	0.459067468	0.367264941	0.581696801	0.443721663	0.66478924	3.099493126	2
k-medians	0.624559228	0.536232749	0.464664196	0.751186514	0.46281103	0.751657275	3.591110991	4
k-medoids	0.62853431	0.511532816	0.382018631	0.608969639	0.442842651	0.826115881	3.400013929	3
PSO	0.388075477	0.255454376	0.313725763	0.392571331	0.181626161	0.515117691	2.0465708	1

Table 23: Down sampled clustering test Dunn index

	DI							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.209403755	0.195019398	0.230376842	0.171087929	0.901228532	0.333895698	2.041012154	2
k-medians	0.188085922	0.170120117	0.163045359	0.118221833	0.440783784	0.363420285	1.4436773	4
k-medoids	0.286017799	0.227352672	0.226813534	0.238049872	0.616563494	0.203801949	1.798599319	3
PSO	0.505437337	1.00074534	0.820108273	1.282953652	0.824575696	0.298011712	4.731832011	1

Table 24: Down sampled clustering test average silhouette

	AS							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.575830002	0.624098087	0.677318562	0.579277828	0.633707023	0.53893811	3.629169611	1
k-medians	0.561036848	0.592600476	0.580031532	0.512730468	0.581666713	0.480388248	3.308454285	4
k-medoids	0.586629631	0.643980734	0.664479997	0.572614139	0.595665565	0.471107979	3.534478045	2
PSO	0.501639445	0.66668741	0.674422702	0.620037839	0.61964957	0.42645893	3.508895896	3

Table 25: Down sampled clustering test mean intra-cluster distance

	D_intra							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	228.6975203	138.7496312	157.1474153	215.4684145	150.3139604	194.4522492	1084.829191	2
k-medians	197.2748381	128.1250779	160.0347479	201.7270162	148.0328577	200.7557416	1035.950279	1
k-medoids	218.2958683	153.7848504	161.6686526	259.5565334	156.7561479	239.081989	1189.144042	3
PSO	315.8392438	198.846943	174.5877217	254.9888478	262.524793	308.9778961	1515.765445	4

Table 26: Down sampled clustering test mean inter-cluster distance

	D_inter							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	1892.671561	2584.023716	2500.359025	1909.331672	4027.946183	1053.042747	13967.3749	3
k-medians	1579.563648	2320.813995	2444.893719	1344.019361	4051.358077	1056.539063	12797.18786	4
k-medoids	1725.131235	3048.282037	2576.135311	2214.735326	4129.958262	1362.117859	15056.36003	2
PSO	4438.235693	6260.424481	5732.344373	5643.546495	6945.167221	4424.133117	33443.85138	1

The results of the clustering test for the remaining three clustering algorithms are provided in tables 27 to 31. This test was done without down sampling. From table 27 it can be seen that the DBI for the PSO algorithm is much lower than that of the other two clustering algorithms. This indicates that the PSO algorithm creates clusters of data points that are very similar to each other, relative to the data points of other clusters. Keep in mind that the DBI takes the distance between clusters into account as well as the similarity between data points in a cluster. Therefore, a large distance between clusters can result in a low DBI even if some of the clusters are relatively uncompact. This seems to be the case when considering the high DI and large D_inter of PSO in relation to the undesirably large D_intra.

The k-means algorithm is ranked fairly consistently across all five of the evaluations. Consequently, it has the highest AS value, although by a very slight margin. The silhouette indicates how well a data point fits into its assigned cluster. The fact that the k-means algorithm has a decent DBI, DI, D_intra, and D_inter, resulted in its data points being well clustered in general. It did not excel at any one of these evaluations, but overall it produced good clustering results.

The k-medians algorithm is poorly ranked in this test. It achieved the lowest rating in all of the evaluation metrics except D_intra. It can be inferred from this that the k-medians algorithm produces very compact clusters. However, these clusters are not separated well from other clusters, and the data points in these clusters are not necessarily better fitted to its assigned clusters than other clusters.

Table 27: Clustering test Davies-Bouldin index

	DBI							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.467831279	0.364639789	0.455967102	0.405967793	0.515850025	0.396221574	2.606477561	2
k-medians	0.506381218	0.476214247	0.536828396	0.399671128	0.505778826	0.482496262	2.907370077	3
PSO	0.341100189	0.323392581	0.36125419	0.428008747	0.136148659	0.376235733	1.9661401	1

Table 28: Clustering test Dunn index

	DI							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.474777584	0.146022861	0.100026951	0.309212295	0.543176696	0.370217165	1.943433552	2
k-medians	0.441179699	0.09472793	0.053115333	0.22230549	0.599842327	0.468123119	1.879293897	3
PSO	0.795609136	1.132580083	1.008623877	0.43510446	1.110855067	0.984396047	5.467168669	1

Table 29: Clustering test average silhouette

	AS							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	0.642215309	0.653565452	0.640907096	0.626627702	0.641059314	0.599052172	3.803427046	1
k-medians	0.63115701	0.606255568	0.598942086	0.609130912	0.638732675	0.577791206	3.662009458	3
PSO	0.64282867	0.570890062	0.636722571	0.634743992	0.803143312	0.492519116	3.780847724	2

Table 30: Clustering test mean intra-cluster distance

	D_intra							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	249.8971057	142.6732942	134.2549584	113.3484003	94.22975253	118.3170052	852.7205163	2
k-medians	245.2898926	105.9079355	105.2728923	129.7425155	89.46759625	106.0735094	781.7543417	1
PSO	253.7951866	286.452569	198.5569427	184.4746202	232.6599121	197.0292268	1352.968457	3

Table 31: Clustering test mean inter-cluster distance

	D_inter							
	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Total	Rank
k-means	2214.233297	2070.562837	2270.903212	2466.90065	461.5450403	1275.817599	10759.96264	2
k-medians	2155.75816	1339.85545	1405.596127	2340.464014	418.7590665	987.6692208	8648.102038	3
PSO	4110.454957	4842.732602	4234.85073	3085.272326	6156.986409	3134.921184	25565.21821	1

5.3. Model parameters results

The purpose of this section is to present the results of the model parameters test. As discussed in section 4.4, the mean and median values of the user-selected areas were compared to the mean and median values of the feature distributions identified by the geologists. These feature distributions are the same as those used in the feature extraction test in section 5.1.2. For each model value in each image sample the Euclidean distance was calculated between the corresponding mean- and median multivariate values. The results, of which, are presented in the tables 32 to 37. Each table corresponds to one of the sample images. The cells shaded in green highlight the model parameter that related the user areas and ground truth models better.

The median values correlated better for two of the samples and the mean correlated better for four of the samples. However, in the two cases where the median correlated better, the margin of superiority was very small compared to the cases where the mean was better. Consequently, the mean appears to be a better choice to model the highlighted mineral components.

Table 32: Sample 0 model parameter test

Sample 0			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Vitrinite	30.94727386	142.826724
	Mineral	41.05589949	
	Crack	70.82355065	
Median	Vitrinite	30.26202118	149.3483673
	Mineral	42.39710895	
	Crack	76.68923716	

Table 33: Sample 1 model parameter test

Sample 1			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Vitrinite	23.63734027	76.32379874
	Vitrinite plus mineral matter	7.257298543	
	Mudstone	13.36847538	
	Calcite	32.06068454	
Median	Vitrinite	22.94174933	73.54949683
	Vitrinite plus mineral matter	6.473997119	
	Mudstone	10.98902466	
	Calcite	33.14472572	

Table 34: Sample 2 model parameter test

Sample 2			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Vitrinite plus mineral matter	12.59180344	156.3890057
	Vitrinite	38.54736574	
	Pyrite	29.54211052	
	Mudstone	22.85746717	
	Cracks in vitrinite	14.51875094	
	Calcite	38.33150793	
Median	Vitrinite plus mineral matter	10.99338916	156.9390376
	Vitrinite	37.68445351	
	Pyrite	30.60676976	
	Mudstone	22.78297841	
	Cracks in vitrinite	17.28479982	
	Calcite	37.5866469	

Table 35: Sample 3 model parameter test

Sample 3			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Inertinite	15.98937535	111.3348872
	Vitrinite	35.56442098	
	Pyrite	14.46624419	
	Cracks	24.280629	
	Calcite	21.03421765	
Median	Inertinite	15.90951676	116.9665691
	Vitrinite	33.85302069	
	Pyrite	16.32576778	
	Cracks	28.38935953	
	Calcite	22.48890429	

Table 36: Sample 4 model parameter test

Sample 4			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Macerals plus minerals	30.73155259	45.49406705
	Inertinite	14.76251446	
Median	Macerals plus minerals	32.32230597	48.33804791
	Inertinite	16.01574194	

Table 37: Sample 5 model parameter test

Sample 5			
Parameter	Model	Euclidean distance	Sum of distances
Mean	Inertinite	4.959692565	37.91965021
	Vitrinite	2.920244899	
	Minerals plus macerals	7.324658264	
	Calcite	22.71505449	
Median	Inertinite	6.510242393	35.400888
	Vitrinite	4.966445519	
	Minerals plus macerals	6.627171297	
	Calcite	17.29702879	

5.4. Final assignments results

The purpose of this section is to present the results of the final assignments test, defined in section 4.5. The following are some of the key points of discussion:

- The controlled variables in the test
- The results of the test in the form of the confusion matrices
- The ratings achieved by the tested components

5.4.1. Controlled variables

For this test all the controlled variables of the clustering test will be maintained. For each sample image the feature space will be generated, once, and the models will be selected once. The various classifiers (combinations of the clustering algorithms and model assignment methods) will be implemented on the exact same feature space and model values.

5.4.2. Evaluation of final assignments

As previously explained, each of the methods of assigning models to clusters (see figure 54) were implemented with every clustering algorithm on each of the sample images. The results of these classifications were then compared to the ground truth values, and confusion matrices were constructed. Table 38 provides an example of a confusion matrix used to compare three classes (models). The value “a” represents the overall accuracy of the classification. It is presented as a percentage of how many data points that have been correctly classified out of how many data points could have been correctly classified. The value “b” represents the percentage of data points that have been erroneously classified.

The rows represent the ground truth classes and the columns represent the classes that have been identified by the system. If a data point belongs to class 1, according to the ground truths, and has been correctly classified, it will add to the “x_1” value in the matrix. If it has been wrongly classified as belonging to class 2, it will add to the “x_2” value. Therefore, the diagonal of the matrix (shaded in green) identifies the percentage of data points that have been correctly classified, per class.

Table 38: Example confusion matrix

Right %	Wrong %	Predicted ID		
a	b	class 1	class 2	class 3
True ID	class 1	x_1	x_2	x_3
	class 2	y_1	y_2	y_3
	class 3	z_1	z_2	z_3

Tables 39 to 44 present the confusion matrices that had the highest overall accuracy with respect to classifying each sample image. The entire collection of 192 (4 clustering algorithms, 8 methods of assigning models, and 6 sample images) matrices can be found in appendix C. Take note that the overall accuracy is a very general measure of the performance of a classifier, and does not represent the absolute efficacy. If, for example, the classifier assigns all data points to a single class, and that class contains a very large amount of ground truth data points, the classifier would still have a high overall accuracy rating in spite of being unrealistically biased towards a single class. To ensure that this is not the case, the class-specific accuracies (the diagonal) have to be taken into account.

From these confusion matrices it can be observed that there is no clear method of clustering or assigning models that outperforms the rest across all the image samples. However, the following tendencies can be observed:

- The sample images with fewer models/classes are easier to classify. In the extreme case, such as a binary-class sample, the probability of wrongly classifying a data point is lower because there are only two possible classes. More classes increase the probability of erroneously classifying data points.
- Classes with more data points affect the overall accuracy much more than classes with fewer data points. This can invalidate the overall accuracy somewhat.
- Classes with fewer data points are more difficult to classify correctly. This is simply because there are fewer opportunities to classify data points from these classes correctly.
- The colour information in sample 0, and to a lesser extent in sample 2, increases the accuracy of all classifier. The added colour information creates dissimilar model values, which are easier to differentiate.
- The PSO algorithm tends to “over-assign” data points to fewer classes. This can result in high overall accuracy, granted that the classes that it is biased towards contain larger amounts of data points. The cause of this phenomenon is likely the fact that the PSO algorithm places much more emphasis on the inter-cluster distance than the alternative clustering algorithms. The resulting clusters are, then, very dissimilar. This could prove detrimental in data sets such as sample 1 and 3, where several of the models are very similar.
- Assigning the models to the regions seems to produce more consistent classifications. By this is meant that, assigning models to regions results in a higher chance of correctly classifying at least some of the data points from each class. This is opposed to assigning models to clusters where the spatial components of the data points are not taken into account.
- There does not seem to be a noticeable improvement in assigning to models-by-proxy instead of the models directly.
- Additionally, there does not seem to be a very strong correlation between the clustering quality (as indicated by the clustering test) and the quality of the classification. This is indicated, especially, well by the poor performance of the PSO algorithm in this test, in spite of it ranking very high in the clustering test.

Table 39: Sample 0 highest accuracy classification

Sample_0 – mean shift - alternative F				
Right %	Wrong %	Predicted ID		
88.33333	11.66667	Vitrinite	Mineral	Crack
True ID	Vitrinite	95.94272	4.057279	0
	Mineral	31.13208	68.86792	0
	Crack	33.33333	53.33333	13.33333

Table 40: Sample 1 highest accuracy classification

Sample_1 – k-means - alternative E					
Right %	Wrong %	Predicted ID			
56.16798	43.83202	Vitrinite	Vitrinite plus mineral matter	Mudstone	Calcite
True ID	Vitrinite	40.69767	49.4186	7.55814	2.325581
	Vitrinite plus mineral matter	16.66667	72.02381	10.11905	1.190476
	Mudstone	0	23.07692	65.38462	11.53846
	Calcite	0	13.33333	46.66667	40

Table 41: Sample 2 highest accuracy classification

Sample_ 2 – PSO - alternative B							
Right %	Wrong %	Predicted ID					
54.54545	45.45455	Vitrinite	Vitrinite plus mineral matter	Pyrite	Cracks in vitrinite	Calcite	Mudstone
True ID	Vitrinite	20.77922	20.77922	2.597403	4.545455	0	0
	Vitrinite plus mineral matter	15.2381	80.95238	3.333333	0.47619	0	0
	Pyrite	0	18.75	81.25	0	0	0
	Cracks in vitrinite	0	40	40	20	0	0
	Calcite	0	75	25	0	0	0
	Mudstone	0	42.85714	57.14286	0	0	0

Table 42: Sample 3 highest accuracy classification

Sample_3 – k-means - alternative F						
Right %	Wrong %	Predicted ID				
62.63158	37.36842	Vitrinite	Inertinite	Pyrite	Cracks	Calcite
True ID	Vitrinite	56.47059	56.47059	0.588235	8.823529	0
	Inertinite	8.928571	72.02381	2.97619	16.07143	0
	Pyrite	0	12.12121	60.60606	27.27273	0
	Cracks	0	57.14286	28.57143	14.28571	0
	Calcite	0	50	0	50	0

Table 43: Sample 4 highest accuracy classification

Sample_4 – mean shift - alternative f			
Right %	Wrong %	Predicted ID	
92.36842	7.631579	Inertinite	Macerals plus minerals
True ID	Inertinite	30.76923	30.76923
	Macerals plus minerals	0.58651	99.41349

Table 44: Sample 5 highest accuracy classification

Sample_5 – k-medians - alternative f					
Right %	Wrong %	Predicted ID			
52.95699	47.04301	Inertinite	Vitrinite	Calcite	Minerals plus macerals
True ID	Inertinite	44.23077	16.34615	9.615385	29.80769
	Vitrinite	50	35.57692	12.5	1.923077
	Calcite	16.66667	8.333333	75	0
	Minerals plus macerals	23.02632	0.657895	7.236842	69.07895

Table 45 presents a high level indication of the classifier performances throughout the final assignments test. The rows correspond to the clustering algorithm, and the columns correspond to the model assigning method of the classifiers. The values in the table represent the average overall accuracy of the respective classifiers across all the image samples. The colour shading ranks the resulting performances from good (green) to bad (red). Notice that several of the tendencies that were observed in the confusion matrices are supported by this table. In summary, these observations are:

- The PSO classification performed poorly in spite of its highly ranked clustering ability.
- Assigning models to regions (alternative D to G) results in more consistent classifications.
- Assigning models to proxies (alternatives B, C, F, and G) instead of the models directly (baseline, and alternatives A, D, and E) does not increase the performance of the classifiers. It seems to decrease the performance.
- Similarly, the use of multiple user selected areas (alternatives A, C, E, and G) had a negligible effect on the classifier performances.
- The results from the k-means, k-medians, and mean shift algorithms are all in the same ballpark.

Table 45: General performance of classifiers

	baseline	alternative A	alternative B	alternative C	alternative D	alternative E	alternative F	alternative G
kmeans	59.73787235	59.39910455	63.4292006	62.32739305	62.17844011	63.41756851	62.89919295	62.63426295
kmedians	60.24065595	61.84767447	62.63991097	62.50432734	62.32729938	63.28424997	63.20193087	61.96150362
mean shift	59.34099164	60.62930648	62.37318189	63.21118154	60.72735586	63.32977295	63.97726682	63.75717787
pso	56.67693847	59.06660504	58.54440034	46.02328273	58.01932466	60.05151864	59.40760387	46.65744142

The clustering algorithm that appears to work the best with the region-based model assignments is the mean shift algorithm. The mean shift algorithm's inherent use of spatial information is the likely cause. The model assignment method, alternative E, achieved the most success in this test and appears to work reasonably well with all four of the clustering algorithms. For this reason the mean shift algorithm is chosen as the preferred method of clustering and the alternative E model assignment method as the preferred model assigner.

5.5. The system results

The purpose of this section is to present the results of the final system test defined in section 4.6. The following are some of the key points of discussion:

- A brief comparison of the baseline- and alternative system
- The image segmentations obtained during the test
- A comparison of the confusion matrices corresponding to the image segmentations

5.5.1. The alternative system

Table 46 provides a detailed comparison of the baseline and the final system that is made up of the best alternatives in the previous tests.

Table 46: Comparison of baseline- and final system

		Baseline	Alternative
Feature extraction	Luminosity	L*	
	Chromaticity	b*	
	Roughness	Sobel operator + Median filter	Sobel operator + Gaussian filter
	Regularity	Singular value decomposition	Entropy
Models		Median	Mean
Classification	Clustering	k-means	mean shift
	Final assignment	Nearest neighbour cluster to model assignment. Single area per model.	Cluster to region, region to nearest model assignment. Multiple areas per model.

5.5.2. Image segmentations

This test is very similar to the final assignments test. The baseline system and the alternative system will both be implemented on the sample images using the same model selection areas. The baseline test will be implemented ten times to take into account the random initial conditions of the k-means++ algorithm. Recall that the mean shift algorithm in the alternative system does not make use of stochastic elements to be initialised, and therefore, no repeated implementations are necessary. The results for each sample image follows. In each of the resulting image segmentations the sample image with the model selection areas highlighted, is presented to the left, and the image segmentation is presented to the right.

- Sample 0

  The baseline system converged to two possible solutions. The first solution (segmentation A) is depicted in figure 73, and the second (segmentation B) in figure 74. Segmentation A was obtained nine out of the ten times the algorithm was initialised. Figure 75 provides the segmentation obtained by the alternative system. In all three of these figures the blue regions in the image segmentation represent vitrinite, the green regions represent mineral matter, and the orange regions represent the cracks in the sample.

  The baseline segmentation A failed to distinguish between the crack and mineral matter in the sample. This indicates that the clusters that were identified by the k-means algorithm were closer together in terms of the luminosity feature than the chromaticity feature. This is, obviously, an undesirable characteristic. The baseline segmentation B succeeded to make this distinction. However, keep in mind that segmentation B was only obtain one tenth of the times the algorithm was initialised. The alternative segmentation successfully distinguished between the three mineral components present in the sample, although it appears to be slightly more biased towards classifying data points as “cracks” than the baseline segmentations.

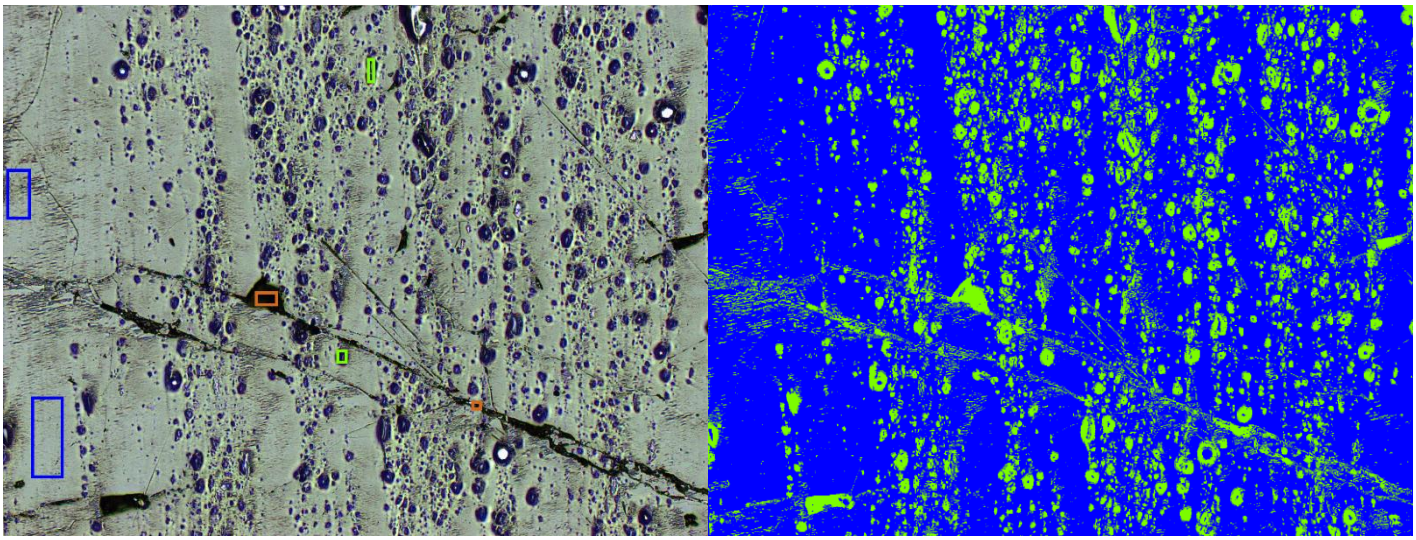


Figure 73: Sample 0 baseline segmentation A

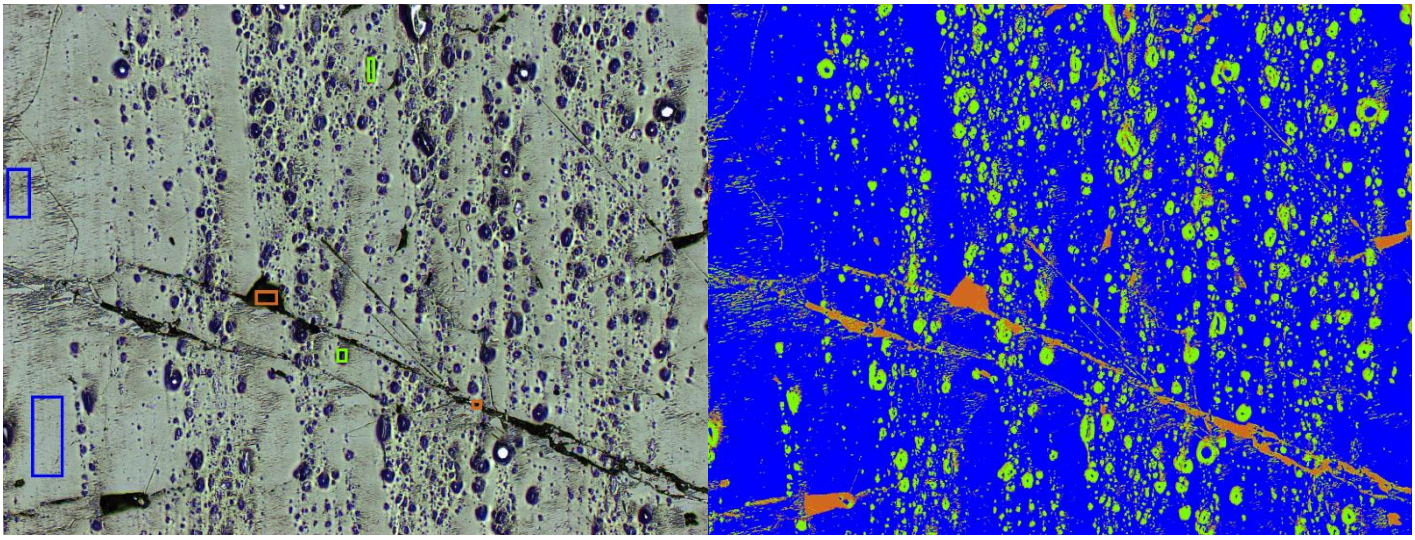


Figure 74: Sample 0 baseline segmentation B

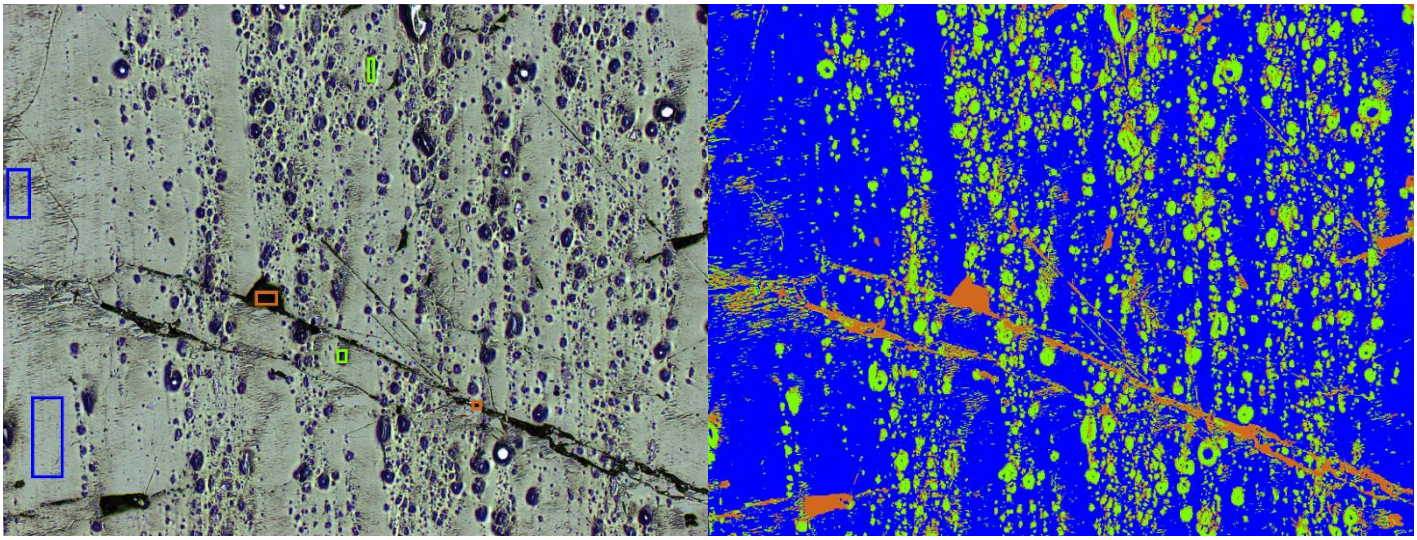


Figure 75: Sample 0 alternative segmentation

- Sample 1

  The baseline system converged to two possible solutions. The first solution (segmentation A) is depicted in figure 76, and the second (segmentation B) in figure 77. Segmentation A was obtained eight out of the ten times the algorithm was initialised. Figure 78 provides the segmentation obtained by the alternative system. In all three of these figures the blue regions in the image segmentation represent vitrinite, the green regions represent vitrinite plus mineral matter, the orange regions represent mudstone, and the red regions represent calcite.

  The two solutions that were obtained by the baseline system are very similar. The only easily discernible difference is the amount of calcite present in the segmentations. Segmentation B appears to be slightly more biased towards calcite than segmentation A. The alternative segmentation is very different to the two baseline segmentations. The segmentation regions are, spatially, very coherent. It does seem to distinguish between “vitrinite” and “vitrinite and mineral matter” with more success than the baseline systems. However, there does not seem to be an increase in identifying the mudstone in the sample.

  Take note of the high concentration of vitrinite plus mineral matter that was identified, by the baseline segmentations, in the upper-left region of the sample. This is a false classification, according to the ground truths. The likely cause is the anomalous red hue present in that region. This was overcome by the alternative system by making use of multiple model area selections.

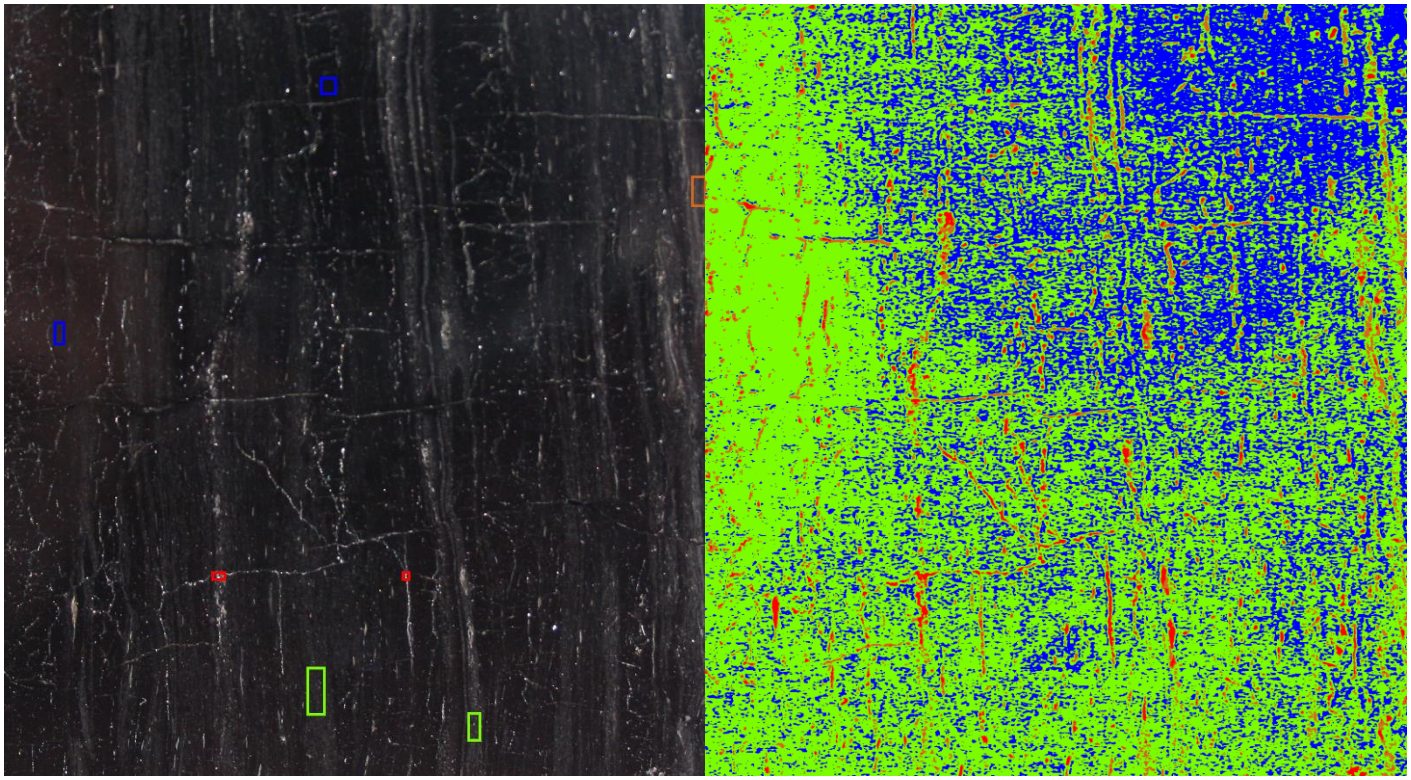


Figure 76: Sample 1 baseline segmentation A

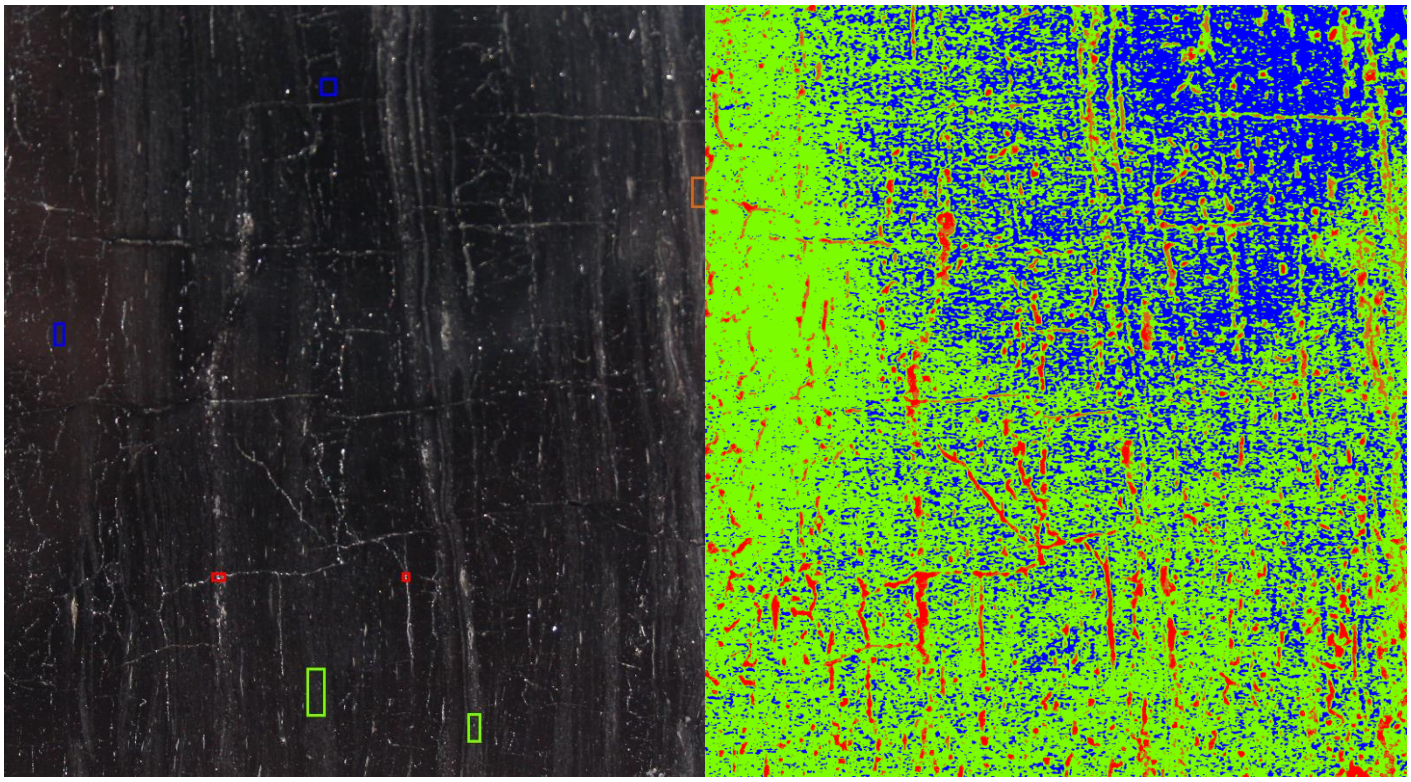


Figure 77: Sample 1 baseline segmentation B

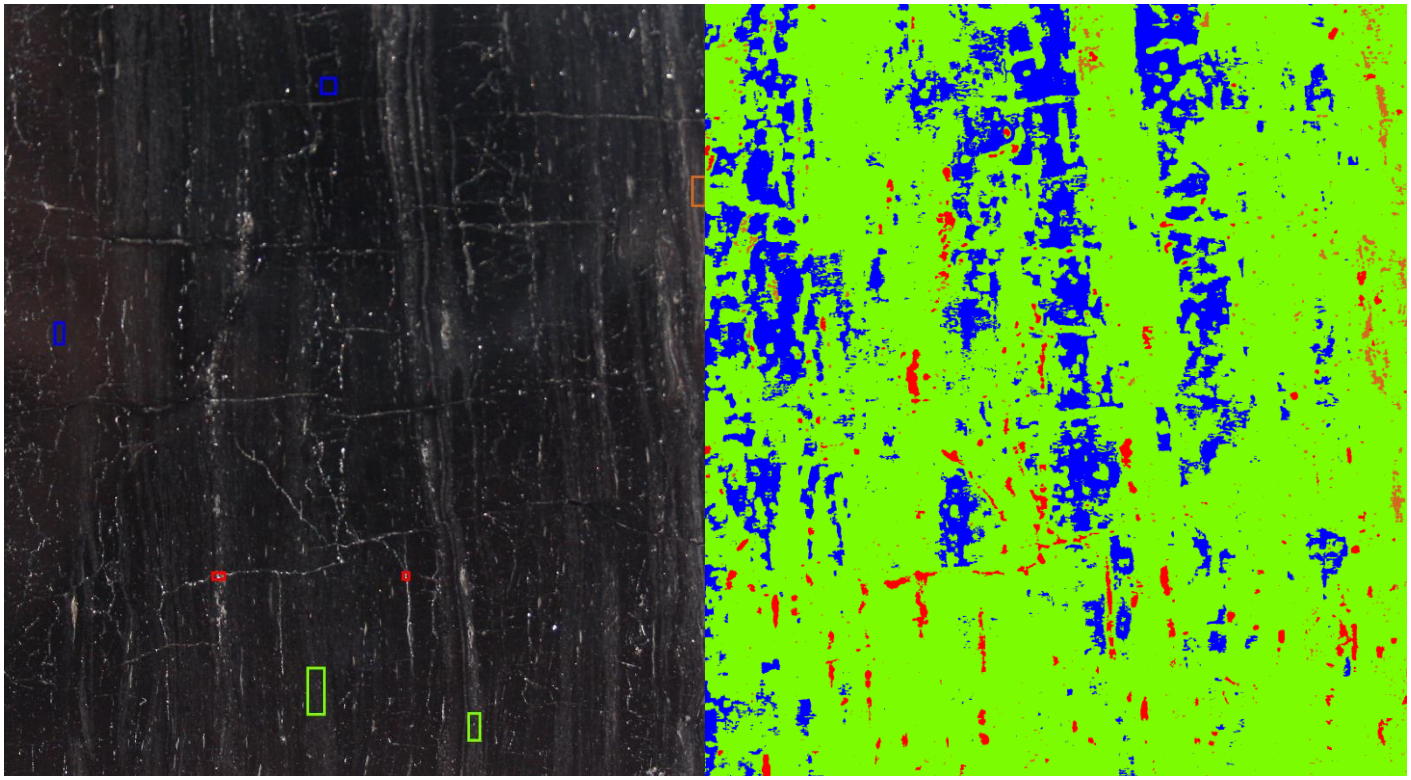


Figure 78: Sample 1 alternative segmentation

- Sample 2

  The baseline system converged to two possible solutions. The first solution (segmentation A) is depicted in figure 79, and the second (segmentation B) in figure 80. Segmentation A was obtained five out of the ten times the algorithm was initialised. Figure 81 provides the segmentation obtained by the alternative system. In all three of these figures the blue regions in the image segmentation represent vitrinite, the green regions represent vitrinite plus mineral matter, the orange regions represent pyrite, the red regions represent cracks in the vitrinite, the grey regions represent the calcite, and the maroon regions represent mudstone.

  The only difference between the baseline- segmentation A and segmentation B is that segmentation A failed to identify any vitrinite mineral components. The final cluster configuration of segmentation A, therefore, contained clusters that were always more similar to the “vitrinite plus mineral matter” component than the vitrinite component. The alternative segmentation, again, contains more spatial coherence. The mineral components, represented by the final segmentation regions are more localised and less spread out across the entire sample. In all three of the segmentations there is a high bias towards “vitrinite plus mineral matter”. This is largely, because of the small difference between “vitrinite” and “vitrinite plus mineral matter” and the large amount of the latter in the sample.

  A final point of observation: The accuracy of classifying the “cracks in vitrinite” is poor for all three segmentations. This is undesirable, yet understandable considering the faintness of the observable cracks in the sample.

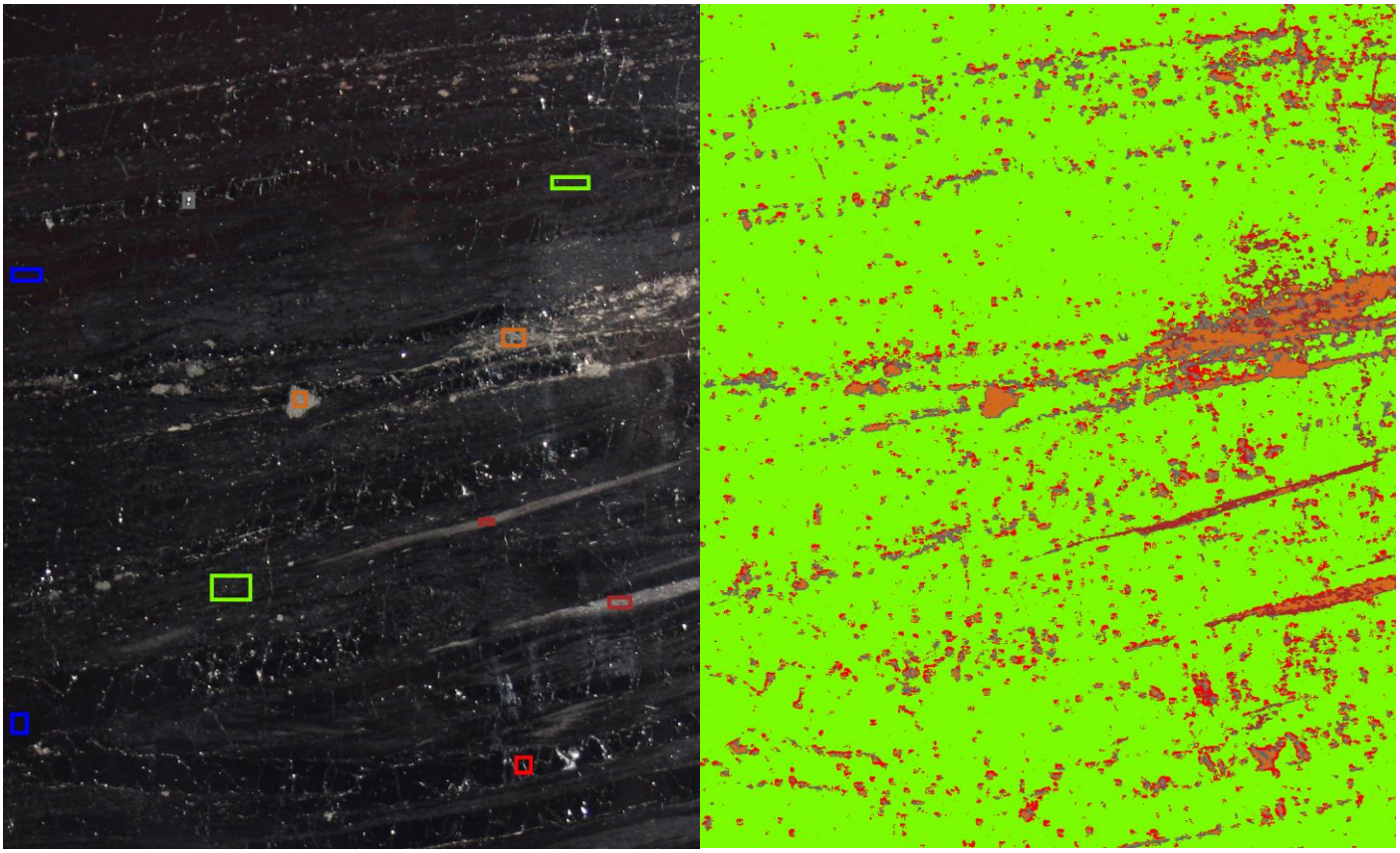


Figure 79: Sample 2 baseline segmentation A

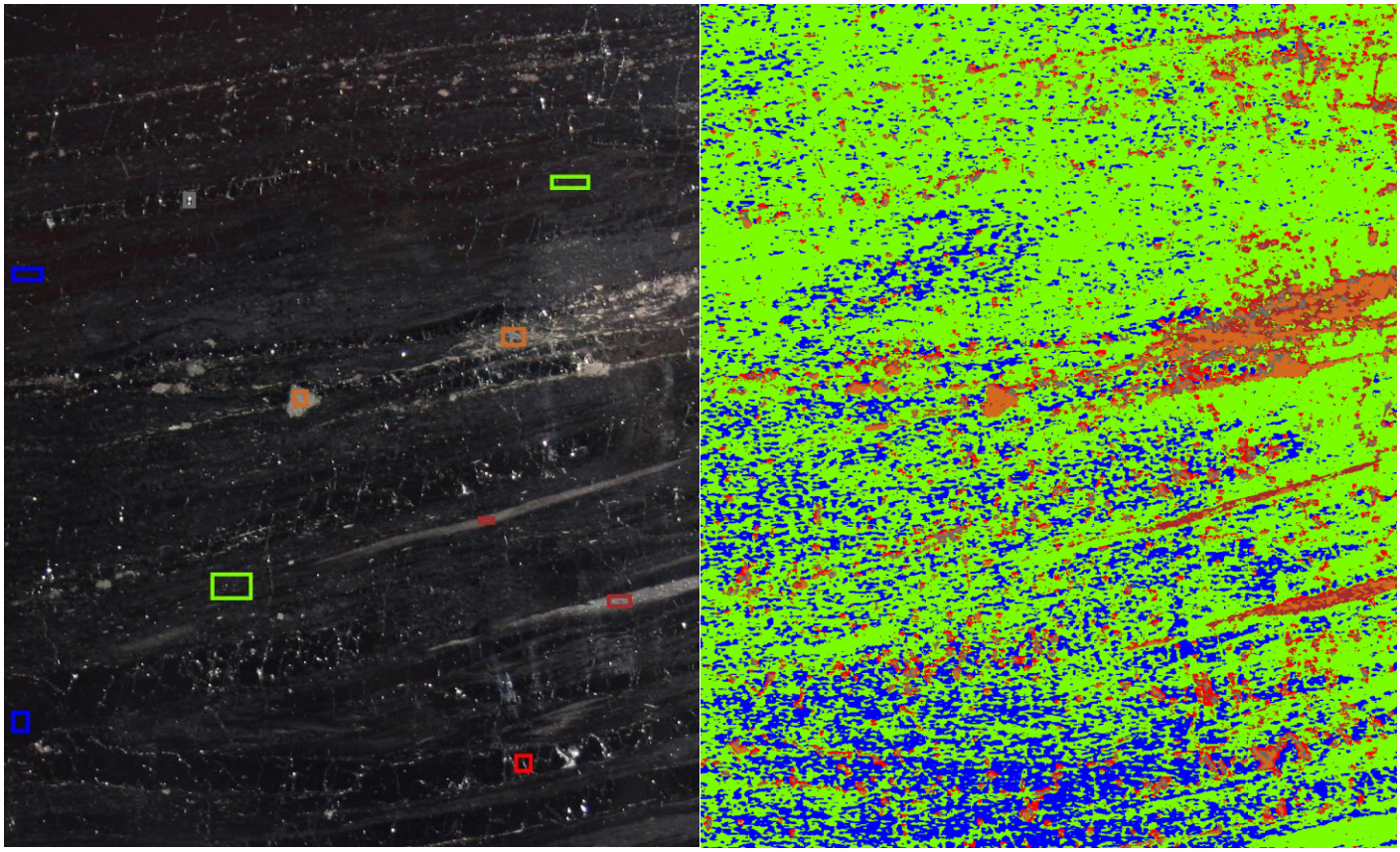


Figure 80: Sample 2 baseline segmentation B

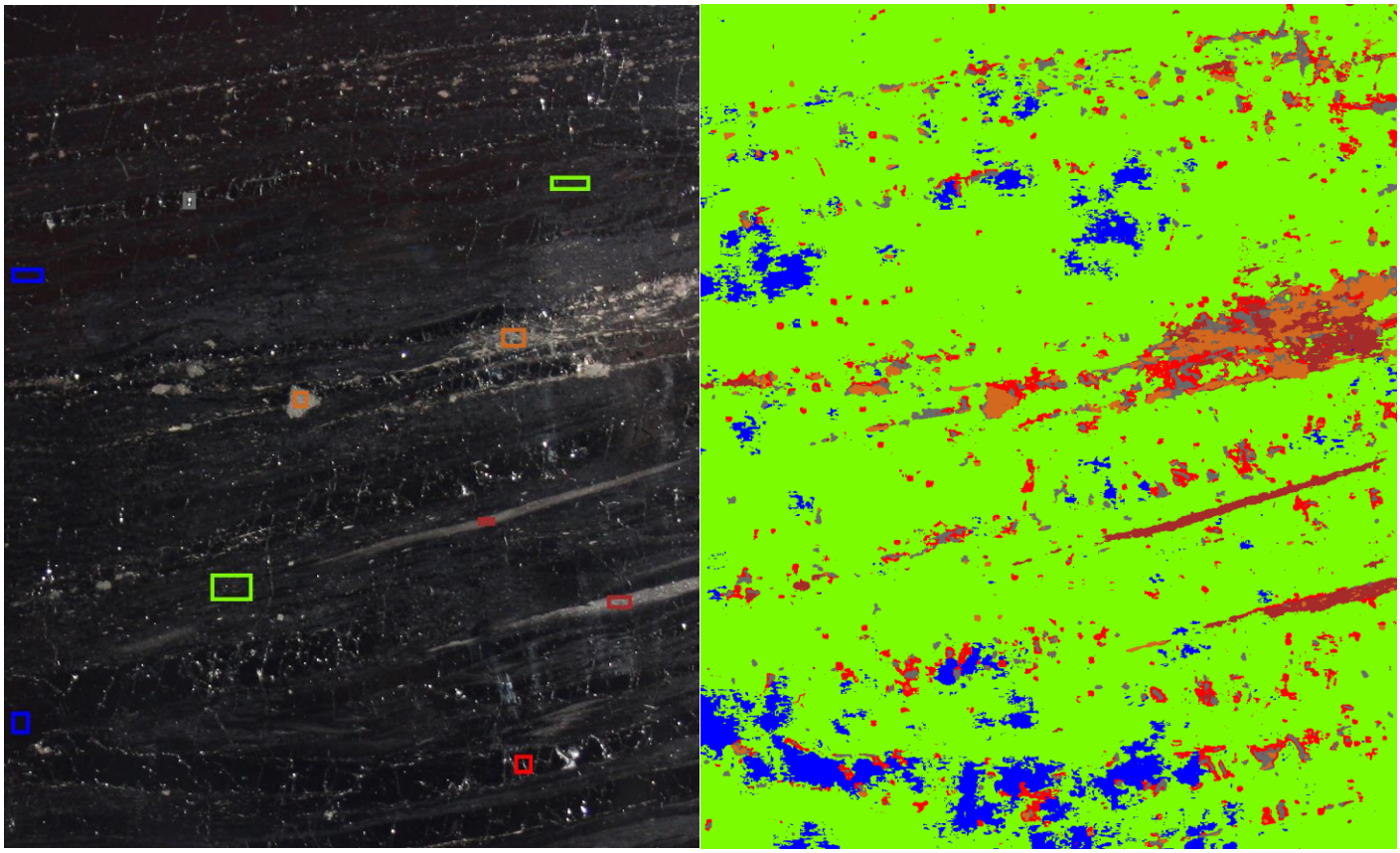


Figure 81: Sample 2 alternative segmentation

- Sample 3

  The baseline system converged to two possible solutions. The first solution (segmentation A) is depicted in figure 82, and the second (segmentation B) in figure 83. Segmentation A was obtained 7 out of the ten times the algorithm was initialised. Figure 84 provides the segmentation obtained by the alternative system. In all three of these figures the blue regions in the image segmentation represent vitrinite, the green regions represent inertinite, the orange regions represent pyrite, the red regions represent cracks, and the grey regions represent the calcite.

  The baseline segmentations for sample 3 are very similar. Segmentation A contains slightly more pyrite and inertinite components, but both are extremely biased towards vitrinite. The alternative segmentation appears to distinguish between the vitrinite-, inertinite-, and pyrite components very well. All three of the segmentations performed poorly in term of classifying the cracks and calcite components. The baseline system failed to assign any data points to these two minerals, and the alternative system appears to mistake some inertinite components for calcite or cracks.

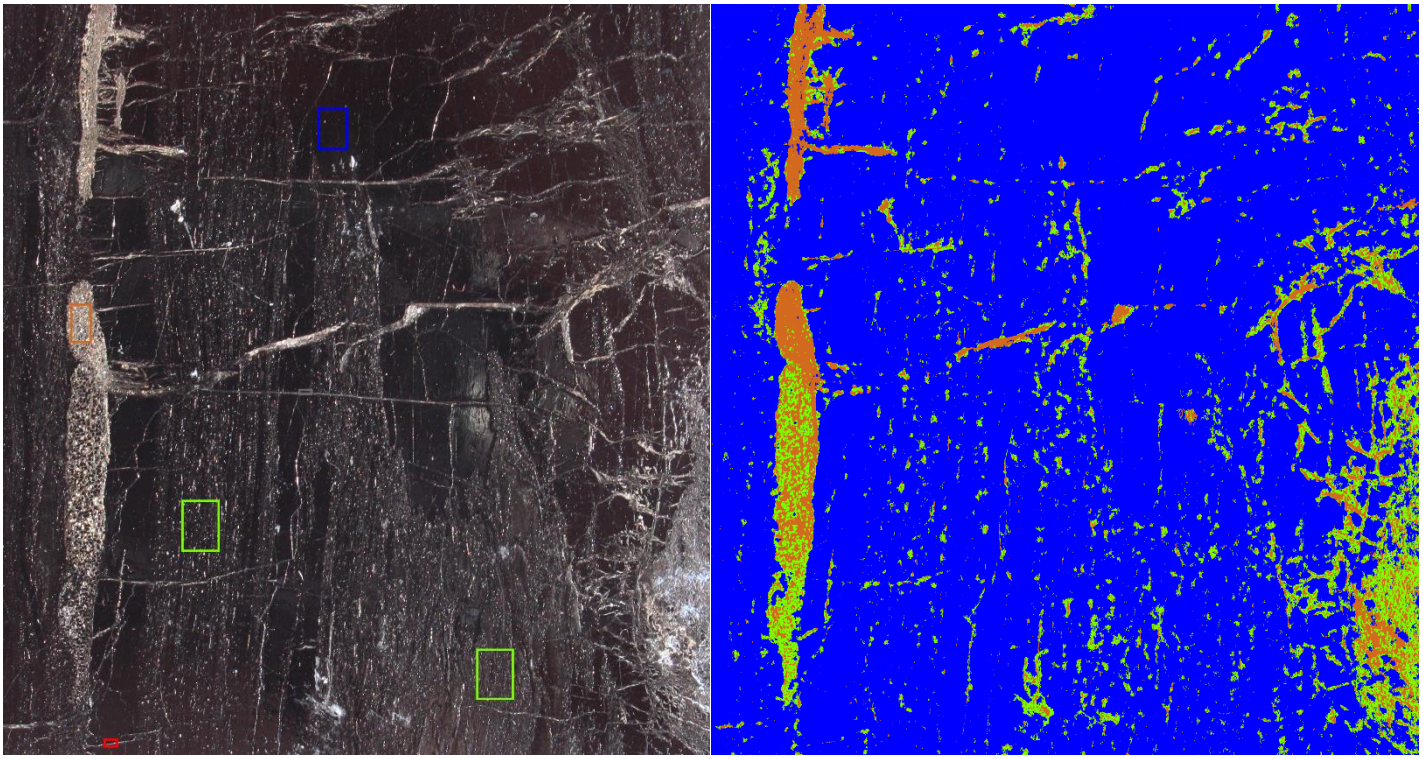


Figure 82: Sample 3 baseline segmentation A

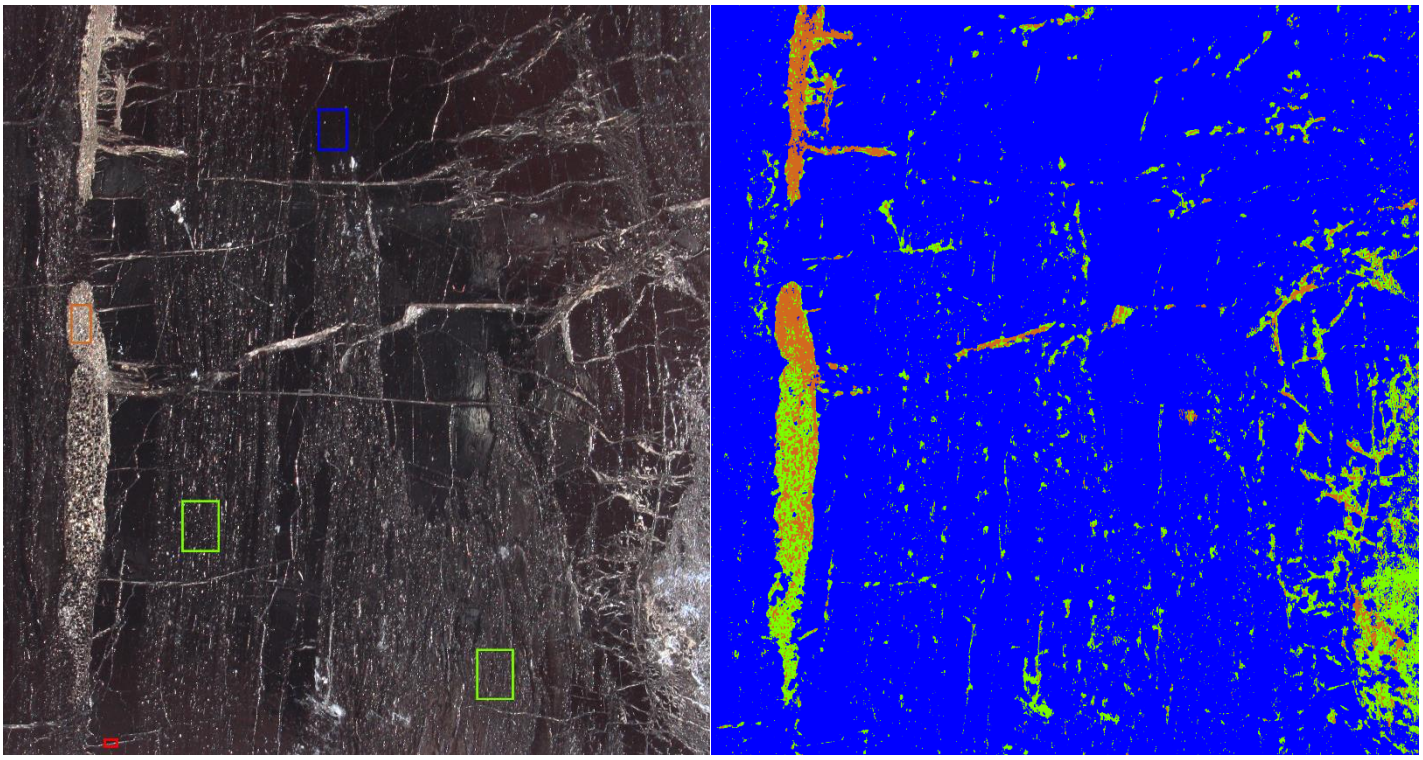


Figure 83: Sample 3 baseline segmentation B

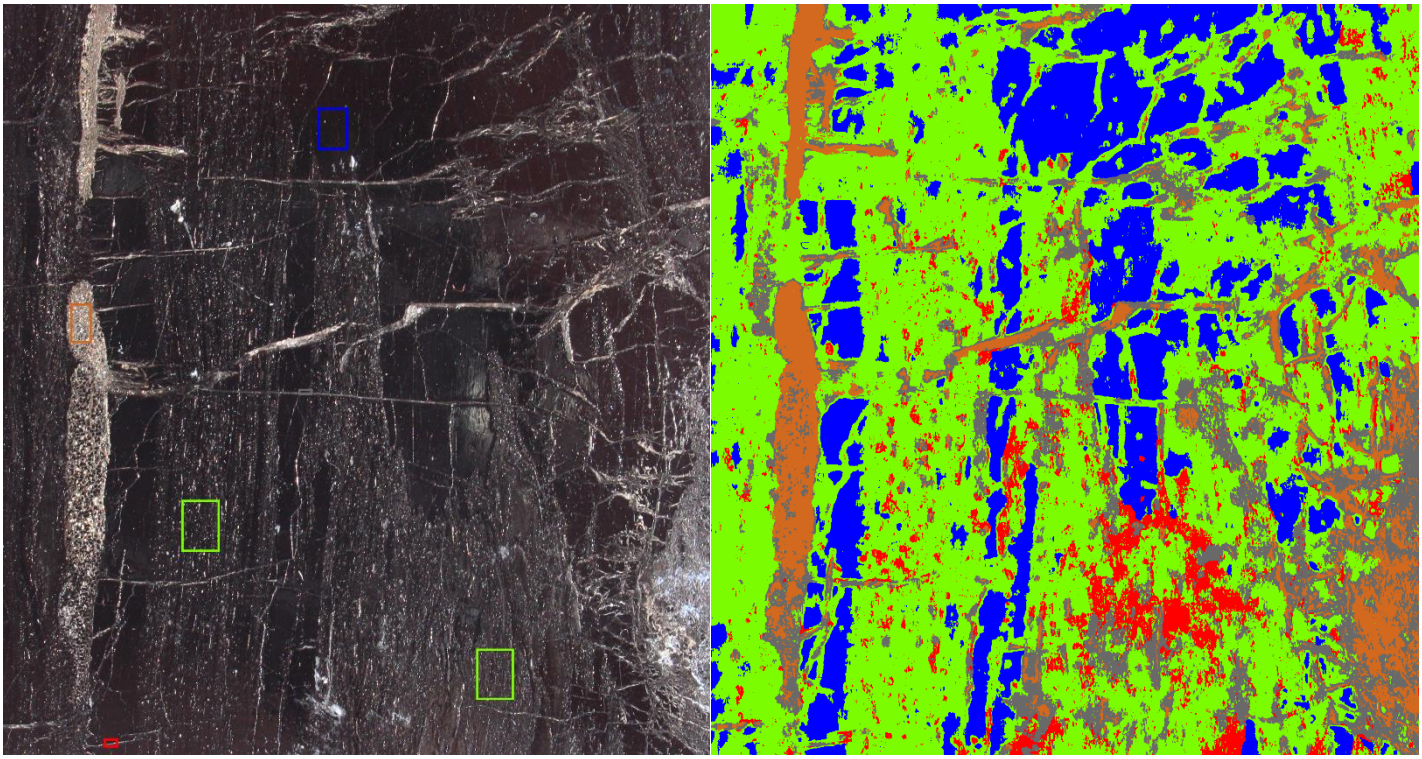


Figure 84: Sample 3 alternative segmentation

- Sample 4

  The baseline system converged to only one solution during the ten repeated initialisations. This baseline solution is depicted in figure 85. Figure 86 provides the segmentation obtained by the alternative system. In both of these figures the blue regions in the image segmentation represent macerals and minerals, and the green regions represent inertinite.

  The baseline system failed to distinguish between the two model classes, and assigned all data points to the “macerals and minerals” class. The alternative system succeeded where the baseline failed. The two classes are very similar in terms of chromaticity, roughness, and regularity, and it is no surprise that the baseline system failed. The additional spatial information of the alternative system helped to differentiate the classes.

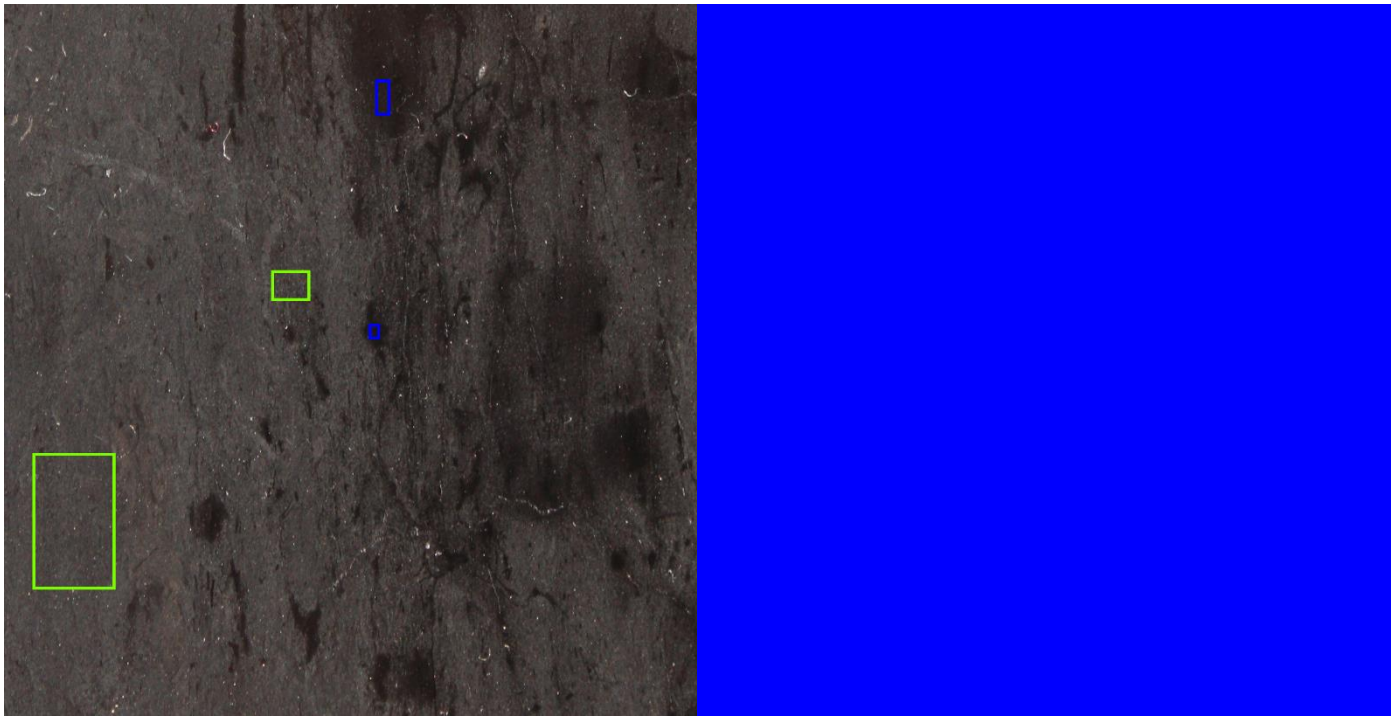


Figure 85: Sample 4 baseline segmentation

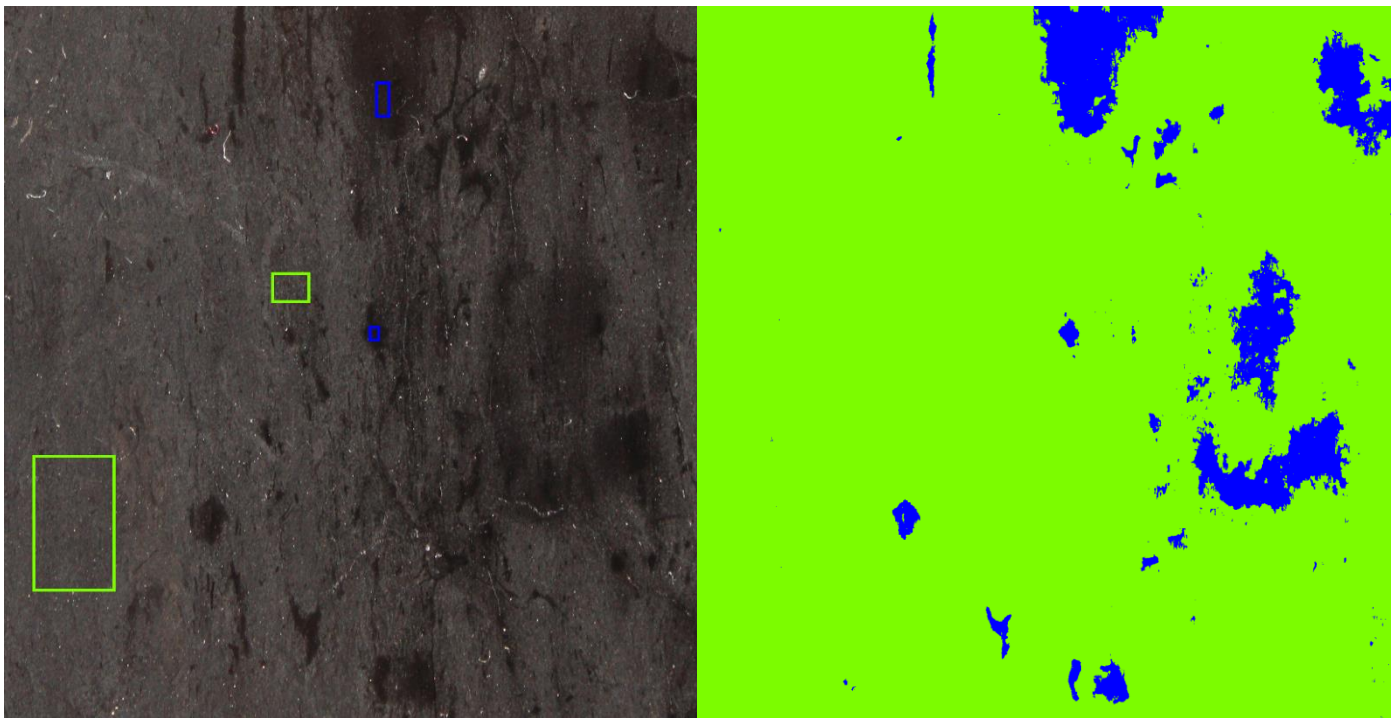


Figure 86: Sample 4 alternative segmentation

- Sample 5

  The baseline system converged to four possible solutions. The first solution (segmentation A) is depicted in figure 87, the second (segmentation B) in figure 88, the third (segmentation C) in figure 89, and the fourth (segmentation D) in figure 90. Segmentation A was obtained 7 out of the ten times the algorithm was initialised, while segmentation B, C, and D was obtained once. Figure 91 provides the segmentation obtained by the alternative system. In all five of these figures the blue regions in the image segmentation represent inertinite, the green regions represent vitrinite, the orange regions represent calcite, and the red regions represent minerals and macerals.

  Notice that the baseline segmentations A, C, and D, are almost identical. All three of them only identified vitrinite and “minerals and macerals”. The baseline segmentation B identified some vitrinite in addition to this. Yet again, the alternative system segmented the sample image more equally among model values and produced regions that are spatially relatively compact. Also take note that the alternative system produced the only segmentation that identified calcite components as well.

  The upper-right region of the sample was classified as minerals and macerals by all five of these segmentations, when it is actually a combination of vitrinite and inertinite. This is due to the intense reflectance in that area of the sample. The excessive brightness produces a similar region to the regions containing minerals and macerals.

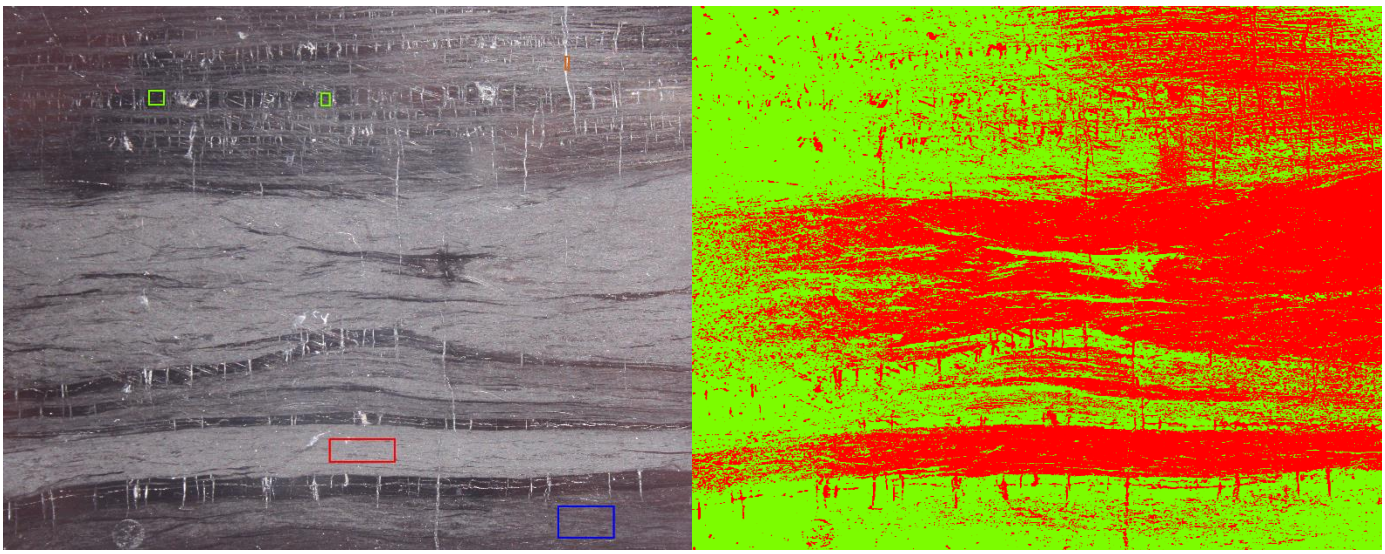


Figure 87: Sample 5 baseline segmentation A

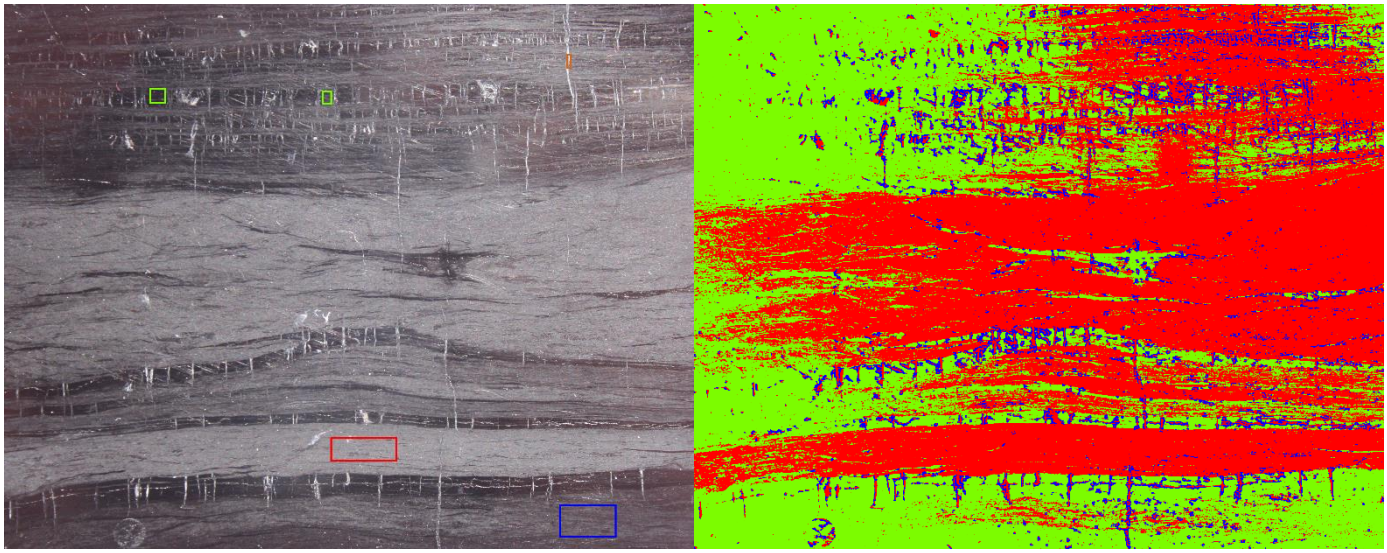


Figure 88: Sample 5 baseline segmentation B

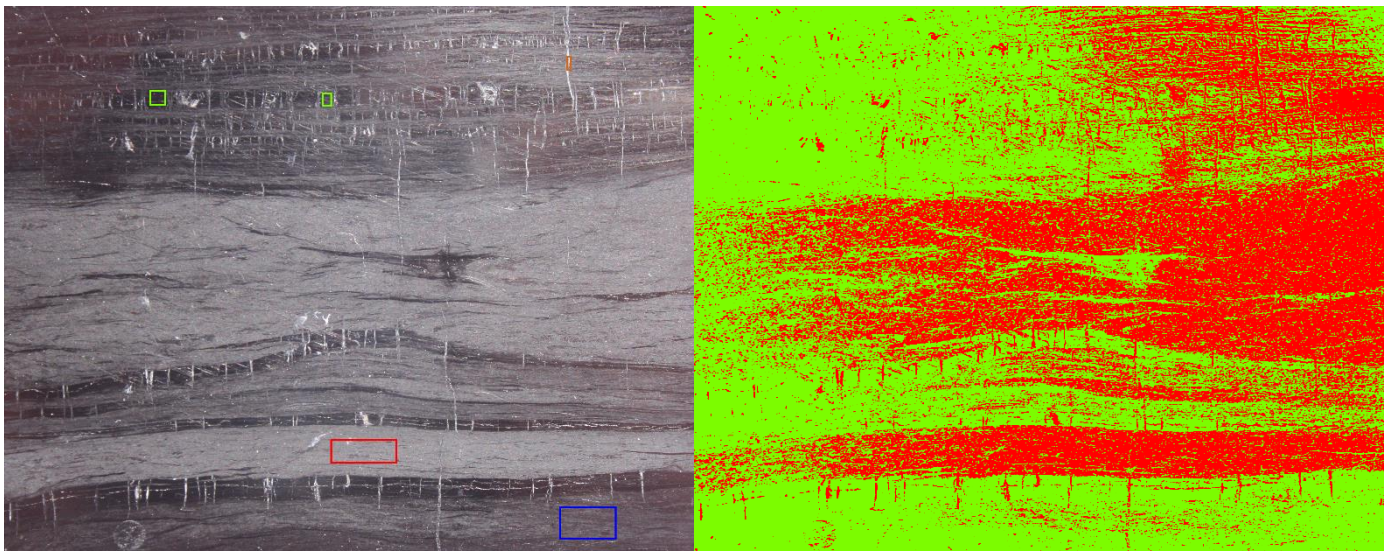


Figure 89: Sample 5 baseline segmentation C

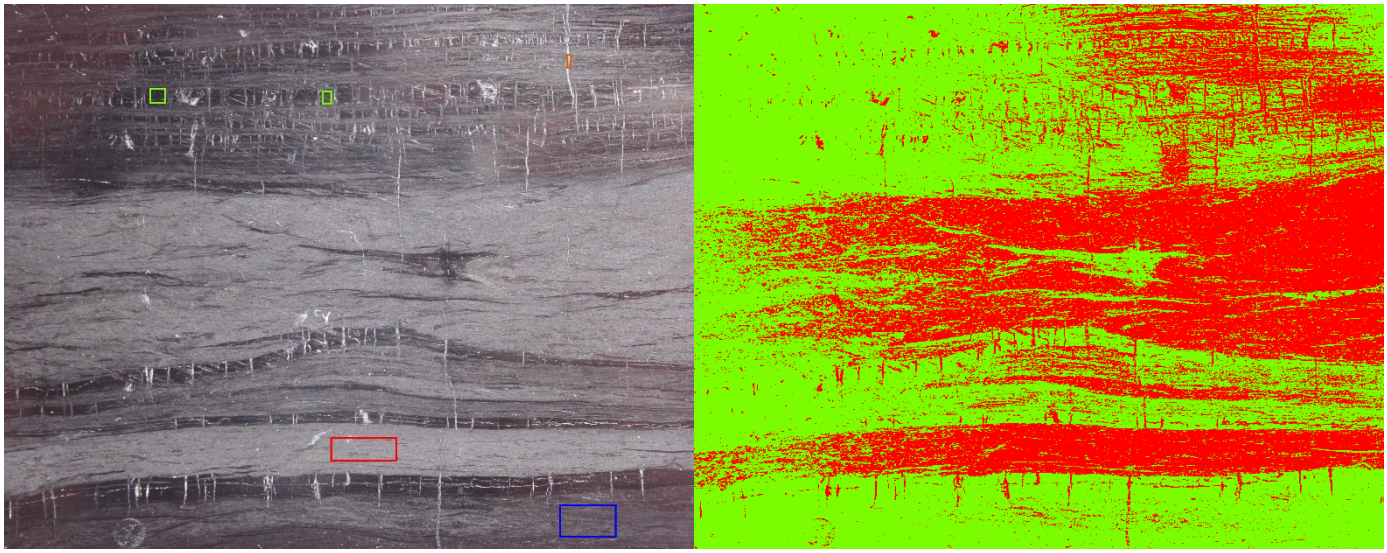


Figure 90: Sample 5 baseline segmentation D

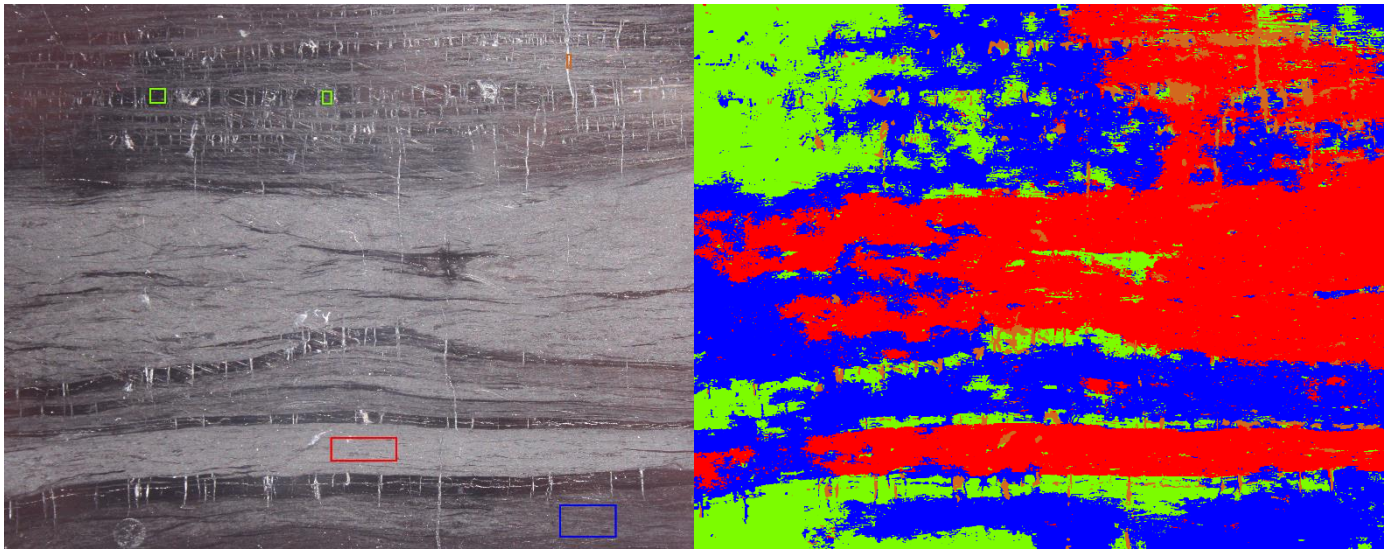


Figure 91: Sample 5 alternative segmentation

5.5.3. Evaluation of the system

The confusion matrices of the classifications obtained in the previous section are presented in the sequel. Refer to appendix C for the definitions of the abbreviations in these matrices. The alternative system consistently obtained better overall accuracies than the baseline system, or at least obtained more decent accuracies across all the model classes.

- Sample 0

Table 47: Sample 0 system test results

baseline segmentation A				
Right %	Wrong %	Predicted ID		
85.37037	14.62963	Vitrinite	Mineral	Crack
True ID	Vitrinite	98.56802	1.431981	0
	Mineral	54.71698	45.28302	0
	Crack	60	40	0

baseline system segmentation B				
Right %	Wrong %	Predicted ID		
85	15	Vitrinite	Mineral	Crack
True ID	Vitrinite	98.09069	0.71599	1.193317
	Mineral	38.67925	36.79245	24.5283
	Crack	26.66667	13.33333	60

alternative system segmentation				
Right %	Wrong %	Predicted ID		
85.18519	14.81481	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.51074	2.863962	2.625298
	Mineral	17.92453	50.9434	31.13208
	Crack	26.66667	6.666667	66.66667

- Sample 1

Table 48: Sample 1 system test results

baseline system segmentation A					
Right %	Wrong %	Predicted ID			
43.30709	56.69291	V	VPMM	M	C
True ID	V	31.39535	64.53488	3.488372	0.581395
	VpMM	34.52381	61.90476	3.571429	0
	M	19.23077	53.84615	15.38462	11.53846
	C	0	53.33333	26.66667	20

baseline system segmentation B					
Right %	Wrong %	Predicted ID			
45.93176	54.06824	V	VPMM	M	C
True ID	V	31.39535	63.37209	3.488372	1.744186
	VpMM	30.95238	65.47619	3.571429	0
	M	15.38462	50	15.38462	19.23077
	C	0	26.66667	26.66667	46.66667

alternative system segmentation					
Right %	Wrong %	Predicted ID			
54.85564	45.14436	V	VPMM	M	C
True ID	V	25	70.93023	1.744186	2.325581
	VpMM	4.761905	94.04762	0.595238	0.595238
	M	0	76.92308	15.38462	7.692308
	C	0	73.33333	0	26.66667

- Sample 2

Table 49: Sample 2 system test results

baseline system segmentation A								baseline system segmentation B							
Right %	Wrong %	Predicted ID						Right %	Wrong %	Predicted ID					
53.28283	46.71717	V	VPMM	P	CIV	C	M	55.05051	44.94949	V	VPMM	P	CIV	C	M
True ID	V	0	0	1.298701	7.792208	1.948052	1.298701	True ID	V	27.92208	27.92208	1.298701	7.142857	0.649351	1.298701
	VPMM	0	92.85714	0	3.333333	2.380952	1.428571		VPMM	18.09524	76.66667	0	1.904762	1.428571	1.904762
	P	0	6.25	56.25	0	18.75	18.75		P	0	25	56.25	0	6.25	12.5
	CIV	0	40	20	20	20	0		CIV	0	40	20	20	20	0
	C	0	50	0	0	50	0		C	25	50	0	0	25	0
	M	0	42.85714	0	0	0	57.14286		M	0	57.14286	0	0	0	42.85714

alternative system segmentation							
Right %	Wrong %	Predicted ID					
57.82828	42.17172	V	VPMM	P	CIV	C	M
True ID	V	8.441558	8.441558	0	6.493506	0.649351	3.246753
	VPMM	2.857143	95.2381	0.47619	0.47619	0.952381	0
	P	0	18.75	56.25	0	12.5	12.5
	CIV	20	20	0	40	20	0
	C	0	75	0	0	25	0
	M	0	42.85714	0	0	0	57.14286

- Sample 3

Table 50: Sample 3 system test results

baseline system segmentation A							baseline system segmentation B						
Right %	Wrong %	Predicted ID					Right %	Wrong %	Predicted ID				
50.52632	49.47368	V	I	P	Cr	Ca	46.57895	53.42105	V	I	P	Cr	Ca
True ID	V	94.70588	94.70588	0.588235	0	0	True ID	V	95.88235	95.88235	0	0	0
	I	90.47619	7.738095	1.785714	0	0		I	97.61905	1.785714	0.595238	0	0
	P	15.15152	30.30303	54.54545	0	0		P	24.24242	42.42424	33.33333	0	0
	Cr	71.42857	14.28571	14.28571	0	0		Cr	71.42857	28.57143	0	0	0
	Ca	100	0	0	0	0		Ca	100	0	0	0	0

alternative system segmentation						
Right %	Wrong %	Predicted ID				
56.31579	43.68421	V	I	P	Cr	Ca
True ID	V	45.88235	45.88235	1.176471	3.529412	8.235294
	I	1.190476	68.45238	2.380952	6.547619	21.42857
	P	0	12.12121	63.63636	0	24.24242
	Cr	0	71.42857	0	0	28.57143
	Ca	0	100	0	0	0

- Sample 4

Table 51: Sample 4 system test results

baseline system segmentation				alternative system segmentation			
Right %	Wrong %	Predicted ID		Right %	Wrong %	Predicted ID	
10.26316	89.73684	I	MPM	90.26316	9.736842	I	MPM
True ID	I	100	100	True ID	I	35.89744	35.89744
	MPM	100	0		MPM	3.519062	96.48094

- Sample 5

Table 52: Sample 5 system test results

baseline system segmentation A						baseline system segmentation B					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
52.95699	47.04301	I	V	C	MPM	56.45161	43.54839	I	V	C	MPM
True ID	I	0	59.61538	0	40.38462	True ID	I	4.807692	47.11538	0	48.07692
	V	0	84.61538	0	15.38462		V	11.53846	75.96154	0	12.5
	C	0	25	0	75		C	25	16.66667	0	58.33333
	MPM	0	28.28947	0	71.71053		MPM	2.631579	14.47368	0	82.89474

baseline system segmentation C						baseline system segmentation D					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
50.26882	49.73118	I	V	C	MPM	50.26882	49.73118	I	V	C	MPM
True ID	I	0	66.34615	0	33.65385	True ID	I	0	67.30769	0	32.69231
	V	0	90.38462	0	9.615385		V	0	91.34615	0	8.653846
	C	0	41.66667	0	58.33333		C	0	33.33333	0	66.66667
	MPM	0	38.81579	0	61.18421		MPM	0	39.47368	0	60.52632

alternative system segmentation					
Right %	Wrong %	Predicted ID			
55.10753	44.89247	I	V	C	MPM
True ID	I	48.07692	12.5	1.923077	37.5
	V	34.61538	50.96154	6.730769	7.692308
	C	16.66667	16.66667	50	16.66667
	MPM	34.86842	1.315789	0.657895	63.15789

To summarise the findings of the system test the following points are presented:

- The alternative system is less biased towards individual model classes and can, therefore, more reliably classify larger amounts of classes.
- The spatial coherence of the segmentations produced by the alternative system increases the system accuracy.
- The baseline system can converge to arbitrarily poor solutions while the alternative system always produces a deterministic solution.
- The overall accuracy of the alternative system is not much higher than that of the baseline system.
- There are certain abnormalities in the sample images that reduce the accuracy of both the systems. However, the alternative system can compensate for some of them using spatial information.

6. DISCUSSION

6.1. Feature extraction discussion

During the feature extraction test it was found that using the local image entropy to estimate regularity differentiates between the model parameters better than the singular value decomposition used by the original system. There is an additional advantage to using this alternative. The roughness and regularity of the original system correlated extremely well. This might seem like a desirable aspect of features in general, however this is not the case within a feature space.

If the data point values with regards to two feature dimensions are too similar, one of the features is redundant. The effect of having two identical features in a feature space would be that they act as a single feature with double the importance during distance considerations. This can cause misperceptions about the true level of similarity between data points. Figure 92 shows the baseline roughness and baseline regularity distributions for sample 1. Notice that the feature distributions are extremely similar.

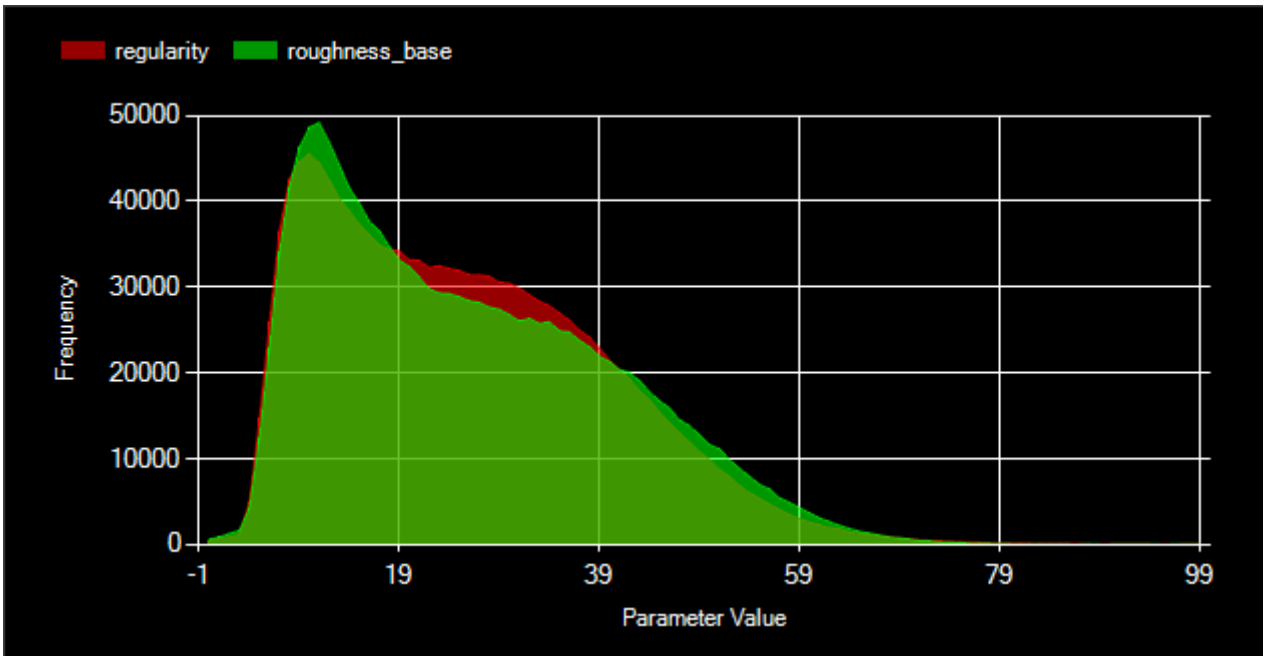


Figure 92: Baseline roughness compared to baseline regularity

It was decided to use the Gaussian filter in the roughness estimate as opposed to the median filter. This decision was made empirically, however there is an intuition behind using the Gaussian filter to sample the local neighbourhoods. Using a square sample window typically results in artefacts in the resulting data. However, the Gaussian kernel uses a weighted average of pixels in the sample window, and the effect of the square window will, therefore, be less intense.

While the alternative feature values do differentiate between the model values better than the original features, they are still ineffective in differentiating between some of the minerals in the sample images. A good example of such models is the “Cracks” and “Calcite veins” in sample 3. Figure 93 provides radar charts of the mineral models in sample 3, with accompanying model area selections. The selected areas are the exact areas that have been identified by the geologists in figure 45. Note the high level of similarity between the two right-most models.

Regardless of the efficacy of the classifier used to assign the data points to these models, if two models are almost identical, there will be fallacious classifications. This does not seem like an issue that can be solved using the current system of classification. Insufficient information is available about nuances regarding the spatial relations between mineral materials. The system will not be able to differentiate between a crack in vitrinite and a crack in inertinite, or a small crack and a small calcite vein.

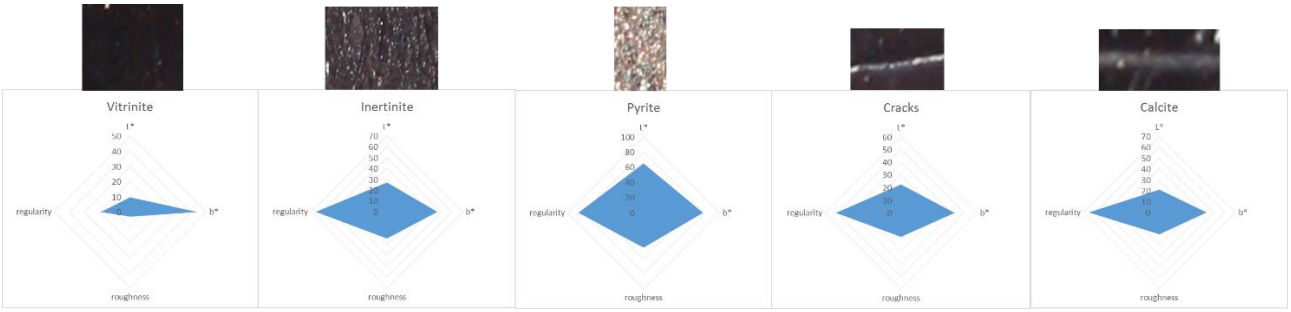


Figure 93: Problematic minerals

In addition to the uncertainty in differentiating between some of the minerals in the coal there is also a glaring issue with the types of images that are to be classified. The presence of blurring, excessive light reflection, and poor photo quality will have to be reduced if this technology is ever to reach an acceptable level of accuracy for all samples. Refer to figure 94. This figure is an annotated version of sample 1.

In the region indicated by the red circle there is a reddish hue present in the photograph. That region is largely comprised of vitrinite, but the presence of the anomalous red pigment makes that region very distinct from the standard regions of vitrinite in this photograph. This could only be compensated for by selecting that region as an additional model for vitrinite in the alternative system.

The two regions that are indicated by the green circles contain very similar mineral components. However, due to the large amount of blurring in the upper half of the image the top-most region appears much smoother and regular in the feature space. The effect of this is also observable when comparing the two blue circles. The “Vitrinite plus mineral matter” in the top-most circle resembles normal “vitrinite” due to the blurring effect.

The feature extraction test was conducted with sampled model values from the hand classifications obtained from the geologists. This method of obtaining ground truth values was sufficient for the tests done in this dissertation. However, some of the ground truth data sets contained very few model values for some of the minerals. One such a case is in sample 3 where there are only two calcite data points in the ground truth data set. The lack of information about the calcite mineral in this sample limits the certainty with which the tests can determine the optimal features for that mineral.

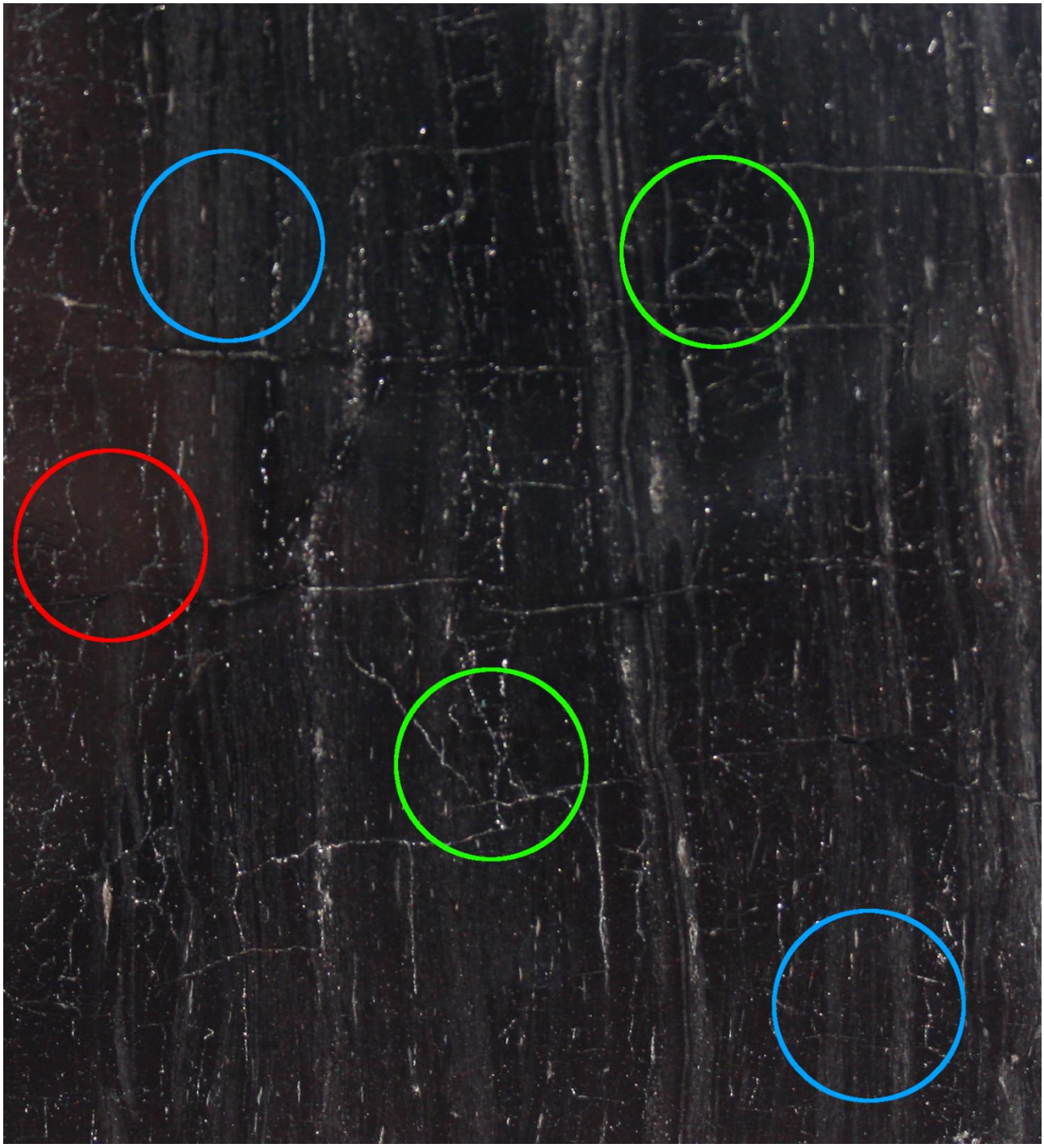


Figure 94: Issues with photo quality

6.2. Clustering discussion

The initial four clustering algorithms were:

- k-means
- k-medians
- k-medoids
- Mean shift
- Particle swarm optimisation (PSO)

The k-medoids algorithm proved to be much too computationally complex to be a viable choice. During the clustering test it was found that the PSO algorithm separates clusters much more efficiently than the k-means or k-median algorithms. Overall, it appeared to be the best method of clustering the datasets. The k-means algorithm rated consistently in this test, only excelling with regards to the average silhouette values. It was expected that the PSO- or k-means algorithm would result in the best classification. However, in the final assignments test it was observed that the most efficient clustering algorithms do not, necessarily, result in a superior classifier. This indicates that the spatial information in the image is more important than initially predicted.

6.3. Models and final assignments discussion

The model parameter test determined that using the mean feature value, in the user selected areas, as the model value, approximates the ground truth feature values better than the median value. However, the difference was slight, and it seems that the actual region that is selected as model is much more important than the parameter used to approximate it.

In the final assignments test 32 different classifiers were compared by comparing their results with the ground truth model values. No classifier performed exceptionally well across all the sample images. However, the classifiers that incorporated spatial information by assigning models to regions instead of clusters performed better in general. This lead to the decision to use the mean shift algorithm, which makes extensive use of spatial information. The method of assigning models to clusters that was decided upon was alternative E (assigning models to region, and using multiple areas as models).

6.4. The system discussion

The alternative system performed better than the baseline system in general. As expected, the resulting image segmentations were, spatially, much more coherent. There was also an increase in the “spread” of classifications. By this is meant that the baseline system tends to assign a large amount of data points to one or two classes, somewhat neglecting the other classes, where the alternative system appears to assign the data points over a wider range of classes while still maintaining a decent amount of accuracy across all classes. The use of a non-stochastic algorithm, such as the mean shift algorithm has the added improvement of consistent results. The results are deterministic and there is limited probability of a random poor classification, which is sometimes the case with the k-means algorithm.

The apparent importance of spatial information and the abandonment of the classical clustering algorithms in the final assignments test promotes a different approach to the image segmentation problem. It is clear from these results, that if a higher level of accuracy is to be achieved, without calibrating the system to every sample image individually, a method of classification that emphasises spatial information and the relation between minerals in the coal ore is necessary.

7. CONCLUSION

The research presented in this document was done with the goal of developing a system that could classify a digital image of a raw coal ore sample into the constituent mineral components. It was motivated that, in practical terms, this amounts to an image segmentation problem. An existing system was investigated and implemented independently. Thereafter a literature survey was done in order to identify alternative techniques that can be used to address the research problem. It was found that the existing classifier is well suited to the problem and the subcomponents were investigated to determine if the performance can be improved by alternatives in the literature.

Several alternatives were implemented and tested to empirically rate their performances. The top rated alternative components were combined and a full system test was done to compare the new system to the existing one. The newer system achieved slightly higher accuracies than the existing one, but the most advantageous characteristic was that it classified the mineral components more consistently regardless of the unevenness of the model distributions in the sample data sets. This was achieved by an emphasis on spatial information in combination with range information.

Internal statistical analyses and performance evaluations were done as a means of verification. The alternative subcomponents were compared to the baseline components in terms of metrics such as distribution comparisons and hypothesis testing for the alternative feature space and cluster evaluation metrics for the clustering algorithms. Validation was done by comparing the classification results to ground truth values and the investigation of the resulting confusion matrices.

The ultimate result of the research was an alternative system that performed better than the baseline system in terms of a set of sample images. The alternative system is superior in general, however, certain shortcomings were identified that prevented it from achieving the accuracies needed to be called absolutely superior. These shortcomings are inherent to the existing system as well, and an entirely different approach to the classification problem is needed to further increase the system's accuracy and reliability. A major limiting factor of the research was the lack of reliable training data. This refers to the scarcity of some of the data points used to determine the model values in the ground truth data set.

Nevertheless, as the problem statement in chapter 1 states, the goal was to improve upon the existing system. This was achieved and proven to be true. If the system is to be improved any further it is advised to:

- Obtain more training data such as the ideal data set described in section 4.1.2. This will enable more accurate testing and rating of the algorithms in the system.
- Conduct an extensive investigation into the nuances related to the relationship between minerals in the coal ore, and determine how to incorporate this information into the per-point feature values.
- Ensure that the training data contains no problematic artefacts such as those described in section 6.1.
- Look into the possibility of using deep learning techniques to determine the underlying correlations between the feature values and the mineral components. However, this would require vast amounts of labelled training data.

REFERENCES

[1] P. v. Vuuren, “Using visual texture analysis to classify raw coal,” *International Conference on Systems, Signals and Image Processing (IWSSIP),* pp. 212-215, 2015.

[2] U. E. I. Administration, “International Energy Outlook 2016,” U.S. Energy Information Administration, Washington, 2016.

[3] I. E. Agency, “World energy outlook 2016,” International Energy Agency, France, 2016.

[4] I. E. Agency, “Key world energy statistics 2016,” International Energy Agency, France, 2016.

[5] I. E. Agency, “IEA Atlas of Energy,” International Energy Agency, 2017. [Online]. Available: http://energyatlas.iea.org/#!/tellmap/-1002896040. [Accessed 19 August 2017].

[6] C. Fu, R. Anantharaman, K. Jordal and T. Gundersen, “Thermal efficiency of coal-fired power plants: From theoretical to practical assessments,” *Energy conversion and management,* vol. 105, pp. 530-544, 2015.

[7] Q. Campbell, M. I. Roux and I. Smith, “Water-only laboratory coal fractionation using the reflux classifier,” *Minerals Engineering,* vol. 83, pp. 59-63, 2015.

[8] K. Galvin, “Options for washability analysis of Coal-A literature review,” *International Journal of Coal Preparation and Utilization,* vol. 26, no. 4, pp. 209-234, 2006.

[9] C. Lin, J. Miller, G. Luttrill and G. Adel, “On-line washability analysis for the control of coarse coal cleaning circuits,” *High efficiency coal preparation: An international symposium, Society for Mining Metallurgy,* pp. 369-378, 1995.

[10] Z. Ze-lin, Y. Jian-guo, W. Yu-ling, X. Wen-Cheng and L. Xiang-yang, “A study on Fast Predicting the Washability Curve of Coal,” *Procedia Environmental Sciences,* vol. 11, pp. 1580-1584, 2011.

[11] Z. Zhang, J. Yang, L. Ding and Y. Zhao, “Estimation of coal particle size distribution by image segmentation,” *International journal of mining science and technology,* vol. 22, no. 5, p. 739–744, 2012.

[12] Z. Zhang, J. Yang, Y. Wang, D. Dou and a. W. Xia, “Ash content prediction of coarse coal by image analysis and ga-svm,” *Powder technology,* vol. 268, p. 429–435, 2014.

[13] H. C. Dorland, Q. P. Campbell, M. L. Roux, K. McMillan, M. I. Dorland, P. Erasmus and N. Wagner, “Simulation of washability and liberation information from photographs,” *XVIII International Coal Preparation Congress,* pp. 249-254, 2016.

[14] N. R. Pal and S. K. Pal, “A review on image segmentation techniques,” *Pattern Recognition,* vol. 26, no. 9, pp. 1277-1294, 1993.

[15] A. M. Khan and R. S, “Image Segmentation Methods: A Comparative Study,” *International Journal of Soft Computing and Engineering,* vol. 3, no. 4, pp. 2231-2307, 2013.

[16] L. Lucchese and S. Mitra, “Colour image segmentation: A state-of-the-art survey,” *PINSA,* vol. 67, no. 2, pp. 207-221, 2001.

[17] N. Sharma, M. Mishra and M. Shrivastava, “Colour image segmentation techniques and issues: An approach,” *International Journal of Scientific & Technology Research,* vol. 1, no. 4, pp. 2277-8616, 2012.

[18] M. G. H. Omran, “Particle Swarm Optimization Methods for Pattern Recognition and Image Processing,” Pretoria, 2004.

[19] R. H. Turi, “Clustering-Based Colour Image Segmentation,” Australia, 2001.

[20] K. Fu and J. Mui, “A survey on image segmentation,” *Pattern Recognition,* vol. 13, no. 1, pp. 3-16, 1981.

[21] H. Cheng, X. Jiang, Y. Sun and J. Wang, “Color image segmentation: advances and prospects,” *Pattern Recognition,* vol. 34, p. 2259}2281, 2001.

[22] M. Sezgin and B. Sankur, “Survey over image thresholding techniques and quantitative performance evaluation,” *Journal of electronic imaging,* vol. 13, no. 1, p. 146–165, 2004.

[23] M. I. Sezan, “A peak Detection Algorithm and Its Application to Histogram-Based Image data Reduction,” *Computer vision, graphics, and image processing,* vol. 49, pp. 36-51, 1990.

[24] Y. Zhang and L. Wu, “Optimal Multi-Level Thresholding Based on Maximum Tsallis Entropy via an Artificial Bee Colony Approach,” *Entropy,* vol. 13, pp. 841-859, 2011.

[25] A. Guida, A. W. Nienow and M. Barigou, “Shannon entropy for local and global description of mixing by Lagrangian particle tracking,” *Chemical Engineering Science,* vol. 65, pp. 2865-2883, 2010.

[26] S. K. Mitra, “Entropy,” in *Digital signal processing: A computer based approach*, Singapore, McGraw-Hill Education, 2006, pp. 901-902.

[27] L. Hertz and R. W. Schafer, “Multilevel thresholding Using Edge Matching,” *Computer vision, graphics, and image processing,* vol. 44, pp. 279-295, 1988.

[28] R. L. Kirby and A. Rosenfeld, “A note on the used of (Gray level, local average gray level) space as an aid in threshold selection),” *IEEE transactions on systems, man, and cybernetics,* vol. 9, no. 12, pp. 860-864, 1979.

[29] A. S. Abutaleb, “Automatic thresholding of gray-level pictures using two-dimensional entropy,” *Computer vision, graphics, and image processing,* vol. 47, pp. 22-32, 1989.

[30] A. K. Chaubey, “Comparison of The Local and Global Thresholding Methods in Image Segmentation,” *World Journal of Research and Review,* vol. 2, no. 1, pp. 01-04, 2016.

[31] W. Niblack, in *An introduction to Digital Image Processing*, Prentice Hall, 1986, pp. 115-116.

[32] T. Uchiyama and M. A. Arbib, “Color Image Segmentation Using Competitive Learning,” *IEEE Transactions on pattern analysis and machine intelligence,* vol. 16, no. 12, pp. 1197-1206, 1994.

[33] Z. Ye and H. Mohamadian, “Intelligent control based multilevel and multiband digital image thresholding using fuzzy entropy and particle swarm optimization,” *International Journal of Advanced Computational Engineering and Networking,* vol. 3, no. 7, pp. 24-30, 2015.

[34] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[35] R. Muthikrishnan and M. Radha, “Edge detection techniques for image segmentation,” *International journal of computer science & information technology,* vol. 3, no. 6, pp. 259-267, 2011.

[36] L. S. Davis, “A survey of edge detection techniques,” *Computer graphics and image processing,* vol. 4, pp. 248-270, 1975.

[37] L. Roberts, “Machine perception of three-dimensional solids,” *Optical and Electro-Optical Information Processing,* pp. 159-197, 1965.

[38] J. Prewitt, “Object enhancement and extraction,” *Picture Processing and Psychopictorics,* pp. 75-149, 1970.

[39] D. Marr and E. Hildreth, “Theory of edge detection,” *Proceedings of the Royal Society of London. Series B, Biological Sciences,* vol. 207, no. 1167, pp. 187-217, 1980.

[40] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vols. PAMI-8, no. 6, pp. 679-698, 1986.

[41] B. Wang and S. Fan, “An improved Canny edge detection algorithm,” in *2009 Second International Workshop on Computer Science and Engineering*, Qingdao, China, 2009.

[42] D. Antolovic, “Review of the Hough Transform Method, With an Implementation of the Fast Hough Variant for Line Detection,” Department of Computer Science, Indiana University,and IBM Corparation, Bloomington, Indiana, 2008.

[43] R. Nevatia, “A Color Edge Detector and Its Use in Scene Segmentation,” *IEEE transactions on systems, man, and cybernetics,* Vols. SMC-7, no. 11, pp. 820-826, 1977.

[44] R. Adams and L. Bischof, “Seeded Region Growing,” *IEEE Transactions on pattern analysis and machine intelligence,* vol. 16, no. 6, pp. 641-647, 1994.

[45] M. M. S. J. Preetha, L. P. Suresh and M. J. Bosco, “Image segmentation using seeded region growing,” in *IEEE*, Kumaracoil, India, 2012.

[46] Z. Lin, J. Jin and H. Talbot, “Unseeded region growing for 3D image segmentation,” *VIP '00 Selected papers from the Pan-Sydney workshop on Visualisation,* vol. 2, pp. 31-37, 2000.

[47] A. Jain, M. Murty and P. Flynn, “Data Clustering: A Review,” *ACM COMPUTING SURVEYS,* vol. 31, no. 3, pp. 264-323, 1999.

[48] C. J. Veenman, M. J. Reinders and E. Backer, “A Cellular Coevolutionary Algorithm for Image Segmentation,” *IEEE TRANSACTIONS ON IMAGE PROCESSING,* vol. 12, no. 3, pp. 304-316, 2003.

[49] S. Ray, R. H. Turi and P. E. Tischer, “Clustering-based colour image segmentation: An evaluation study,” in *Proceedings of the DICTA-95 conference Brisbane*, Brisbane, 1995.

[50] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient graph-based image segmentation,” *International Journal of Computer Vision,* vol. 59, no. 2, pp. 167-181, 2004.

[51] T. M. Nquyen, W. J. Wu and S. Ahuja, “An Extension of the Standard Mixture Model for Image Segmentation,” *IEEE TRANSACTIONS ON NEURAL NETWORKS,* vol. 21, no. 8, pp. 1326-1338, 2010.

[52] Y. Cheng, “Mean shift, Mode Seeking, and Clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 17, no. 8, pp. 790-799, 1995.

[53] D. Arthur and S. Vassilvitskii, “K-Means++: The Advantages of Careful Seeding,” in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, Louisiana, 2007.

[54] D. Comaniciu and P. Meer, “Mean Shift Analysis and Applications,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Kerkyra, Greece, 1999.

[55] H. Frigui and R. Krishnapuram, “A Robust Competitive Clustering Algorithm With Applications in Computer Vision,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,* vol. 21, no. 5, pp. 450-465, 1999.

[56] C. Klein and B. Dutrow, in *Mineral Science*, New Mexico, Luisiana, Wiley, 2007, pp. 19-36.

[57] A. Ford and A. Roberts, “Colour Space Conversions,” Westminster University, 1998.

[58] K. I. Laws, “Textured image segmentation,” University of Southern California, Los Angeles, California, 1980.

[59] M. S. Landy and N. Graham, “Visual Perception of Texture,” *The visual neurosciences,* pp. 1106-1118, 2004.

[60] A. Kassner and R. Thornhill, “Texture Analysis: A Review of Neurologic MR Imaging Applications,” *American Journal of Neurology,* vol. 31, no. 5, pp. 809-816, 2010.

[61] R. Gonzalez and R. Woods, in *Digital Image Processing*, Addison Wesley, 1993, pp. 507-512.

[62] N. Bhushan, A. Rao and G. Lohse, “The texture lexicon: unerstanding the categorization of visual texture terms and their relationship to texture images,” *Cognitive Science,* vol. 21, no. 2, pp. 219-246, 1997.

[63] X. Guo, C. M. Asano, A. Asano and T. Kurita, “Visual Complexity Perception and Texture Image Characteristics,” in *2011 International Conference on Biometrics and Kansei Engineering*, Takamatsu, Kagawa, Japan, 2011.

[64] R. A. Sadek, “SVD Based Image Processing Applications: State of The Art, Contributions and Research Challenges,” *International Journal of Advanced Computer Science and Applications,* vol. 3, no. 7, pp. 26-34, 2012.

[65] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu, “An Efficient k-Means Clustering Algorithm: Analysis and Implementation,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,* vol. 24, no. 7, pp. 881-892, 2002.

[66] G. Hamerly and C. Elkan, “Alternatives to the k-means algorithm,” in *Proceedings of the eleventh international conference on Information and knowledge management*, New York, 2002.

[67] G. J. Hamerly, “Learning structure and concepts in data through data clustering,” UNIVERSITY OF CALIFORNIA, SAN DIEGO, San Diego, California, 2003.

[68] M. Halkidi, Y. Batistakis and M. Vazirgiannis, “On Clustering Validation Techniques,” *Journal of Intelligent Information Systems,* vol. 17, no. 2/3, pp. 107-145, 2001.

[69] L. Kaufman and P. J. Rousseeuw, “Clustering by means of medoids,” *Statistical Data Analysis Based on the L1–Norm and Related Methods,* pp. 405-416, 1987.

[70] H.-S. Park and C.-H. Jun, “A simple and fast algorithm for K-medoids clustering,” *Expert Systems with Applications,* vol. 36, no. 2, pp. 3336-3341, 2009.

[71] S. Har-Peled and A. Kushal, “Smaller Coresets for k-Median and k-Means Clustering,” *Discrete & Computational Geometry,* vol. 37, no. 1, pp. 3-19, 2007.

[72] Y. Ding, X. Dang, H. Peng and D. Wilkins, “Robust clustering in high dimensional data using statistical depths,” *Proceedings of the Fourth Annual MCBIOS Conference. Computational Frontiers in Biomedicine,* vol. 8, pp. 1-16, 2007.

[73] D. Arthur and S. Vassilvitskii, “k-means++: The Advantages of Careful Seeding,” in *SODA '07 Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* , New Orleans, Louisiana, 2007.

[74] A. Esmin, D. Pereira and F. d. Araujo, “Study of different approach to clustering data by using the Particle Swarm Optimization Algorithm,” in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)*, Hong Kong, China, 2008.

[75] F. v. d. Bergh and A. Engelbrecht, “A New Locally Convergent Particle Swarm Optimiser,” in *2002 IEEE International Conference on Systems, Man and Cybernetics*, Yasmine Hammamet, Tunisia, Tunisia, 2002.

[76] F. v. d. Bergh, “An Analysis of Particle Swarm Optimizers,” University of Pretoria, Pretoria, 2001.

[77] J. Kennedy, “Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance,” in *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99*, Washington, DC, USA, USA , 1999.

[78] J. Kennedy and R. Mendes, “Population structure and particle swarm performance,” in *Proceedings of the 2002 Congress on Evolutionary Computation, 2002. CEC '02*, Honolulu, HI, USA, USA , 2002.

[79] K. Fukunaga and L. D. Hostetler, “The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition,” *IEEE Transactions on information theory,* vol. 21, no. 1, pp. 32-40, 1975.

[80] D. Comaniciu and P. Meer, “Mean Shift: A Robust Approach Toward Feature Space Analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, no. 5, pp. 603-619, 2002.

[81] B. Georgescu, I. Shimshoni and P. Meer, “Mean Shift Based Clustering in High Dimensions: A Texture Classification Example,” in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, Nice, France, France, 2003.

[82] W. Tao, H. Jin and Y. Zhang, “Color Image Segmentation Based on Mean Shift and Normalized Cuts,” *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS,* vol. 37, no. 5, pp. 1382-1389, 2007.

[83] D. Bouldin and D. L. Davies, “A Cluster Separation Measure,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 224-227, 1979.

[84] J. Dunn, “Well-Separated Clusters and Optimal Fuzzy Partitions,” *Journal of Cybernetics,* vol. 4, no. 1, pp. 95-104, 1974.

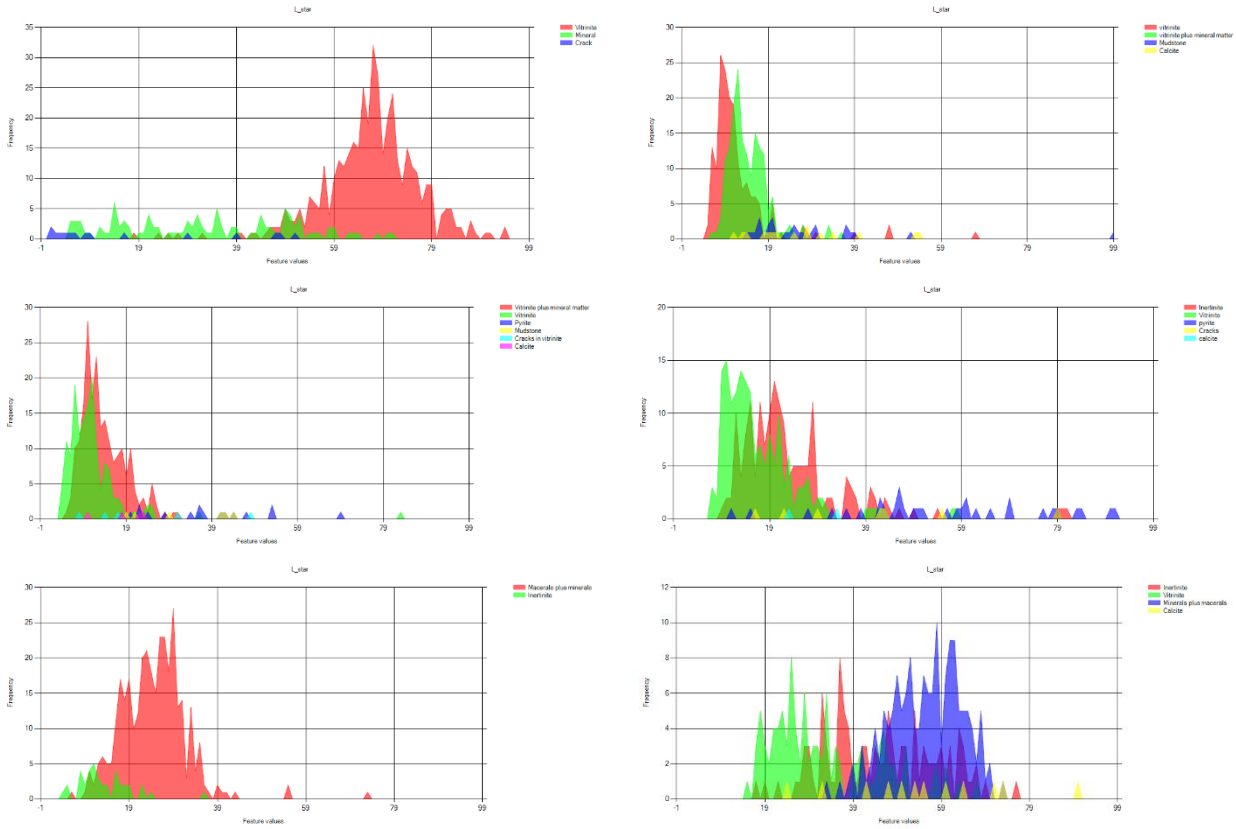
[85] P. J. Rousseeuw, “Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis,” *Computational and Applied Mathematics,* vol. 20, pp. 53-65, 1987.

APPENDIX

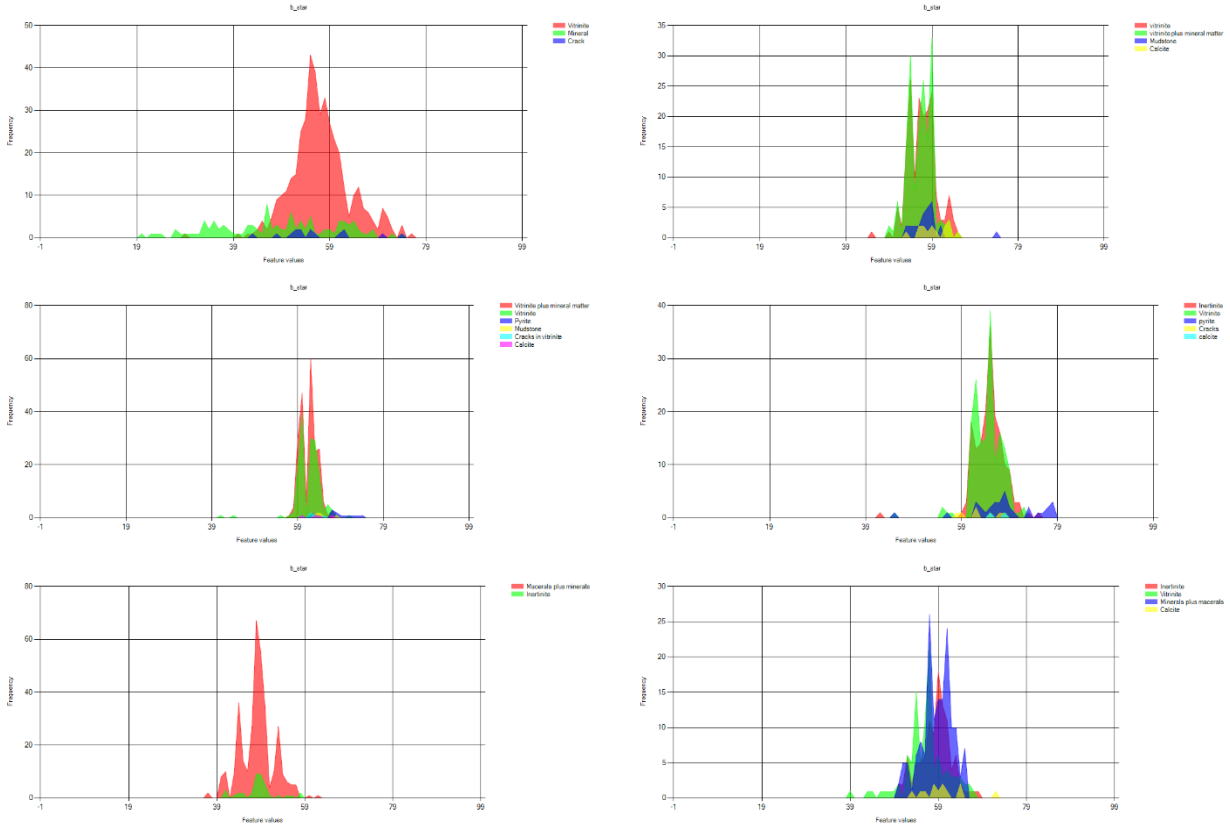
A. Model feature value distributions

The following histograms represent the feature distributions of the ground truths, in terms of each sample image. For each feature value, six histograms are provided (one for each sample image). The figures are ordered from the top-left to the bottom-right, and from sample 0 to sample 5.

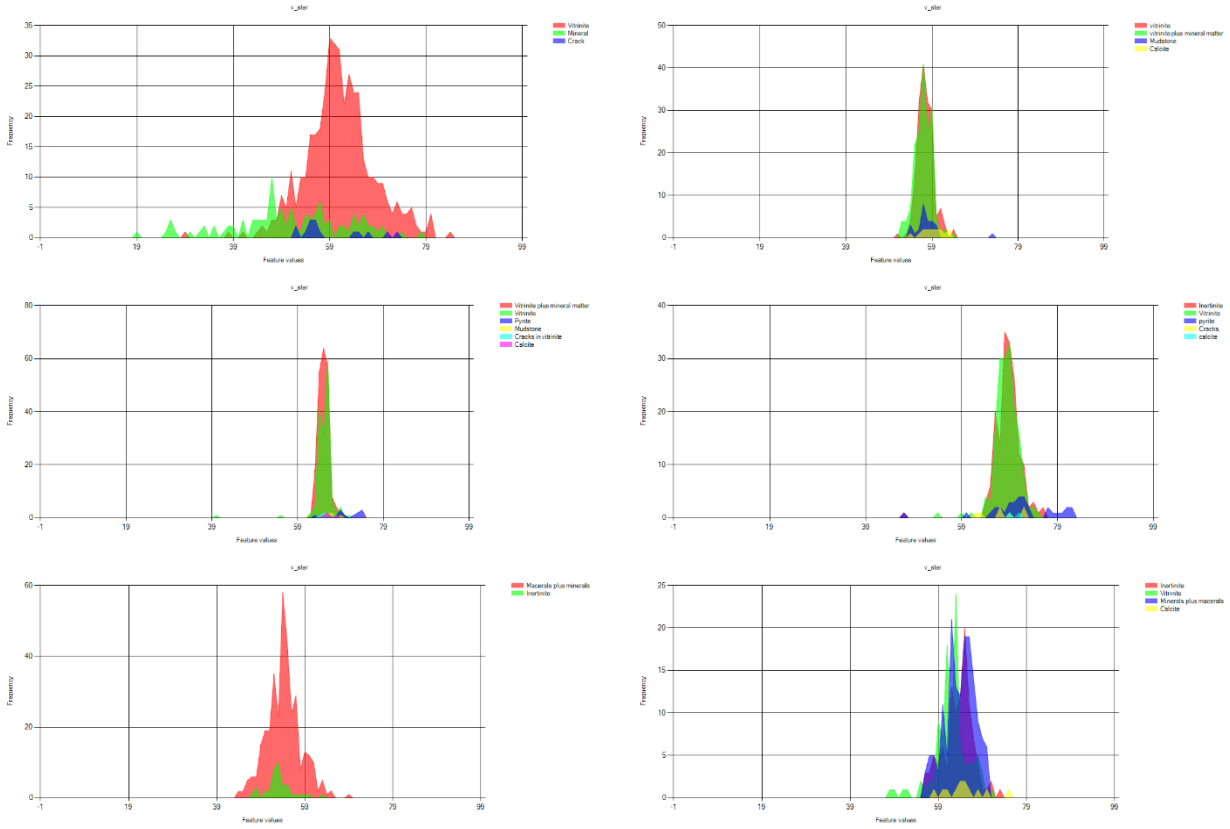
L* feature



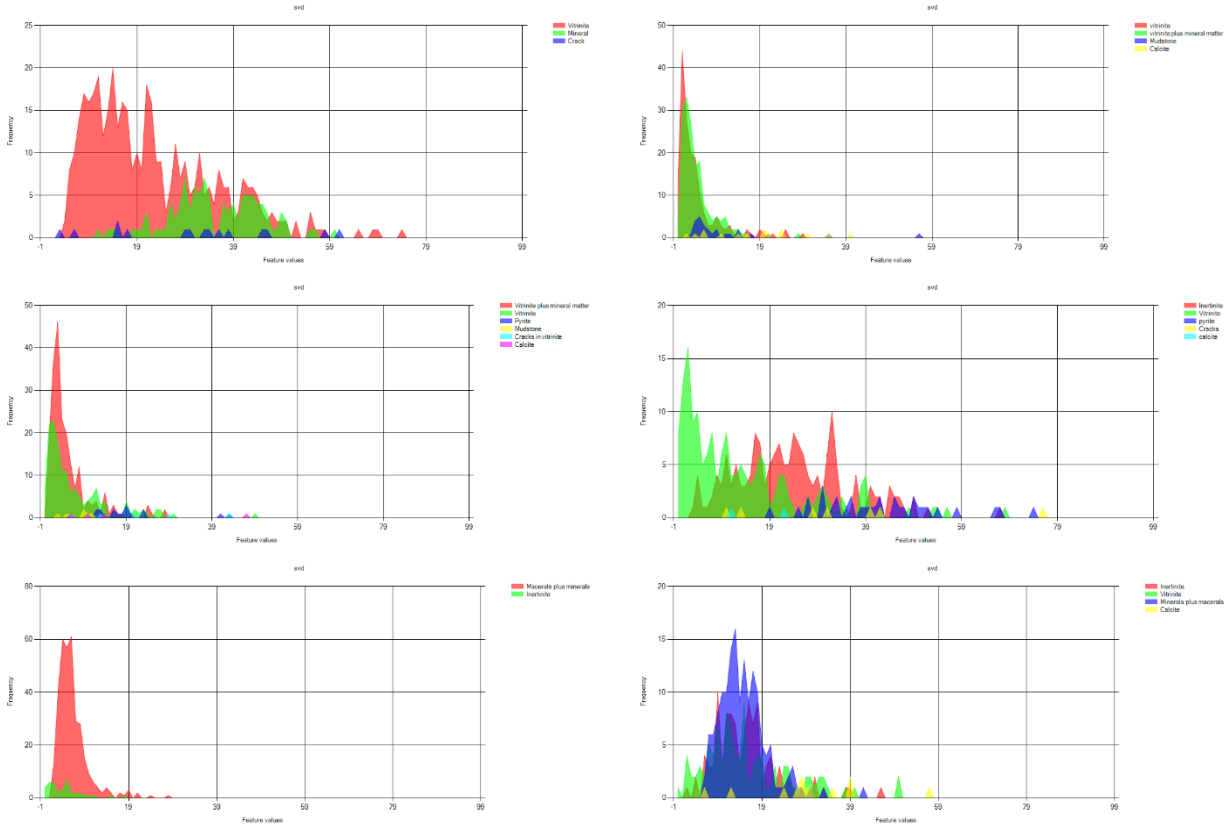
b* feature



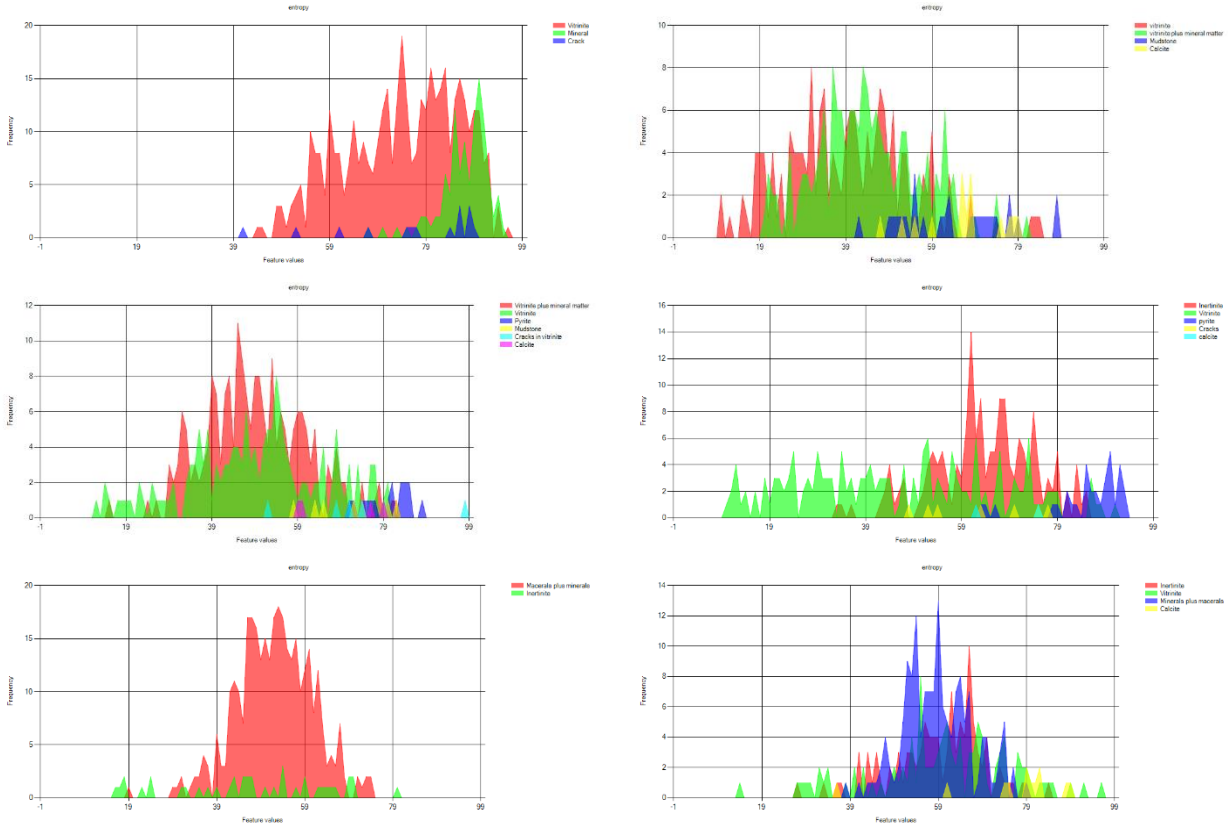
v* feature



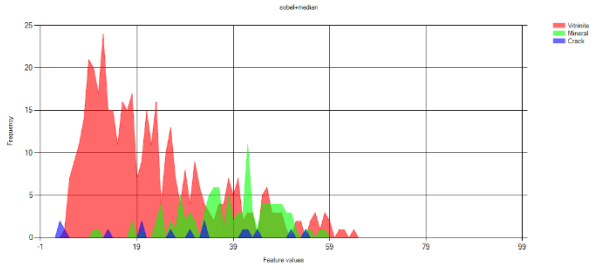
SVD feature

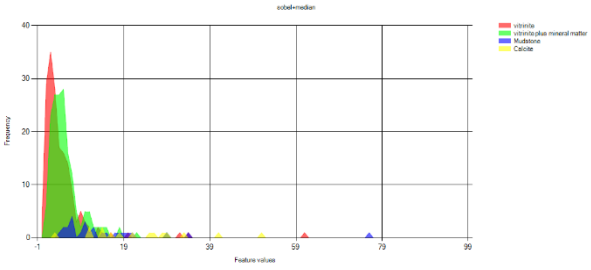


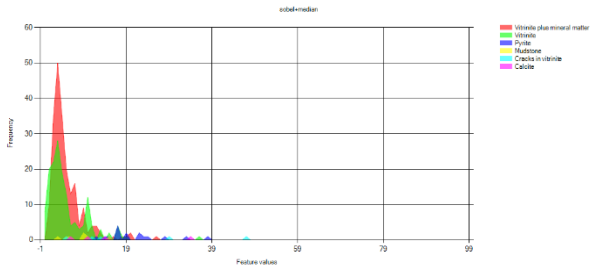
Entropy feature

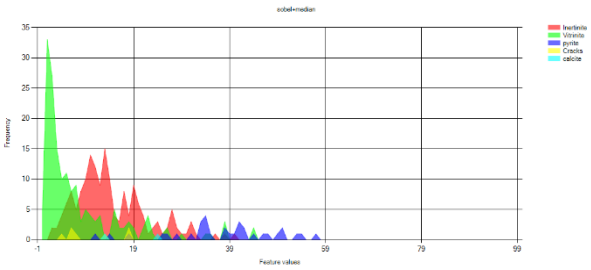


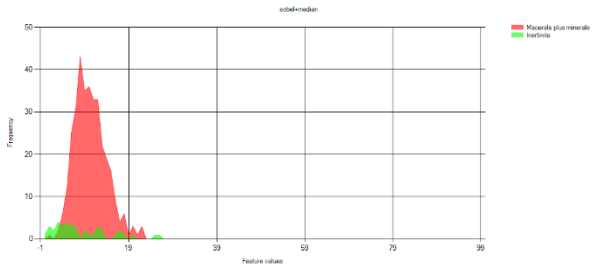
Sobel + median feature

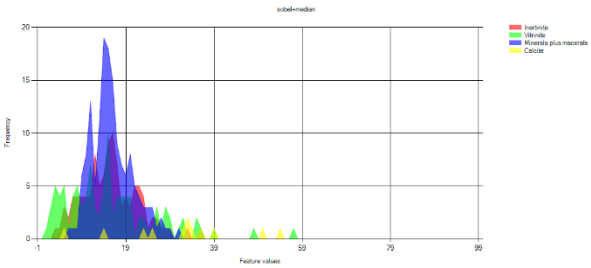




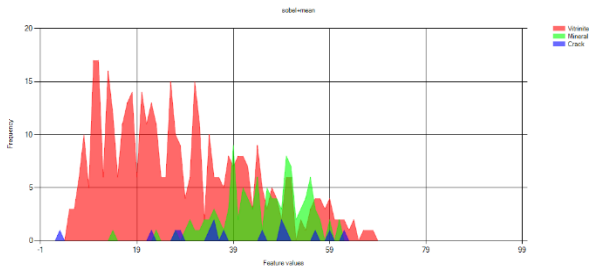


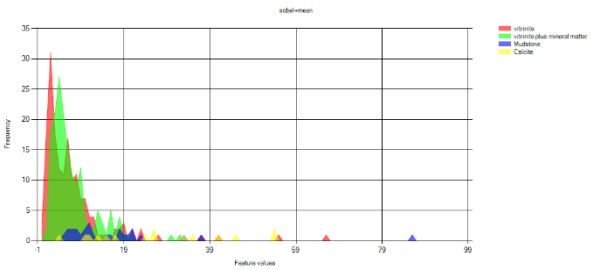


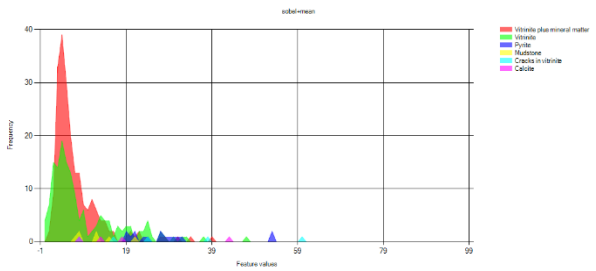


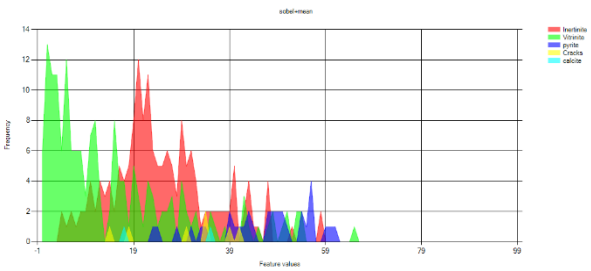


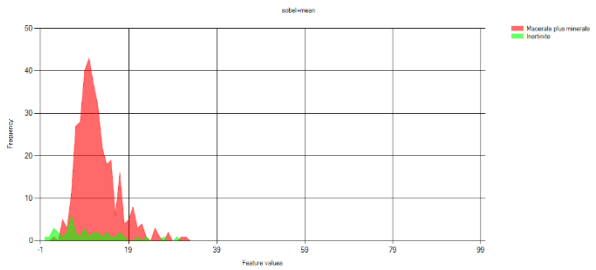
Sobel + mean feature

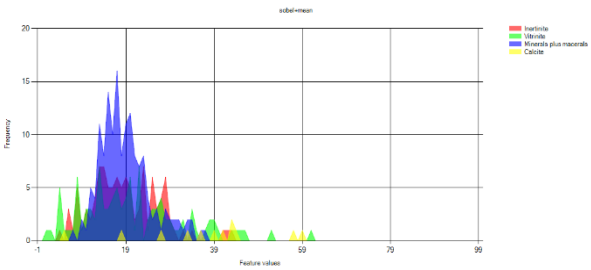




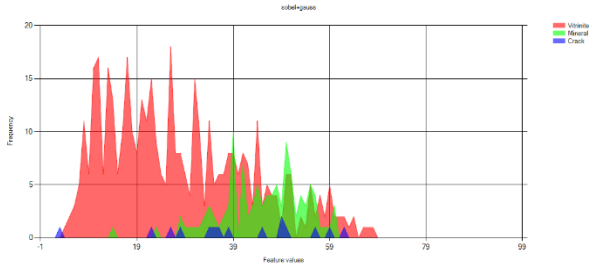


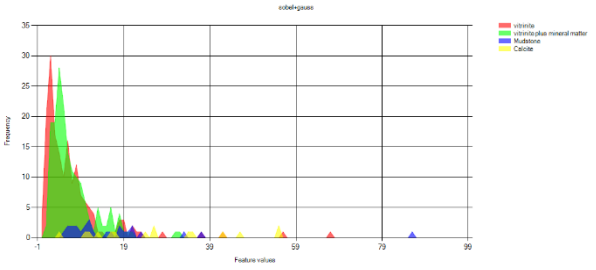


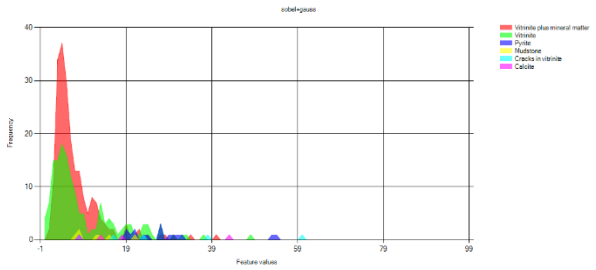


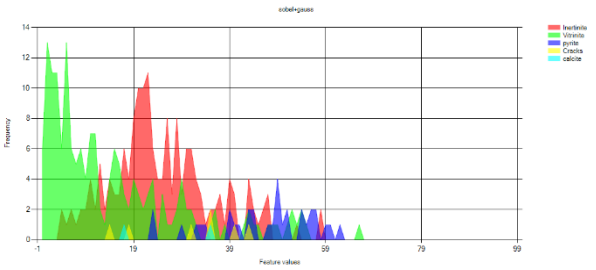


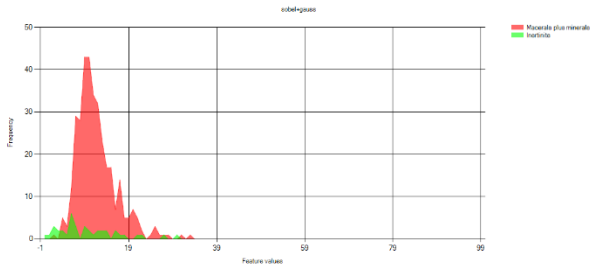
Sobel + gauss feature

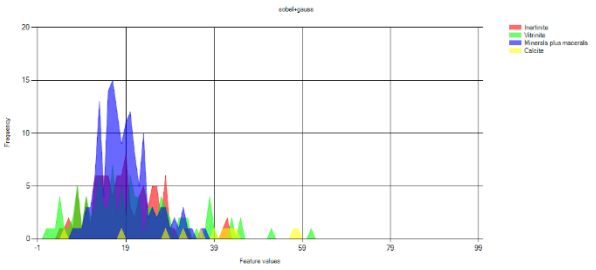




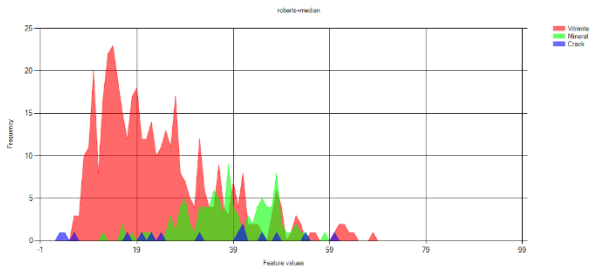


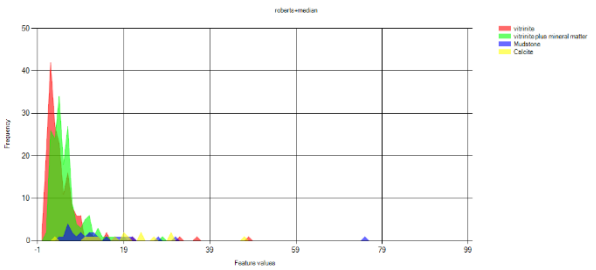


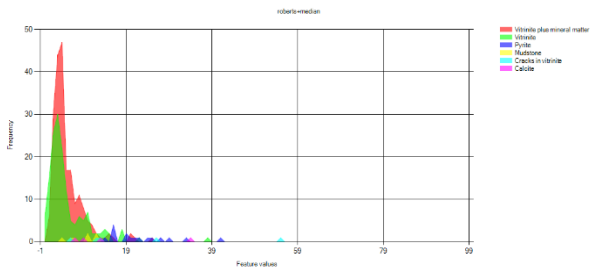


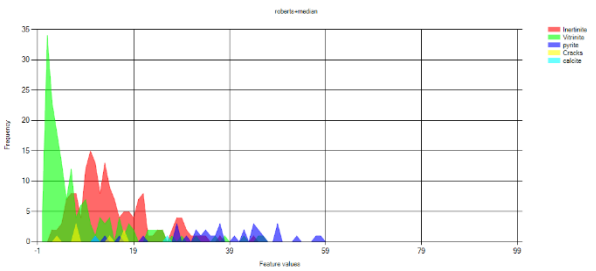


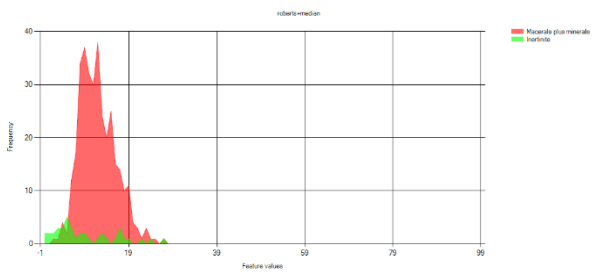
Roberts + median feature

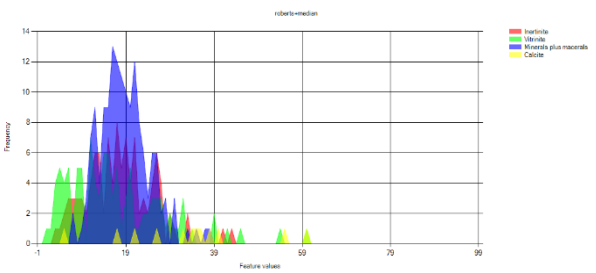




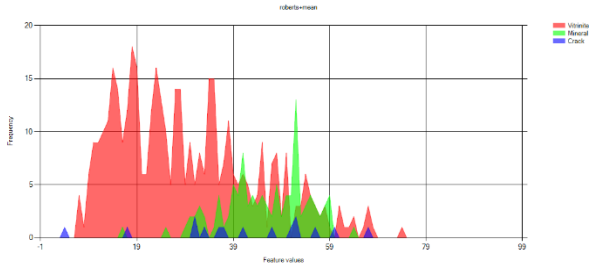


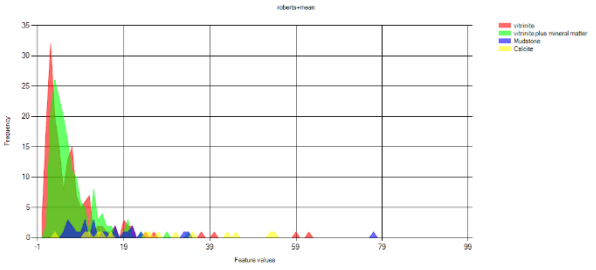


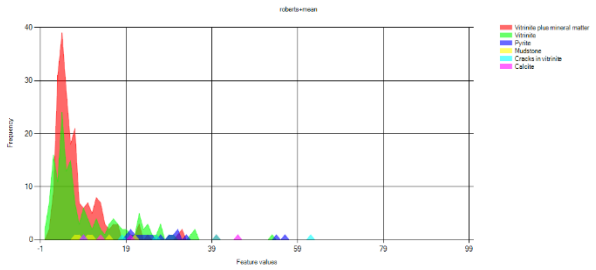


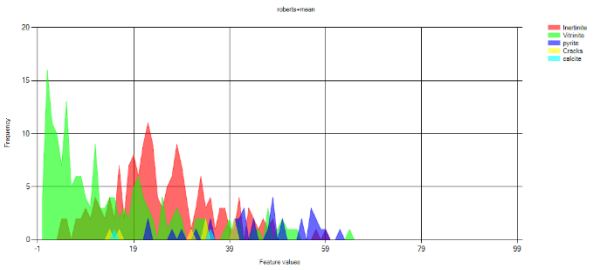


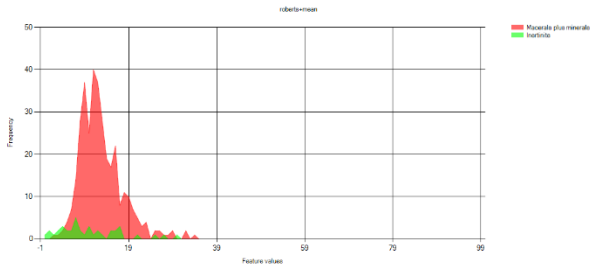
Roberts + mean feature

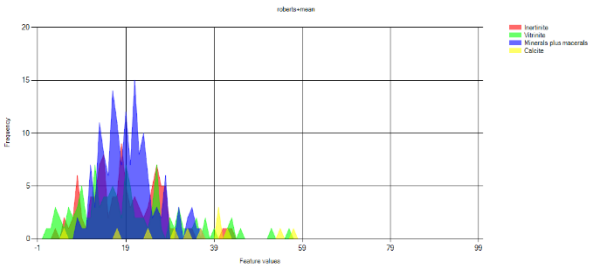




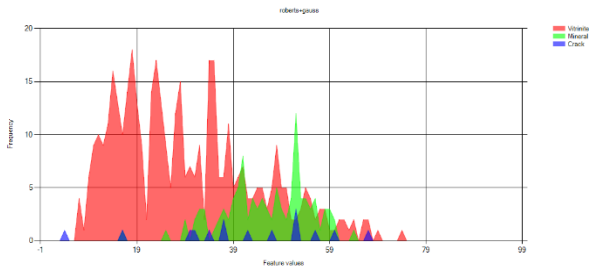


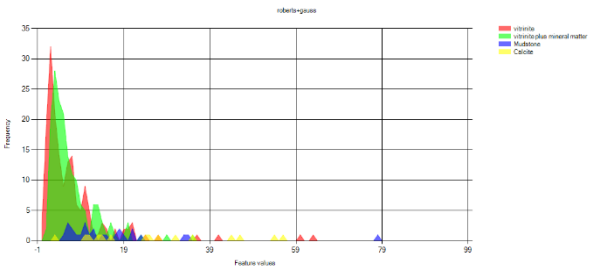


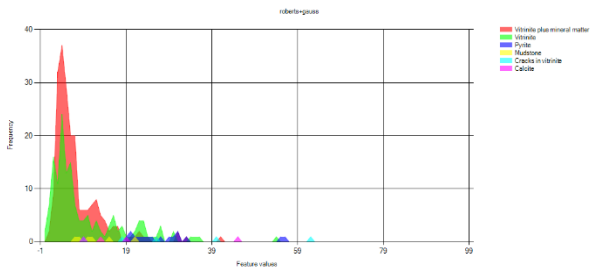


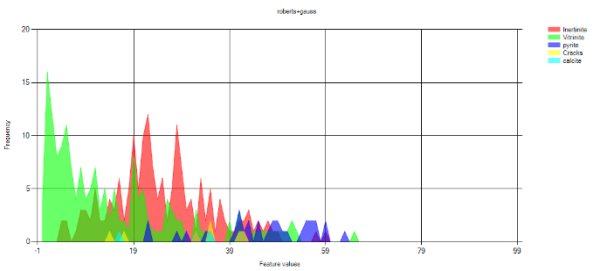


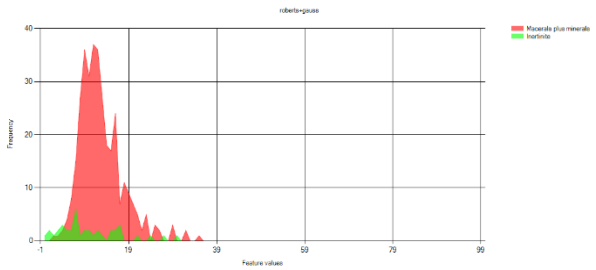
Roberts + gauss feature

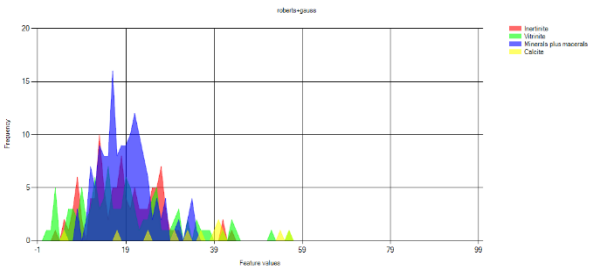




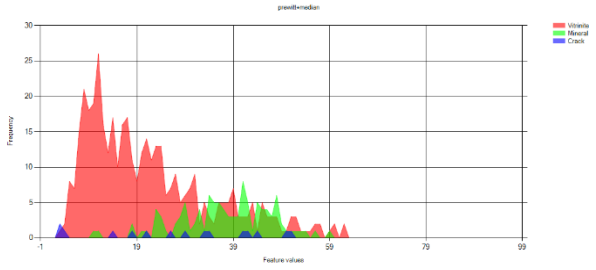


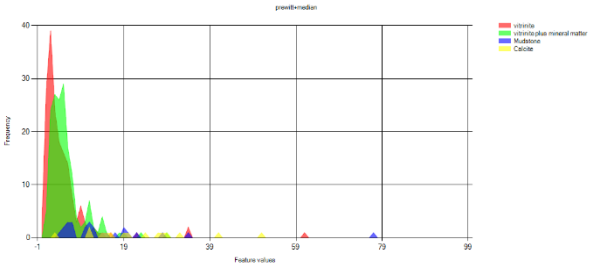


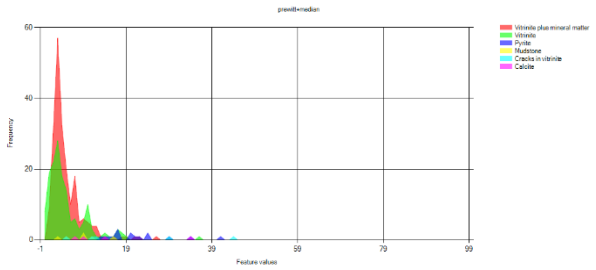


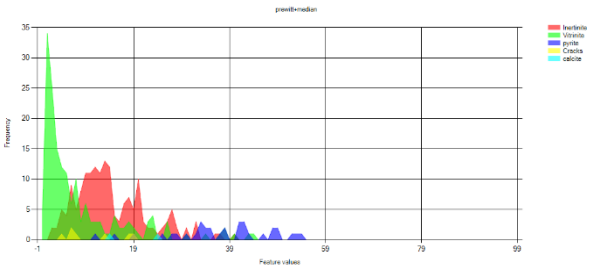


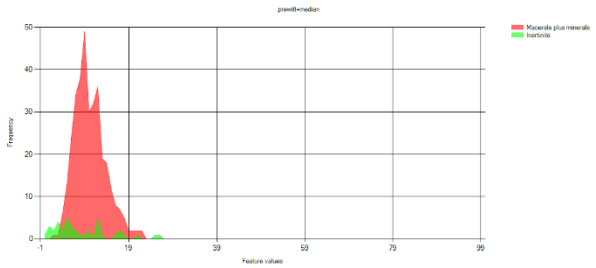
Prewitt + median feature

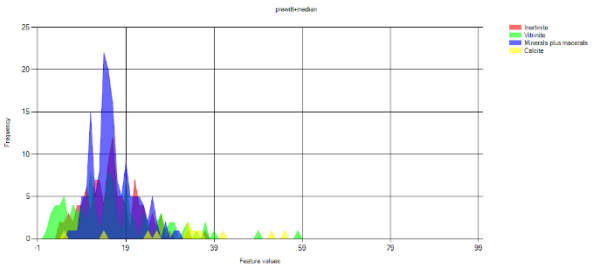




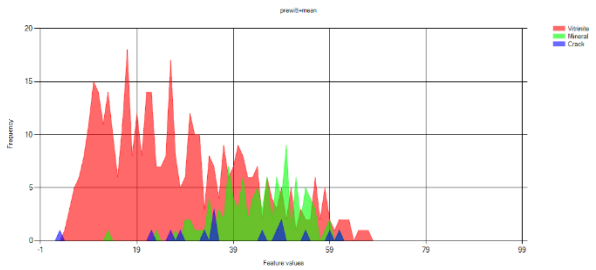


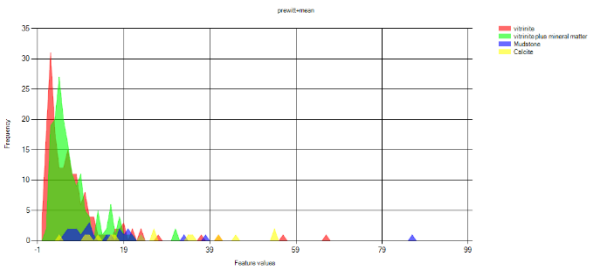


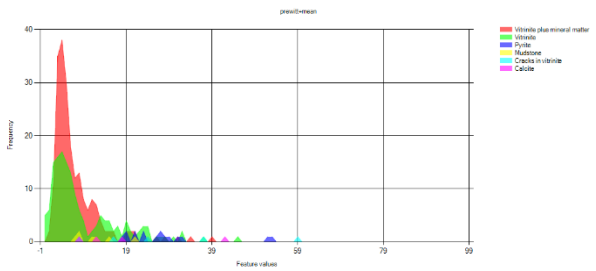


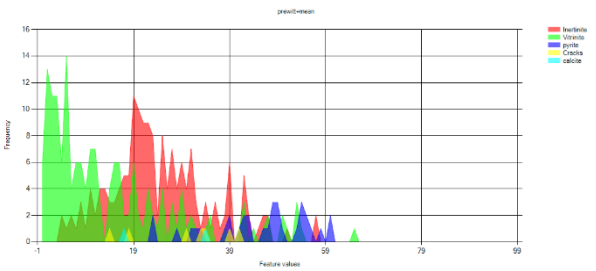


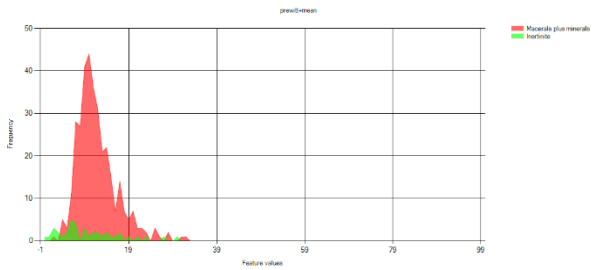
Prewitt + mean feature

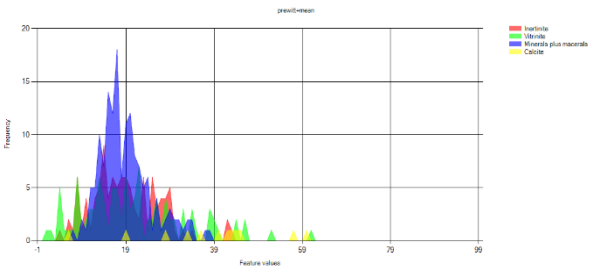




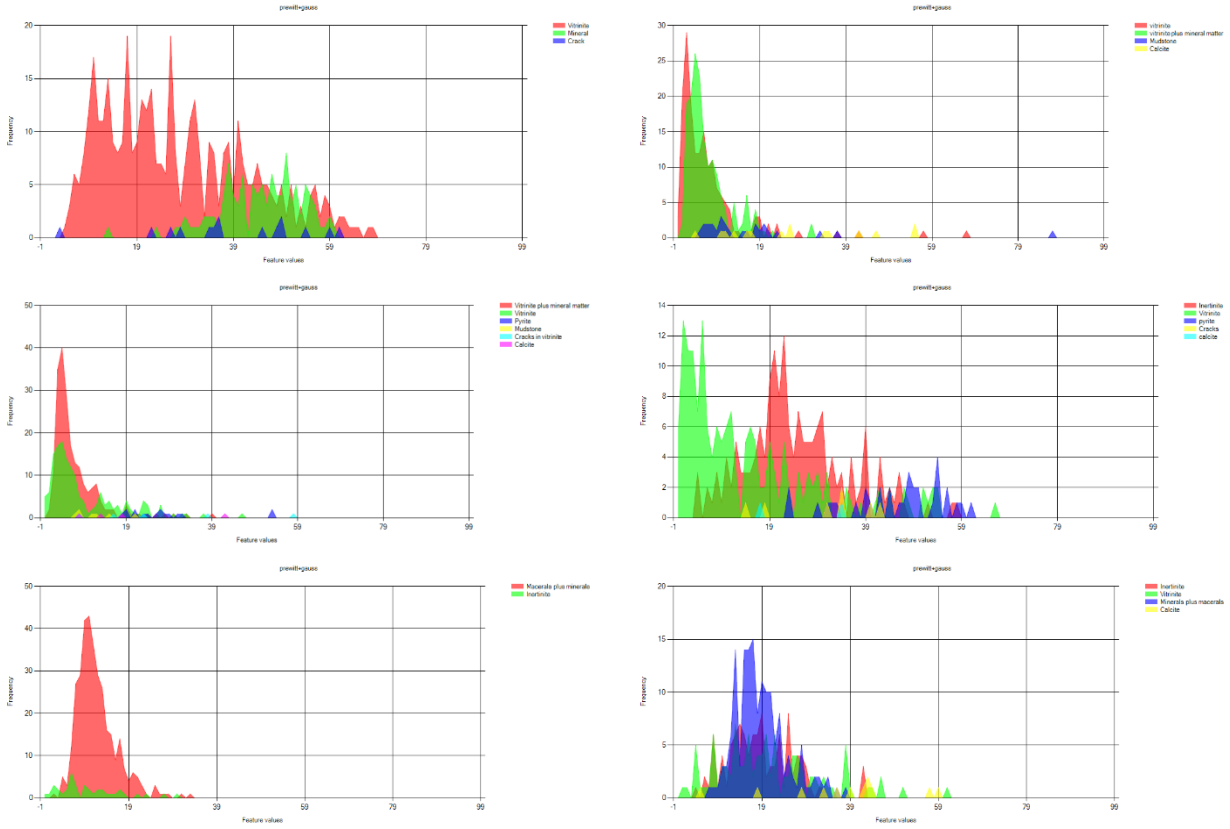




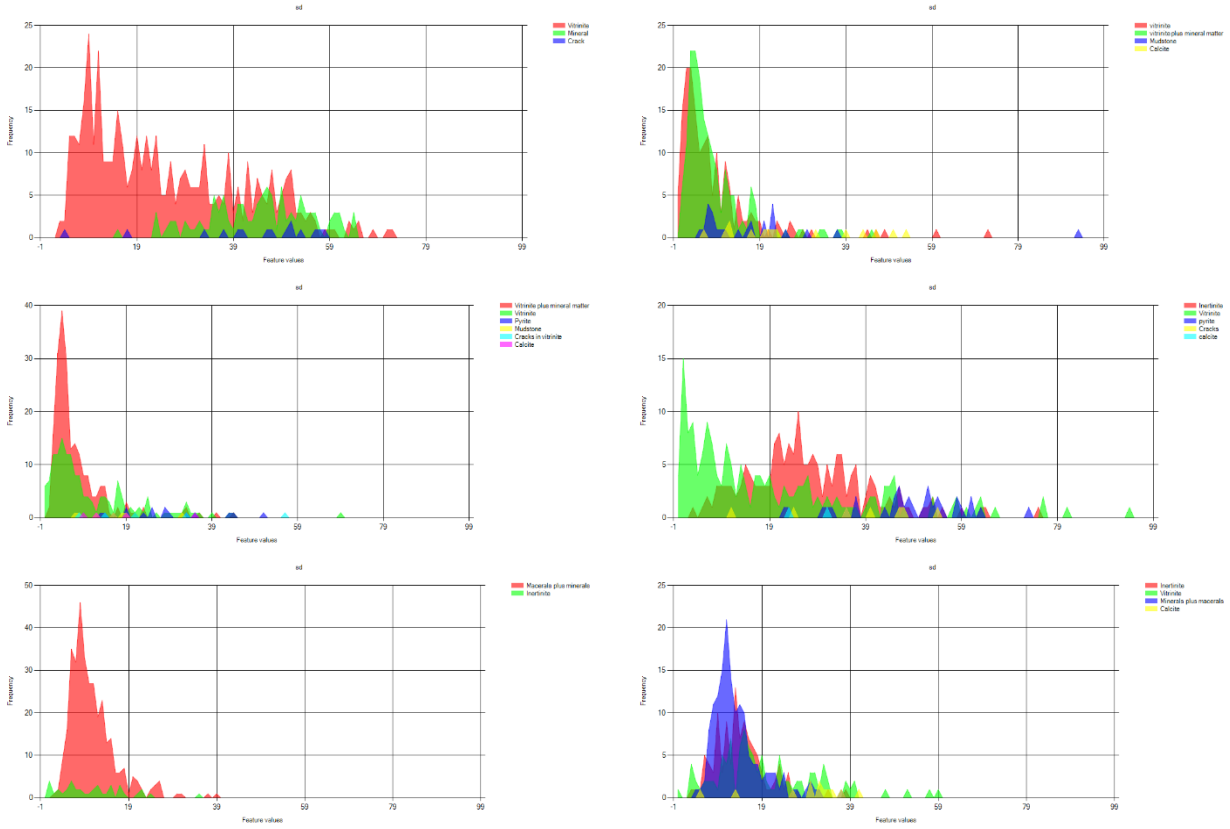




Prewitt + gauss feature



Standard deviation (SD) feature



B. Feature value t-tests

Chromaticity

Sample 0	*b* Vitrinite*	*v* Vitrinite*
Mean	57.56531939	61.56697339
Variance	35.91323063	49.57339329
Observations	419	419
Hypothesized Mean Difference	0	
df	815	
t Stat	-8.859260331	
P(T<=t) one-tail	2.49091E-18	
t Critical one-tail	1.646725422	
P(T<=t) two-tail	4.98183E-18	
t Critical two-tail	1.962879001	

Sample 0	*b* Mineral*	*v* Mineral*
Mean	48.54611724	51.35685617
Variance	152.4880915	138.5485561
Observations	106	106
Hypothesized Mean Difference	0	
df	210	
t Stat	-1.696288043	
P(T<=t) one-tail	0.045656574	
t Critical one-tail	1.652141981	
P(T<=t) two-tail	0.091313148	
t Critical two-tail	1.971324793	

	b Crack*	*v* Crack*
Mean	57.07097263	59.61932776
Variance	68.17842236	47.72139125
Observations	15	15
Hypothesized Mean Difference	0	
df	27	
t Stat	-0.916778058	
P(T<=t) one-tail	0.18368782	
t Critical one-tail	1.703288446	
P(T<=t) two-tail	0.36737564	
t Critical two-tail	2.051830516	

Regularity

Sample 0	*svd Vitrinite*	*entropy Vitrinite*
Mean	22.74426597	74.43492734
Variance	173.9976356	136.5627873
Observations	419	419
Hypothesized Mean Difference	0	
df	824	
t Stat	-60.0407413	
P(T<=t) one-tail	1.8875E-303	
t Critical one-tail	1.646704955	
P(T<=t) two-tail	3.775E-303	
t Critical two-tail	1.962847116	

Sample 0	*svd Mineral*	*entropy Mineral*
Mean	35.64417832	87.29063904
Variance	90.99692895	25.77483295
Observations	106	107
Hypothesized Mean Difference	0	
df	160	
t Stat	-49.2575711	
P(T<=t) one-tail	6.7149E-99	
t Critical one-tail	1.654432901	
P(T<=t) two-tail	1.34E-98	
t Critical two-tail	1.97490156	

Sample 0	*svd Crack*	*entropy Crack*
Mean	31.45004556	76.7855405
Variance	308.0892572	221.8338748
Observations	15	15
Hypothesized Mean Difference	0	
df	27	
t Stat	-7.627418085	
P(T<=t) one-tail	1.66459E-08	
t Critical one-tail	1.703288446	
P(T<=t) two-tail	3.32918E-08	
t Critical two-tail	2.051830516	

Sample 1	*svd Vitrinite*	*entropy Vitrinite*
Mean	4.807269124	39.97458823
Variance	26.44733733	204.882614
Observations	172	172
Hypothesized Mean Difference	0	
df	214	
t Stat	-30.32407433	
P(T<=t) one-tail	1.02165E-79	
t Critical one-tail	1.652005156	
P(T<=t) two-tail	2.04331E-79	
t Critical two-tail	1.971111258	

Sample 1	*svd Mudstone*	*Entropy Mudstone*
Mean	11.33753005	64.91814693
Variance	131.7281752	148.8561294
Observations	26	26
Hypothesized Mean Difference	0	
df	50	
t Stat	-16.31032807	
P(T<=t) one-tail	5.83779E-22	
t Critical one-tail	1.675905025	
P(T<=t) two-tail	1.16756E-21	
t Critical two-tail	2.008559112	

Sample 1	*svd Vitrinite+mineral matter*	*entropy Vitrinite+mineral matter*
Mean	4.886819224	44.70283442
Variance	14.64306415	143.1183469
Observations	168	168
Hypothesized Mean Difference	0	
df	201	
t Stat	-41.08772008	
P(T<=t) one-tail	4.7665E-100	
t Critical one-tail	1.652469842	
P(T<=t) two-tail	9.533E-100	
t Critical two-tail	1.971836507	

Sample 1	*svd Calcite*	*entropy Calcite*
Mean	18.5886577	66.52345111
Variance	130.891693	90.99641277
Observations	15	15
Hypothesized Mean Difference	0	
df	27	
t Stat	-12.46319753	
P(T<=t) one-tail	5.19069E-13	
t Critical one-tail	1.703288446	
P(T<=t) two-tail	1.03814E-12	
t Critical two-tail	2.051830516	

Sample 2	*svd Vitrinite+mineral matter*	*entropy Vitrinite+mineral matter*
Mean	5.979683687	49.67985899
Variance	26.32083827	140.3701091
Observations	210	210
Hypothesized Mean Difference	0	
df	285	
t Stat	-49.04975236	
P(T<=t) one-tail	2.8362E-141	
t Critical one-tail	1.650217713	
P(T<=t) two-tail	5.6723E-141	
t Critical two-tail	1.968322603	

Sample 2	*svd Mudstone*	*entropy Mudstone*
Mean	8.112553115	70.89531461
Variance	7.650032673	81.06725293
Observations	7	7
Hypothesized Mean Difference	0	
df	7	
t Stat	-17.63539987	
P(T<=t) one-tail	2.32256E-07	
t Critical one-tail	1.894578605	
P(T<=t) two-tail	4.64513E-07	
t Critical two-tail	2.364624252	

Sample 2	*svd Vitrinite*	*entropy Vitrinite*
Mean	7.596870656	49.65710187
Variance	58.16217864	243.2836146
Observations	154	154
Hypothesized Mean Difference	0	
df	222	
t Stat	-30.06265981	
P(T<=t) one-tail	1.61812E-80	
t Critical one-tail	1.651746359	
P(T<=t) two-tail	3.23624E-80	
t Critical two-tail	1.970707395	

Sample 2	*svd Pyrite*	*entropy Pyrite*	*Sample 2*	*svd Crack in Vitrinite*	*entropy Crack in vitrinite*
Mean	20.71221243	80.56285814	Mean	22.77054389	73.09267823
Variance	83.70978252	26.17668989	Variance	206.8580385	271.0530084
Observations	16	16	Observations	5	5
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	24		df	8	
t Stat	-22.83792885		t Stat	-5.147193798	
P(T<=t) one-tail	4.32121E-18		P(T<=t) one-tail	0.000438671	
t Critical one-tail	1.71088208		t Critical one-tail	1.859548038	
P(T<=t) two-tail	8.64241E-18		P(T<=t) two-tail	0.000877342	
t Critical two-tail	2.063898562		t Critical two-tail	2.306004135	
Sample 2	*svd Calcite*	*entropy Calcite*	*Sample 3*	*svd Inertinite*	*entropy Inertinite*
Mean	19.60961744	68.89505757	Mean	24.34355953	64.17851187
Variance	353.9168486	107.5248619	Variance	132.1399703	112.3684984
Observations	4	4	Observations	168	168
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	5		df	332	
t Stat	-4.588705762		t Stat	-33.01961383	
P(T<=t) one-tail	0.002950276		P(T<=t) one-tail	3.2358E-107	
t Critical one-tail	2.015048373		t Critical one-tail	1.649456205	
P(T<=t) two-tail	0.005900552		P(T<=t) two-tail	6.4716E-107	
t Critical two-tail	2.570581836		t Critical two-tail	1.967135057	
Sample 3	*svd Vitrinite*	*entropy Vitrinite*	*Sample 3*	*svd Cracks*	*entropy Cracks*
Mean	13.94995392	46.54469761	Mean	34.9659239	63.19389834
Variance	172.2317811	412.0528511	Variance	493.8179825	138.0224888
Observations	170	170	Observations	7	7
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	289		df	9	
t Stat	-17.58165684		t Stat	-2.971152993	
P(T<=t) one-tail	7.35224E-48		P(T<=t) one-tail	0.007836529	
t Critical one-tail	1.650143229		t Critical one-tail	1.833112933	
P(T<=t) two-tail	1.47045E-47		P(T<=t) two-tail	0.015673057	
t Critical two-tail	1.968206436		t Critical two-tail	2.262157163	
Sample 3	*svd Pyrite*	*entropy Pyrite*	*Sample 4*	*svd Macerals+mineral matter*	*entropy Macerals+mineral matte*
Mean	42.4553683	85.76522069	Mean	6.727531205	52.60281446
Variance	160.1510554	61.81222066	Variance	12.57921217	72.32947153
Observations	33	33	Observations	341	341
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	53		df	455	
t Stat	-16.69948625		t Stat	-91.93479121	
P(T<=t) one-tail	4.57697E-23		P(T<=t) one-tail	2.6099E-296	
t Critical one-tail	1.674116237		t Critical one-tail	1.648209441	
P(T<=t) two-tail	9.15394E-23		P(T<=t) two-tail	5.2198E-296	
t Critical two-tail	2.005745995		t Critical two-tail	1.965191428	
Sample 3	*svd Calcite*	*entropy Calcite*	*Sample 4*	*svd Inertinite*	*entropy Inertinite*
Mean	16.79119348	68.84009386	Mean	5.097197235	47.60089867
Variance	58.80365274	86.72681317	Variance	16.00153006	294.3445341
Observations	2	2	Observations	39	39
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	2		df	42	
t Stat	-6.101682377		t Stat	-15.06731732	
P(T<=t) one-tail	0.012911893		P(T<=t) one-tail	7.66256E-19	
t Critical one-tail	2.91998558		t Critical one-tail	1.681952357	
P(T<=t) two-tail	0.025823787		P(T<=t) two-tail	1.53251E-18	
t Critical two-tail	4.30265273		t Critical two-tail	2.018081703	
Sample 5	*svd Inertinite*	*entropy Inertinite*	*Sample 5*	*svd Macerals+Minerals*	*entropy Macerals+Minerals*
Mean	15.8768961	59.63015547	Mean	15.24026741	59.45182044
Variance	55.63148766	108.3119719	Variance	30.53434548	60.34009162
Observations	106	106	Observations	152	152
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	190		df	273	
t Stat	-35.18165747		t Stat	-57.17901871	
P(T<=t) one-tail	1.91113E-85		P(T<=t) one-tail	2.8584E-154	
t Critical one-tail	1.652912949		t Critical one-tail	1.650454303	
P(T<=t) two-tail	3.82225E-85		P(T<=t) two-tail	5.7168E-154	
t Critical two-tail	1.972528182		t Critical two-tail	1.96869162	
Sample 5	*svd Vitrinite*	*entropy Vitrinite*	*Sample 5*	*svd Calcite*	*entropy Calcite*
Mean	16.24824372	60.62633245	Mean	30.09006182	76.05021901
Variance	100.5772557	232.5589257	Variance	170.1127079	203.8640693
Observations	104	104	Observations	12	12
Hypothesized Mean Difference	0		Hypothesized Mean Difference	0	
df	178		df	22	
t Stat	-24.79558518		t Stat	-8.232842573	
P(T<=t) one-tail	6.1716E-60		P(T<=t) one-tail	1.82722E-08	
t Critical one-tail	1.653459126		t Critical one-tail	1.717144374	
P(T<=t) two-tail	1.23432E-59		P(T<=t) two-tail	3.65445E-08	
t Critical two-tail	1.973380889		t Critical two-tail	2.073873068	

Roughness

Sample 0	*SM Vitrinite*	*SG Vitrinite*
Mean	22.61497359	28.41006001
Variance	178.058395	222.8052251
Observations	419	419
Hypothesized Mean Difference	0	
df	826	
t Stat	-5.924730578	
P(T<=t) one-tail	2.29455E-09	
t Critical one-tail	1.646700467	
P(T<=t) two-tail	4.5891E-09	
t Critical two-tail	1.962840125	

Sample 0	*SM Mineral*	*SG Mineral*
Mean	38.61548981	45.49290046
Variance	93.75313836	74.69787798
Observations	106	106
Hypothesized Mean Difference	0	
df	207	
t Stat	-5.455581796	
P(T<=t) one-tail	6.92907E-08	
t Critical one-tail	1.652248086	
P(T<=t) two-tail	1.38581E-07	
t Critical two-tail	1.971490392	

Sample 0	*SM Crack*	*SG Crack*
Mean	28.2174018	40.03819852
Variance	291.0022807	248.9185008
Observations	15	15
Hypothesized Mean Difference	0	
df	28	
t Stat	-1.970277313	
P(T<=t) one-tail	0.029384204	
t Critical one-tail	1.701130934	
P(T<=t) two-tail	0.058768408	
t Critical two-tail	2.048407142	

Sample 1	*SM Vitrinite+mineral matter*	*SG Vitrinite+mineral matter*
Mean	5.963003326	7.386325066
Variance	15.77203779	26.56048171
Observations	168	168
Hypothesized Mean Difference	0	
df	314	
t Stat	-2.83544133	
P(T<=t) one-tail	0.002436576	
t Critical one-tail	1.649720831	
P(T<=t) two-tail	0.004873152	
t Critical two-tail	1.967547698	

Sample 1	*SM Calcite*	*SG Calcite*
Mean	23.16188238	28.07086799
Variance	158.0952487	267.3929688
Observations	15	15
Hypothesized Mean Difference	0	
df	26	
t Stat	-0.921708556	
P(T<=t) one-tail	0.182577412	
t Critical one-tail	1.70561792	
P(T<=t) two-tail	0.365154823	
t Critical two-tail	2.055529439	

Sample 1	*SM Vitrinite*	*SG Vitrinite*
Mean	5.173303837	7.527383976
Variance	37.90668419	74.76405423
Observations	172	172
Hypothesized Mean Difference	0	
df	309	
t Stat	-2.908572519	
P(T<=t) one-tail	0.001946999	
t Critical one-tail	1.649799826	
P(T<=t) two-tail	0.003893998	
t Critical two-tail	1.967670885	

Sample 2	*SM Vitrinite*	*SG Vitrinite*
Mean	5.573456925	9.586616435
Variance	24.04089188	71.8707539
Observations	154	154
Hypothesized Mean Difference	0	
df	245	
t Stat	-5.085235974	
P(T<=t) one-tail	3.65011E-07	
t Critical one-tail	1.65109682	
P(T<=t) two-tail	7.30022E-07	
t Critical two-tail	1.969693921	

Sample 1	*SM Mudstone*	*SG Mudstone*
Mean	15.51165649	17.67081339
Variance	212.7619738	266.0505075
Observations	26	26
Hypothesized Mean Difference	0	
df	49	
t Stat	-0.503139189	
P(T<=t) one-tail	0.308560339	
t Critical one-tail	1.676550893	
P(T<=t) two-tail	0.617120677	
t Critical two-tail	2.009575237	

Sample 2	*SM Mudstone*	*SG Mudstone*
Mean	8.971103427	12.37836723
Variance	15.762431	26.15307053
Observations	7	7
Hypothesized Mean Difference	0	
df	11	
t Stat	-1.392411004	
P(T<=t) one-tail	0.095652253	
t Critical one-tail	1.795884819	
P(T<=t) two-tail	0.191304505	
t Critical two-tail	2.20098516	

Sample 2	*SM Vitrinite+mineral matter*	*SG Vitrinite+mineral matter*
Mean	5.394955013	7.625765652
Variance	13.8456222	32.96923887
Observations	210	210
Hypothesized Mean Difference	0	
df	358	
t Stat	-4.724767964	
P(T<=t) one-tail	1.65751E-06	
t Critical one-tail	1.649121068	
P(T<=t) two-tail	3.31502E-06	
t Critical two-tail	1.966612519	

Sample 2	*SM Calcite*	*SG Calcite*
Mean	16.48242911	20.97568577
Variance	153.8996064	235.3836245
Observations	4	4
Hypothesized Mean Difference	0	
df	6	
t Stat	-0.455468529	
P(T<=t) one-tail	0.332395082	
t Critical one-tail	1.943180281	
P(T<=t) two-tail	0.664790163	
t Critical two-tail	2.446911851	

Sample 2	*SM Pyrite*	*SG Pyrite*
Mean	21.55277623	29.08331672
Variance	49.73750408	111.9861795
Observations	16	16
Hypothesized Mean Difference	0	
df	26	
t Stat	-2.368641463	
P(T<=t) one-tail	0.012783759	
t Critical one-tail	1.70561792	
P(T<=t) two-tail	0.025567517	
t Critical two-tail	2.055529439	

Sample 2	*SM Crack in vitrinite*	*SG Crack in vitrinite*
Mean	21.50508981	29.5507311
Variance	282.5698701	419.0889986
Observations	5	5
Hypothesized Mean Difference	0	
df	8	
t Stat	-0.679176513	
P(T<=t) one-tail	0.258095316	
t Critical one-tail	1.859548038	
P(T<=t) two-tail	0.516190633	
t Critical two-tail	2.306004135	

Sample 3	*SM Inertinite*	*SG Inertinite*
Mean	14.80337484	25.35940813
Variance	56.22800606	116.8826891
Observations	168	168
Hypothesized Mean Difference	0	
df	297	
t Stat	-10.39904462	
P(T<=t) one-tail	4.18817E-22	
t Critical one-tail	1.650000301	
P(T<=t) two-tail	8.37635E-22	
t Critical two-tail	1.967983525	

Sample 3	*SM Vitrinite*	*SG Vitrinite*
Mean	8.168921498	15.47677943
Variance	83.81259899	213.7532436
Observations	170	170
Hypothesized Mean Difference	0	
df	284	
t Stat	-5.523610134	
P(T<=t) one-tail	3.753E-08	
t Critical one-tail	1.650236662	
P(T<=t) two-tail	7.50601E-08	
t Critical two-tail	1.968352158	

Sample 3	*SM Pyrite*	*SG Pyrite*
Mean	37.87823764	46.54141146
Variance	118.4655778	106.7502613
Observations	33	33
Hypothesized Mean Difference	0	
df	64	
t Stat	-3.316152796	
P(T<=t) one-tail	0.000753463	
t Critical one-tail	1.669013025	
P(T<=t) two-tail	0.001506926	
t Critical two-tail	1.997729654	

Sample 3	*SM Cracks*	*SG Cracks*
Mean	10.75158377	31.1974945
Variance	35.98580844	118.7851502
Observations	7	7
Hypothesized Mean Difference	0	
df	9	
t Stat	-4.348212338	
P(T<=t) one-tail	0.000927544	
t Critical one-tail	1.833112933	
P(T<=t) two-tail	0.001855088	
t Critical two-tail	2.262157163	

Sample 3	*SM Calcite*	*SG Calcite*
Mean	19.06944084	26.47555804
Variance	62.64285319	161.2059003
Observations	2	2
Hypothesized Mean Difference	0	
df	2	
t Stat	-0.700048675	
P(T<=t) one-tail	0.278184124	
t Critical one-tail	2.91998558	
P(T<=t) two-tail	0.556368249	
t Critical two-tail	4.30265273	

Sample 5	*SM inertinite*	*SG inertinite*
Mean	16.05451918	19.20118971
Variance	39.67573018	58.95961213
Observations	106	106
Hypothesized Mean Difference	0	
df	202	
t Stat	-3.262029818	
P(T<=t) one-tail	0.0006493	
t Critical one-tail	1.652431964	
P(T<=t) two-tail	0.0012986	
t Critical two-tail	1.971777385	

Sample 5	*SM Macerals+Minerals*	*SG Macerals+Minerals*
Mean	16.33741252	19.28183307
Variance	22.17731511	33.28280997
Observations	152	152
Hypothesized Mean Difference	0	
df	290	
t Stat	-4.874512758	
P(T<=t) one-tail	8.99538E-07	
t Critical one-tail	1.650124931	
P(T<=t) two-tail	1.79908E-06	
t Critical two-tail	1.968177896	

Sample 4	*SM macerals+minerals*	*SG macerals+minerals*
Mean	10.83741917	12.14754731
Variance	12.86341957	20.75363235
Observations	341	341
Hypothesized Mean Difference	0	
df	644	
t Stat	-4.172642268	
P(T<=t) one-tail	1.71216E-05	
t Critical one-tail	1.647223156	
P(T<=t) two-tail	3.42433E-05	
t Critical two-tail	1.963654451	

Sample 4	*SM inertinite*	*SG inertinite*
Mean	8.940521096	10.381743
Variance	44.01622168	50.272273
Observations	39	39
Hypothesized Mean Difference	0	
df	76	
t Stat	-0.926901971	
P(T<=t) one-tail	0.178455831	
t Critical one-tail	1.665151353	
P(T<=t) two-tail	0.356911663	
t Critical two-tail	1.99167261	

Sample 5	*SM Vitrinite*	*SG Vitrinite*
Mean	16.15326716	20.91757963
Variance	104.5711765	133.2833531
Observations	104	104
Hypothesized Mean Difference	0	
df	203	
t Stat	-3.150367317	
P(T<=t) one-tail	0.000938504	
t Critical one-tail	1.65239446	
P(T<=t) two-tail	0.001877009	
t Critical two-tail	1.971718848	

Sample 5	*SM Calcite*	*SG Calcite*
Mean	31.95974629	37.47959908
Variance	186.9321518	223.007112
Observations	12	12
Hypothesized Mean Difference	0	
df	22	
t Stat	-0.944405178	
P(T<=t) one-tail	0.177609929	
t Critical one-tail	1.717144374	
P(T<=t) two-tail	0.355219857	
t Critical two-tail	2.073873068	

C. Classification tests

The following tables are the confusion matrices obtained from comparing the classifications obtained through the use of each of the four clustering methods, and each of the eight methods of assigning final model values. The colour shading represents the rank of the shaded accuracy compared to the corresponding accuracy in all other classifiers implemented on the same image. Greener shades represent a higher rank and redder shades represent a lower rank.

Sample 0

k-means baseline				
Right %	Wrong %	Predicted ID		
83.7037	16.2963	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.98807	5.011933	0
	Mineral	49.0566	50.9434	0
	Crack	86.66667	13.33333	0

k-means alternative A				
Right %	Wrong %	Predicted ID		
84.62963	15.37037	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.03341	5.011933	0.954654
	Mineral	23.58491	50.9434	25.4717
	Crack	26.66667	13.33333	60

k-means alternative B				
Right %	Wrong %	Predicted ID		
84.62963	15.37037	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.03341	5.011933	0.954654
	Mineral	23.58491	50.9434	25.4717
	Crack	26.66667	13.33333	60

k-means alternative C				
Right %	Wrong %	Predicted ID		
84.62963	15.37037	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.03341	5.011933	0.954654
	Mineral	23.58491	50.9434	25.4717
	Crack	26.66667	13.33333	60

k-means alternative D				
Right %	Wrong %	Predicted ID		
86.11111	13.88889	Vitrinite	Mineral	Crack
True ID	Vitrinite	97.13604	2.625298	0.238663
	Mineral	48.11321	51.88679	0
	Crack	26.66667	53.33333	20

k-means alternative E				
Right %	Wrong %	Predicted ID		
84.44444	15.55556	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.79475	4.534606	1.670644
	Mineral	21.69811	50.9434	27.35849
	Crack	26.66667	13.33333	60

k-means alternative F				
Right %	Wrong %	Predicted ID		
80.74074	19.25926	Vitrinite	Mineral	Crack
True ID	Vitrinite	89.02148	4.295943	6.682578
	Mineral	16.98113	50	33.01887
	Crack	20	13.33333	66.66667

k-means alternative G				
Right %	Wrong %	Predicted ID		
80.74074	19.25926	Vitrinite	Mineral	Crack
True ID	Vitrinite	89.02148	4.295943	6.682578
	Mineral	16.98113	50	33.01887
	Crack	20	13.33333	66.66667

k-medians baseline				
Right %	Wrong %	Predicted ID		
82.59259	17.40741	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.55609	6.443914	0
	Mineral	49.0566	50.9434	0
	Crack	86.66667	13.33333	0

k-medians alternative A				
Right %	Wrong %	Predicted ID		
82.59259	17.40741	Vitrinite	Mineral	Crack
True ID	Vitrinite	91.16945	6.443914	2.386635
	Mineral	16.98113	50.9434	32.07547
	Crack	20	13.33333	66.66667

k-medians alternative B				
Right %	Wrong %	Predicted ID		
82.59259	17.40741	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.55609	6.443914	0
	Mineral	49.0566	50.9434	0
	Crack	86.66667	13.33333	0

k-medians alternative C				
Right %	Wrong %	Predicted ID		
82.59259	17.40741	Vitrinite	Mineral	Crack
True ID	Vitrinite	91.16945	6.443914	2.386635
	Mineral	16.98113	50.9434	32.07547
	Crack	20	13.33333	66.66667

k-medians alternative D				
Right %	Wrong %	Predicted ID		
85.37037	14.62963	Vitrinite	Mineral	Crack
True ID	Vitrinite	97.3747	2.386635	0.238663
	Mineral	52.83019	47.16981	0
	Crack	33.33333	46.66667	20

k-medians alternative E				
Right %	Wrong %	Predicted ID		
83.33333	16.66667	Vitrinite	Mineral	Crack
True ID	Vitrinite	91.88544	5.727924	2.386635
	Mineral	16.98113	50.9434	32.07547
	Crack	13.33333	13.33333	73.33333

k-medians alternative F				
Right %	Wrong %	Predicted ID		
85.92593	14.07407	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.79475	6.205251	0
	Mineral	33.01887	66.98113	0
	Crack	40	60	0

k-medians alternative G				
Right %	Wrong %	Predicted ID		
76.66667	23.33333	Vitrinite	Mineral	Crack
True ID	Vitrinite	84.00955	4.534606	11.45585
	Mineral	10.37736	48.11321	41.50943
	Crack	13.33333	13.33333	73.33333

mean shift baseline				
Right %	Wrong %	Predicted ID		
86.2963	13.7037	Vitrinite	Mineral	Crack
True ID	Vitrinite	97.61337	2.147971	0.238663
	Mineral	48.11321	51.88679	0
	Crack	40	46.66667	13.33333

mean shift alternative A				
Right %	Wrong %	Predicted ID		
85.74074	14.25926	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.07876	5.48926	1.431981
	Mineral	31.13208	62.26415	6.603774
	Crack	33.33333	20	46.66667

mean shift alternative B				
Right %	Wrong %	Predicted ID		
87.96296	12.03704	Vitrinite	Mineral	Crack
True ID	Vitrinite	95.46539	4.295943	0.238663
	Mineral	31.13208	68.86792	0
	Crack	33.33333	53.33333	13.33333

mean shift alternative C				
Right %	Wrong %	Predicted ID		
81.48148	18.51852	Vitrinite	Mineral	Crack
True ID	Vitrinite	88.06683	5.250597	6.682578
	Mineral	19.81132	56.60377	23.58491
	Crack	6.666667	20	73.33333

mean shift alternative D				
Right %	Wrong %	Predicted ID		
87.77778	12.22222	Vitrinite	Mineral	Crack
True ID	Vitrinite	97.61337	2.386635	0
	Mineral	40.56604	59.43396	0
	Crack	33.33333	53.33333	13.33333

mean shift alternative E				
Right %	Wrong %	Predicted ID		
84.81481	15.18519	Vitrinite	Mineral	Crack
True ID	Vitrinite	93.31742	4.295943	2.386635
	Mineral	19.81132	54.71698	25.4717
	Crack	26.66667	13.33333	60

mean shift alternative F				
Right %	Wrong %	Predicted ID		
88.33333	11.66667	Vitrinite	Mineral	Crack
True ID	Vitrinite	95.94272	4.057279	0
	Mineral	31.13208	68.86792	0
	Crack	33.33333	53.33333	13.33333

mean shift alternative G				
Right %	Wrong %	Predicted ID		
83.14815	16.85185	Vitrinite	Mineral	Crack
True ID	Vitrinite	91.16945	4.057279	4.77327
	Mineral	16.03774	51.88679	32.07547
	Crack	13.33333	6.666667	80

PSO baseline				
Right %	Wrong %	Predicted ID		
85.92593	14.07407	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.98807	5.011933	0
	Mineral	37.73585	62.26415	0
	Crack	80	20	0

PSO alternative A				
Right %	Wrong %	Predicted ID		
86.85185	13.14815	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.27208	5.011933	0.71599
	Mineral	26.41509	62.26415	11.32075
	Crack	26.66667	20	53.33333

PSO alternative B				
Right %	Wrong %	Predicted ID		
85.92593	14.07407	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.98807	5.011933	0
	Mineral	37.73585	62.26415	0
	Crack	80	20	0

PSO alternative C				
Right %	Wrong %	Predicted ID		
86.85185	13.14815	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.27208	5.011933	0.71599
	Mineral	26.41509	62.26415	11.32075
	Crack	26.66667	20	53.33333

PSO alternative D				
Right %	Wrong %	Predicted ID		
87.96296	12.03704	Vitrinite	Mineral	Crack
True ID	Vitrinite	96.42005	3.579952	0
	Mineral	35.84906	64.15094	0
	Crack	33.33333	46.66667	20

PSO alternative E				
Right %	Wrong %	Predicted ID		
87.03704	12.96296	Vitrinite	Mineral	Crack
True ID	Vitrinite	94.27208	4.77327	0.954654
	Mineral	24.5283	62.26415	13.20755
	Crack	26.66667	13.33333	60

PSO alternative F				
Right %	Wrong %	Predicted ID		
87.40741	12.59259	Vitrinite	Mineral	Crack
True ID	Vitrinite	96.42005	3.579952	0
	Mineral	35.84906	64.15094	0
	Crack	33.33333	66.66667	0

PSO alternative G				
Right %	Wrong %	Predicted ID		
87.22222	12.77778	Vitrinite	Mineral	Crack
True ID	Vitrinite	95.22673	3.341289	1.431981
	Mineral	24.5283	60.37736	15.09434
	Crack	26.66667	20	53.33333

Sample 1

Class name abbreviations:

- V: Vitrinite
- VpMM: Vitrinite plus mineral matter
- M: Mudstone
- C: Calcite

k-means baseline					
Right %	Wrong %	Predicted ID			
53.54331	46.45669	V	VPMM	M	C
True ID	V	36.62791	52.90698	8.139535	2.325581
	VpMM	16.07143	73.21429	9.52381	1.190476
	M	0	34.61538	50	15.38462
	C	0	13.33333	53.33333	33.33333

k-means alternative A					
Right %	Wrong %	Predicted ID			
53.54331	46.45669	V	VPMM	M	C
True ID	V	36.62791	52.90698	8.139535	2.325581
	VpMM	16.07143	73.21429	9.52381	1.190476
	M	0	34.61538	50	15.38462
	C	0	13.33333	53.33333	33.33333

k-means alternative B					
Right %	Wrong %	Predicted ID			
50.13123	49.86877	V	VPMM	M	C
True ID	V	59.30233	30.23256	8.139535	2.325581
	VpMM	47.02381	42.2619	9.52381	1.190476
	M	3.846154	30.76923	50	15.38462
	C	0	13.33333	53.33333	33.33333

k-means alternative C					
Right %	Wrong %	Predicted ID			
50.13123	49.86877	V	VPMM	M	C
True ID	V	59.30233	30.23256	8.139535	2.325581
	VpMM	47.02381	42.2619	9.52381	1.190476
	M	3.846154	30.76923	50	15.38462
	C	0	13.33333	53.33333	33.33333

k-means alternative D					
Right %	Wrong %	Predicted ID			
55.11811	44.88189	V	VPMM	M	C
True ID	V	33.72093	56.39535	7.55814	2.325581
	VpMM	11.90476	76.78571	10.11905	1.190476
	M	0	23.07692	65.38462	11.53846
	C	0	13.33333	46.66667	40

k-means alternative E					
Right %	Wrong %	Predicted ID			
56.16798	43.83202	V	VPMM	M	C
True ID	V	40.69767	49.4186	7.55814	2.325581
	VpMM	16.66667	72.02381	10.11905	1.190476
	M	0	23.07692	65.38462	11.53846
	C	0	13.33333	46.66667	40

k-means alternative F					
Right %	Wrong %	Predicted ID			
51.70604	48.29396	V	VPMM	M	C
True ID	V	49.4186	39.53488	8.72093	2.325581
	VpMM	33.33333	54.16667	11.30952	1.190476
	M	0	23.07692	61.53846	15.38462
	C	0	13.33333	53.33333	33.33333

k-means alternative G					
Right %	Wrong %	Predicted ID			
51.70604	48.29396	V	VPMM	M	C
True ID	V	49.4186	39.53488	8.72093	2.325581
	VpMM	33.33333	54.16667	11.30952	1.190476
	M	0	23.07692	61.53846	15.38462
	C	0	13.33333	53.33333	33.33333

k-medians baseline					
Right %	Wrong %	Predicted ID			
46.98163	53.01837	V	VPMM	M	C
True ID	V	14.53488	76.74419	8.72093	0
	VpMM	2.380952	82.14286	15.47619	0
	M	0	38.46154	61.53846	0
	C	0	26.66667	73.33333	0

k-medians alternative A					
Right %	Wrong %	Predicted ID			
50.65617	49.34383	V	VPMM	M	C
True ID	V	47.09302	44.18605	8.72093	0
	VpMM	27.38095	57.14286	15.47619	0
	M	0	38.46154	61.53846	0
	C	0	26.66667	73.33333	0

k-medians alternative B					
Right %	Wrong %	Predicted ID			
44.35696	55.64304	V	VPMM	M	C
True ID	V	33.72093	45.34884	16.86047	4.069767
	VpMM	16.07143	54.16667	25	4.761905
	M	0	15.38462	42.30769	42.30769
	C	0	6.666667	33.33333	60

k-medians alternative C					
Right %	Wrong %	Predicted ID			
45.40682	54.59318	V	VPMM	M	C
True ID	V	47.09302	31.97674	16.86047	4.069767
	VpMM	27.38095	42.85714	25	4.761905
	M	0	15.38462	42.30769	42.30769
	C	0	6.666667	33.33333	60

k-medians alternative D					
Right %	Wrong %	Predicted ID			
51.1811	48.8189	V	VPMM	M	C
True ID	V	21.51163	69.76744	6.395349	2.325581
	VpMM	6.547619	78.57143	14.88095	0
	M	0	15.38462	76.92308	7.692308
	C	0	33.33333	26.66667	40

k-medians alternative E					
Right %	Wrong %	Predicted ID			
54.85564	45.14436	V	VPMM	M	C
True ID	V	47.09302	44.18605	6.395349	2.325581
	VpMM	24.40476	60.71429	14.88095	0
	M	0	15.38462	76.92308	7.692308
	C	0	33.33333	26.66667	40

k-medians alternative F					
Right %	Wrong %	Predicted ID			
42.78215	57.21785	V	VPMM	M	C
True ID	V	30.23256	48.25581	18.02326	3.488372
	VpMM	16.66667	53.57143	25	4.761905
	M	0	11.53846	46.15385	42.30769
	C	0	6.666667	33.33333	60

k-medians alternative G					
Right %	Wrong %	Predicted ID			
44.35696	55.64304	V	VPMM	M	C
True ID	V	40.69767	37.7907	18.02326	3.488372
	VpMM	23.80952	46.42857	25	4.761905
	M	0	11.53846	46.15385	42.30769
	C	0	6.666667	33.33333	60

mean shift baseline						mean shift alternative A					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
45.66929	54.33071	V	VPMM	M	C	52.75591	47.24409	V	VPMM	M	C
True ID	V	2.325581	90.69767	5.813953	1.162791	True ID	V	42.44186	50.5814	5.813953	1.162791
	VpMM	1.190476	89.88095	8.333333	0.595238		VpMM	26.19048	64.88095	8.333333	0.595238
	M	0	42.30769	53.84615	3.846154		M	0	42.30769	53.84615	3.846154
	C	0	33.33333	33.33333	33.33333		C	6.666667	26.66667	33.33333	33.33333

mean shift alternative B						mean shift alternative C					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
45.40682	54.59318	V	VPMM	M	C	55.64304	44.35696	V	VPMM	M	C
True ID	V	2.325581	91.86047	4.069767	1.744186	True ID	V	53.48837	40.69767	4.069767	1.744186
	VpMM	1.190476	90.47619	7.142857	1.190476		VpMM	30.35714	61.30952	7.142857	1.190476
	M	0	42.30769	46.15385	11.53846		M	0	42.30769	46.15385	11.53846
	C	0	33.33333	33.33333	33.33333		C	13.33333	20	33.33333	33.33333

mean shift alternative D						mean shift alternative E					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
50.3937	49.6063	V	VPMM	M	C	54.59318	45.40682	V	VPMM	M	C
True ID	V	13.95349	77.90698	5.813953	2.325581	True ID	V	42.44186	49.4186	5.813953	2.325581
	VpMM	3.571429	88.69048	7.142857	0.595238		VpMM	23.21429	69.04762	7.142857	0.595238
	M	0	38.46154	53.84615	7.692308		M	0	38.46154	53.84615	7.692308
	C	0	26.66667	40	33.33333		C	6.666667	20	40	33.33333

mean shift alternative F						mean shift alternative G					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
51.9685	48.0315	V	VPMM	M	C	55.11811	44.88189	V	VPMM	M	C
True ID	V	26.16279	66.86047	4.069767	2.906977	True ID	V	52.90698	40.11628	4.069767	2.906977
	VpMM	13.09524	79.16667	5.357143	2.380952		VpMM	33.33333	58.92857	5.357143	2.380952
	M	0	38.46154	53.84615	7.692308		M	0	38.46154	53.84615	7.692308
	C	6.666667	20	33.33333	40		C	13.33333	13.33333	33.33333	40

PSO baseline						PSO alternative A					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
50.3937	49.6063	V	VPMM	M	C	53.80577	46.19423	V	VPMM	M	C
True ID	V	25.5814	73.25581	0	1.162791	True ID	V	44.18605	54.65116	0	1.162791
	VpMM	13.09524	86.30952	0	0.595238		VpMM	24.40476	75	0	0.595238
	M	0	92.30769	0	7.692308		M	0	92.30769	0	7.692308
	C	0	80	0	20		C	0	80	0	20

PSO alternative B						PSO alternative C					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
53.80577	46.19423	V	VPMM	M	C	53.80577	46.19423	V	VPMM	M	C
True ID	V	44.18605	54.65116	0	1.162791	True ID	V	44.18605	54.65116	0	1.162791
	VpMM	24.40476	75	0	0.595238		VpMM	24.40476	75	0	0.595238
	M	0	92.30769	0	7.692308		M	0	92.30769	0	7.692308
	C	0	80	0	20		C	0	80	0	20

PSO alternative D						PSO alternative E					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
51.70604	48.29396	V	VPMM	M	C	54.33071	45.66929	V	VPMM	M	C
True ID	V	28.48837	69.18605	1.744186	0.581395	True ID	V	41.86047	55.81395	1.744186	0.581395
	VpMM	13.09524	86.30952	0.595238	0		VpMM	20.83333	78.57143	0.595238	0
	M	0	92.30769	0	7.692308		M	0	92.30769	0	7.692308
	C	0	80	0	20		C	0	80	0	20

PSO alternative F						PSO alternative G					
Right %	Wrong %	Predicted ID				Right %	Wrong %	Predicted ID			
54.85564	45.14436	V	VPMM	M	C	54.85564	45.14436	V	VPMM	M	C
True ID	V	43.60465	55.81395	0	0.581395	True ID	V	43.60465	55.81395	0	0.581395
	VpMM	22.02381	77.97619	0	0		VpMM	22.02381	77.97619	0	0
	M	0	92.30769	0	7.692308		M	0	92.30769	0	7.692308
	C	0	80	0	20		C	0	80	0	20

Sample 2

Class name abbreviations:

- V: Vitrinite
- VPMM: Vitrinite plus mineral matter
- P: Pyrite
- CIV: Cracks in vitrinite
- C: Calcite
- M: Mudstone

baseline							
Right %	Wrong %	Predicted ID					
46.9697	53.0303	V	VPMM	P	CIV	C	M
True ID	V	31.81818	31.81818	0	15.58442	1.298701	1.298701
	VPMM	28.09524	62.85714	0	8.095238	0.47619	0.47619
	P	0	0	0	31.25	18.75	50
	CIV	0	20	0	40	40	0
	C	0	50	0	25	25	0
	M	0	28.57143	0	42.85714	0	28.57143

k-means alternative A							
Right %	Wrong %	Predicted ID					
48.48485	51.51515	V	VPMM	P	CIV	C	M
True ID	V	31.81818	31.81818	1.298701	15.58442	1.298701	0
	VPMM	28.09524	62.85714	0.47619	8.095238	0.47619	0
	P	0	0	50	31.25	18.75	0
	CIV	0	20	0	40	40	0
	C	0	50	0	25	25	0
	M	0	28.57143	28.57143	42.85714	0	0

k-means alternative B							
Right %	Wrong %	Predicted ID					
48.73737	51.26263	V	VPMM	P	CIV	C	M
True ID	V	31.81818	31.81818	1.948052	15.58442	0.649351	0
	VPMM	28.09524	62.85714	0.47619	8.095238	0.47619	0
	P	0	0	56.25	31.25	12.5	0
	CIV	0	20	20	40	20	0
	C	0	50	0	25	25	0
	M	0	28.57143	28.57143	42.85714	0	0

k-means alternative C							
Right %	Wrong %	Predicted ID					
49.49495	50.50505	V	VPMM	P	CIV	C	M
True ID	V	31.81818	31.81818	1.948052	9.74026	0.649351	5.844156
	VPMM	28.09524	62.85714	0.47619	4.761905	0.47619	3.333333
	P	0	0	56.25	0	12.5	31.25
	CIV	0	20	20	40	20	0
	C	0	50	0	0	25	25
	M	0	28.57143	28.57143	0	0	42.85714

k-means alternative D							
Right %	Wrong %	Predicted ID					
49.74747	50.25253	V	VPMM	P	CIV	C	M
True ID	V	27.27273	27.27273	0	16.23377	1.298701	1.298701
	VPMM	20.95238	70	0	8.095238	0.952381	0
	P	0	0	0	31.25	18.75	50
	CIV	0	20	0	60	20	0
	C	0	50	0	25	25	0
	M	0	28.57143	0	14.28571	0	57.14286

k-means alternative E							
Right %	Wrong %	Predicted ID					
52.27273	47.72727	V	VPMM	P	CIV	C	M
True ID	V	27.27273	27.27273	1.298701	15.58442	1.298701	1.298701
	VPMM	20.95238	70	0	6.666667	0.952381	1.428571
	P	0	0	56.25	25	12.5	6.25
	CIV	0	20	0	60	20	0
	C	0	50	0	25	25	0
	M	0	28.57143	0	0	0	71.42857

k-means alternative F							
Right %	Wrong %	Predicted ID					
49.74747	50.25253	V	VPMM	P	CIV	C	M
True ID	V	30.51948	30.51948	1.298701	14.93506	1.948052	0
	VPMM	25.71429	65.2381	0.952381	7.619048	0.47619	0
	P	0	0	62.5	18.75	18.75	0
	CIV	0	20	20	40	20	0
	C	0	50	0	25	25	0
	M	0	28.57143	57.14286	14.28571	0	0

k-means alternative G							
Right %	Wrong %	Predicted ID					
50	50	V	VPMM	P	CIV	C	M
True ID	V	30.51948	30.51948	1.298701	9.74026	1.298701	8.441558
	VPMM	25.71429	64.7619	0.47619	4.761905	0.47619	3.809524
	P	0	0	56.25	0	12.5	31.25
	CIV	0	20	20	40	20	0
	C	0	50	0	0	25	25
	M	0	28.57143	28.57143	0	0	42.85714

k-medians baseline							
Right %	Wrong %	Predicted ID					
49.74747	50.25253	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	0	13.63636	0	1.298701
	VPMM	15.71429	76.19048	0	8.095238	0	0
	P	0	0	0	37.5	0	62.5
	CIV	0	20	0	60	0	20
	C	0	50	0	50	0	0
	M	0	28.57143	0	42.85714	0	28.57143

k-medians alternative A							
Right %	Wrong %	Predicted ID					
51.76768	48.23232	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	1.298701	13.63636	0	0
	VPMM	15.71429	76.19048	0	8.095238	0	0
	P	0	0	62.5	37.5	0	0
	CIV	0	20	20	60	0	0
	C	0	50	0	50	0	0
	M	0	28.57143	28.57143	42.85714	0	0

k-medians alternative B							
Right %	Wrong %	Predicted ID					
51.76768	48.23232	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	1.298701	13.63636	0	0
	VPMM	15.71429	76.19048	0	8.095238	0	0
	P	0	0	62.5	37.5	0	0
	CIV	0	20	20	60	0	0
	C	0	50	0	50	0	0
	M	0	28.57143	28.57143	42.85714	0	0

k-medians alternative C							
Right %	Wrong %	Predicted ID					
52.27273	47.72727	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	1.298701	9.74026	0	3.896104
	VPMM	15.71429	76.19048	0	4.285714	0	3.809524
	P	0	0	62.5	0	0	37.5
	CIV	0	20	20	40	0	20
	C	0	50	0	0	0	50
	M	0	28.57143	28.57143	0	0	42.85714

k-medians alternative D

Right %	Wrong %	Predicted ID					
49.49495	50.50505	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	0	13.63636	0.649351	0.649351
	VPMM	16.19048	74.7619	0	8.571429	0.47619	0
	P	0	0	0	37.5	6.25	56.25
	CIV	0	20	0	60	20	0
	C	0	50	0	50	0	0
	M	0	28.57143	0	14.28571	0	57.14286

k-medians alternative E

Right %	Wrong %	Predicted ID					
52.0202	47.9798	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	0.649351	13.63636	0.649351	0
	VPMM	16.19048	74.7619	0	6.666667	0.47619	1.904762
	P	0	0	56.25	25	6.25	12.5
	CIV	0	20	0	60	20	0
	C	0	50	0	50	0	0
	M	0	28.57143	0	0	0	71.42857

k-medians alternative F

Right %	Wrong %	Predicted ID					
52.0202	47.9798	V	VPMM	P	CIV	C	M
True ID	V	20.12987	20.12987	1.298701	14.28571	0	0
	VPMM	13.80952	77.14286	0.47619	8.571429	0	0
	P	0	0	62.5	37.5	0	0
	CIV	0	20	20	60	0	0
	C	0	50	0	50	0	0
	M	0	28.57143	28.57143	42.85714	0	0

k-medians alternative G

Right %	Wrong %	Predicted ID					
52.52525	47.47475	V	VPMM	P	CIV	C	M
True ID	V	20.12987	20.12987	1.298701	10.38961	0	3.896104
	VPMM	13.80952	77.14286	0	5.238095	0	3.809524
	P	0	0	62.5	0	0	37.5
	CIV	0	20	20	40	0	20
	C	0	50	0	0	0	50
	M	0	28.57143	28.57143	0	0	42.85714

mean shift baseline

Right %	Wrong %	Predicted ID					
50	50	V	VPMM	P	CIV	C	M
True ID	V	18.83117	18.83117	0	17.53247	1.298701	0
	VPMM	12.85714	78.09524	0	8.095238	0.47619	0.47619
	P	0	0	0	43.75	6.25	50
	CIV	0	20	0	60	20	0
	C	0	25	0	50	0	25
	M	0	28.57143	14.28571	28.57143	0	28.57143

mean shift alternative A

Right %	Wrong %	Predicted ID					
52.77778	47.22222	V	VPMM	P	CIV	C	M
True ID	V	18.83117	18.83117	0.649351	15.58442	0.649351	1.948052
	VPMM	12.85714	78.09524	0.47619	8.095238	0.47619	0
	P	0	0	62.5	25	6.25	6.25
	CIV	0	20	0	60	20	0
	C	0	25	25	50	0	0
	M	0	28.57143	14.28571	14.28571	0	42.85714

mean shift alternative B

Right %	Wrong %	Predicted ID					
51.26263	48.73737	V	VPMM	P	CIV	C	M
True ID	V	18.18182	18.18182	0	17.53247	0.649351	0.649351
	VPMM	11.42857	79.52381	0	8.095238	0.47619	0.47619
	P	0	0	18.75	43.75	6.25	31.25
	CIV	0	20	0	60	20	0
	C	0	25	0	50	0	25
	M	0	28.57143	14.28571	28.57143	0	28.57143

mean shift alternative C

Right %	Wrong %	Predicted ID					
52.27273	47.72727	V	VPMM	P	CIV	C	M
True ID	V	18.18182	18.18182	0.649351	15.58442	0.649351	1.948052
	VPMM	11.42857	79.52381	0	8.095238	0.47619	0.47619
	P	0	0	37.5	37.5	6.25	18.75
	CIV	0	20	0	60	20	0
	C	0	25	0	50	0	25
	M	0	28.57143	14.28571	14.28571	0	42.85714

mean shift alternative D

Right %	Wrong %	Predicted ID					
50.50505	49.49495	V	VPMM	P	CIV	C	M
True ID	V	19.48052	19.48052	0	18.83117	0.649351	1.298701
	VPMM	14.28571	77.14286	0	7.619048	0.952381	0
	P	0	0	6.25	37.5	18.75	37.5
	CIV	0	20	0	60	20	0
	C	0	25	0	50	25	0
	M	0	28.57143	0	28.57143	0	42.85714

mean shift alternative E

Right %	Wrong %	Predicted ID					
52.77778	47.22222	V	VPMM	P	CIV	C	M
True ID	V	19.48052	19.48052	0.649351	18.18182	0.649351	1.298701
	VPMM	14.28571	77.14286	0	7.619048	0.952381	0
	P	0	0	50	25	12.5	12.5
	CIV	0	20	0	60	20	0
	C	0	25	0	50	25	0
	M	0	28.57143	0	0	0	71.42857

mean shift alternative F

Right %	Wrong %	Predicted ID					
50.75758	49.24242	V	VPMM	P	CIV	C	M
True ID	V	19.48052	19.48052	0	18.18182	0.649351	1.298701
	VPMM	14.28571	77.14286	0	7.619048	0.47619	0.47619
	P	0	0	18.75	43.75	6.25	31.25
	CIV	0	20	0	60	20	0
	C	0	25	0	75	0	0
	M	0	28.57143	0	28.57143	0	42.85714

mean shift alternative G

Right %	Wrong %	Predicted ID					
52.52525	47.47475	V	VPMM	P	CIV	C	M
True ID	V	19.48052	19.48052	0	18.18182	0.649351	1.298701
	VPMM	14.28571	77.14286	0	7.619048	0.47619	0.47619
	P	0	0	50	37.5	0	12.5
	CIV	0	20	0	60	20	0
	C	0	25	0	75	0	0
	M	0	28.57143	0	0	0	71.42857

PSO baseline

Right %	Wrong %	Predicted ID					
52.52525	47.47475	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	0	4.545455	0.649351	1.948052
	VPMM	15.2381	80.95238	0	0.47619	0	3.333333
	P	0	18.75	6.25	0	6.25	68.75
	CIV	0	40	0	20	0	40
	C	0	75	0	0	0	25
	M	0	42.85714	0	0	0	57.14286

PSO alternative A

Right %	Wrong %	Predicted ID					
52.52525	47.47475	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	0	4.545455	0.649351	1.948052
	VPMM	15.2381	80.95238	0	0.47619	0	3.333333
	P	0	18.75	6.25	0	6.25	68.75
	CIV	0	40	0	20	0	40
	C	0	75	0	0	0	25
	M	0	42.85714	0	0	0	57.14286

PSO alternative B

Right %	Wrong %	Predicted ID					
54.54545	45.45455	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	2.597403	4.545455	0	0
	VPMM	15.2381	80.95238	3.333333	0.47619	0	0
	P	0	18.75	81.25	0	0	0
	CIV	0	40	40	20	0	0
	C	0	75	25	0	0	0
	M	0	42.85714	57.14286	0	0	0

PSO alternative C

Right %	Wrong %	Predicted ID					
54.54545	45.45455	V	VPMM	P	CIV	C	M
True ID	V	20.77922	20.77922	2.597403	4.545455	0	0
	VPMM	15.2381	80.95238	3.333333	0.47619	0	0
	P	0	18.75	81.25	0	0	0
	CIV	0	40	40	20	0	0
	C	0	75	25	0	0	0
	M	0	42.85714	57.14286	0	0	0

PSO alternative D							
Right %	Wrong %	Predicted ID					
53.28283	46.71717	V	VPMM	P	CIV	C	M
True ID	V	20.12987	20.12987	0	7.792208	1.298701	1.298701
	VPMM	12.85714	82.85714	0	2.857143	0.952381	0.47619
	P	0	18.75	6.25	0	25	50
	CIV	0	40	0	20	40	0
	C	0	75	0	25	0	0
	M	0	42.85714	0	0	0	57.14286

PSO alternative E							
Right %	Wrong %	Predicted ID					
54.0404	45.9596	V	VPMM	P	CIV	C	M
True ID	V	20.12987	20.12987	0	7.792208	1.298701	1.298701
	VPMM	12.85714	82.85714	0	2.857143	0.952381	0.47619
	P	0	18.75	25	0	25	31.25
	CIV	0	40	0	20	40	0
	C	0	75	0	25	0	0
	M	0	42.85714	0	0	0	57.14286

PSO alternative F							
Right %	Wrong %	Predicted ID					
54.29293	45.70707	V	VPMM	P	CIV	C	M
True ID	V	21.42857	21.42857	1.948052	7.792208	0	0
	VPMM	16.19048	80	3.333333	0.47619	0	0
	P	0	18.75	81.25	0	0	0
	CIV	0	40	40	20	0	0
	C	0	75	25	0	0	0
	M	0	42.85714	57.14286	0	0	0

PSO alternative G							
Right %	Wrong %	Predicted ID					
54.29293	45.70707	V	VPMM	P	CIV	C	M
True ID	V	21.42857	21.42857	1.948052	7.792208	0	0
	VPMM	16.19048	80	3.333333	0.47619	0	0
	P	0	18.75	81.25	0	0	0
	CIV	0	40	40	20	0	0
	C	0	75	25	0	0	0
	M	0	42.85714	57.14286	0	0	0

Sample 3

Class name abbreviations:

- V: Vitrinite
- I: Inertinite
- P: Pyrite
- Cr: Cracks
- Ca: Calcite

k-means baseline						
Right %	Wrong %	Predicted ID				
34.47368	65.52632	V	I	P	Cr	Ca
True ID	V	55.29412	55.29412	0.588235	10.58824	31.17647
	I	9.52381	9.52381	2.380952	12.5	66.07143
	P	0	6.060606	60.60606	27.27273	6.060606
	Cr	0	42.85714	14.28571	0	42.85714
	Ca	0	0	0	50	50

k-means alternative A						
Right %	Wrong %	Predicted ID				
33.42105	66.57895	V	I	P	Cr	Ca
True ID	V	30	30	0.588235	10.58824	19.41176
	I	0.595238	32.7381	2.380952	12.5	51.78571
	P	0	9.090909	60.60606	27.27273	3.030303
	Cr	0	57.14286	14.28571	0	28.57143
	Ca	0	0	0	50	50

k-means alternative B						
Right %	Wrong %	Predicted ID				
62.10526	37.89474	V	I	P	Cr	Ca
True ID	V	55.29412	55.29412	0.588235	10	0
	I	9.52381	70.83333	2.380952	17.2619	0
	P	0	27.27273	60.60606	12.12121	0
	Cr	0	42.85714	14.28571	42.85714	0
	Ca	0	50	0	50	0

k-means alternative C						
Right %	Wrong %	Predicted ID				
54.73684	45.26316	V	I	P	Cr	Ca
True ID	V	30	30	0.588235	10	0
	I	0.595238	79.7619	2.380952	17.2619	0
	P	0	27.27273	60.60606	12.12121	0
	Cr	0	42.85714	14.28571	42.85714	0
	Ca	0	50	0	50	0

k-means alternative D						
Right %	Wrong %	Predicted ID				
42.36842	57.63158	V	I	P	Cr	Ca
True ID	V	53.52941	53.52941	0.588235	10	20
	I	7.738095	28.57143	2.380952	13.09524	48.21429
	P	0	0	60.60606	33.33333	6.060606
	Cr	0	14.28571	14.28571	28.57143	42.85714
	Ca	0	50	0	50	0

k-means alternative E						
Right %	Wrong %	Predicted ID				
48.94737	51.05263	V	I	P	Cr	Ca
True ID	V	44.11765	44.11765	0.588235	10	8.823529
	I	1.190476	52.97619	2.380952	13.09524	30.35714
	P	0	3.030303	60.60606	33.33333	3.030303
	Cr	0	42.85714	14.28571	28.57143	14.28571
	Ca	0	50	0	50	0

k-means alternative F						
Right %	Wrong %	Predicted ID				
62.63158	37.36842	V	I	P	Cr	Ca
True ID	V	56.47059	56.47059	0.588235	8.823529	0
	I	8.928571	72.02381	2.97619	16.07143	0
	P	0	12.12121	60.60606	27.27273	0
	Cr	0	57.14286	28.57143	14.28571	0
	Ca	0	50	0	50	0

k-means alternative G						
Right %	Wrong %	Predicted ID				
60.78947	39.21053	V	I	P	Cr	Ca
True ID	V	44.11765	44.11765	0.588235	8.823529	0
	I	1.190476	79.7619	2.97619	16.07143	0
	P	0	9.090909	63.63636	27.27273	0
	Cr	0	57.14286	28.57143	14.28571	0
	Ca	0	50	0	50	0

k-medians baseline						
Right %	Wrong %	Predicted ID				
41.57895	58.42105	V	I	P	Cr	Ca
True ID	V	53.52941	53.52941	1.764706	8.235294	21.17647
	I	8.928571	24.40476	4.166667	11.30952	51.19048
	P	0	15.15152	75.75758	3.030303	6.060606
	Cr	14.28571	42.85714	14.28571	0	28.57143
	Ca	0	0	0	50	50

k-medians alternative A						
Right %	Wrong %	Predicted ID				
48.68421	51.31579	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	1.764706	8.235294	11.17647
	I	2.380952	52.38095	4.166667	11.30952	29.7619
	P	0	18.18182	75.75758	3.030303	3.030303
	Cr	0	71.42857	14.28571	0	14.28571
	Ca	0	0	0	50	50

k-medians alternative B						
Right %	Wrong %	Predicted ID				
59.73684	40.26316	V	I	P	Cr	Ca
True ID	V	53.52941	53.52941	1.764706	11.17647	0
	I	8.928571	64.28571	4.166667	22.61905	0
	P	0	6.060606	75.75758	18.18182	0
	Cr	14.28571	28.57143	14.28571	42.85714	0
	Ca	0	50	0	50	0

k-medians alternative C						
Right %	Wrong %	Predicted ID				
57.36842	42.63158	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	1.764706	11.17647	0
	I	2.380952	70.83333	4.166667	22.61905	0
	P	0	6.060606	75.75758	18.18182	0
	Cr	0	42.85714	14.28571	42.85714	0
	Ca	0	50	0	50	0

k-medians alternative D						
Right %	Wrong %	Predicted ID				
45.26316	54.73684	V	I	P	Cr	Ca
True ID	V	50.58824	50.58824	1.176471	10	18.82353
	I	4.761905	36.90476	2.97619	16.07143	39.28571
	P	0	0	66.66667	27.27273	6.060606
	Cr	0	14.28571	14.28571	28.57143	42.85714
	Ca	0	50	0	50	0

k-medians alternative E						
Right %	Wrong %	Predicted ID				
50	50	V	I	P	Cr	Ca
True ID	V	44.70588	44.70588	1.176471	10	8.235294
	I	2.97619	53.57143	2.97619	16.07143	24.40476
	P	0	3.030303	66.66667	27.27273	3.030303
	Cr	0	42.85714	14.28571	28.57143	14.28571
	Ca	0	50	0	50	0

k-medians alternative F						
Right %	Wrong %	Predicted ID				
60.52632	39.47368	V	I	P	Cr	Ca
True ID	V	51.76471	51.76471	1.764706	11.17647	0
	I	4.761905	69.04762	4.761905	21.42857	0
	P	0	6.060606	75.75758	18.18182	0
	Cr	14.28571	28.57143	42.85714	14.28571	0
	Ca	0	50	0	50	0

k-medians alternative G						
Right %	Wrong %	Predicted ID				
60.26316	39.73684	V	I	P	Cr	Ca
True ID	V	48.82353	48.82353	1.764706	10	0
	I	4.761905	70.83333	4.761905	19.64286	0
	P	0	6.060606	78.78788	15.15152	0
	Cr	0	42.85714	42.85714	14.28571	0
	Ca	0	50	0	50	0

mean shift baseline						
Right %	Wrong %	Predicted ID				
39.21053	60.78947	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	0.588235	9.411765	21.76471
	I	1.190476	35.71429	1.785714	8.333333	52.97619
	P	0	6.060606	51.51515	30.30303	12.12121
	Cr	14.28571	28.57143	14.28571	14.28571	28.57143
	Ca	0	50	0	50	0

mean shift alternative A						
Right %	Wrong %	Predicted ID				
39.21053	60.78947	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	0.588235	9.411765	21.17647
	I	1.190476	35.71429	1.785714	8.333333	52.97619
	P	0	6.060606	51.51515	30.30303	12.12121
	Cr	14.28571	28.57143	14.28571	14.28571	28.57143
	Ca	0	50	0	50	0

mean shift alternative B						
Right %	Wrong %	Predicted ID				
54.73684	45.26316	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	4.705882	4.705882	7.647059
	I	1.190476	69.04762	2.97619	7.142857	19.64286
	P	0	6.060606	60.60606	15.15152	18.18182
	Cr	14.28571	42.85714	14.28571	14.28571	14.28571
	Ca	0	50	0	50	0

mean shift alternative C						
Right %	Wrong %	Predicted ID				
55	45	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	4.705882	4.705882	7.647059
	I	1.190476	69.04762	2.97619	7.142857	19.64286
	P	0	6.060606	63.63636	15.15152	15.15152
	Cr	14.28571	42.85714	14.28571	14.28571	14.28571
	Ca	0	50	0	50	0

mean shift alternative D						
Right %	Wrong %	Predicted ID				
38.42105	61.57895	V	I	P	Cr	Ca
True ID	V	42.94118	42.94118	0.588235	11.17647	25.88235
	I	1.785714	31.54762	1.190476	13.09524	52.38095
	P	0	0	54.54545	33.33333	12.12121
	Cr	14.28571	14.28571	14.28571	28.57143	28.57143
	Ca	0	50	0	50	0

mean shift alternative E						
Right %	Wrong %	Predicted ID				
52.10526	47.89474	V	I	P	Cr	Ca
True ID	V	41.76471	41.76471	0.588235	11.17647	9.411765
	I	0.595238	63.09524	1.785714	12.5	22.02381
	P	0	3.030303	57.57576	30.30303	9.090909
	Cr	0	42.85714	14.28571	28.57143	14.28571
	Ca	0	50	0	50	0

mean shift alternative F						
Right %	Wrong %	Predicted ID				
54.73684	45.26316	V	I	P	Cr	Ca
True ID	V	42.94118	42.94118	1.176471	10.58824	8.235294
	I	1.785714	66.07143	1.785714	9.52381	20.83333
	P	0	6.060606	66.66667	21.21212	6.060606
	Cr	14.28571	42.85714	14.28571	28.57143	0
	Ca	0	50	0	50	0

mean shift alternative G						
Right %	Wrong %	Predicted ID				
53.94737	46.05263	V	I	P	Cr	Ca
True ID	V	39.41176	39.41176	1.176471	10.58824	8.235294
	I	0.595238	67.85714	1.785714	9.52381	20.2381
	P	0	6.060606	66.66667	21.21212	6.060606
	Cr	0	57.14286	14.28571	28.57143	0
	Ca	0	50	0	50	0

PSO baseline						
Right %	Wrong %	Predicted ID				
28.68421	71.31579	V	I	P	Cr	Ca
True ID	V	25.88235	25.88235	0	17.64706	14.11765
	I	1.785714	34.52381	0	41.07143	22.61905
	P	0	3.030303	6.060606	87.87879	3.030303
	Cr	0	0	0	71.42857	28.57143
	Ca	0	50	0	50	0

PSO alternative A						
Right %	Wrong %	Predicted ID				
38.68421	61.31579	V	I	P	Cr	Ca
True ID	V	25.88235	25.88235	0	17.64706	0
	I	1.785714	57.14286	0	41.07143	0
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	28.57143	0	71.42857	0
	Ca	0	50	0	50	0

PSO alternative B						
Right %	Wrong %	Predicted ID				
38.68421	61.31579	V	I	P	Cr	Ca
True ID	V	25.88235	25.88235	0	17.64706	0
	I	1.785714	57.14286	0	41.07143	0
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	28.57143	0	71.42857	0
	Ca	0	50	0	50	0

PSO alternative C						
Right %	Wrong %	Predicted ID				
38.68421	61.31579	V	I	P	Cr	Ca
True ID	V	25.88235	25.88235	0	17.64706	0
	I	1.785714	57.14286	0	41.07143	0
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	28.57143	0	71.42857	0
	Ca	0	50	0	50	0

PSO alternative D						
Right %	Wrong %	Predicted ID				
32.10526	67.89474	V	I	P	Cr	Ca
True ID	V	34.11765	34.11765	0	16.47059	15.88235
	I	2.380952	34.52381	0	30.95238	32.14286
	P	0	3.030303	6.060606	87.87879	3.030303
	Cr	0	14.28571	0	57.14286	28.57143
	Ca	0	50	0	50	0

PSO alternative E						
Right %	Wrong %	Predicted ID				
41.84211	58.15789	V	I	P	Cr	Ca
True ID	V	31.17647	31.17647	0	16.47059	3.529412
	I	1.785714	59.52381	0	30.95238	7.738095
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	42.85714	0	57.14286	0
	Ca	0	50	0	50	0

PSO alternative F						
Right %	Wrong %	Predicted ID				
41.05263	58.94737	V	I	P	Cr	Ca
True ID	V	32.35294	32.35294	0	17.05882	0
	I	2.380952	56.54762	0	41.07143	0
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	42.85714	0	57.14286	0
	Ca	0	50	0	50	0

PSO alternative G						
Right %	Wrong %	Predicted ID				
41.05263	58.94737	V	I	P	Cr	Ca
True ID	V	32.35294	32.35294	0	17.05882	0
	I	2.380952	56.54762	0	41.07143	0
	P	0	6.060606	6.060606	87.87879	0
	Cr	0	42.85714	0	57.14286	0
	Ca	0	50	0	50	0

Sample 4

Class name abbreviations:

- I: Inertinite
- MPM: Macerals plus minerals

k-means baseline			
Right %	Wrong %	Predicted ID	
89.73684	10.26316	I	MPM
True ID	I	0	0
	MPM	0	100

k-means alternative A			
Right %	Wrong %	Predicted ID	
86.31579	13.68421	I	MPM
True ID	I	41.02564	41.02564
	MPM	8.504399	91.4956

k-means alternative B			
Right %	Wrong %	Predicted ID	
86.31579	13.68421	I	MPM
True ID	I	41.02564	41.02564
	MPM	8.504399	91.4956

k-means alternative C			
Right %	Wrong %	Predicted ID	
86.31579	13.68421	I	MPM
True ID	I	41.02564	41.02564
	MPM	8.504399	91.4956

k-means alternative D			
Right %	Wrong %	Predicted ID	
90.26316	9.736842	I	MPM
True ID	I	25.64103	25.64103
	MPM	2.346041	97.65396

k-means alternative E			
Right %	Wrong %	Predicted ID	
89.21053	10.78947	I	MPM
True ID	I	38.46154	38.46154
	MPM	4.985337	95.01466

k-means alternative F			
Right %	Wrong %	Predicted ID	
85.52632	14.47368	I	MPM
True ID	I	43.58974	43.58974
	MPM	9.677419	90.32258

k-means alternative G			
Right %	Wrong %	Predicted ID	
85.52632	14.47368	I	MPM
True ID	I	43.58974	43.58974
	MPM	9.677419	90.32258

k-medians baseline			
Right %	Wrong %	Predicted ID	
89.73684	10.26316	I	MPM
True ID	I	0	0
	MPM	0	100

k-medians alternative A			
Right %	Wrong %	Predicted ID	
86.57895	13.42105	I	MPM
True ID	I	53.84615	53.84615
	MPM	9.677419	90.32258

k-medians alternative B			
Right %	Wrong %	Predicted ID	
86.57895	13.42105	I	MPM
True ID	I	53.84615	53.84615
	MPM	9.677419	90.32258

k-medians alternative C			
Right %	Wrong %	Predicted ID	
86.57895	13.42105	I	MPM
True ID	I	53.84615	53.84615
	MPM	9.677419	90.32258

k-medians alternative D			
Right %	Wrong %	Predicted ID	
91.57895	8.421053	I	MPM
True ID	I	25.64103	25.64103
	MPM	0.879765	99.12023

k-medians alternative E			
Right %	Wrong %	Predicted ID	
88.42105	11.57895	I	MPM
True ID	I	38.46154	38.46154
	MPM	5.865103	94.1349

k-medians alternative F			
Right %	Wrong %	Predicted ID	
85	15	I	MPM
True ID	I	58.97436	58.97436
	MPM	12.02346	87.97654

k-medians alternative G			
Right %	Wrong %	Predicted ID	
85	15	I	MPM
True ID	I	58.97436	58.97436
	MPM	12.02346	87.97654

mean shift baseline			
Right %	Wrong %	Predicted ID	
91.05263	8.947368	I	MPM
True ID	I	20.51282	20.51282
	MPM	0.879765	99.12023

mean shift alternative A			
Right %	Wrong %	Predicted ID	
89.47368	10.52632	I	MPM
True ID	I	30.76923	30.76923
	MPM	3.812317	96.18768

mean shift alternative B			
Right %	Wrong %	Predicted ID	
91.05263	8.947368	I	MPM
True ID	I	20.51282	20.51282
	MPM	0.879765	99.12023

mean shift alternative C			
Right %	Wrong %	Predicted ID	
91.05263	8.947368	I	MPM
True ID	I	20.51282	20.51282
	MPM	0.879765	99.12023

mean shift alternative D			
Right %	Wrong %	Predicted ID	
92.10526	7.894737	I	MPM
True ID	I	28.20513	28.20513
	MPM	0.58651	99.41349

mean shift alternative E			
Right %	Wrong %	Predicted ID	
90.52632	9.473684	I	MPM
True ID	I	35.89744	35.89744
	MPM	3.225806	96.77419

mean shift alternative F			
Right %	Wrong %	Predicted ID	
92.36842	7.631579	I	MPM
True ID	I	30.76923	30.76923
	MPM	0.58651	99.41349

mean shift alternative G			
Right %	Wrong %	Predicted ID	
92.10526	7.894737	I	MPM
True ID	I	30.76923	30.76923
	MPM	0.879765	99.12023

PSO baseline			
Right %	Wrong %	Predicted ID	
89.73684	10.26316	I	MPM
True ID	I	0	0
	MPM	0	100

PSO alternative A			
Right %	Wrong %	Predicted ID	
89.73684	10.26316	I	MPM
True ID	I	0	0
	MPM	0	100

PSO alternative B			
Right %	Wrong %	Predicted ID	
86.31579	13.68421	I	MPM
True ID	I	10.25641	10.25641
	MPM	4.985337	95.01466

PSO alternative C			
Right %	Wrong %	Predicted ID	
10.26316	89.73684	I	MPM
True ID	I	100	100
	MPM	100	0

PSO alternative D			
Right %	Wrong %	Predicted ID	
90.26316	9.736842	I	MPM
True ID	I	12.82051	12.82051
	MPM	0.879765	99.12023

PSO alternative E			
Right %	Wrong %	Predicted ID	
90.26316	9.736842	I	MPM
True ID	I	12.82051	12.82051
	MPM	0.879765	99.12023

PSO alternative F			
Right %	Wrong %	Predicted ID	
86.57895	13.42105	I	MPM
True ID	I	12.82051	12.82051
	MPM	4.985337	95.01466

PSO alternative G			
Right %	Wrong %	Predicted ID	
10.26316	89.73684	I	MPM
True ID	I	100	100
	MPM	100	0

Sample 5

- I: Inertinite
- V: Vitrinite
- C: Calcite
- MPM: Minerals plus macerals

k-means baseline					
Right %	Wrong %	Predicted ID			
50	50	I	V	C	MPM
True ID	I	57.69231	13.46154	6.730769	22.11538
	V	71.15385	19.23077	6.730769	2.884615
	C	33.33333	8.333333	58.33333	0
	MPM	29.60526	0	5.263158	65.13158

k-means alternative A					
Right %	Wrong %	Predicted ID			
50	50	I	V	C	MPM
True ID	I	57.69231	13.46154	6.730769	22.11538
	V	71.15385	19.23077	6.730769	2.884615
	C	33.33333	8.333333	58.33333	0
	MPM	29.60526	0	5.263158	65.13158

k-means alternative B					
Right %	Wrong %	Predicted ID			
48.65591	51.34409	I	V	C	MPM
True ID	I	21.15385	13.46154	6.730769	58.65385
	V	53.84615	19.23077	6.730769	20.19231
	C	8.333333	8.333333	58.33333	25
	MPM	7.894737	0	5.263158	86.84211

k-means alternative C					
Right %	Wrong %	Predicted ID			
48.65591	51.34409	I	V	C	MPM
True ID	I	21.15385	13.46154	6.730769	58.65385
	V	53.84615	19.23077	6.730769	20.19231
	C	8.333333	8.333333	58.33333	25
	MPM	7.894737	0	5.263158	86.84211

k-means alternative D					
Right %	Wrong %	Predicted ID			
49.46237	50.53763	I	V	C	MPM
True ID	I	47.11538	13.46154	3.846154	35.57692
	V	66.34615	22.11538	10.57692	0.961538
	C	25	8.333333	58.33333	8.333333
	MPM	27.63158	0	3.289474	69.07895

k-means alternative E					
Right %	Wrong %	Predicted ID			
49.46237	50.53763	I	V	C	MPM
True ID	I	47.11538	13.46154	3.846154	35.57692
	V	66.34615	22.11538	10.57692	0.961538
	C	25	8.333333	58.33333	8.333333
	MPM	27.63158	0	3.289474	69.07895

k-means alternative F					
Right %	Wrong %	Predicted ID			
47.04301	52.95699	I	V	C	MPM
True ID	I	29.80769	15.38462	8.653846	46.15385
	V	58.65385	21.15385	11.53846	8.653846
	C	16.66667	8.333333	66.66667	8.333333
	MPM	19.07895	0	5.921053	75

k-means alternative G					
Right %	Wrong %	Predicted ID			
47.04301	52.95699	I	V	C	MPM
True ID	I	29.80769	15.38462	8.653846	46.15385
	V	58.65385	21.15385	11.53846	8.653846
	C	16.66667	8.333333	66.66667	8.333333
	MPM	19.07895	0	5.921053	75

k-medians baseline					
Right %	Wrong %	Predicted ID			
50.80645	49.19355	I	V	C	MPM
True ID	I	46.15385	12.5	7.692308	33.65385
	V	68.26923	17.30769	8.653846	5.769231
	C	16.66667	8.333333	58.33333	16.66667
	MPM	15.78947	0	7.894737	76.31579

k-medians alternative A					
Right %	Wrong %	Predicted ID			
50.80645	49.19355	I	V	C	MPM
True ID	I	46.15385	12.5	7.692308	33.65385
	V	68.26923	17.30769	8.653846	5.769231
	C	16.66667	8.333333	58.33333	16.66667
	MPM	15.78947	0	7.894737	76.31579

k-medians alternative B					
Right %	Wrong %	Predicted ID			
50.80645	49.19355	I	V	C	MPM
True ID	I	46.15385	12.5	7.692308	33.65385
	V	68.26923	17.30769	8.653846	5.769231
	C	16.66667	8.333333	58.33333	16.66667
	MPM	15.78947	0	7.894737	76.31579

k-medians alternative C					
Right %	Wrong %	Predicted ID			
50.80645	49.19355	I	V	C	MPM
True ID	I	46.15385	12.5	7.692308	33.65385
	V	68.26923	17.30769	8.653846	5.769231
	C	16.66667	8.333333	58.33333	16.66667
	MPM	15.78947	0	7.894737	76.31579

k-medians alternative D					
Right %	Wrong %	Predicted ID			
51.07527	48.92473	I	V	C	MPM
True ID	I	50	12.5	2.884615	34.61538
	V	63.46154	22.11538	12.5	1.923077
	C	25	8.333333	58.33333	8.333333
	MPM	26.31579	0	2.631579	71.05263

k-medians alternative E					
Right %	Wrong %	Predicted ID			
51.07527	48.92473	I	V	C	MPM
True ID	I	50	12.5	2.884615	34.61538
	V	63.46154	22.11538	12.5	1.923077
	C	25	8.333333	58.33333	8.333333
	MPM	26.31579	0	2.631579	71.05263

k-medians alternative F					
Right %	Wrong %	Predicted ID			
52.95699	47.04301	I	V	C	MPM
True ID	I	44.23077	16.34615	9.615385	29.80769
	V	50	35.57692	12.5	1.923077
	C	16.66667	8.333333	75	0
	MPM	23.02632	0.657895	7.236842	69.07895

k-medians alternative G					
Right %	Wrong %	Predicted ID			
52.95699	47.04301	I	V	C	MPM
True ID	I	44.23077	16.34615	9.615385	29.80769
	V	50	35.57692	12.5	1.923077
	C	16.66667	8.333333	75	0
	MPM	23.02632	0.657895	7.236842	69.07895

mean shift baseline					
Right %	Wrong %	Predicted ID			
43.8172	56.1828	I	V	C	MPM
True ID	I	65.38462	13.46154	0	21.15385
	V	60.57692	27.88462	4.807692	6.730769
	C	41.66667	16.66667	25	16.66667
	MPM	57.89474	0.657895	0	41.44737

mean shift alternative A					
Right %	Wrong %	Predicted ID			
43.8172	56.1828	I	V	C	MPM
True ID	I	65.38462	13.46154	0	21.15385
	V	60.57692	27.88462	4.807692	6.730769
	C	41.66667	16.66667	25	16.66667
	MPM	57.89474	0.657895	0	41.44737

mean shift alternative B					
Right %	Wrong %	Predicted ID			
43.8172	56.1828	I	V	C	MPM
True ID	I	64.42308	14.42308	0	21.15385
	V	62.5	29.80769	0.961538	6.730769
	C	50	16.66667	16.66667	16.66667
	MPM	57.89474	0.657895	0	41.44737

mean shift alternative C					
Right %	Wrong %	Predicted ID			
43.8172	56.1828	I	V	C	MPM
True ID	I	64.42308	14.42308	0	21.15385
	V	62.5	29.80769	0.961538	6.730769
	C	50	16.66667	16.66667	16.66667
	MPM	57.89474	0.657895	0	41.44737

mean shift alternative D					
Right %	Wrong %	Predicted ID			
45.16129	54.83871	I	V	C	MPM
True ID	I	59.61538	15.38462	0.961538	24.03846
	V	58.65385	29.80769	7.692308	3.846154
	C	33.33333	16.66667	41.66667	8.333333
	MPM	53.94737	0	0	46.05263

mean shift alternative E					
Right %	Wrong %	Predicted ID			
45.16129	54.83871	I	V	C	MPM
True ID	I	59.61538	15.38462	0.961538	24.03846
	V	58.65385	29.80769	7.692308	3.846154
	C	33.33333	16.66667	41.66667	8.333333
	MPM	53.94737	0	0	46.05263

mean shift alternative F					
Right %	Wrong %	Predicted ID			
45.69892	54.30108	I	V	C	MPM
True ID	I	61.53846	15.38462	0	23.07692
	V	58.65385	32.69231	5.769231	2.884615
	C	33.33333	16.66667	41.66667	8.333333
	MPM	55.26316	0.657895	0	44.07895

mean shift alternative G					
Right %	Wrong %	Predicted ID			
45.69892	54.30108	I	V	C	MPM
True ID	I	61.53846	15.38462	0	23.07692
	V	58.65385	32.69231	5.769231	2.884615
	C	33.33333	16.66667	41.66667	8.333333
	MPM	55.26316	0.657895	0	44.07895

PSO baseline					
Right %	Wrong %	Predicted ID			
32.7957	67.2043	I	V	C	MPM
True ID	I	95.19231	4.807692	0	0
	V	80.76923	18.26923	0.961538	0
	C	75	8.333333	16.66667	0
	MPM	98.68421	0	0	1.315789

PSO alternative A					
Right %	Wrong %	Predicted ID			
32.7957	67.2043	I	V	C	MPM
True ID	I	95.19231	4.807692	0	0
	V	80.76923	18.26923	0.961538	0
	C	75	8.333333	16.66667	0
	MPM	98.68421	0	0	1.315789

PSO alternative B					
Right %	Wrong %	Predicted ID			
31.98925	68.01075	I	V	C	MPM
True ID	I	95.19231	4.807692	0	0
	V	80.76923	18.26923	0.961538	0
	C	83.33333	8.333333	8.333333	0
	MPM	100	0	0	0

PSO alternative C					
Right %	Wrong %	Predicted ID			
31.98925	68.01075	I	V	C	MPM
True ID	I	95.19231	4.807692	0	0
	V	80.76923	18.26923	0.961538	0
	C	83.33333	8.333333	8.333333	0
	MPM	100	0	0	0

PSO alternative D					
Right %	Wrong %	Predicted ID			
32.7957	67.2043	I	V	C	MPM
True ID	I	94.23077	5.769231	0	0
	V	79.80769	19.23077	0.961538	0
	C	75	8.333333	16.66667	0
	MPM	98.68421	0	0	1.315789

PSO alternative E					
Right %	Wrong %	Predicted ID			
32.7957	67.2043	I	V	C	MPM
True ID	I	94.23077	5.769231	0	0
	V	79.80769	19.23077	0.961538	0
	C	75	8.333333	16.66667	0
	MPM	98.68421	0	0	1.315789

PSO alternative F					
Right %	Wrong %	Predicted ID			
32.25806	67.74194	I	V	C	MPM
True ID	I	94.23077	5.769231	0	0
	V	78.84615	20.19231	0.961538	0
	C	83.33333	8.333333	8.333333	0
	MPM	100	0	0	0

PSO alternative G					
Right %	Wrong %	Predicted ID			
32.25806	67.74194	I	V	C	MPM
True ID	I	94.23077	5.769231	0	0
	V	78.84615	20.19231	0.961538	0
	C	83.33333	8.333333	8.333333	0
	MPM	100	0	0	0