

3-22-2018

## Enhanced Grain Partitioning of X-Ray Microtomography Segmented Images

Nicholas C. Skrivanos II

*Louisiana State University and Agricultural and Mechanical College*

Follow this and additional works at: [https://digitalcommons.lsu.edu/gradschool\\_theses](https://digitalcommons.lsu.edu/gradschool_theses)



Part of the [Other Engineering Commons](#)

---

### Recommended Citation

Skrivanos, Nicholas C. II, "Enhanced Grain Partitioning of X-Ray Microtomography Segmented Images" (2018). *LSU Master's Theses*. 4625.

[https://digitalcommons.lsu.edu/gradschool\\_theses/4625](https://digitalcommons.lsu.edu/gradschool_theses/4625)

This Thesis is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Master's Theses by an authorized graduate school editor of LSU Digital Commons. For more information, please contact [gradetd@lsu.edu](mailto:gradetd@lsu.edu).

ENHANCED GRAIN PARTITIONING OF X-RAY MICROTOMOGRAPHY  
SEGMENTED IMAGES

A Thesis

Submitted to the Graduate Faculty of the  
Louisiana State University and  
Agricultural and Mechanical College  
in partial fulfillment of the  
requirements for the degree of  
Master of Science

in

The Craft and Hawkins Department of Petroleum Engineering

by  
Nicholas Christopher Skrivanos II  
B.S. and B.A., Trinity University, 2015  
May 2018

## ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Karsten Thompson, for his continued guidance and support throughout my work on this project. From him, I learned much regarding both technical and non-technical matters, and I am very grateful to have had the opportunity to work with him. I also would like to thank my graduate committee members, Dr. Clint Wilson and Dr. Ipsita Gupta. These individuals showed an earnest interest in my research, which helped inspire me to push the bounds for what I was able to accomplish.

I would like to thank Tim Thibodeaux, who was a great resource for all things related to the visualization software Avizo, machine learning, and all PoreSim in-house programs. I would additionally like to thank the other individuals with whom I have worked closely, namely, Jack Blears, Godfrey Mills, Paula Sanematsu, Masoud Safari Zanjani, Dongxing Liu, and Bin Wang.

Thank you, to all my friends and family, for the support you have shown me over the past several years. Specifically, I would like to thank my girlfriend, Chloe Kirk, who encouraged me to chase this goal of mine, even though it meant living several hundred miles away from one another.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS .....	ii
ABSTRACT.....	iv
CHAPTER 1. INTRODUCTION .....	1
CHAPTER 2. BACKGROUND .....	3
2.1. Analysis Methods.....	3
2.2. Grain Partitioning Technologies .....	6
CHAPTER 3. PROGRAM DEVELOPMENT .....	14
3.1. VOX2GRAINS Initial State .....	14
3.2. Main Program Alterations and Additions .....	22
3.3. Post-Processing Refinements .....	25
CHAPTER 4. RESULTS .....	36
4.1. VOX2GRAINS Body Improvements .....	36
4.2. Post Processing Refinements .....	57
CHAPTER 5. CONCLUSION AND RECOMMENDATIONS .....	72
5.1. Conclusions.....	72
5.2. Recommended Future Work .....	72
REFERENCES .....	74
APPENDIX. MACHINE LEARNING OUTPUTS.....	80
VITA .....	92

## ABSTRACT

In the field of petroleum engineering, rock samples are often taken from wells during the drilling process. Grain partitioning of digital three-dimensional microtomography segmented images obtained from these samples provides valuable in-situ properties and statistics that allow for accurate particle and structure characterization. This information can be used directly in detailed production and reservoir analysis, and can also be used to generate realistic packing models for advanced simulation. Additionally, the partitioned image can be used as a building block for realistic hydraulic fracture modeling. This technology has applications in other fields as well, such as core analysis in soil sciences and developing novel structures in material science. Several automatic and manual partitioning algorithms have been developed, but these algorithms often perform poorly for irregularly shaped or consolidated rocks. The objective of this work is to improve the accuracy, reliability, and control of the grain partitioning algorithm VOX2GRAINS. The program is broken into two separate categories: initial partitioning, and post processing refinement. The initial partitioning methodology assumes that the true particle interfaces coincide with watershed surfaces. In order to combat the over partitioning that is common with this methodology and to provide smoother interfaces, several new iterative techniques have been implemented into the particle assembly stage along with an adjustment to the distance map generation. Once the initial partitioning is completed, the user has the opportunity to interact with the three-dimensional partitioned image. Bulk and individual properties, such as porosity, particle volumes, surface areas, contact areas, and aspect ratios are calculated and displayed. Results show that many of program's new additions and alterations provide more accurate and realistic grain-grain interfaces. The program's post processing options have been expanded to include planar regression of grain-grain contact surfaces and the reduction of over partitioned grains through machine learning via logistic regression. The machine learning refinement option was found to be a particularly effective method that combines user control, automation, and time efficiency to create a more accurately partitioned image.

## CHAPTER 1. INTRODUCTION

In petroleum engineering, very little is initially known about the subsurface formations. Operators frequently rely solely on core samples in order to predict reservoir properties, which in turn can impact future production estimates and hydrocarbon recovery strategies. These samples can be very costly to obtain, so it is important to maximize the amount of information generated from each sample, and to ensure that this information is accurate. This is the overarching objective of rock analysis.

Rock analysis is used to determine accurate particle and structural characterizations. Particle characterization refers to individual granular properties, such as size, shape, porosity, and density, whereas structural characterization refers to the microstructure of the granular material and granular assemblies (Williams, 2002). These properties provide a general idea of the subsurface formations, and can be directly used in production and reservoir analysis, such as reservoir simulation and nodal analysis. Additionally, this information can be useful for more advanced analysis like generating realistic particle packing models for flow analysis and geo-mechanical models for hydraulic fracturing.

Although different rock analysis methods have been explored, X-ray microtomography is one of the only methods that provides the opportunity for obtaining accurate particle and structural characterizations simultaneously. In this process, a three-dimensional digital representation of a given core sample is generated. In an attempt to maximize the utility of these three-dimensional X-ray microtomographic digital images, computer algorithms have been developed that allow for both manual and automatic identification of individual grains within the image, each providing corresponding spatial statistics (Thompson, 2006, Ketcham, 2005, Sheppard and Sok, 2006, ThermoFisher, 2017). Although current automatic algorithms have shown the ability to arrive at accurate statistics from packs with simple geometries and little to no consolidation, there is a need for more robust algorithms to more accurately partition irregularly shaped particles that are commonly found in core samples. This thesis proposes and evaluates new methods to improve the accuracy and reliability of grain partitioning of preexisting algorithms in the program VOX2GRAINS (Thompson, 2006).

In order to provide a more broad understanding of where this work fits, a brief background on X-ray microtomography and other rock analysis methods is first presented. After discussing common preprocessing stages, various published partitioning technologies are described in detail. Following, program developments will be described, and the impacts of any meaningful additions or alterations will be shown through partitioned images.

Specifically, this thesis first compares VOX2GRAIN's in-house distance map with a true distance and Euclidean distance map, and investigates how each map impacts partitioning. An accurate distance map is of utmost importance, since this provides the foundation for the grain partition. Next, several iterative techniques added within the granular assembly stage of the program are evaluated along with a new technique for assigning voxels that border multiple grains. Like most partitioning algorithms, VOX2GRAINS relies upon a watershed transform, and often results in over partitioning grains, especially in cases of irregularly shaped grains or significant consolidation. To provide an efficient method for combating this issue, machine learning via

logistic regression is implemented and evaluated. Lastly, two separate methods of applying planar regression to grain-grain contact interfaces are assessed.

The algorithms presented here were designed to be robust; applications for this particular technology extend far beyond petroleum engineering to any science involving particulate or granular matter, such as soil sciences, materials sciences, pharmaceutical sciences, civil engineering, and environmental engineering.

## **CHAPTER 2. BACKGROUND**

As previously mentioned, often very limited information is initially known about the physical properties of subsurface formations in the field of petroleum engineering. Operators frequently rely on core samples and well logs in order to predict reservoir properties. These properties provide a relative idea of the subsurface to geologists and engineers, and often provide the basis for subsequent analysis ranging from production inflow analysis to reserve estimations. Well logs and core samples can be costly to obtain, so operators seek to maximize the amount of useful information from these tests, and to ensure that this information is as accurate as possible. Occasionally, well logs and core samples are both obtained and analyzed together to corroborate their individual results, but it is not uncommon for operators to obtain only a core sample. While core analysis and well logs provide some overlapping information, there are significant differences between the two methodologies, and advanced core analysis provides information that cannot currently be derived from well logging, such as grain sizes and granular distributions. This thesis focuses on recent advancements in core analysis algorithms, but before detailing these methods, a historical overview of common core analysis techniques is presented.

### **2.1. Analysis Methods**

An ideal analysis method is one that provides a broad spectrum of accurate rock characteristics on both the micro and macro scales. In petroleum engineering and geosciences, the common properties of interest for understanding and predicting flow dynamics and characterizing the subsurface are porosity, permeability, tortuosity, density, grain size and shape, degree of consolidation, anisotropy, and heterogeneity. Keeping in mind that these rock samples are often costly and timely to obtain, methodology that does not require destroying or dispersing the sample is greatly preferred so that the sample can be retested for verification purposes and used in other experiments. The methodology should also be able to determine sufficient parameter conclusions from a relatively small sample of rock, since typical core samples are usually only millimeters in diameter.

Over the past seventy years, a number of rock analysis techniques have been developed, each carrying its own advantages and disadvantages. Initial techniques were relatively simple but provided very limited information. More recently, advanced techniques have shown promising results in determining a range of accurate information.

Particle sieving requires dispersing the bulk sample, and passing the individual grains through a sieve (Ballard, 2013). While this process can provide accurate size statistics for spherically shaped particles, it provides no information about how the grains were organized within the rock sample. Thin section analysis also requires dispersing the rock sample, but additionally provides some measure of structural characterization. The core is cut into thin sections, and observed under a microscope. Methods have been proposed for analyzing particle and structural characterizations for rock images by combining the individual analysis of each two dimensional slice (Krumbein, 1951). Thin section analysis can be very timely, and requires carrying out analysis in two dimensions many times. Afterward, the individual results from each slice can be combined with one another to generate a three dimensional picture, but this three dimensional image analysis is based upon findings from two dimensional images, and is therefore prone to errors due to the lack

of complete spatial comprehension. X-ray diffraction is a common method for determining chemical compositions for rock samples, but it requires crushing the sample into a fine powder, thus providing no understanding of particle shapes and how they originally were oriented (Hosterman et al, 1981). While these methods are useful, they do not simultaneously provide the opportunity for accurate particle and structural information to be captured.

Alternatively, X-ray Microtomography (XMT, or micro-CT) is a fairly recently developed non-destructive and non-invasive analysis method that is well suited for the micro-scale, meaning that, for most rock samples, the resolution provided by this methodology is fine enough that one is able to distinguish between individual pores and individual grains (Wildenschild, 2012). From a given core sample, this technique produces a reliable, in situ, three dimensional digital image that is fully representative, and, because this technique is nondestructive, provides the ability for the sample to be retested for validation.

XMT offers the advantage of preserving information about particle orientations, spatial correlations, and particle distributions in situ, just as they appear in the core sample. Simultaneously, the methodology retains individual particle information, which opens the door for advanced statistical calculations for particulate properties. Concretely, this method provides the opportunity to calculate accurate sample density distributions, particle distributions, porosity, heterogeneity, anisotropy, grain sizes, locations, volumes, aspect ratios, surface areas, orientations, particle contact areas, and particle contact locations. The disadvantages of this process are the cost associated with generating the digital image, and the limited software analysis packages.

There are alternative methods that can produce results similar to X-ray microtomography, such as transmission electron microscopy (TEM) and focused ion beam/scanning electron microscope (Wildenschild, 2012). Although this thesis focuses on partitioning grains in XMT images, the logic behind the algorithms should be applicable to all image types, assuming the image contains the full representative elementary volume.

An example of a typical XMT experimental setup is shown in Figure 2.1. XMT works by passing a radiation source through a rock sample and measuring the corresponding X-ray attenuation downstream of the radiation source with a sensor (Wildenschild, 2012). The rock sample is rotated, and the sensor captures a series of two dimensional attenuation maps that are later reconstructed into a single three dimensional greyscale image. In the resulting three dimensional greyscale image, each voxel (three dimensional pixel) holds a numerical value that corresponds directly to the X-ray attenuation through the representative portion of the sample. Attenuation level is directly proportional to the density, atomic number, and energy of the incident X-ray. By keeping the energy level of the incident X-ray constant with a monochromator, materials with similar densities and atomic numbers will display similar greyscale values in the reconstructed image. This technology is capable of producing images up to approximately  $2000^3$  voxels, with a resolution of several microns (Blunt et al., 2013).

Before the digital XMT image can be partitioned, it is common to apply preprocessing measures to remove or reduce the noise and any image artifacts. Once minimized or removed, the phases within the image can be segmented from one another. Although many segmentation processes exist, thresholding according to voxel intensity is a straightforward and fairly common

methodology (Iassonov et al., 2009). As previously mentioned, X-ray microtomography gives a reconstructed grey scale image, with each voxel corresponding to the X-ray attenuation, and therefore the density and atomic number of the material. Similar materials will provide similar densities, and have similar grey scale values. By applying attenuation thresholds corresponding to the grey scale values observed in the image, the separate phases of the grey scale image can be segmented to create a new image with voxel values representing the separate phases.

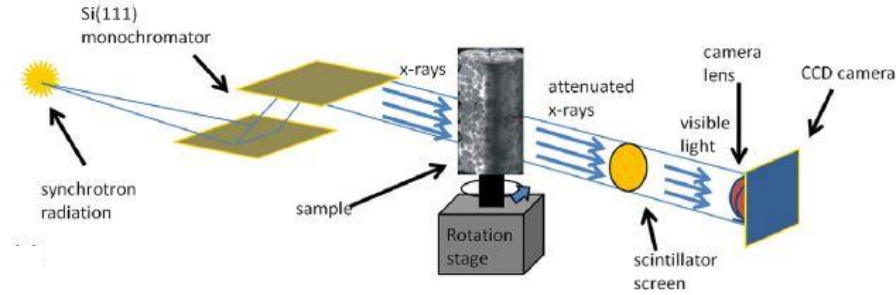


Figure 2.1. Typical X-ray microtomography setup (Wildenschild, 2012)

Take, for example, a grey scale image of a porous material that appears to have only one solid phase. The grey scale image is segmented by a selected intensity value, and the resulting image is binary, meaning all voxels hold one of two possible values. All voxels with a value of 1 may represent the solid phase, whereas all voxels with a value of 0 may represent the void phase. This same process can be carried out for images with multiple solid phases present by specifying additional threshold intensity criteria. Attempts have been made to automate this process by using machine learning to select thresholds, but because this process is so subjective, segmentation is most frequently still carried out by manually selecting threshold values (Wildenschild, 2012). An example three phase segmentation is provided in Figure 2.2 (Mills, 2016).

Understanding the segmentation process sheds light on the problems that can arise from failing to properly remove noise or artifacts. Each step of this process builds upon the last, and any error is exasperated as the process moves forward. For these reasons, a clean segmented image is crucial for an accurate grain partitioning.

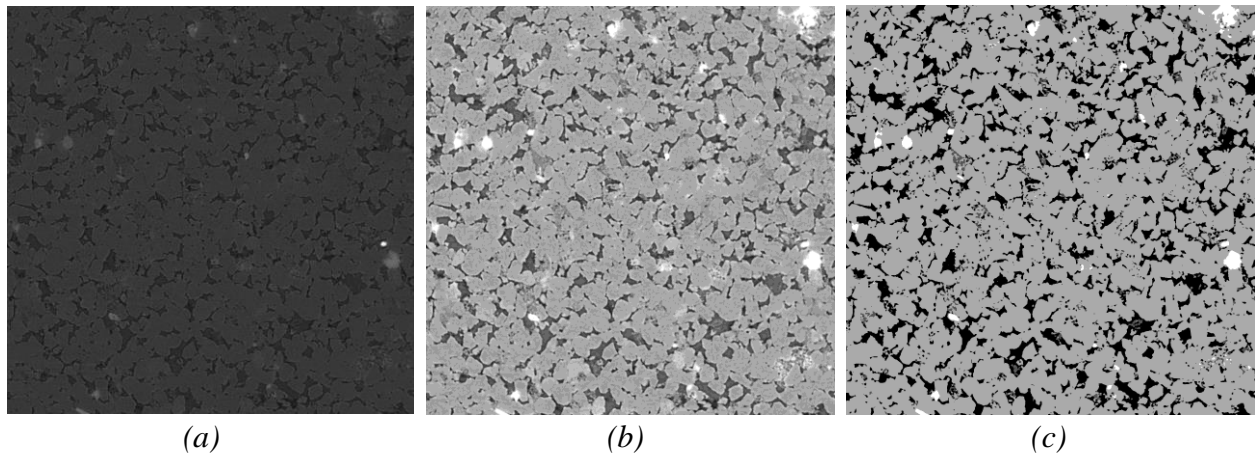


Figure 2.2. XMT raw image (a), after anisotropic diffusion (b), and then after segmentation (c). 3.5cm per side (Mills, 2016)

## 2.2. Grain Partitioning Technologies

Once the grey scale image has been cleaned and segmented, the individual grains can be partitioned. Several research groups have worked to progress the analysis capabilities and accuracies for grain partitioning, but there is still a need for robust algorithms that are able to properly partition grains which are not spherical, and rock samples with a high degree of consolidation.

### 2.2.1. Seidler et al.

In 2000, Seidler et al. detail how they successfully processed a synchrotron X-ray microtomography image of a disordered pack of glass spheres granule-by-granule (Seidler et al, 2000). From their reconstructed image, they were able to determine individual granular locations and sphere connectivity by using object recognition software. Due to this methodology, this algorithm is only valid for packs of spherical granules, and would not yield quality results if applied to granular materials with irregular shapes, unless new object recognition algorithms were developed. Furthermore, if this technology were to be expanded for analyzing true rock images, this technology would require information about expected grain shapes prior to analysis.

### 2.2.2. BLOB3D

First published in 2005, BLOB3D was the first computer program that partitioned individual grains from high resolution X-ray computed tomography images in three-dimensional space (Ketcham, 2005). Prior to this program, analysis had been performed on individual two dimensional slices, and recreated in order to form a three-dimensional representation. As mentioned previously, analyzing a three-dimensional image via a series of two-dimensional representations omits valuable information, is less efficient, and can be a cause of inaccuracies. BLOB3D was primarily developed with the intention of analyzing porphyroblastic metamorphic rocks, and is capable of identifying up a few thousand features in a given three-dimensional image.

The algorithm is broken into three primary portions: segment, separate, and extract. Unlike many other grain partitioning algorithms, BLOB3D segments the distinct phases internally. In the separate stage, the programs displays each individual continuous cluster of the phase of interest, and prompts the user to decide whether it is a single contiguous blob or multiple individual blobs. Statistics about each blob are given to the user in order to aid in the decision making. If the image has several hundred or thousand initial clusters, then the separate process could be extremely lengthy. Also, for a sample with any degree of consolidation or even barely touching particles, the user would be required to spend a significant amount of time separating the individual blobs into their individual partitions, since the majority of the image would be defined as one contiguous cluster. Once all separations are completed, the final stage, extract, is carried out. Here, statistics based upon the quantity and orientation of the voxels in the each previously defined cluster are presented. These statistics include center of mass, volume, and surface area. This final stage also provides aspect ratio and axis orientation by fitting the outside surface of the cluster with a least squares ellipsoid. Lastly, the contact areas and contact orientations between particles are also determined. The results shown are promising, though it should be mentioned that the images used for testing were composed solely of spheres barely touching one another. Since the partitioning of

any touching particles is completely manual, a significant amount of time would be required to partition a true rock image.

### 2.2.3. VOX2GRAINS

Thompson et al. (Thompson, 2006) created a program that automatically identifies and partitions separate grains in three-dimensional space from a segmented image. This program, referred to in this thesis as VOX2GRAINS, was the first to offer a solution to identifying and partitioning grains when no previous knowledge of particulate shape is available. Since many core samples obtained in industry are composed of angular and odd shape grains, this is a powerful program. This thesis focuses on improving the accuracies, reliability, and capabilities of VOX2GRAINS.

The overall algorithm is broken into five separate stages. In the first, a dual-phase burn map is calculated. This terminology burn map is used because this process visually resembles a grassfire burn, starting at the boundary of the solid and void phases and expanding both outward (into the void) and inward (into the solid) until all voxels of the image have been covered. This is a step-wise procedure, and each voxel in the burn map is assigned an integer value corresponding to the number of incremental steps that were taken before the voxel was reached. Voxels representing the solid phase would be assigned a positive integer, whereas voxels representing the void phase would be assigned a negative integer. For two voxels representing the solid phase, a voxel closer to the void phase would receive a smaller burn map value than a voxel further away from the void phase. This map provides information about each voxel's distance to the nearest boundary.

The second stage of the automated partitioning algorithm is identifying the location of the local extrema. Within the burn map, voxels that are solely surrounded by smaller absolute values are identified as the initial grain center locations. If there are multiple voxels that hold the same local absolute maximum value, then the center of mass of these voxels is identified as the initial grain center.

The grain centers can be further refined in the third step of this process. Thompson et al. note that for images of spherical objects with a significant number of voxels per sphere diameter, this option need not be used. However, for irregularly shaped grains, it can provide improved partitioned results. If refinement is selected, then an iterative optimization process of determining the largest possible inscribed sphere in each local maxima of the solid phase is performed. The center of the sphere acts as the grain center. If, after determining all maximum inscribed spheres, a sphere center is located within another sphere, the two are merged together.

The fourth stage of VOX2GRAINS is particle assembly, in which each particle is constructed outward from the identified centers in a restricted manner. The assembly process starts at the identified grain centers and spreads outward until another particle or void phase is reached in a manner similar to the previously described burn. It begins by setting the minimum burn value equal to the largest global burn. A full voxel loop of the domain is performed, and any voxel that holds a value equal to or greater than the minimum burn value, and also touches a voxel that has already been assigned to a grain, is assigned to the same grain as its neighbor. The voxel loop of the domain with the same minimum burn value is iterated until an iteration with no additional assignments takes place. At this point, the minimum burn value is decreased, and the process repeats itself. This

continues until all voxels representing the phase of interest are assigned to a particle. The assumption for this stage is that the true particle-particle interfaces correspond with the watershed distances of the burn map. More in depth analysis of the watershed algorithm is provided by Beucher et al. and Vincent (Beucher et al., 1979, Beucher, 2000, Vincent et al., 1991). Often, this process leads to over partitioning of the grain space.

The fifth and final stage is to compute physical parameters on the individually partitioned particles, such as volume, inscribed radius, surface area, particle aspect ratio, and contact area with other particles. In order to combat the previously mentioned calculation errors due to boxy nature of voxels, methods proposed by Lindblad (Lindblad, 2005) and Dalla et al. (Dalla et al., 2002) are implemented.

Along with the quantitative statistics, the resulting partitioned image can be viewed and interacted with via a visualization package. This provides a clearer understanding of the particle size and spatial distributions, as well as the general shapes of the particulate matter. Being able to interact with the image also allows the user the opportunity to evaluate the results of the automated partitioning. Understanding that the use of the watershed transform in particle assembly often results in over partitioning, this program offers the user both manual and automated refinement options for merging over partitioned grains. In the case of manual refinement, the user specifies the grains he or she wishes to merge together by their identifying numeric values. In automated refinement, the user specifies a thresholding contact area to surface area ratio. This threshold value is compared against all neighboring pairs, and if the calculated ratio for a grain pair is greater than the threshold value, then the grains will automatically be merged. All statistics are updated after any manual or automatic merges take place.

#### 2.2.4. Australian Research Group

Saadatfar et al. (Saadatfar, 2005) have shown good results for grain partitioning of three-dimensional images with two separate methodologies, one of which is very similar to that of Thompson et al. (Thompson, 2006). Similarly, these methodologies are robust enough to identify and partition grains of spherical and irregular shape. The two separate processes and their capabilities are summarized below.

The first methodology described begins with a three-dimensional segmented image and applies an erosion of the outer boundaries of the phase of interest. This erosion breaks apart the individual grains. This process is continued until a maximum value is reached, at which point each cluster of the solid phase that has not eroded is assumed to be fully isolated from all other particles, and therefore represents the base of a single grain. Each cluster is labeled accordingly, with a specified location at the center of mass of the non-eroded voxel cluster, and the image is dilated back to the original size. A Voronoi tessellation then separates the entire space such that each individual compartment contains a single identified grain seed, or what was formerly the isolated cluster's center of mass. The particles are grown outward from their individual grain seeds to fill the Voronoi partitioned space at a constant rate. Since the grains are only dilated outward in their Voronoi partitioned space, the resultant grain boundaries are planes. Saadatfar et al. mention that this process works well, but can lead to losing smaller grains due to the nature of the erosion. For

this reason, it is likely not well suited for rock samples with large size distributions, or when there are grains only represented by a limited number of voxels, possibly due to a poor resolution.

The second method described in this article begins by calculating a Euclidean distance map, where each individual voxel of a selected phase holds a value equal to the minimum distance from that voxel to any voxel of another phase. In other words, distance map values represent the distance from a particular voxel to the nearest surface. The result is a three dimensional image similar to burn map described by Thompson et al. (Thompson, 2006). The local maxima are taken as the grain centers, and the regions are grown outward from the center voxels in accordance with the watershed transformation. Voxels nearer a particular grain center will be assigned to that grain before voxels further away from the grain center. If a particular voxel is found to touch two separate grains, then the voxel is assigned to the grain it first contacted. The author mentions that the resulting partitioning from this watershed method provides similar results when compared to the Voronoi partitioning method in many cases, but this watershed method maintains the ability to determining accurate information about smaller particles. This algorithm is parallelized in order to reduce computational time.

The partitioned image from either method is used to subsequently determine grain fabric and texture properties, such as grain size, grain size distribution, sphericity, roundness, shape class, grain sorting, and grain contacts. The results shown in this article are promising, but there are limited images of partitioned grains, and the images that are included show little contact area between grains.

Soon after, two follow up articles were published (Sheppard, Arns, et al., 2006, Sheppard et al., 2005). The first describes merging grain centers if one grain center is located within another grains inscribed sphere. The second implements a region-merging algorithm to combat the over partitioning that commonly results from the watershed transform in particle assembly. Sheppard et al. mention that the watershed partitioning methodology has been tested for many different rock types successfully. From the partitioned images shown, the technology looks impressive. However, there does appear to be some room for improvement in the images containing irregularly shaped grains. The images presented show that it becomes increasingly difficult to accurately partition the space as grain shape irregularity increases and consolidation increases.

#### 2.2.5. PerGeos

PerGeos is a newly released commercially available software package that specializes in digital rock and core analysis (ThermoFisher, 2017). This package targets reservoir engineers and geologists as consumers with primary applications such as pore network extraction, determination of pore statistics, flow simulations, and the ability to correlate digital cores with petrophysical logs. This software package also provides the ability to partition grains from a segmented image, and provides statistics on these partitioned grains. The partitioning can be performed on both two-dimensional and three-dimensional data sets. This software is not open sourced, meaning that the specific partitioning algorithms are unspecified.

### 2.2.6. Parameter Calculation Improvements

There have also been a number of technological advancements in calculating accurate particle statistics from X-ray tomographic images. Several of the partitioning programs mentioned in the prior sections implement variations of the techniques described here to ensure accurate calculations.

Voxels are discrete and inherently follow a cubic structure. This can lead to difficulties in attempting to calculate accurate properties like surface area and contact area for objects that have curvature. Several methods have been proposed for circumventing this issue and determining accurate area calculations.

Lin and Miller showed how particle shape and size could be quantitatively measured from a segmented image (Lin et al., 2005). They propose two methods that they call assemblage and boundary method, and show how these can be applied for volume, surface area, aspect ratios of the principle axis, and sphericity calculations. In this work by Lin and Miller, all calculations were performed on isolated objects. Around the same time, Lindblad also proposed a methodology for calculating surface areas by using weighted local configurations (Lindblad, 2005). These methods were accurate when predicting surface areas exposed to void, but were not capable of determining total surface areas when two solid phases contact one another. Soon after, Thompson (Thompson, 2007) proposed a methodology that built upon the method proposed by Lindblad (Lindblad, 2005) and allowed for accurate surface and contact areas to be calculated for granular packs. The calculations allow for elementwise surface areas that include both surface areas exposed to the void and surface areas that contact other solid elements.

### 2.2.7. Custom Refinement and Machine Learning Via Logistic Regression

Most, if not all, current partitioning algorithms incorporate the watershed transform during granular assembly. This inevitably leads to over partitioning, meaning the number of identified grains is greater than the true number of grains. The degree for which a data set is over partitioned is primarily dependent upon the irregularity of the shape of the grains and the amount of consolidation. Various programs offer different solutions to this issue, such as automatically merging specific regions towards the end of particle assembly (Sheppard, 2005) and, in the case of VOX2GRAINS, post partitioning manual or automated refinement. To the knowledge of this author, machine learning has not been applied to grain partitioning in any form.

Machine learning has advanced rapidly over the past few decades, and has been applied to a variety of industries in many different forms. Examples include advanced data mining, handwriting recognition, product recommendations for online shopping websites, and the control of autonomous vehicles. The author believed that supervised machine learning could be applied to grain partitioning to yield more accurate partitions for irregularly shaped and consolidated grains. This thesis investigates this possibility through the application of logistic regression as a refinement method.

Since, in this case, the objective is to classify a given pair of grains as sufficiently merged or needing to be merged, a classification algorithm is well suited. Logistic regression and linear

regression are two common classification methods. For the particular situation described here, the dependent variable is binary in that the decision is only to either merge, or to not merge a specified pair. For binary classification, it is common notation for the value 0 to represent the negative class, and 1 to represent the positive class. Represented symbolically,

$$y \in \{0,1\}$$

where  $y$  represents the decision, 0 represents not merging, and 1 represented merging the grains. It is possible to have a multiclass classification, but it is unnecessary for this particular application.

Supervised machine learning uses a hypothesis function to map the dependent variable from the independent variable (Ng, 2017). For a specified independent variable, the hypothesis function will return the probability that the output is of class 1. Different regression models have different hypotheses equations. Equation 2.1 provides the hypothesis for linear regression whereas Equation 2.2, 2.3 and 2.4 give the hypothesis for logistic regression.  $X$  represents the array of independent features and the  $\theta$  values represent the learned parameter array. The superscript  $T$  represents a transpose of the preceding variable, as these calculations are shown out in vector form.

$$h_{\theta}(x) = \theta_0 + \theta_1 x \quad (2.1)$$

$$h_{\theta}(x) = g(\theta^T x) \quad (2.2)$$

$$z = \theta^T x \quad (2.3)$$

$$g(z) = \frac{1}{1+e^{-z}} \quad (2.4)$$

Equation 2.4 is commonly referred to as the Sigmoid Function, or Logistic Function. For Logistic Regression, the Sigmoid Function adds the restriction that

$$0 \leq h_{\theta}(x) \leq 1$$

In order to arrive at merge or do not merge decision, sample data must be collected with at least one independent variable, commonly referred to as a feature. For illustrative purposes, consider the following oversimplified example with only one independent variable. Suppose a total of fifteen merge and do not merge decisions were recorded, and for each decision, the grain pair's original contact area was recorded. Ten of the fifteen data points were randomly gathered and plotted as shown in Figure 2.3. Let us assume that this information is sufficient for determining whether or not a merge should take place.

From visual inspection, there is a fairly obvious threshold contact area in this example that separates the decision to merge and to not merge between the 5<sup>th</sup> and 6<sup>th</sup> data points. If linear regression were applied here, as represented by the solid line, then the resulting hypothesis equation would not represent the data very well. Assuming a common decision boundary of 0.5, the algorithm's hypothesis would determine anything to the left of the 0.5 value to be classified as do not merge, and anything to the right of the 0.5 value to be classified as merge. This leads to a poor hypothesis function, since it could be erroneously classifying a significant number of pairs. The hypothesis function could also output values significantly greater than 1 and significantly less than 0, which does not make sense for binary classification. Alternatively, logistic regression is

represented by Equation 2.4, which only provides outputs between 0 and 1. When applied to the sample data, the logistic function fits much better, as shown in Figure 2.4. For these reasons, logistic regression is far more suitable than linear regression for this application.

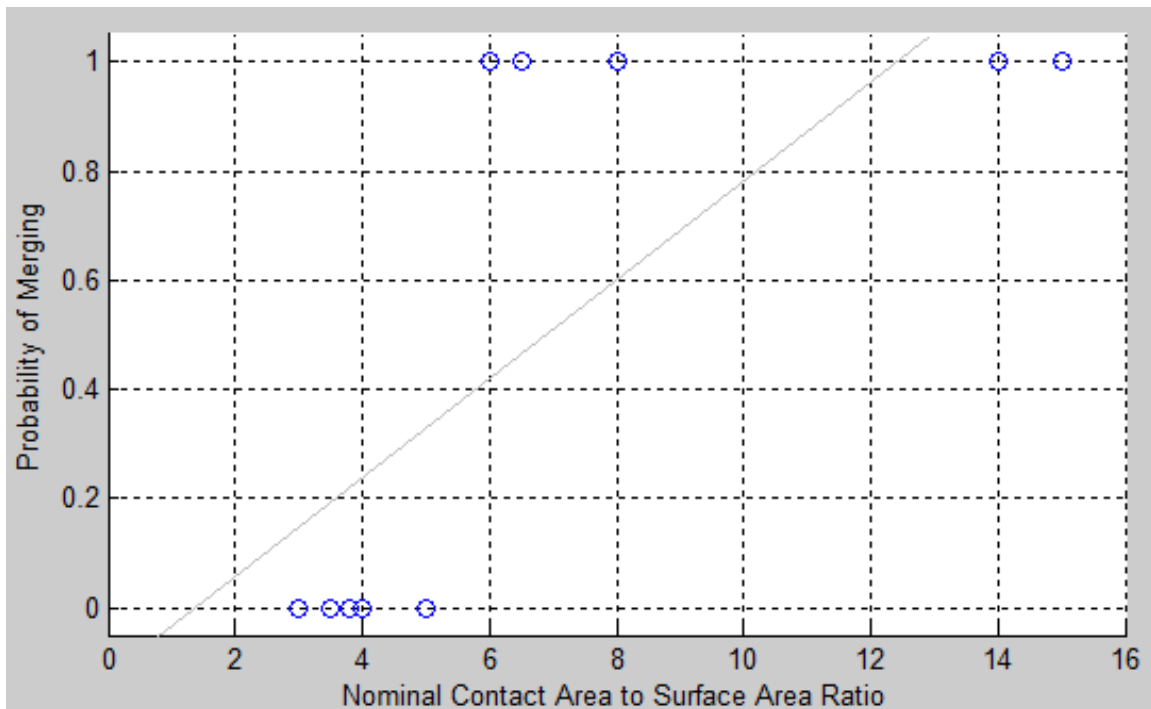


Figure 2.3. Illustrative Example of Merge or Do Not Merge Decision with Linear Regression

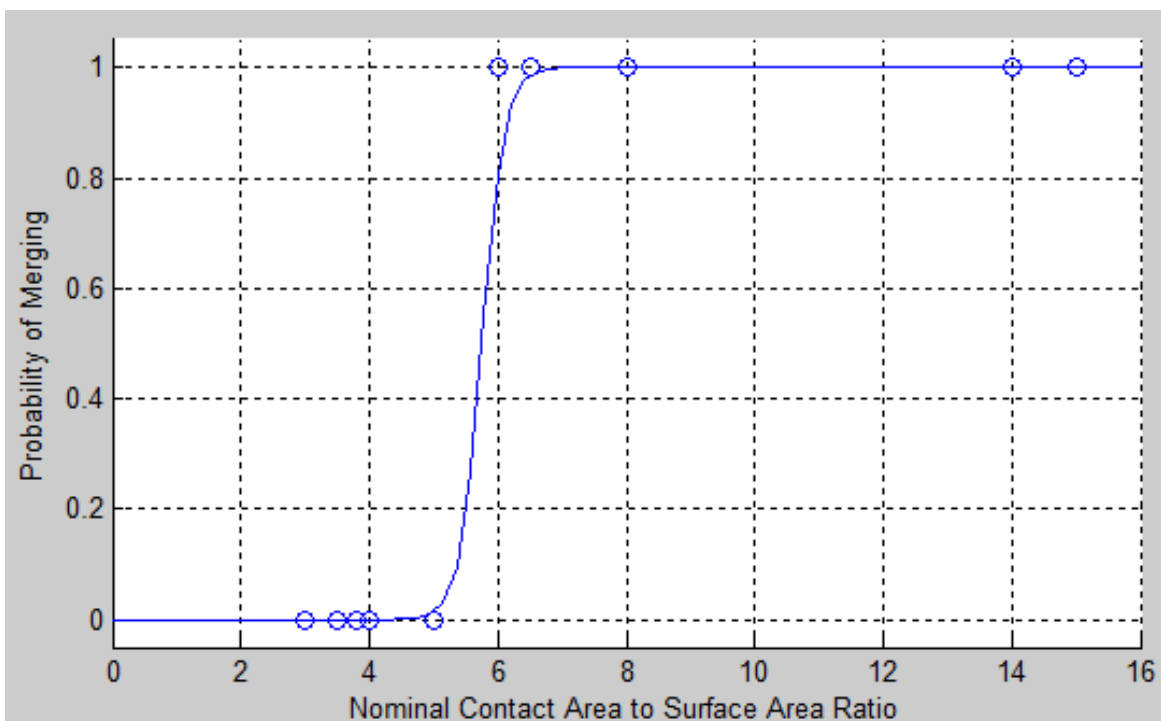


Figure 2.4. Illustrative Example of Merge or Do Not Merge Decision with Logistic Regression

In this example, the ten points selected from the pool of fifteen data points comprise the training set data and the remaining five data points comprise the cross validation data set. Each data point includes an independent variable value (also known as a feature) and the corresponding dependent variable (the decision). In order to arrive at an accurate hypothesis equation, the appropriate  $\theta$  parameter values must be determined. The value of these parameters are determined by the training set data via a cost function, which calculates the difference between the hypothesis values and the training set example values. By minimizing the cost function, the most appropriate parameters are determined. The equation for the cost function for logistic regression is provided in Equation 2.5. Here,  $m$  represents the number of training set data points.

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m \left[ y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) \right] \quad (2.5)$$

Equation 2.5 can be minimized using gradient descent or a number of well known, faster, and more efficient minimization methods, such as conjugate gradient descent or the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS) (Ng, 2017). Once minimized, the  $\theta$  parameter values can be obtained, and the resulting hypothesis should be compared against the cross validation data points for verification. If the hypothesis can be used to correctly predict the dependent variable to a specified degree of certainty, then the algorithm can be used for further predictions when the results is unknown, assuming the data sets were diverse and fully representative. If, however, the hypothesis erroneously predicts a significant portion of the cross validation data set, then the hypothesis is a poor predictor. There are a number of causes that could lead to a poor hypothesis equation in this situation, such as not enough representative data, too few predictive features, overfitting the training set data, or features not properly or wholly representing the dependent variable.

## CHAPTER 3. PROGRAM DEVELOPMENT

As mentioned in Chapter 2, the VOX2GRAINS program is fairly developed. The program had two separate versions, though they both relied on the same core FORTRAN90 programming. The first version was runs on Linux via Louisiana State University's supercomputer Philip. This version supplies no direct visualization, but has the advantage of more processing power and RAM. The other version runs on a desktop computer through the visualization software package Avizo and provides direct visualization after processing.

### 3.1. VOX2GRAINS Initial State

To summarize, VOX2GRAINS is a multistage process that starts by creating a burn map, upon which the entire grain partitioning is based. Next, the local extrema of the burn map are used as grain seeds. The option exists for these seed locations to be further refined by inscribed sphere optimization. Next, the particles are assembled from the centers outward in accordance with the burn map. This technique restricts the smaller particles from growing until larger particles have already begun to develop. This process is continued until all solid phase voxels are assigned to individual grains, at which point various statistics about each grain are calculated. The program outputs a PSN file, as well as an image file. The image file is a voxel image with each voxel holding an integer value that corresponds to its grain number; the void space is assigned a voxel value of 0. The PSN file is a text file that contains information about the domain size, number of particles, particle locations, grain contacts, and many other statistics. With both the image file and the text file together, all information can be saved and revisited at any point.

There are several detailed operational parameters that can be adjusted for various images such as periodic boundaries, optimization of grain centers by inscribed spheres, maximum number of re-optimization attempts, merge criteria for two separate inscribed spheres (overlapping or touching), the connectivity criteria for grain contacts, and the connectivity for individual voxels when determining the burn map. The two connectivity parameters arise from the manner in which voxels in a grid can neighbor one another. By the most restrictive definition, only voxels sharing a face are considered to be neighbors; assuming that a particular voxel is completely surrounded by other voxels, then, by this definition, the subject voxel has six neighbors. The definition could be expanded to include voxels with which the subject voxel also shares an entire side. This would mean that there would be a total of eighteen neighbors for any voxel completely engrossed by other voxels. By the most liberal definition, a voxel neighbors twenty six other voxels. This definition includes the faces, the sides, and also the corners of the subject voxel. The options for these connectivity criteria are therefore six, eighteen, and twenty six. The voxel neighbors are determined solely by these connectivity values and selecting different connectivity values can have a significant impacts on the resulting partitions. All algorithms described in this thesis are capable of functioning with all neighbor connectivity values.

#### 3.1.1. Initial Partition Quality

Initially, VOX2GRAINS was applied to a number of different real and artificial segmented images yielding results that were often poor prior to any refinement. As is common with other partitioning programs, VOX2GRAINS tended to over partition and had more trouble properly partitioning

irregularly shaped grains and images with higher degrees of consolidation. Alternatively, the program performed well for images with spherical grains that did not overlapping.

Specifically, this study began by analyzing the partition of a series of images donated by Masoud Safari Zanjani (Zanjani, 2016) showing porous quartz-feldspar and clay. The quartz-feldspar and clay were segmented separately. Quartz-feldspar was the material of interest for this study. The segmented images each were 600 voxels cubed with a voxel resolution of 0.00041cm/voxel, bringing length of each side to 0.246cm. Figure 3.1 shows the segmented image used for testing.

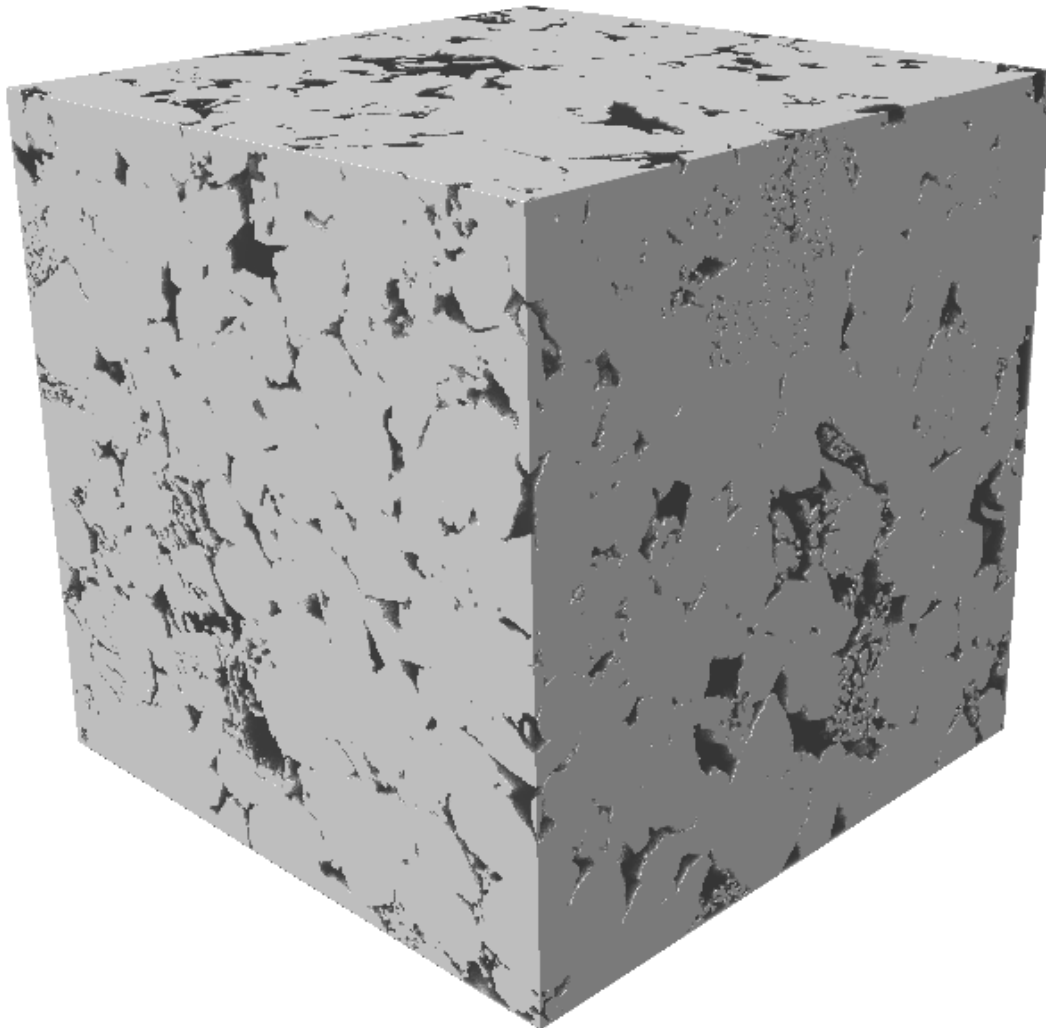


Figure 3.1. Segmented image of quartz-feldspar and clay, 0.246cm per side, 600<sup>3</sup> voxels

These images were analyzed with VOX2GRAINS under a variety of input parameters, namely the connectivity variables. Figures 3.2 through 3.9 present cross sections through a variety of these partitioned images. Black represents the void space, and the other colors are used to distinguish between different grains. When looking at the full partitioned images, it can be difficult to properly observe the individual grain interfaces. Close up images are provided to better depict what many of these grain-grain contacts look like.

Due to the consolidation and irregular shape of the grains, it is understandable that this image would be quite difficult to partition accurately. It should also be mentioned that these images are two-dimensional slices of a three-dimensional image, and do not provide the entirety of the information available. However, these two-dimensional slices do provide significant insight about the partitions, and are a convenient method for displaying examples in this thesis. All conclusions drawn from any two-dimensional images were verified by closely examining the full image in three-dimensions.

Figures 3.2 through 3.9 show that the interfaces between the grains are, for the most part, not ideal. There are a significant number of instances where a piece of a given grain juts out far into another grain. These “slots and drawers” are in no way justified by the granular geometries, and therefore are likely an incorrect partition. These figures also show that many grains are over-partitioned, breaking larger cluster into small grains that, in the physical sample, likely comprise a single grain. This leads to many middle grains that should not be classified as individual grains.

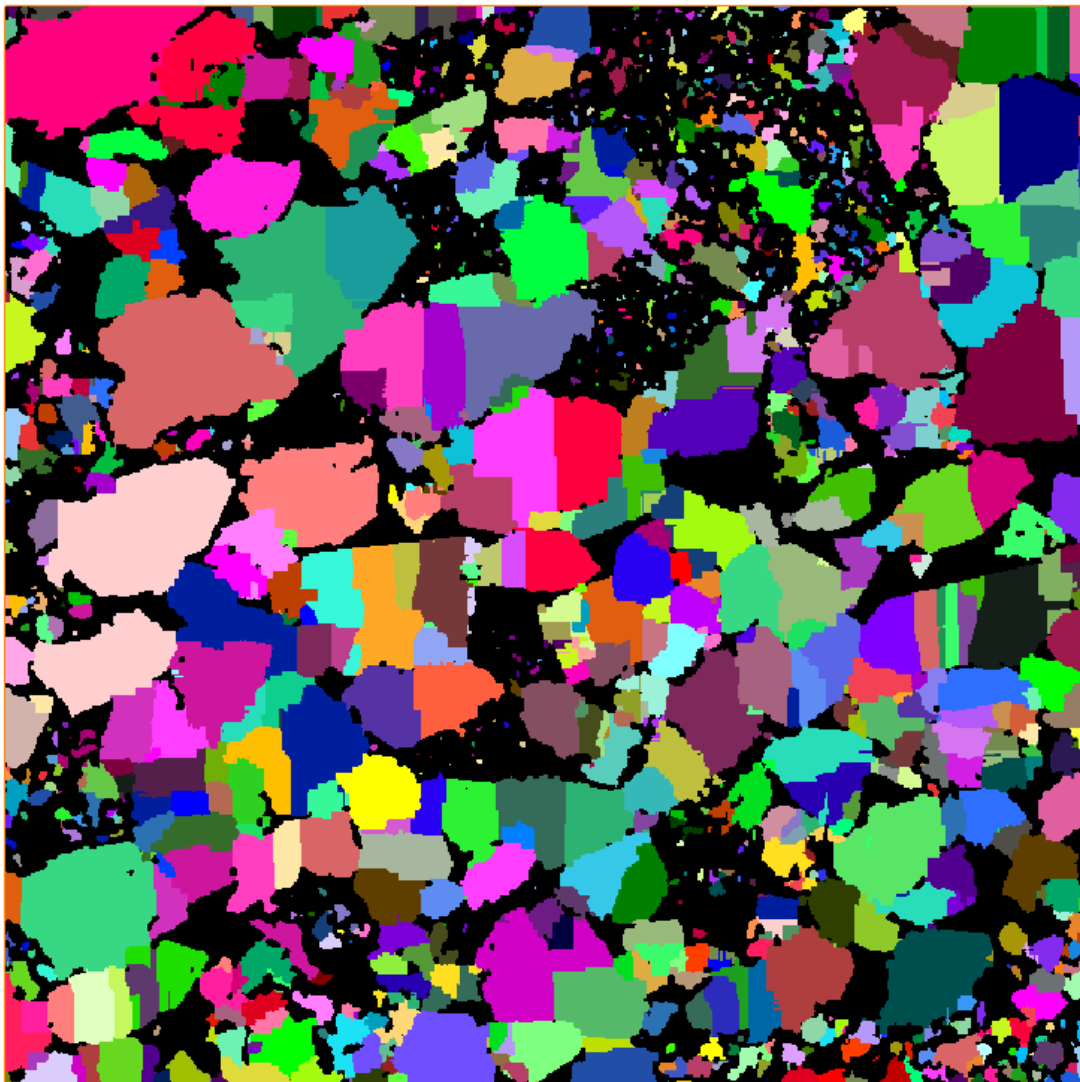


Figure 3.2. Partitioned image of quartz-feldspar rock using connectivity values of six, 0.246cm per side,  $600^3$  voxels

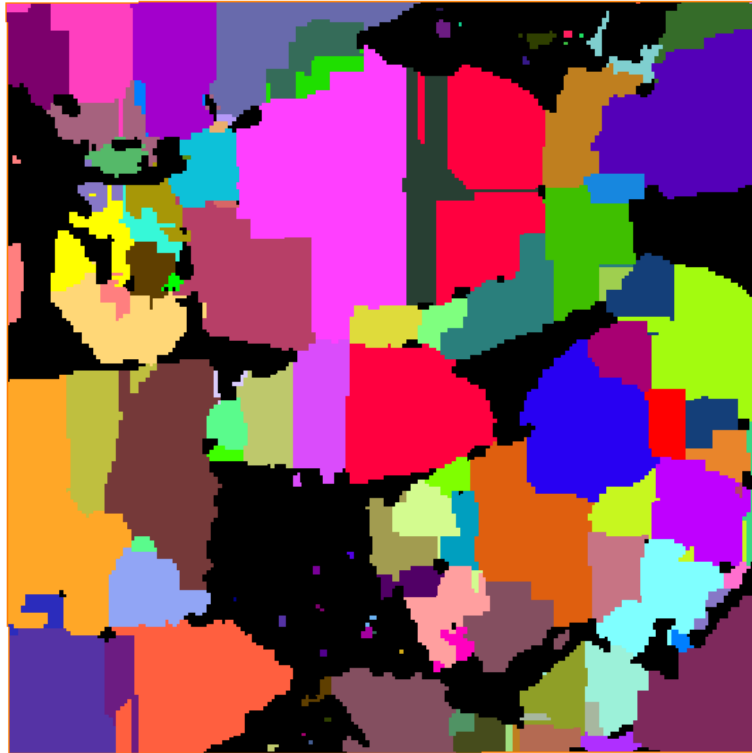


Figure 3.3. Close up view of partitioned image of quartz-feldspar rock using connectivity values of six, 820 $\mu$ m per side, 200<sup>3</sup> voxels



Figure 3.4. Partitioned image of quartz-feldspar rock using connectivity values of six, 820 $\mu$ m per side, 200<sup>3</sup> voxels

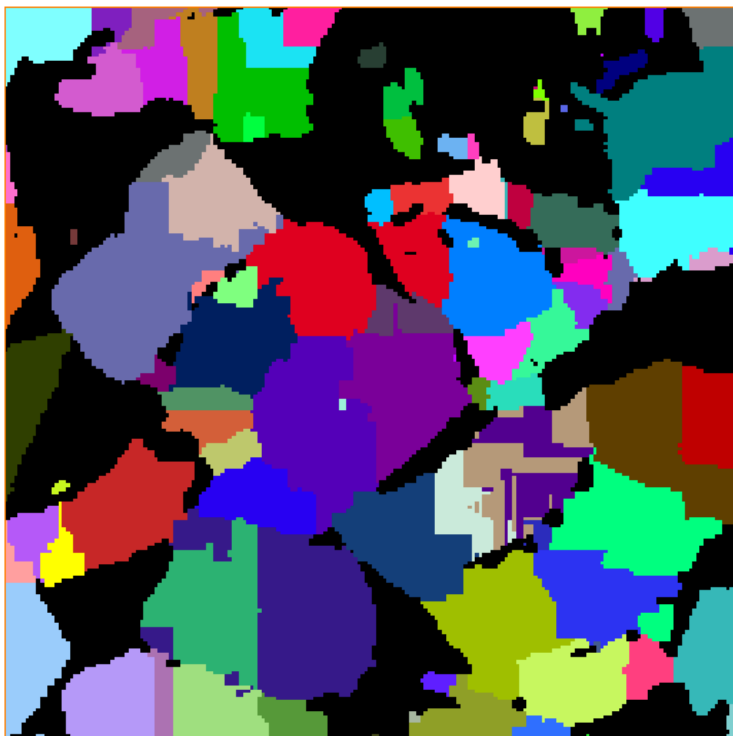


Figure 3.5. Partitioned image of quartz-feldspar rock using connectivity values of six, 820 $\mu$ m per side, 200<sup>3</sup> voxels

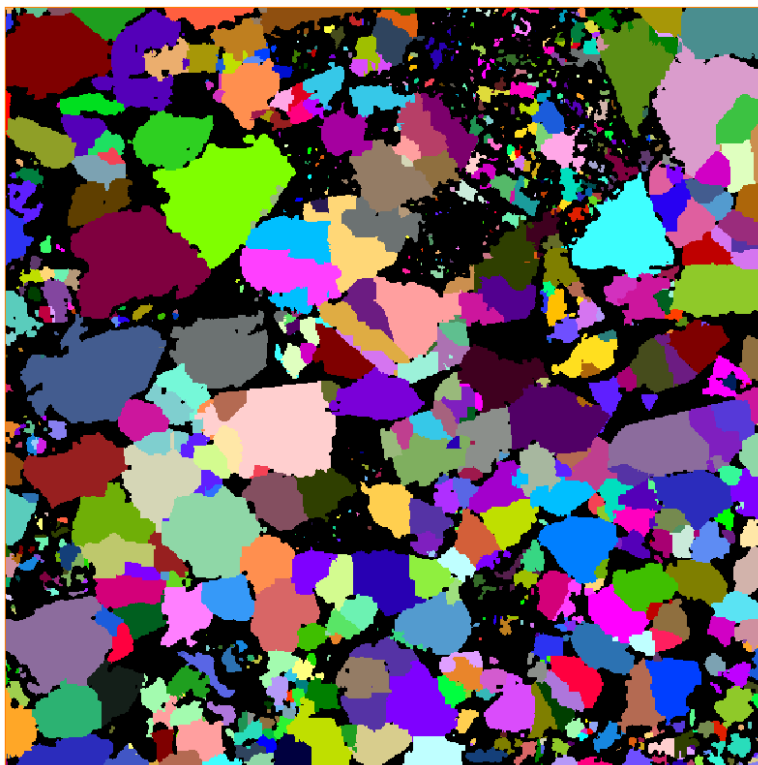


Figure 3.6. Partitioned image of quartz-feldspar rock using connectivity values of twenty-six, 0.246cm per side, 600<sup>3</sup> voxels

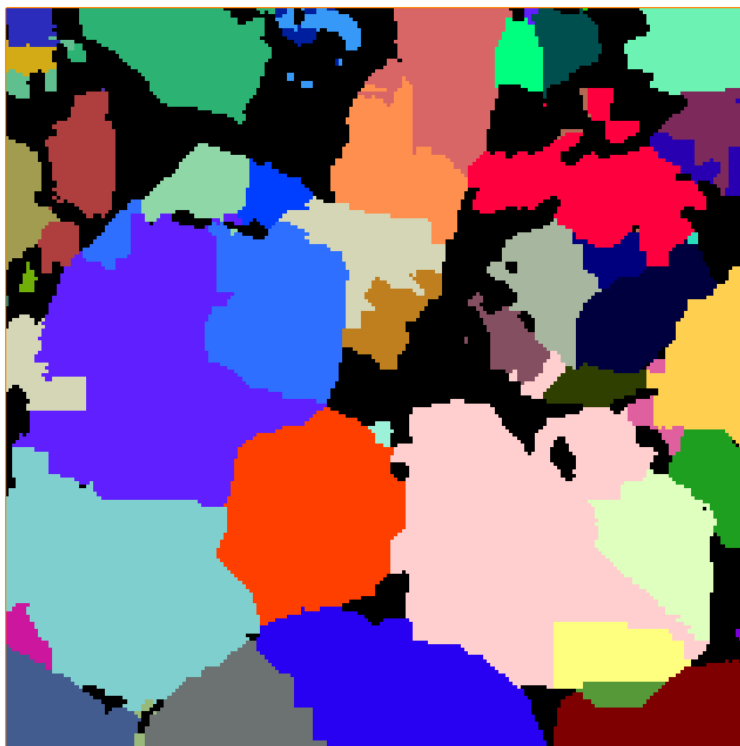


Figure 3.7. Close up view of partitioned image of quartz-feldspar rock using connectivity values of twenty-six, 820 $\mu$ m per side, 200<sup>3</sup> voxels

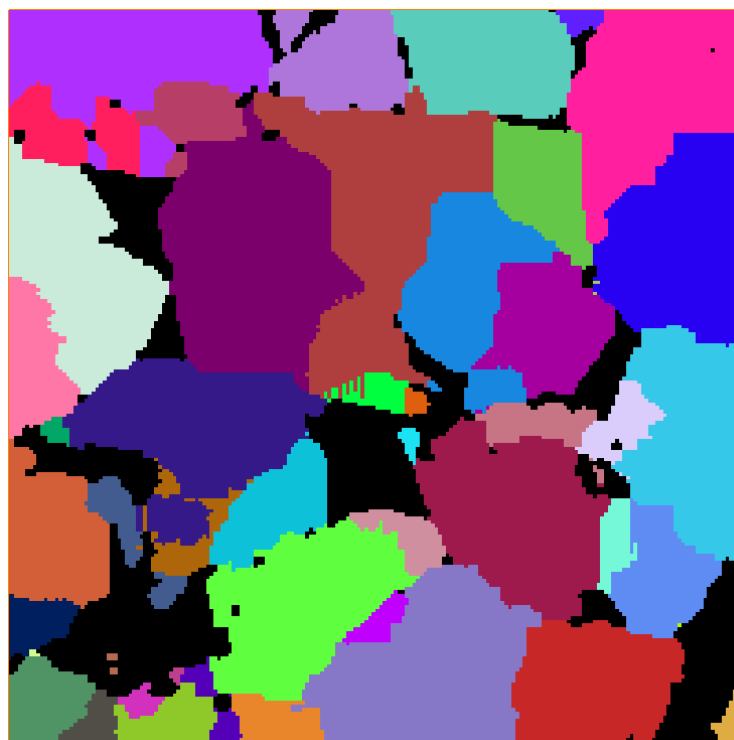


Figure 3.8. Partitioned image of quartz-feldspar rock using connectivity values of six, 820 $\mu$ m per side, 200<sup>3</sup> voxels

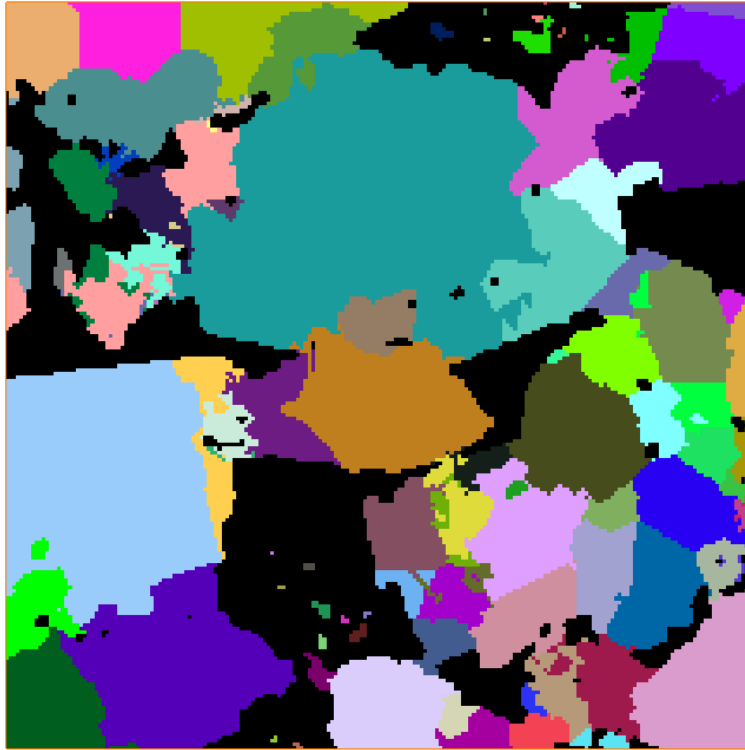


Figure 3.9. Partitioned image of quartz-feldspar rock using burn connectivity value of six and a connectivity for contacts of twenty-six, 820 $\mu$ m per side, 200<sup>3</sup> voxels

Finding specific weaknesses and problem causing scenarios in the program provides insight into what adjustments or alterations could provide remedies. Starting as simple as possible, artificial images with spheres barely touching one another were tested. Visually, this can be thought of as a pack of glass spheres. The program accurately determined these partitions, as shown in Figure 3.10.

Next, the radii of the each sphere was increased, which lead to the spheres to partially overlap one another. In this situation, the program performed well for some of the partitions, but poorly for others. A cross sectional slice through this partition is provided in Figure 3.11.

In order to determine what could be causing these discrepancies, several artificial packs containing only two large overlapping spheres were generated. The sphere sizes remained constant from image to image, but the orientation of the grains was adjusted. In the first image, Figure 3.12, the grains were oriented so that their contact boundary coincided with the Cartesian coordinates of the grid. Figure 3.12 shows that this partition looks accurate both externally and internally. In Figure 3.13, the two grains are intersecting each other a forty-five-degree angle with respect to each of the three Cartesian coordinates. While these grains appeared properly partitioned from an external view, the slice reveals that the interior is erroneous.

These figures show that there was significant need for improved partitioning algorithms within VOX2GRAINS. Critically examining images like those presented in this chapter and understanding the specific situations that give rise to erroneous partitions provides information about what types of productive changes could be made.

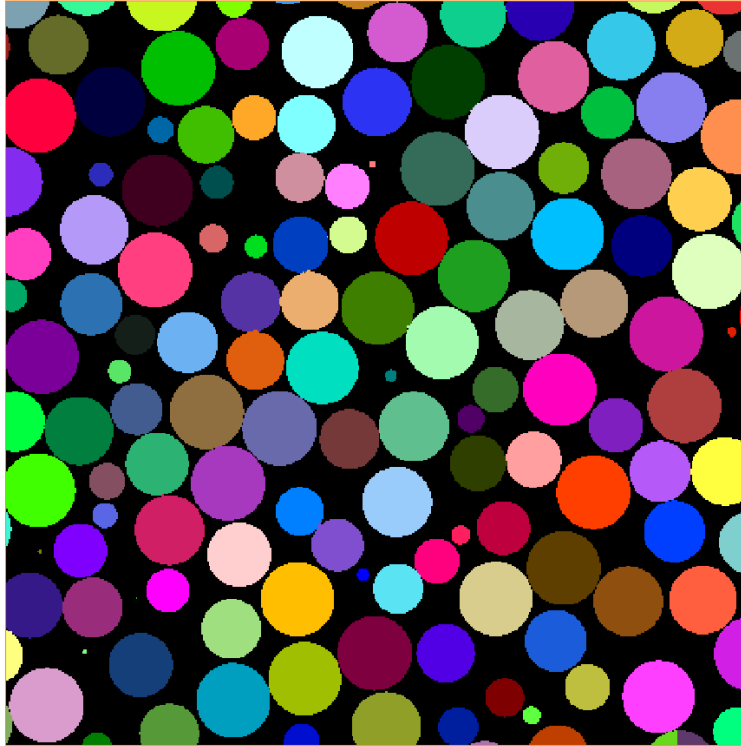


Figure 3.10. VOX2GRAINS partition of unconsolidated sphere pack. 500 voxels per side with a nominal voxel width of 0.01 voxel

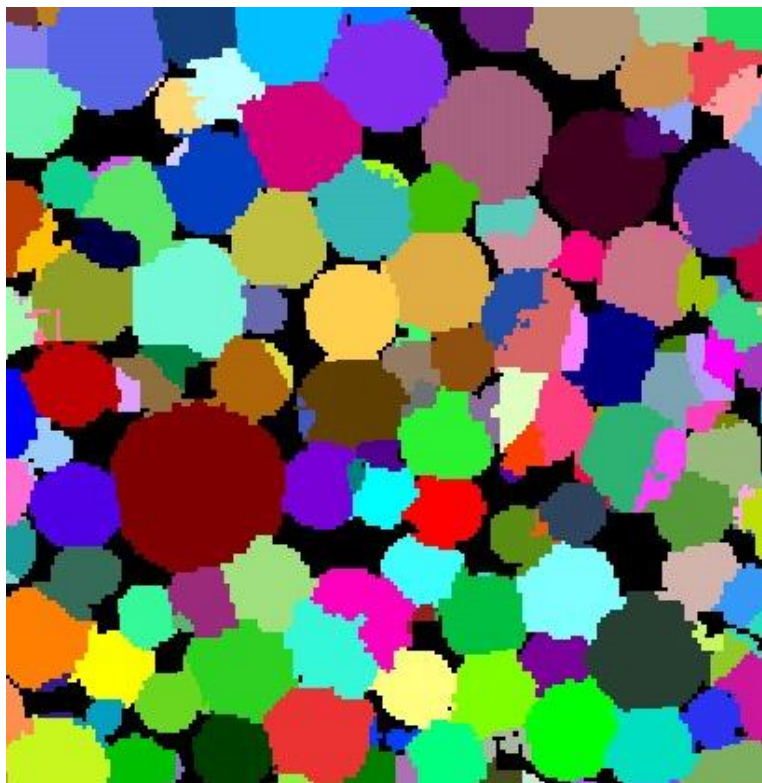


Figure 3.11. Artificial sphere pack with increased radii

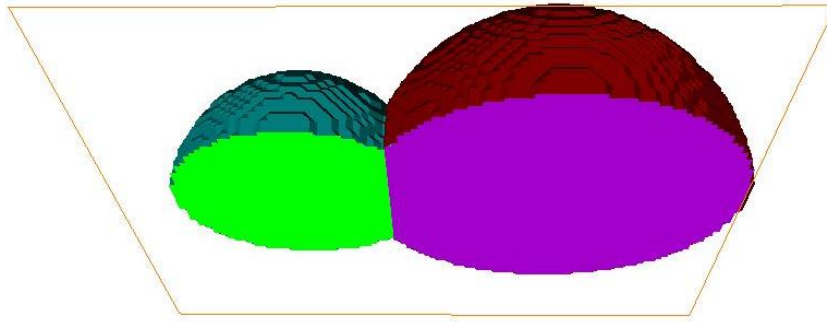


Figure 3.12. Two spheres with an intersection along the Cartesian coordinate direction, 100 voxels per side with a nominal voxel width of 0.01 cm

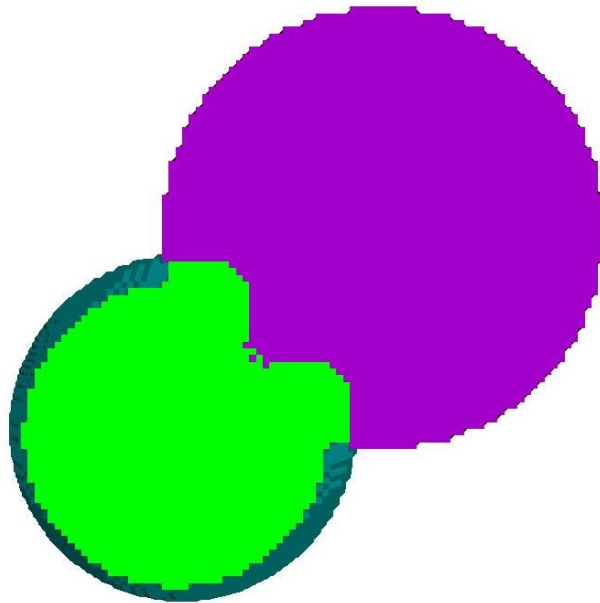


Figure 3.13. Two spheres with an intersection offset by 45 degrees in all Cartesian coordinate directions, 100 voxels per side with a nominal voxel width of 0.01 cm

### 3.2. Main Program Alterations and Additions

A number of alterations and additions were proposed for this program in an attempt to generate a more accurate and robust partitioning algorithm. These changes can be broken into two separate categories: VOX2GRAINS body adjustments and additions, and post-processing refinement adjustments and additions. Although both categories target the same overarching objective of a satisfactory partitioned image, an important distinction is that the post-processing refinement options are geared toward customizability, whereas the body adjustments and alterations are designed to be robust and universal.

### 3.2.1. Burn Map Alterations

The burn map lays the foundation for the partition. Any errors generated within the burn map manifest themselves when the grains are assembled. Ideally, each voxel of the burn map should correspond to the distance that each voxel is from the nearest surface. A sample burn map generated from the previous example of two overlapping spheres is shown in Figure 3.14. In order to determine the accuracy of the burn map, there must be a verified true distance map to use as a comparison. For this purpose, a true distance map generator for two overlapping spheres with specified radii and center locations was created.

The calculations were broken into two separate categories depending on the subject voxel locations: the two conical sections extending from the grain centers to the plane of intersection, and the remaining spheres. For the voxels not within the conical sections, the distances were determined by simply subtracting the total grain radius from the distance of the grain center to the subject voxel. For voxels within the conical sections, the closest surface voxel lies somewhere on the ring of intersection between the two grains. First, the circle of intersect was determined from the grain center locations and the corresponding radii. Then, a plane was created from the subject voxel and the two grain centers. This plane intersects the circle of intersect in two separate locations. The distance from the subject voxel to both intersection points were calculated, and the smaller value was determined to be the minimum distance to the surface.

Since the algorithm calculating the true distance map was solely designed for spheres with known centers and radii, it was not possible to use this program to calculate a true distance map of real rock images. This true distance map simply allowed for an unbiased evaluation tool.

### 3.2.2. Grain Assembly Looping Direction and Neighbor Checking Direction

During grain assembly, the program loops through all voxels in the image and checks if they meet both of two criteria: having a corresponding burn map value either equal to or greater than the current minimum, and neighboring a voxel that has already been assigned to a grain. If the subject voxel meets these two criteria, then the voxel is added to the grain that it neighbors. Initially, the program did not alter looping direction. It was believed that always starting the voxel loop with the same corner favor grains closer to the corner, and therefore extend their boundaries significantly further than it should. This issue of bidirectional looping was thought to be a manifesting itself in the many slots and drawers seen throughout the internal slices of the partitions.

Several different voxel loop adjustments were investigated. As a control, partitions were created with unidirectional looping directions. To compare, two additional scenarios were generated. In the first, the looping direction was altered between two opposite corners. In the second, the starting corner was randomly selected from among the eight available domain corners for each iteration.

For each individual voxel that is passed through the domain loop, the neighboring voxels are checked to see if they are assigned to a grain. In the event that the neighboring voxel is already assigned to a grain, and the subject voxel has a burn map value that is greater than or equal to the current minimum burn value, then the voxel is added to that grain. If the subject voxel is found to be touching two separate grains, then the program assigns the voxel to whichever grain's center is

closer. It was believed that not altering the neighbor check starting location between two corners was insufficient, and was providing favoritism for grains closer to the starting corner. For this reason, the neighbor voxel checking direction was adjusted to alternate between several corners, and later randomized. In order to examine the full impact of randomization, the clause stating that if a new voxel touches more than one grain, it is added to whichever grain it is closest to, was removed for this testing. This provided more power to the inner looping direction pattern or randomization.



Figure 3.14. Burn map for two overlapping spheres, 100 voxels per side with a nominal voxel width of 0.01 cm

### 3.2.3. Assigning Voxels Based Upon Neighbor Counting

In another attempt to further reduce the number of slots and drawers and provide cleaner interfaces, another piece of the grain assembly algorithm was adjusted. As previously mentioned, whenever a voxel has a burn value greater than or equal to the current minimum burn value and also neighbors a voxel already assigned to a grain, then that voxel is assigned to the grain it neighbors. Adjusting the neighbor check looping direction, in theory, minimizes any consistent favoritism. To ensure this reduction, a caveat was added to voxels that neighbor two or more pre-established grains. Originally, the program assigned the voxel to whichever grain had a closer center location. The algorithm was adjusted to assign the voxel to the grain that is neighbors the most number of times. Although this caveat reduces the impact of the randomization or pattern alterations of starting direction when checking the neighbor voxels, it was believed that this would generate more realistic interfaces. It is worth noting that corner adjustments for checking neighbor voxels does maintain some impact through this caveat in situations where the voxel touches multiple

neighbors an equal number of times. In this scenario, whichever neighbor was discovered first is the neighbor to which the voxel will be assigned.

#### 3.2.4. Overwriting Voxels After Grain Assembly Based Upon Neighbor Counting

With all previous additions and alterations, the possibility still arose for smaller particle peninsulas to form during grain assembly. To reduce this phenomenon, a similar approach to assigning voxels based upon neighbor counting was implemented directly following the completion of granular assembly. After all voxels belonging to the phase of interest have been partitioned into grains, a full voxel loop is repeated. For each individual voxel, a tally of which grains neighbor the voxel and how many times each grain neighbors the subject voxel is recorded. If the subject voxel is assigned to grain A, but the program finds only two neighboring voxels belong to grain A, and four neighboring voxels belong to grain B, then the program will overwrite the value of the voxel and reassign it to grain B. If there is a tie between two or more grains, the original assignment remains. As voxels are overwritten, they in turn, could change the status of the neighbors, meaning that often times, this process can be iterated and provide slightly different results after each iteration. Eventually, the image becomes stable, and no necessary overwrites are discovered within the full iteration. For this reason, the overwriting process was designed to be controlled by the user. An input parameter directly controls how many overwrite iterations are performed.

#### 3.2.5. Iterative Grain Assembly

The final addition to the body of VOX2GRAINS implemented a way to iterate over the entire grain assembly process. Although randomization is good in that it reduces consistent favoritism, it could occasionally produce undesired partitions. It was believed that performing grain assemblies several different times and averaging the final image would, overall, reduce any consistent favoritism while minimizing any undesired partitions. This relies on each individual grain assembly iteration creating an image that, for the most part, shows a proper partition. Additionally, it relies on each iteration making different partitioning mistakes. It was believed that this would be the case, due to the impacts of randomization previously described. In other words, with randomization, it was unlikely that the same exact mistake would be made consistently. This idea was implemented by iterating over the entirety of granular assembly and storing each iteration temporarily. The final, averaged image was created by examining the same voxel location on each image, and determining which grain it was assigned to the most number of times. In the event of a tie, the final voxel is randomly chosen from the tying grains. Since this method combines several different images, it follows that simply averaging them together could often lead to isolated voxels, peninsulas, and other interface details that are undesirable. For this reason, all voxels in the averaged image are overwritten based upon neighbor counting. This final overwrite value is equal to the overwrite value for each individual iteration of granular assembly, as specified by the user.

### 3.3. Post-Processing Refinements

After VOX2GRAINS has partitioned the grains and provided various statistics, the user is able to interact with the resultant image in the visualization package Avizo. The watershed algorithm has a tendency to cause grains to be over partitioned, and therefore some refinement might be required in order to arrive at a satisfactory image. The additions specified here aim to provide efficient customizability to ensure that a satisfactory image can be achieved relatively quickly.

### 3.3.1. Automerge

VOX2GRAINS initially had a manual refinement feature, where two individual grains could be selected and merged together, but to go through an entire image with thousands of grains using this method would be extremely time intensive. In an effort to combat this, VOX2GRAINS initially also had an automatic merging feature. The user would input a threshold contact area to surface area ratio, and the program would compare all grain pairs in the image. Grain pairs with contact area to surface area ratios that surpass the specified threshold value were subsequently merged together, and all statistics updated accordingly. While, in theory, this is a great idea for an efficient post-processing feature, it was not particularly effective in merging the over partitioned pairs of grains. In the hopes of finding a more accurate classifier based upon the spatial statistics, several different simple equations, derived solely from logic, were implemented into the program. The initial logic for this feature is shown directly below. THRESHOLD is the user specified value, CONTAREA is the shared contact area between the grains, and GSURF is the surface area of the respective grain. All grain pairs in the image were analyzed with the algorithm.

```
IF [(CONTAREA/GSURF1 > THRESHOLD) AND (CONTAREA/GSURF2 > THRESHOLD)] THEN
    IF [(VOLUME(GRAIN1)>0.5) AND (VOLUME(GRAIN2)>0.5)] THEN
        MERGE GRAINS
    END IF
END IF
```

This logic was designed to ensure that grains were not erroneously merged, in situations like a small sphere overlapping a significantly larger sphere. Figure 3.15 shows a cross sectioned partition that contains many intermediate grains, and Figure 3.16 shows the result of applying a very aggressive ratio of 0.15. Note that the partition shown in Figure 3.15 was determined using several VOX2GRAINS body updates (updated distance map and voxel overwrite iterations). While the impact of these body features will be examined later, the point here is to examine the effectiveness of the initial automerge feature. These images show that even with a more aggressive merge criteria, the program is merging many of the wrong pairs and not merging many of the pairs that are over partitioned. Intermediate and low contract area to surface area criteria values were also tested, but found to yield similar results.

The initial logic was slightly altered in an attempt to better automatically target over partitioned grains. The altered logic is shown below. It was believed that requiring both grains to meet the specified criteria was a major restricting factor, and that extending the algorithm to merge pairs where either grain meets the criteria would increase aggressiveness and improve results.

```
IF [(CONTAREA/GSURF1 > THRESHOLD) OR (CONTAREA/GSURF2 > THRESHOLD)] THEN
    MERGE GRAINS
END IF
```

After looking through several partitioned images for various sphere packs, it was discovered that the program tends to identify an extraneous grain in between two spheres, and often this extraneous grain has no surface area exposed to the void, or to any other phase. Although only shown in a two-dimensional plane here, Figures 3.15 and 3.16 contain several such instances. These intermediate grains are unrealistic and prohibit a favorable partition. To remedy this problem, a

separate feature was created to automatically reassign all grains with no exposed surface area to whichever grain shares the most contact area.

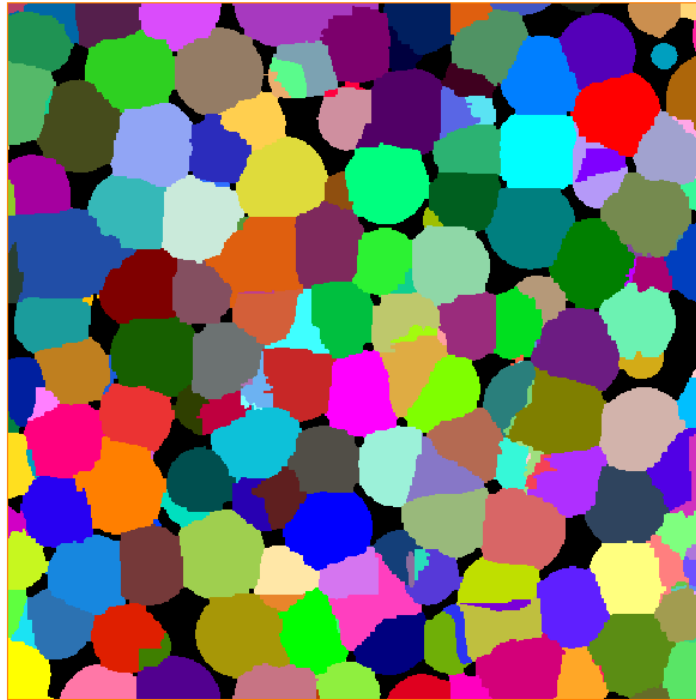


Figure 3.15. VOX2GRAINS output of overlapping sphere pack showing issue of intermediate grains, 500 voxels per side with a nominal voxel width of 0.01 cm/voxel



Figure 3.16. Refined output using the initial automerge logic with a contact area to surface area ratio of 0.15, 500 voxels per side with a nominal voxel width of 0.01 cm/voxel

### 3.3.2. Planar Regression: Single Plane

As shown in many of the previous images, the program would initially often determine the grain interfaces to be jagged, and likely not representative of the true contact. Although there are situations in which a jagged interface between the grains may be correct, there are many instances in which this is known to not be the case. For these situations, the ability for the user to apply a planar regression over the grain-grain interfaces would be extremely powerful, as it would result in a clean interface. Furthermore, it would be beneficial if the plane could be adjusted by both angle and position to minimize the connected voxels between grains, as the local minimized cross sectional area likely constitutes the boundary between two identified grains. The assumption of this post-processing refinement method is very similar to that of the watershed algorithm.

This can be a fairly complex process that is dependent upon the grain shapes, orientations, and coordination numbers (the number of other grains that each individual grain touches). In an attempt to design a program that was robust and could properly handle various grain orientations, two iterations of planar regression were implemented.

The first technique performed one regression plane for each neighboring pair of grains, regardless of whether or not the two grains neighbored one another in separate, isolated contact locations. For example, the grain pair shown in Figure 3.17 would be assigned a single plane for both separate contact locations.

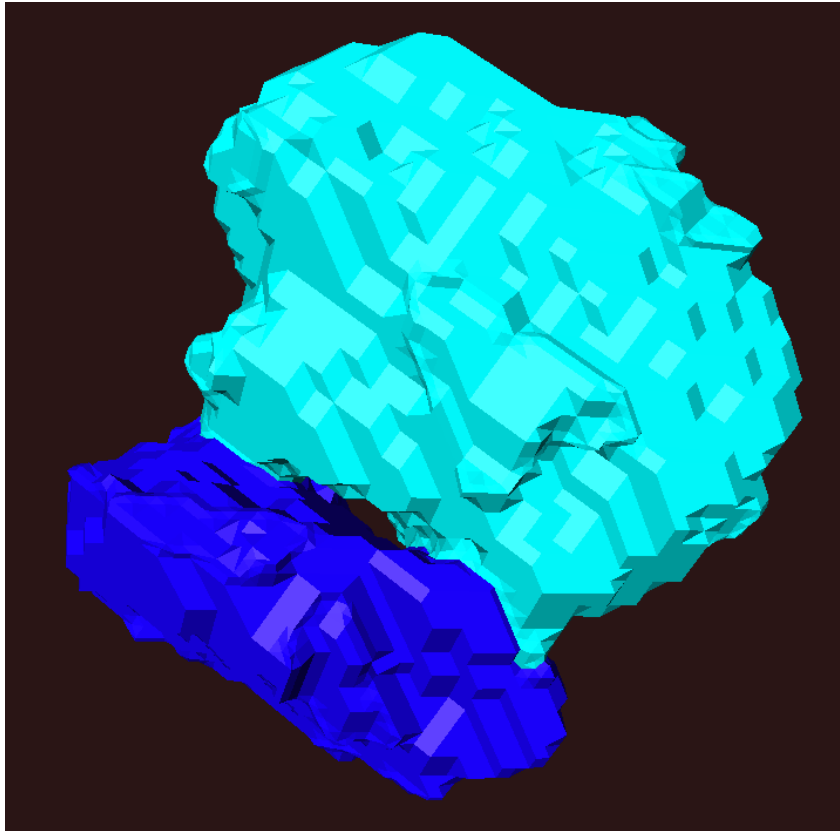


Figure 3.17. A grain pair with more than one isolated contact region

The determination and application of the plane centers and normal vectors for all grain pairs is a multistage process. All grains are looped through in numeric order, according to the material number that each grain is assigned. For each grain, all voxels comprising that grain are identified and stored in an array. To speed up the processing time, instead of looping through the entire domain for each grain, the algorithm begins at the grain's identified center, and expands outward radially until no additional voxels are found. The program gathers the material numbers of the previously identified neighbors for the subject grain. For each neighbor, the algorithm loops through the array containing the subject grain's voxels, and each voxel is checked to determine if it directly borders that grain neighbor. If a voxel does directly touch the grain neighbor, it is added to a second array. Once all voxels touching the current grain neighbor are identified from the list of all voxels of the subject grain, the array of touching voxels is passed to a three-dimensional planar regression solver. This solver (Eberly, 1998) generates the least-squares planar fit from the contact voxels that are input. The result from this regression solver is the plane center of mass, and the unit vector normal to the plane.

Once these parameters are determined, the voxels belonging to both of the grains in the subject pair are stored in an array. Next, the algorithm determines if the center of the grain with the lower material number is on the positive or negative side of the plane. This is calculated as shown in Equation 2.6. *GCLOC* represents the grain contact location, *GPOS* represents the grain position, and *GCVN* represents the grain contact normal vector. The subscripts distinguish between the Cartesian grid components. This calculation is used to determine what material numbers the two sides of the plane will be assigned. If the resultant value is greater than or equal to zero, then the grain center is located on the positive side of the plane, and the corresponding material number will be assigned to all voxels of either grain also on the positive side. If the value is less than zero, it is located on the negative side, and all voxels of either grain also on the negative side will be reassigned the corresponding material number. The voxels on the opposite side of the plane are then reassigned values to correspond with the other grain's material number. It is important to note that this method assumes that the two grain centers are not both on the same side of the plane.

$$(GCLOC_x - GPOS_x) * GCVN_x + (GCLOC_y - GPOS_y) * GCVN_y + (GCLOC_z - GPOS_z) * GCVN_z \quad (2.6)$$

For each individual plane, the contact area (defined as the number of voxels of the smaller material number grain bordering the opposing grain) between the grains is minimized by manipulating the plane's center location and normal vectors, both individually. The center is first adjusted, creating the effect of sliding the plane forward by a specified increment. The new number of contact points with the updated plane location is calculated and compared with the number of contact voxels in the previous location. If fewer contact voxels are found after the slide of one increment, then the new contact voxel number is stored, the plane is pushed further again in the same direction. This process continues until the number of contact voxels no longer decrease. At this point, the minimum contact value and the corresponding plane location are stored, and this process is repeated by sliding the plane in the opposite direction until the contact voxel value increases, at which point the previous step contact value and location are assigned to the plane, and the voxels are overwritten accordingly.

Next, the angle of the plane is altered by adjusting the components of the normal vector. This stage involves individually increasing and decreasing each component of the normal vector individually with a consistent incremental value. For each adjustment, the new contact voxels are determined and compared, just as described when moving the plane center. Once a component reaches a minimum contact value, the corresponding normal vector component is overwritten, and the next component is manipulated. Once all components have been minimized, then the plane is assumed to also have been minimized.

A slight variation of this method was also tested by allowing the minimization check to continue adjusting whenever a slide or angle increase (or decrease) resulted in an equal number of contact voxels. The manipulation of the normal vectors was abandoned after ten full incremental adjustments if a smaller contact voxel value had not been reached. The planar slide was adjusted until the plane arrived at either grain center location, or until the plane arrived at a domain boundary.

### 3.3.3. Planar Regression: Multiplane

The second planar regression method investigated is similar, but has the important distinction of treating the separately isolated contact areas between grains as different interfaces, and regressing and minimizing upon each isolated contact area individually.

The technique begins by identifying all voxels belonging to the subject grain, just as in the previously described method. For each neighbor, the subject grain's voxels are looped through until a contact voxel is found. This voxel is stored in an array, and then the voxel is examined to determine if any of its voxel neighbors also touch the same neighboring grain. If so, the neighboring voxel or voxels is added to the array, and the algorithm continues searching through the list of identified contact clusters for voxel neighbors that link back to the same neighboring grain. This continues until each voxel that has been added to array has been searched for neighbors that meet this criteria. The effect of this is an array that contains all voxels within an isolated contact cluster. The voxel loop through the grain is continued until another voxel that neighbors the same grain is found. The voxel is compared with the previous contact cluster, and if the voxel has already been assigned to a contact cluster, then the voxel loop continues forward to the next voxel. However, if the voxel is not found to be in the previously defined contact cluster, then a new contact cluster array is created, and the algorithm proceeds with the same outward neighboring voxel search, resulting in a second isolated contact cluster. This continues until all voxels of the grain have been looped through, and results in separate voxel position arrays for each isolated contact cluster.

Once the separate contact clusters for each grain pair have been identified, the contact clusters are each regressed individually. The first step is to determine the contact plane center of mass and normal unit vector by passing the subject contact cluster's voxels to the planar regression solver (Eberly, 1998). Next, the algorithm generates a "box" that extends directly above and below all voxels in subject contact cluster, fifteen units in each direction. The cross section of the box is the same shape as the cross section of the contact area between the grains. Any voxel within this region is stored in a third array that will be referred to as the planar box array. In order to assign the material numbers that belong on each side of the contact plane, the program loops through all

voxels within the planar box array and tallies the number of voxels belonging to each material on both sides of the calculated plane. This information is then used to assign which material number will overwrite the positive side of the plane, and which material number will overwrite the voxels on the negative side of the plane. The voxels within the planar box array are overwritten accordingly. For multiplane planar regression, minimization is a specified input. If no minimization is desired, the algorithm stops at this point, and continues to the next grain pair after regressing over each isolated contact cluster. If planar minimization is selected, the program continues as described below.

Since it is possible that resulting plane is not at the true interface, but somewhat receded into one of the two grains, it is important to be able to extend the boundary of the plane in all directions until another phase is met. However, it is important that this plane not be extended too far, or else it could impact the interface between other contact clusters of the same pair. This is an especially crucial step for minimization, because the algorithm seeks to minimize the contact area. If the initial contact area is calculated incorrectly or incompletely, then the resultant interface will likely be far from ideal. An algorithm was developed to allow for restricted growth by individual voxel through this process. The program first identifies and stores all voxels in the planar box array that are within a distance of one voxel from the true plane. The array that these voxels are stored in is referred to here as the planar extension array. The voxels in this planar extension array are looped through, and the voxel neighbors of each are checked. If a voxel neighbor is identified as being within one voxel of the true plane, and if the voxel exactly opposite this voxel with respect to the true plane is identified as either of the subject grains, then the voxel is added to the planar extension array, and the value is overwritten accordingly. This newly added voxel's neighbors will then be searched to extend the plane further. Once all voxels within the planar extension array have had their neighbors checked and no additional voxels are determined to meet the specified criteria, then the planar extension is complete. In theory, this process allows the plane to grow to the complete planar location, but prevents it from interfering with other contact clusters.

After the plane has been extended, there may be isolated chunks of the two grains that need to be overwritten as they were caught under the extension of the plane. For each grain individually, starting at the grain center, the program expands outward, expanding the neighbor search continuously. Unlike the previously described radial expansion from the grain center, this algorithm searches only through direct neighbor contacts. This ensures that every voxel checked has a direct path back to the grain center location. Once this has been completed for both grains and there are two arrays containing the subject grains traceable voxels, the program loops through all voxels marked with either material number and verifies that the locations specified are included in the array leading back to the centers. If a voxel is found to not be traced back to the center, then there is an isolated voxel or cluster, and the voxel will be reassigned the material number of the other grain in the pair. The assumption of this method is that both grains have at least at least one direct path from the grain centers to the voxels aligning the planar contact.

Once the plane has been extended and all voxels have been overwritten accordingly, the user has the ability to automatically adjust the plane so that the contact area between the two grains is minimized. The minimization is with respect to the number of contact voxels of the grain with the smaller material number, as designed in the single plane method. The user has the ability to select between minimization options based upon sliding the plane by adjusting the center of mass, alter

the planar angle by manipulating the normal vector, or both separately. While the basic concept and approach of this minimization technique is similar to the previously described method, there is a difference between the two with respect to how the plane is extended. For each incremental adjustment that is made, the program expands the plane using the same extension method previously described. After the minimization is complete, the voxels of the two grains forming continuous maps back to the centers are stored, and any isolated clusters are once again reassigned. This final step is necessary because during the minimization and extension process, only the voxels directly linked to the new plane are overwritten. This saves computational time by minimizing the number of total voxel overwrites necessary.

#### 3.3.4. Machine Learning

VOX2GRAINS tends to over partition grains, as is common for all programs utilizing the watershed transform. Initially the program was equipped with several tools to try to combat this issue, such as manual merging by specifying two grains and automatic merging by specifying a threshold contact area to surface area ratio. Manual merging is effective, but very time consuming if the image contains a significant number of grains. Automatically merging, on the other hand, is quick, but was not highly effective in merging the over partitioned grains. In an effort to combine the effectiveness and control of manual merging and the efficiency of automatic refinement, an additional option was developed that uses machine learning through logistic regression.

The underlying assumption of the approach taken is that the spatial statistics available in VOX2GRAINS can reasonably determine whether or not a grain pair should be merged. For a given image, a collection of examples displaying both merge and do not merge decisions were recorded with all corresponding available statistics for the subject pair. These spatial statistics and corresponding merge or do not merge decisions were input into a logistic regression algorithm, and the output produced an equation that could be used to predict whether or not two grains should be merged together, based solely upon their spatial statistics. The advantage of this method is that the user is able to control which grains should or shouldn't be merged by selecting the pairs for the training set, and the program will, in theory, automatically merge similar grains across the full image and other images of similar rock type. This provides customizability, control, and is relatively efficient with respect to time required, especially if there are many images of the same rock type. The process and the required program development is described in detail below.

VOX2GRAINS initially performed very well regarding individual grain statistics like surface areas, contact areas, grain volumes, inscribed radii, center of mass positions, major axes, and aspect ratios. This program was expanded to also include statistics about the grain contact locations and grain contact normal vector components. Additionally, a numeric value indicating whether or not the grain touches any of the outer domain boundaries was also added to the grain statistics. This indicator was included because grains touching the outer boundary are not fully representative. The grains touching the domain boundary are chunks of various sizes of grains, and have often presented significant problems for partitioning. Although the program does not partition the interior grains perfectly, the nature of the partitioning issue appears far different for interior and domain boundary touching grains. For this reason, it was this author's belief that these grains should include an identifying distinguished feature in the grains' spatial statistics, so that these data points could be filtered or removed from the training and cross validation sets.

In order to arrive at an accurate logistic regression equation, a significant number of training set examples must be provided. Alternatively, in order to verify that the regression equation can be used to accurately predict a merge or do not merge decision, a significant number of additional examples, often referred to as the cross validation set, must also be gathered.

To aid in data collection, a new feature through the visualization software Avizo was developed. The user selects a button labeled “ML Data Generation” in the created visualization module, and the program automatically shows a particular pair of grains and all of their neighbors in three dimensions. All other grains not directly touching the subject pair are removed from view. The subject grains are always displayed in black and white, and the corresponding material numbers, which can be checked by clicking on the grain of interest, are displayed in the corner of the module to avoid any possible confusion. The grain cluster can be examined in detail by rotating, translating, zooming in and out, and viewing slices through the visible grains. By visually inspection of the pair, the user is prompted to decide if the grain pair should be merged or not merged. Once a decision has been made, the subject grains are assigned random colors, a new pair of subject grains and their neighbors are displayed, and the user is again prompted for a decision. This continues until the user chooses to process the merges by selecting the indicated button. If the user decides to merge a particular pair, and selects the merge option accordingly, the grain values are stored in an array until the user processes the merges. The grains-to-merge array is capable of handling up to one thousand grain pairs. Once it is full, the user is notified and instructed to process the merges. The program was set up this way to make the data generation process quicker for the user. Whenever a merge is processed, it requires reloading the entire updated image, which can be timely for large images. If multiple merges are processed at the same time, the image only needs to be updated once after all merges have been internally processed. After a grain has been marked to be merged but the program has not yet processed the merge, the program will not prompt the user for any subsequent merges pertaining the subject grain. This was implemented to circumvent the issue of possibly merging grain A into B, and grain A into C all in one stage, only to find that grain A is not able to be merged with C since it has already been merged with B. With this system in place, grain A can be merged into B, and after this merge is process, grain B can be merged with grain C.

For the first iteration of this feature, the program looped through all grain pairs in numerical order according to the individual grain’s material numbers. A slight restriction was later added to display only internal grain pairs, where neither grain touched the domain boundary. Additionally, the manual merge feature was adapted to also export all grain statistics with the merge decision to the text file, so that this refinement method could also allow for machine learning data collection. This was useful since the number of do not merge decisions in the automatically prompted method typically far outweighs the number of merge decisions. Although the total number of each decision for the example sets do not need to be equal or even approximately equal, there needs to be a sufficient number of both in order to achieve meaningful and useful results. Manual refinement provided the opportunity to generate a reasonable amount of merge decisions in the training set.

The three dimensional segmented images used to evaluate this refinement option were graciously donated by Dr. Ning Lu of the Colorado School of Mines Department of Civil and Environmental Engineering. These images have been analyzed before using a previous iteration of VOX2GRAINS (Willson, 2012), though never with this particular refinement method. These

images are ideal for this sort of testing because all five samples were of the same rock type, Ottawa (F-75) sand. This is important because it is believed that a logistic regression equation determining merge probability would only be applicable to the same or similar rock types. Additionally, as discovered in (Willson, 2012) the grain sizes are fairly consistent throughout all the separate images, and the rock shapes are angular, which provides additional difficulty for VOX2GRAINS. The five images were cropped so that each was a cube with 275 voxels, each side representing a length of 3.025 mm. Smaller images have the advantage of displaying grain pairs faster during the decision process, thus saving time, but also come with the disadvantage of having fewer interior grains, which results in fewer data points. Cropping the images to 275 voxels per side was found to be fairly quick when updating the visualization (approximately 5-10 seconds for each pair) while simultaneously providing enough interior data points.

Once a merge or do not merge decision is made, a text file is appended in the working directory with the grain numbers, all available corresponding spatial statistics, and the decision to merge or to not merge. Each decision constitutes one example. A total of 1985 examples were gathered, taken from each of the five similar images. After filtering out all examples that contained a grain touching the domain boundary, the examples were randomized and divided into a training set and a cross validation set made up of approximately 60% and 40%. Tables 3.1 and 3.2 show the origin and breakdown of the training and cross validation sets.

Once the data sets were separated, a logistic regression equation was generated from the training set data through a script in MATLAB. The resulting output was the  $\theta$  array that minimized the difference between the logistic regression equation's prediction and the training sets recorded decision value (0 for do not merge, and 1 for merge). This logistic regression equation was then used to predict whether or not each pair should be merged for the cross validation set. The predictive accuracy with respect to the cross validation set was recorded. This process was repeated several times, after adjusting which original features were and were not included, adding interaction features, applying regularization to the equation to minimize overfitting of the training set, and feature scaling to determine the relative weight of each variable. Table 3.3 summarizes the different specifications for the different trials.

Table 3.1. Machine learning data origin

<b>Image ID</b>	<b>Merge Count</b>	<b>Do Not Merge Count</b>
CSM1	76	185
CSM3	68	148
CSM5	65	386
CSM7	99	368
CSM10	67	305
Total	375	1392

Table 3.2. Training and cross validation set breakdown

	<b>Total</b>	<b>Merge Count</b>	<b>Do Not Merge Count</b>	<b>% Merged</b>
<b>Training Set</b>	1060	206	854	19.43%
<b>Cross Validation</b>	707	169	538	23.90%
<b>Combined</b>	1767	375	1392	21.22%

Table 3.3. Specifications for each machine learning logistic regression Trials 1.3

<b>Trail Number</b>	<b>Trial 1</b>	<b>Trial 2</b>	<b>Trial 3</b>
<b>Description</b>	No Feature Scaling, No Regularization, Only Interior Grains	Regularization Included, Only Interior Grains	Feature scaling included, interior grains only
<b>Features</b>	Surface Area Grain 1	Surface Area Grain 1	Surface Area Grain 1
	Surface Area Grain 2	Surface Area Grain 2	Surface Area Grain 2
	Volume Grain 1	Volume Grain 1	Volume Grain 1
	Volume Grain 2	Volume Grain 2	Volume Grain 2
	Inscribed Radius Grain 1	Inscribed Radius Grain 1	Inscribed Radius Grain 1
	Inscribed Radius Grain 2	Inscribed Radius Grain 2	Inscribed Radius Grain 2
	Number of Contacts Grain 1	Number of Contacts Grain 1	Number of Contacts Grain 1
	Number of Contacts Grain 2	Number of Contacts Grain 2	Number of Contacts Grain 2
	X Location Grain 1	X Location Grain 1	X Location Grain 1
	Y Location Grain 1	Y Location Grain 1	Y Location Grain 1
	Z Location Grain 1	Z Location Grain 1	Z Location Grain 1
	X Location Grain 2	X Location Grain 2	X Location Grain 2
	Y Location Grain 2	Y Location Grain 2	Y Location Grain 2
	Z Location Grain 2	Z Location Grain 2	Z Location Grain 2
	Contact Area	Contact Area	Contact Area
	X Major Axis Grain 1	X Major Axis Grain 1	X Major Axis Grain 1
	X Major Axis Grain 2	X Major Axis Grain 2	X Major Axis Grain 2
	Y Major Axis Grain 1	Y Major Axis Grain 1	Y Major Axis Grain 1
	Y Major Axis Grain 2	Y Major Axis Grain 2	Y Major Axis Grain 2
	Z Major Axis Grain 1	Z Major Axis Grain 1	Z Major Axis Grain 1
	Z Major Axis Grain 2	Z Major Axis Grain 2	Z Major Axis Grain 2
	Aspect Ratio Grain 1	Aspect Ratio Grain 1	Aspect Ratio Grain 1
	Aspect Ratio Grain 2	Aspect Ratio Grain 2	Aspect Ratio Grain 2
	Contact Center Location X	Contact Center Location X	Contact Center Location X
	Contact Center Location Y	Contact Center Location Y	Contact Center Location Y
	Contact Center Location Z	Contact Center Location Z	Contact Center Location Z
	Contact Normal Vector X	Contact Normal Vector X	Contact Normal Vector X
	Contact Normal Vector Y	Contact Normal Vector Y	Contact Normal Vector Y
	Contact Normal Vector Z	Contact Normal Vector Z	Contact Normal Vector Z

## CHAPTER 4. RESULTS

Grain partitioning can sometimes be subjective, especially in situations where the individual grains are more irregularly shaped and there is significant consolidation. The evaluation of all results presented here are based purely upon visual inspection, unless otherwise stated. In an attempt to minimize any personal bias of the author and to accurately portray the progression and impact of each portion of the code, many before and after images are presented. Unless otherwise stated, the “before” images were generated using all techniques that the results have previously presented. In other words, the program continued to use each addition and alteration, so the “before” images include each technique for which the results have previously been shown. This allows for a relative understanding of the impact for each individual technique. Also, many of the images presented here are two-dimensional slices of a full three-dimensional image. Although it has been noted that two dimensional representations and slices do not provide the full understanding as they lack some information, they are useful and necessary for detailed analysis of the grain-grain interfaces. The conclusions drawn and trends observed from any two-dimensional images were verified throughout the full three-dimensional images.

A majority of the initial development testing was performed on artificially generated sphere packs. When partitioning spheres, there is little room for argument about what constitutes a single grain, and what constitutes multiple grains, assuming the spheres do not overlap extensively. This allowed for objective testing for various algorithms. Also, it is relatively easy and quick to manipulate sphere packs to meet certain specifications, such as sphere overlap and orientation of sphere-sphere contacts. Lastly, it was known that the program was capable of accurately partitioning sphere packs in which the spheres barely touched one another, however it was also known that program had trouble with more consolidated sphere packs. It was believed that if there were robust partitioning improvements that objectively showed enhanced capabilities through the partitions of overlapping sphere packs, then the program would also perform better when partitioning more realistic images with angular and irregularly shaped grains.

### 4.1. VOX2GRAINS Body Improvements

Overall, the VOX2GRAINS body alterations and additions combine to yield significantly more realistic particle partitions for the images that have been tested.

#### 4.1.1. Burn Map Alterations

The burn map lays the foundation for all grain partitioning, since the particle assembly phase is restricted according to the individual burn map value for each voxel. The VOX2GRAINS generated burn map and the geometrically calculated true distance map were compared to one another for an image containing two overlapping spheres, offset by 45 degrees in all three Cartesian grid dimensions. Figure 4.1 shows a slice through the VOX2GRAIN generated burn map, and Figure 4.2 shows the same slice through the geometrically calculated true distance map. Each color represents a different burn level that, in theory, corresponds the distance from that voxel to the nearest surface.



Figure 4.1. Burn map for two overlapping spheres, 100 voxels per side with a nominal voxel width of 0.01 cm

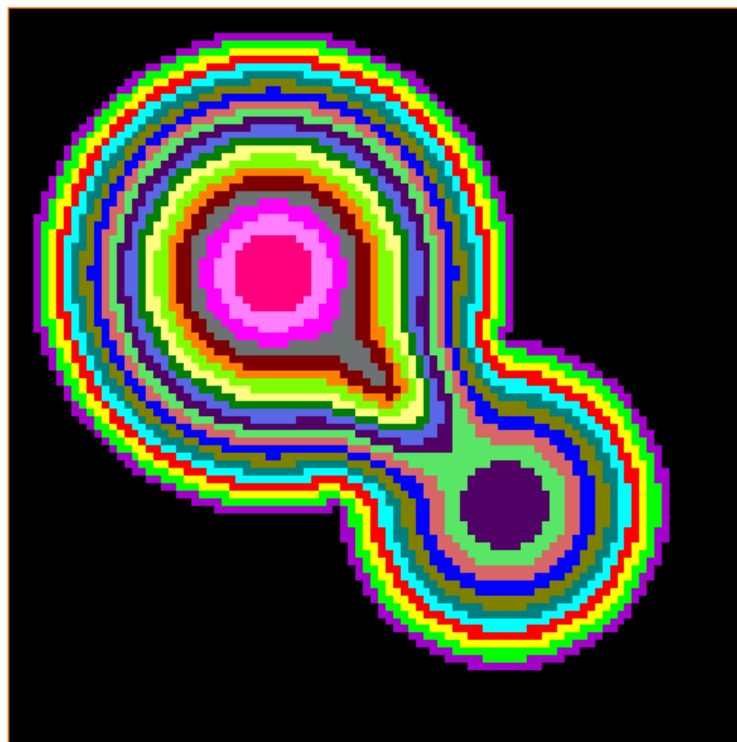


Figure 4.2. True distance map for two overlapping spheres, 100 voxels per side with a nominal voxel width of 0.01 cm

From a cursory examination, there are very obvious differences between the burn map and the true distance map. The burn map appears boxier in parts, and has a bullet shaped artifact protruding from the top left grain into the bottom right grain near the overlapping region. The true distance map, on the other hand, appears far smoother through all parts of the image.

The true distance map calculations were designed only for the case of two overlapping spheres with known radii and center locations. While the true distance map algorithm was useful for pointing out flaws of the burn algorithm, it was not robust enough to generate a true distance map for more realistic rock images, or even larger sphere pack images. For this reason, a procedural way to generate accurate distance maps for segmented images containing particulate matter of any shape was required. Once generated, the distance map could be passed into the program and replace the burn map. Avizo, the visualization software used throughout this work, has the capability of calculating a Euclidean distance map. Since the Avizo distance map was generated solely from the segmented image, and required no prior knowledge of geometry or center locations, the tool was considered as a universal distance map generator. The Avizo Euclidean distance map is shown in Figure 4.3 for the same overlapping two spheres. The map is very similar to the true distance map, which means that the Avizo Euclidean distance map provides a sufficiently accurate and robust map that can be passed into the program and used as the burn map. The partitions generated from the burn, true distance, and Euclidean distance map are shown in Figure 4.4.

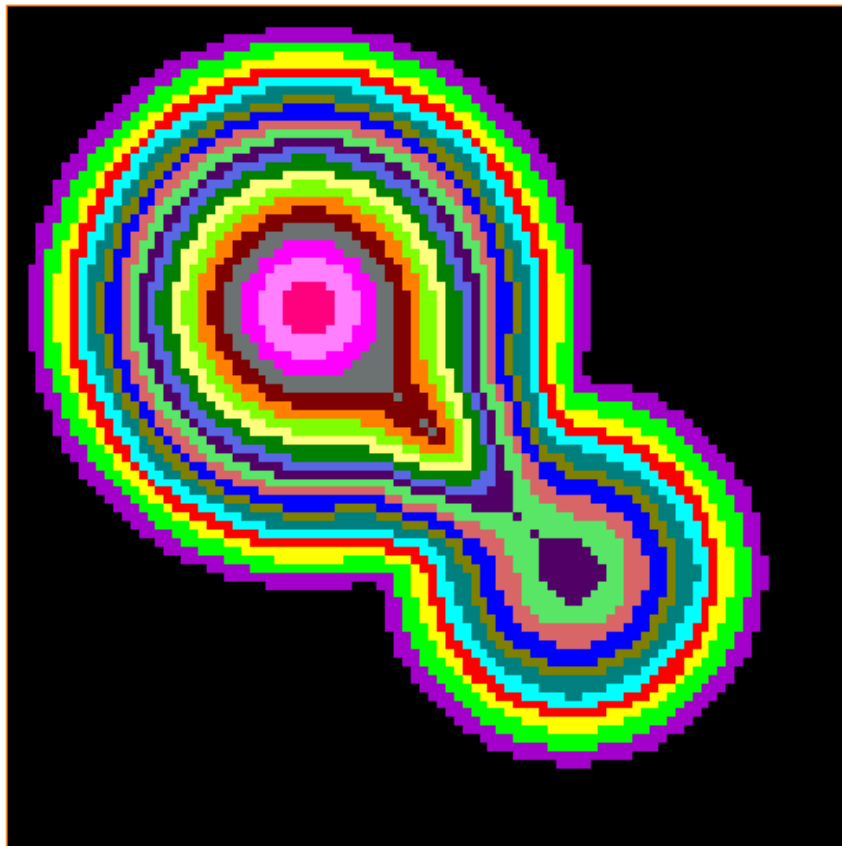


Figure 4.3. Avizo Euclidean distance map for two overlapping spheres, 100 voxels per side with a nominal voxel width of 0.01 cm

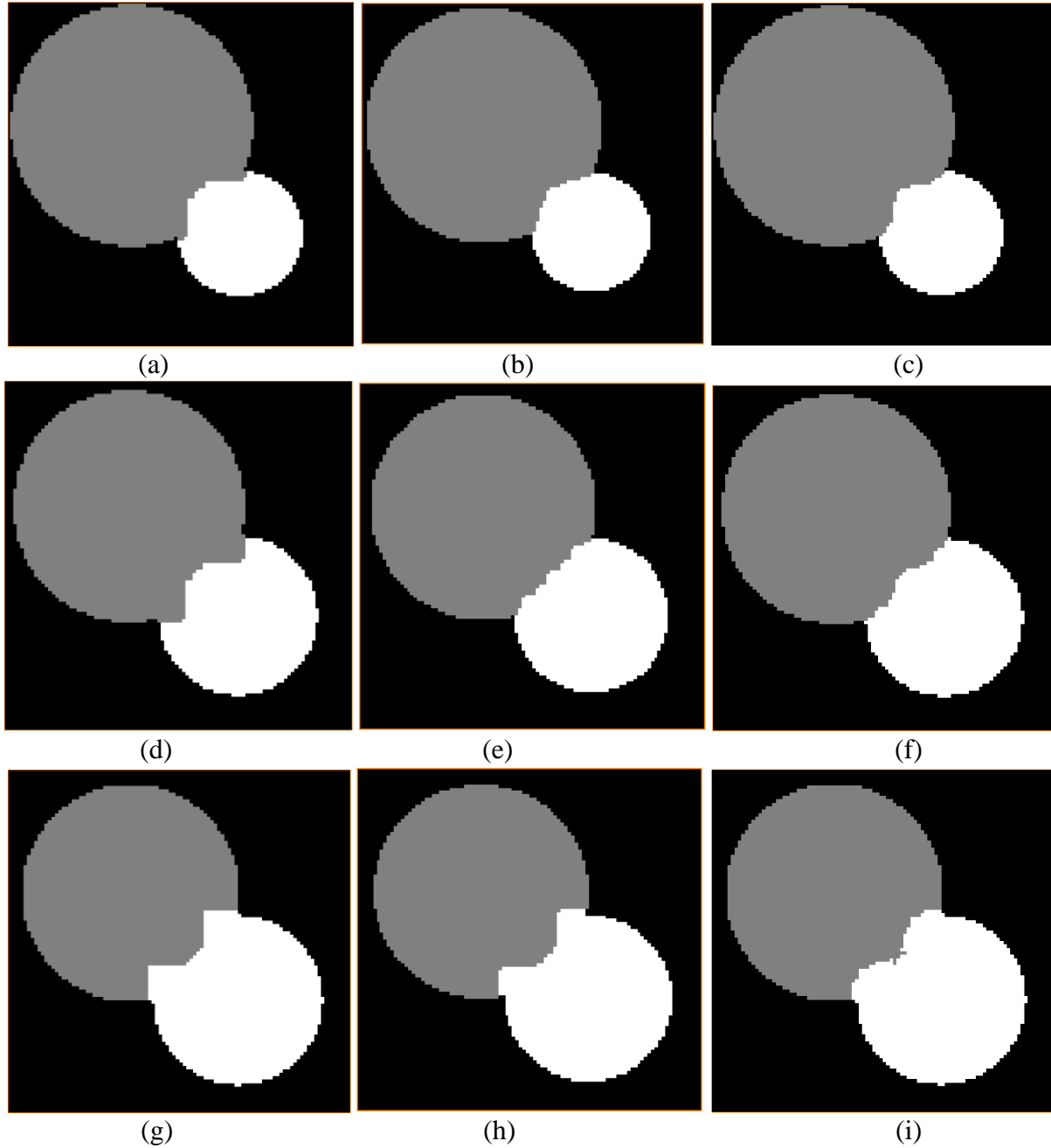


Figure 4.4. Three slices through partitions for two overlapping spheres generated by burn map (a, d, g), true distance map (b, e, h), and Euclidean distance map (c, f, i) holding all other parameters constant, 100 voxels per side with a nominal voxel width of 0.01 cm

The partitions generated from the true distance map and Euclidean distance map are far more favorable than the partition generated from the burn map. Though still not ideal, these results show that a more accurate distance map is able to eliminate and reduce some of the peaks and valleys that appear in the partition.

In order to make general use of the Avizo Euclidean distance map, several issues in the raw distance map needed to be accounted for. By default, the Euclidean distance map produced by

Avizo erodes the most external layer of the phase of interest. The program rounds the real distances, so if the distance is less than 0.5, then the voxel is assigned a value of zero. In order to combat this problem, all distance map values initially holding a value are increased by one, and all distance map values that have a value of zero are compared with the corresponding voxel in the segmented image. If the corresponding voxel in the segmented image holds the value of the phase of choice, then the distance map voxel is given a value of one. This process ensures that the distance map does not erode the image, and that the image is also not over-enlarged.

Although the HPC version of this program was able to handle images with multiple phases, the Avizo version could not. This functionality was expanded, but added an additional hurdle for calculating the distance map. When calculating a distance map, Avizo lumps all material values greater than zero into one category, meaning the resultant distance map would be for two or more phases lumped together. This issue is circumvented by a short algorithm that overwrites the voxel values of the phases not of interest to a value of zero, which represents void. Once this was completed, the only positive value that remains is the phase of interest, and the distance map can be calculated as normal. This is strictly internal, so no damage is done to the original image since VOX2GRAINS creates and exports the final partitioned image by default. Finally, one last distance map issue arises when the void phase is input as a positive number in the segmented image, and the phase of interest is zero. In this situation, all voxels belonging to the phase of interest are assigned an extremely large dummy value, and all voxels that are not of the phase of interest are assigned a value of 0. This reorders all the voxels so that the phase of interest is positive, all other phases are zero, and the distance map can now correctly be calculated.

In order to make quicker and more efficient use of the Euclidean distance map capabilities as they pertain to grain partitioning, a user defined module within Avizo was created that rearranges any necessary phases and provides an isolated distance map for the phase of interest. The output can easily be exported so that it may be passed into the HPC version of the program, just as the segmented image is passed. If performing grain partitioning within Avizo, this process will be carried out automatically, and the Euclidean distance map will be passed into the program to act as the distance map.

#### 4.1.2. Grain Assembly Looping Direction and Neighbor Checking Direction

During the grain assembly process, the voxel looping direction of the program initially never changed. The program was expanded to alternate between two opposite starting corners and also to randomly select the starting position of each iteration from the eight available corners. Similarly, each voxel's neighbors were checked originally in the same direction with each iteration. The program was expanded to alter between two and four separate corners. The impact of these expansions were tested on an artificially generated sphere pack containing only two spheres. Recalling that the program originally did not struggle significantly with determining the interface locations on spheres as seen from the exterior, attention here is focused on the interior interface. Figures 4.6 shows three slices through two overlapping spheres. The left images (a, d, g) show the partition generated by keeping the looping direction constant, the middle (b, e, h) was created by alternating between two inner and outer opposite starting corners, and the right image (c, f, i) was generated by randomizing both the interior and the exterior starting positions.

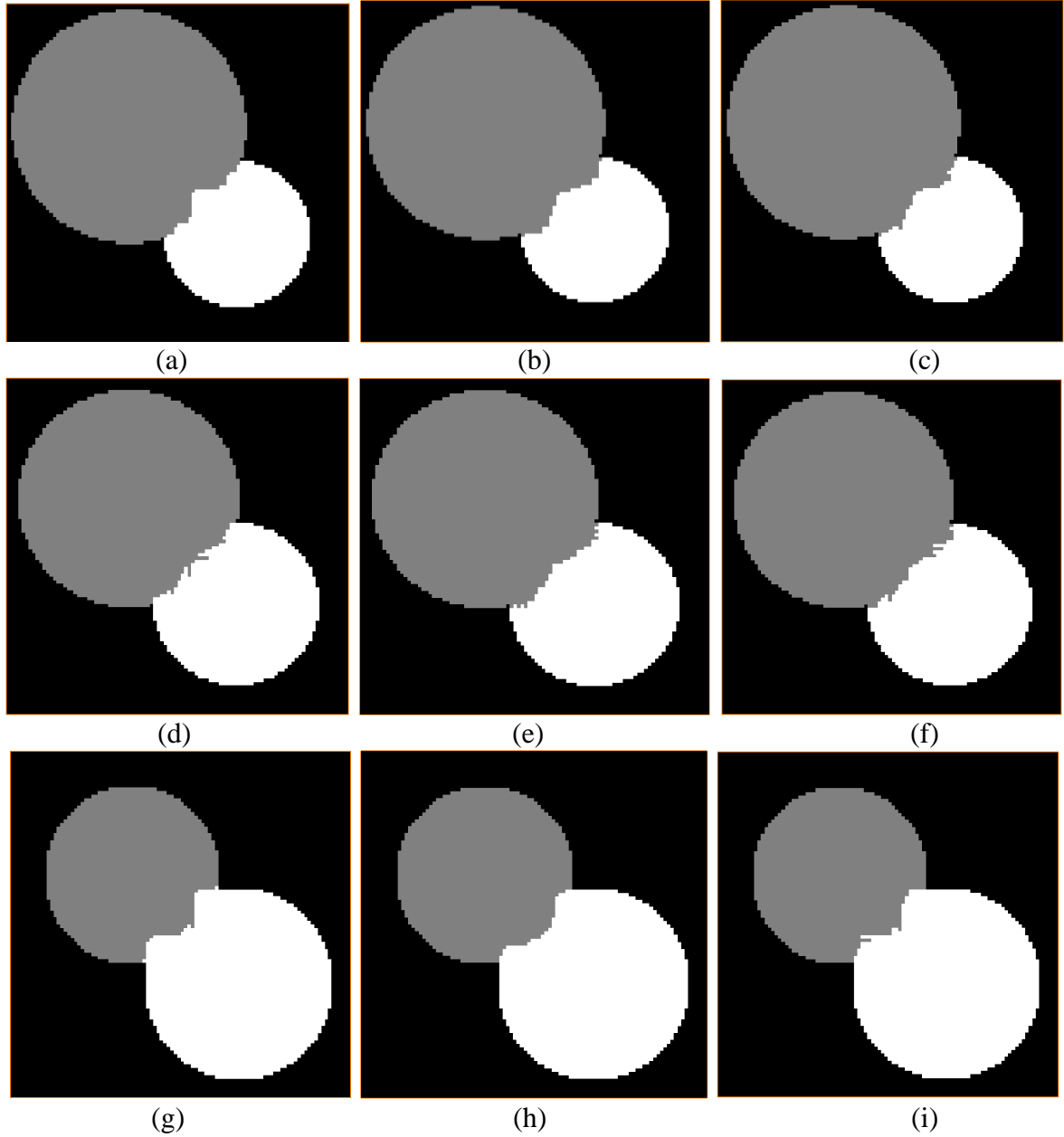


Figure 4.6. Partitioned impact of always using the same looping corner (a, d, g), alternating between two corners (b, e, h), and randomizing the corners (c, f, i) for two overlapping spheres, 100 voxels per side with a nominal voxel width of 0.01cm

From Figure 4.6, the results show that two alternating corners appears to yield slightly better partitions. When applied to more difficult geometries that are found in true rock samples, as shown in Figure 4.7, the results are less conclusive.

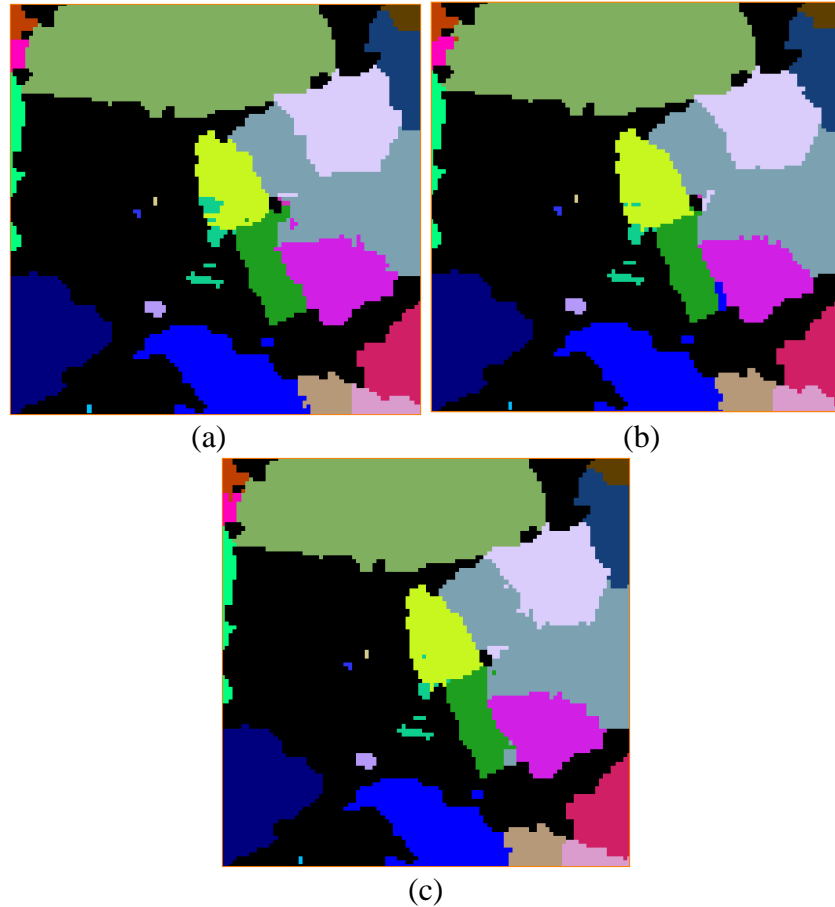


Figure 4.7. Partitioned impact of always using the same looping corner (a), alternating between two corners (b), and randomizing the corners (c) for quartz-feldspar image, 820 $\mu$ m per side, 200<sup>3</sup> voxels

#### 4.1.3. Assigning Voxels Based Upon Neighbor Counting

During the grain assembly stage, the program initially assigned voxels that touched two or more grains to whichever grain had the closest center. As mentioned previously, this was removed to implement randomization into neighbor checks. Instead of randomly assigning every voxel with multiple grain neighbors, all neighbors are counted, and the voxel is assigned to the grain the voxel neighbors the most. In the event of a tie, the grain is assigned to whichever grain was first found to be a contact. Since there are usually a significant number of voxels to be assigned that have equal number of neighbors, randomization of the neighbor check direction remained in place to hinder any continuous bias development. Figure 4.8 compares several slices through the partitions generated from this neighbor counting method with randomization and the original method with randomization. The images on the left show the partition generated by assigning the voxel to whichever identified center is closest, and the images on the right show the same slices through the partition generated by neighbor counting. These images can also be compared with the complete randomized partitions showed in Figure 4.6. Although still not completely ideal, the images show that assigning voxels with multiple grain contacts using an algorithm based on the number of neighboring voxels each grain provides the most accurate grain-grain interfaces for the images tested.

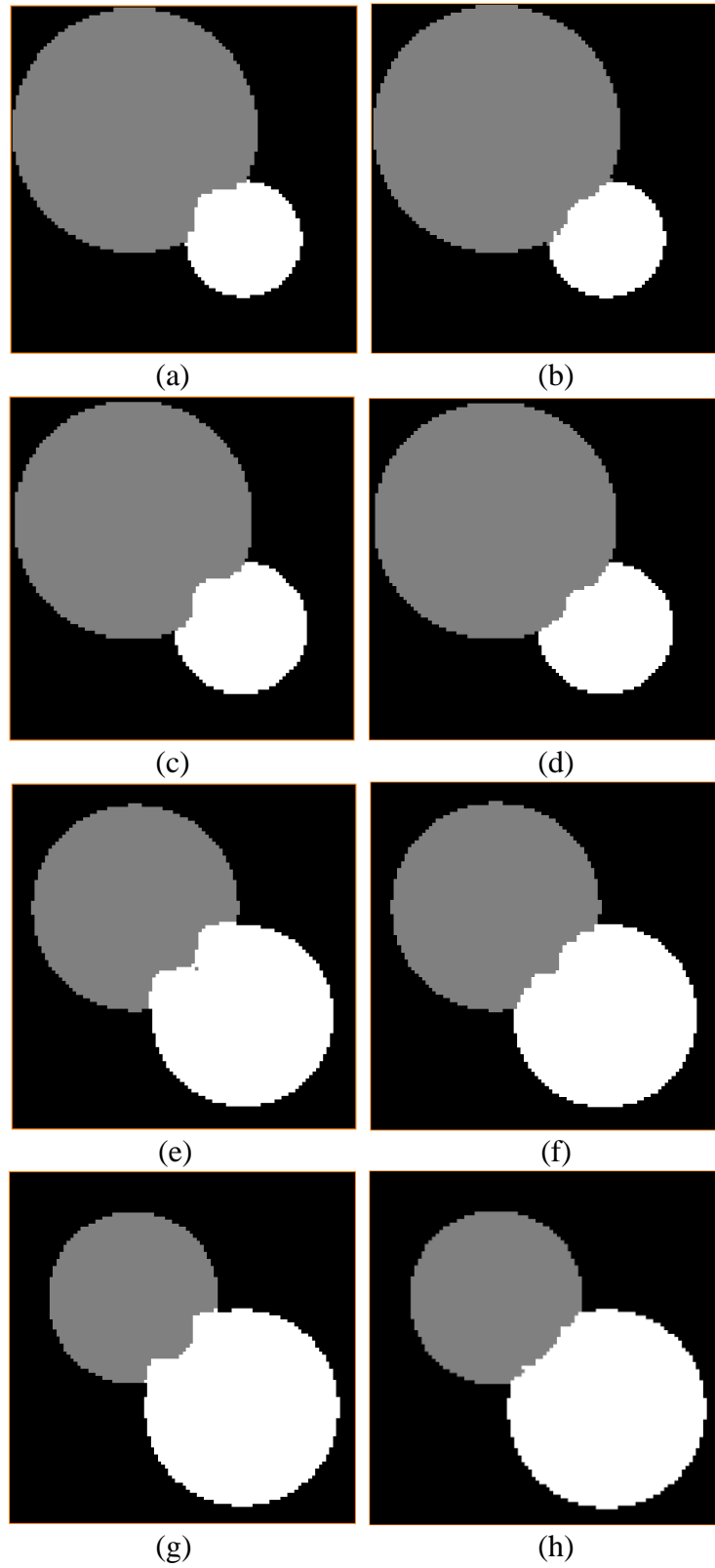


Figure 4.8. Slices through partitions generated by using the closest center method (a, c, e, g) and neighbor counting (b, d, f, h), 100 voxels per side with a nominal voxel width of 0.01cm

#### 4.1.4. Overwriting Voxels After Grain Assembly Based Upon Neighbor Counting

Overwriting each voxel after grain assembly based upon the values of the neighboring voxels was implemented in an effort to reduce the slots and drawers at grain-grain interfaces. This user controls the number of voxel overwrite iterations through an input. Values analyzed range from zero to ten. To best illustrate the impact of this feature on the overall partition, a cropped version of the previously described quartz-feldspar and clay image from (Zanjani, 2016) was tested with various iterations of voxel overwrites, holding all other parameters constant. Figure 4.9 shows a cross section of this segmented image. The black voxels represent the void phase, the grey voxels represent the clay phase, and the white voxels represent the quartz-feldspar phase, which, for the tests performed, was the phase of interest. This image is a total of 200 voxels cubed, and each voxel has an individual length of 4.1  $\mu\text{m}$ , meaning that each side of this cubed image is 820  $\mu\text{m}$ .

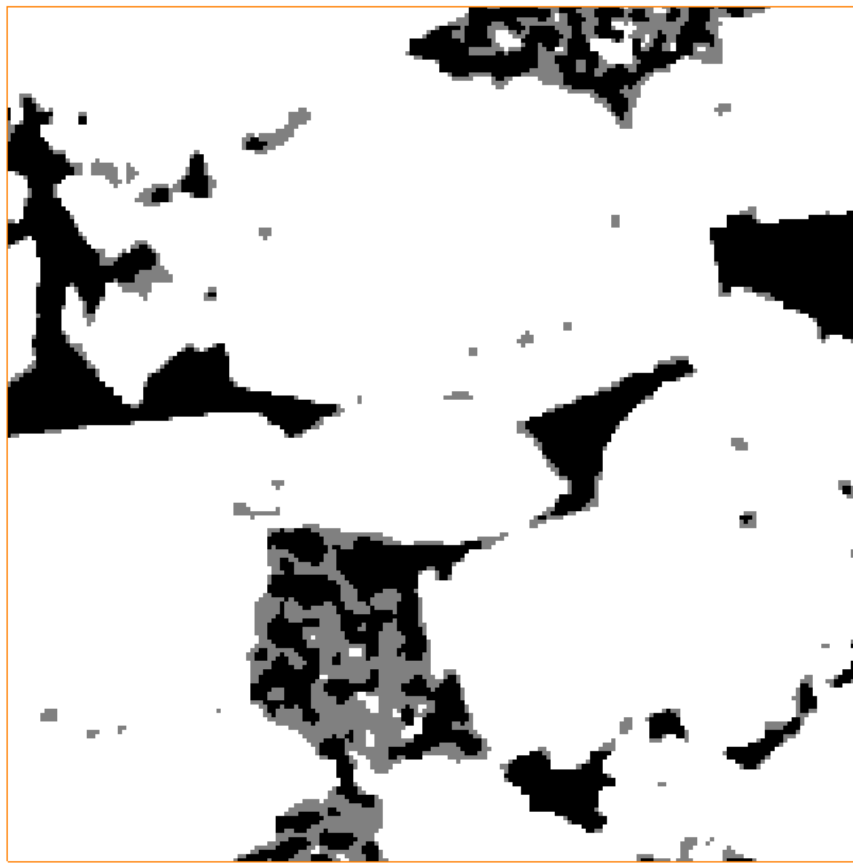


Figure 4.9. Segmented image prior to partitioning, 820 $\mu\text{m}$  per side, 200<sup>3</sup> voxels

Figure 4.10 shows a slice through the partitioned image with zero voxel overwrite iterations. Many of the grain-grain interfaces of this slice show that there are slots and drawers present, and that the interfaces are not very smooth.

Figure 4.11 shows the same slice through the partitioned image, whenever there are three voxel overwrite iterations. The interfaces between the grains are significantly smoother than in Figure 4.10, and there are no longer any slots or drawers in this cross section. The colors representing the grains are different in Figures 4.9 and 4.10. This is because there are fewer grains in Figure 4.10

and the color map has been shifted accordingly. Zero voxel overwrite iterations resulted in a partition containing 1190 total grains, whereas three voxel overwrite iterations resulted in 1041 grains.



Figure 4.10. Partitioned quartz-feldspar image after 0 voxel overwrite iterations, 820 $\mu$ m per side, 200<sup>3</sup> voxels

Figure 4.12 shows the same cross section of the partition resulting from ten voxel overwrite iterations. Once again, fewer grains are identified, as hinted by the different colors. Ten voxel overwrite iterations results with a total of 1037 partitioned grains, only four fewer grains than the result from three voxel overwrite iterations. Comparing Figures 4.12 with 4.11, we see that the interfaces are, for the most part, slightly more refined and smoother when ten iterations are applied. Over partitioning still exists, but has been reduced, and the grain-grain interfaces look far more realistic than those of Figure 4.10.



Figure 4.11. Partitioned quartz-feldspar image after 3 voxel overwrite iterations, 820 $\mu$ m per side, 200<sup>3</sup> voxels

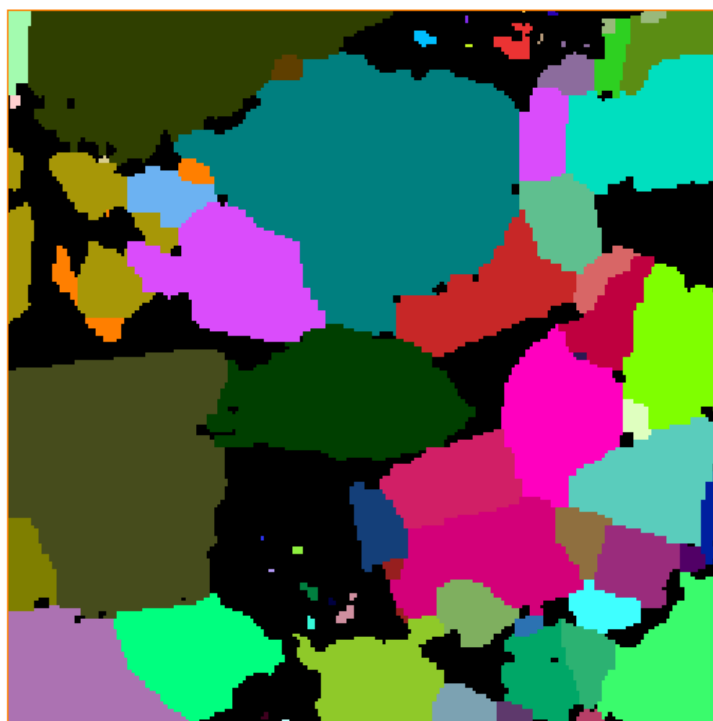


Figure 4.12. Partitioned quartz-feldspar image after 10 voxel overwrite iterations, 820 $\mu$ m per side, 200<sup>3</sup> voxels

#### 4.1.5. Iterative Grain Assembly

Iterating upon grain assembly and averaging the resulting image has proven to yield sharper interfaces between the grains. Figure 4.13 illustrates slices through a two sphere pack with (a) one grain assembly iteration, (b) three grain assembly iterations, and (c) five grain assembly iterations. There were 0 voxel overwrite iterations based upon neighbor checks for this test, and all other parameters were held constant.

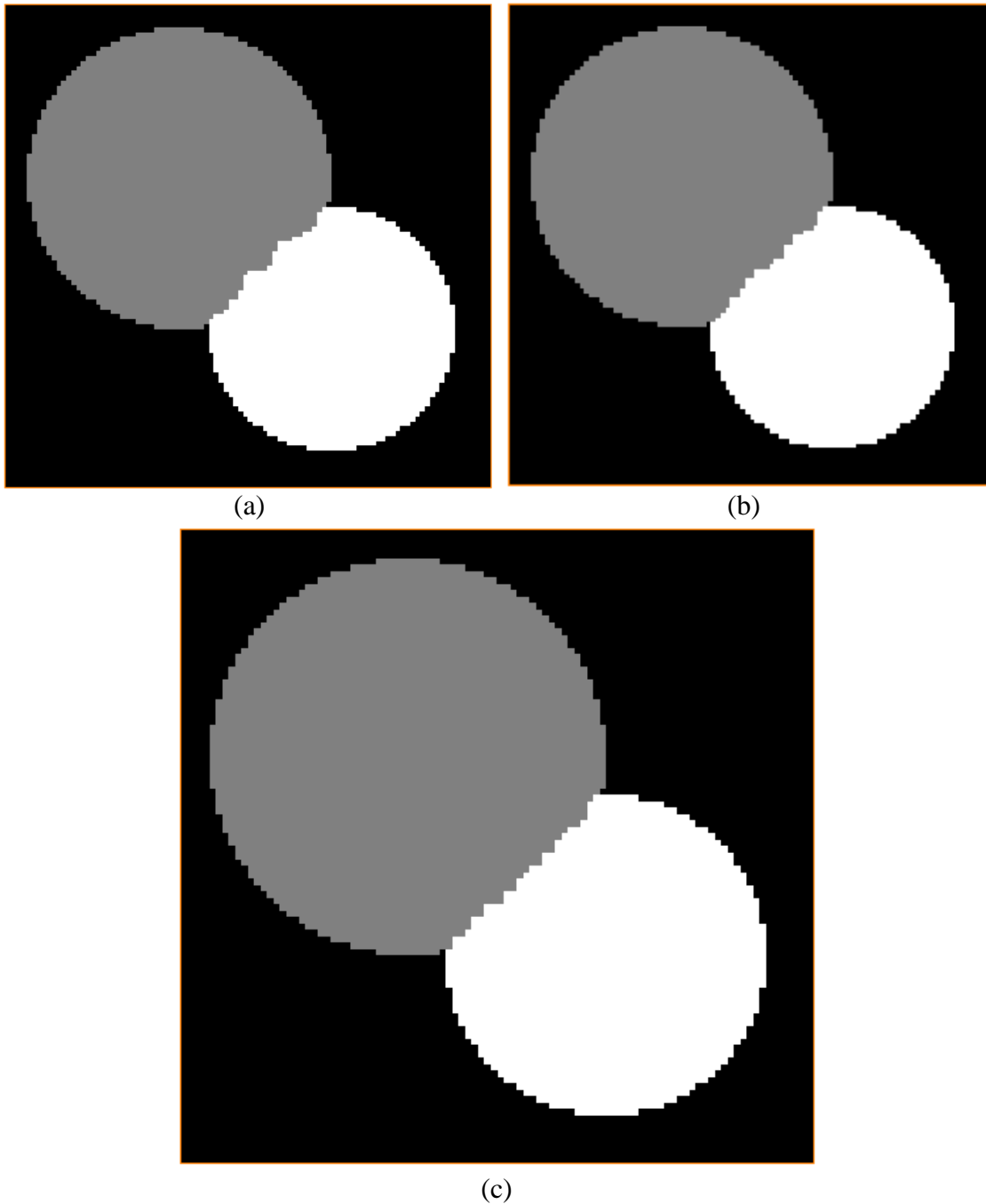


Figure 4.13. One (a), three (b) and five (c) grain assembly iterations for a two sphere pack, 100 voxels per side with a nominal voxel width of 0.01cm

Although these results show that increasing the number of grain assembly iterations creates a more realistic interface, one could argue that the additional computational expense exhausted through these iterations is not the additional increase in image quality. While this may be true for these particular images, it should be noted that spheres have a very simple geometry, and any minute signs of deviation from an ideal interface in this scenario could very quickly compound within the algorithm for more realistic grain shapes.

To better illustrate the relative impact of iterating within then granular assembly, the quartz-feldspar image was once again analyzed. Two versions of this image were processed. In the first image, Figure 4.14, there was only one iteration of grain assembly. The resultant partition is the same image that would result prior to adding the iterative grain assembly feature. For this reason, this image can be thought of as a control. In the second image, Figure 4.15, there were ten iterations of granular assembly. In both images, there were no grain overwrite iterations.



Figure 4.14. Partitioned quartz-feldspar image with 0 voxel overwrites and 1 grain assembly iteration, 820 $\mu$ m per side, 200<sup>3</sup> voxels

The image improvement from Figure 4.14 to Figure 4.15 appears fairly minute. Both zero and ten grain assembly iterations result in the same number of partitioned grains, and the grain-grain interfaces improve slightly in some scenarios, but appear to slightly worsen in others. The

conclusion from these images is that iterative grain assembly, by itself, is not highly advantageous. However, it must be stated again that for these images, there were zero voxel overwrite iterations based upon neighbor counting. If iterative grain assembly is used in conjunction with iterative voxel overwrites based upon neighbor counting, then the resulting image quality improves significantly, as is shown in the following section.



Figure 4.15. Partitioned quartz-feldspar image with 0 voxel overwrites and 10 grain assembly iterations, 820 $\mu$ m per side, 200<sup>3</sup> voxels

#### 4.1.6. Combined Impact of Voxel Overwrite and Grain Assembly Iterations

So far, results have been shown for images generated from grain assembly iterations, with no voxel overwrite iterations based upon neighbor counting, and from voxel overwrite iterations, with no grain assembly iterations. In this section, the interactive impact of these two parameters is investigated.

Figure 4.16 shows the resultant partition from one voxel overwrite iteration based upon neighbor counting and one grain assembly iterations. In this image, the program identifies a total of 1044 grains. The interfaces are not very refined, and there are some jagged grain boundaries that do not appear to be justified by the grain shapes. There is a voxel cluster that, on all sides, is surrounded by the same grain. Furthermore, this cross section shows that the grains are likely over partitioned.

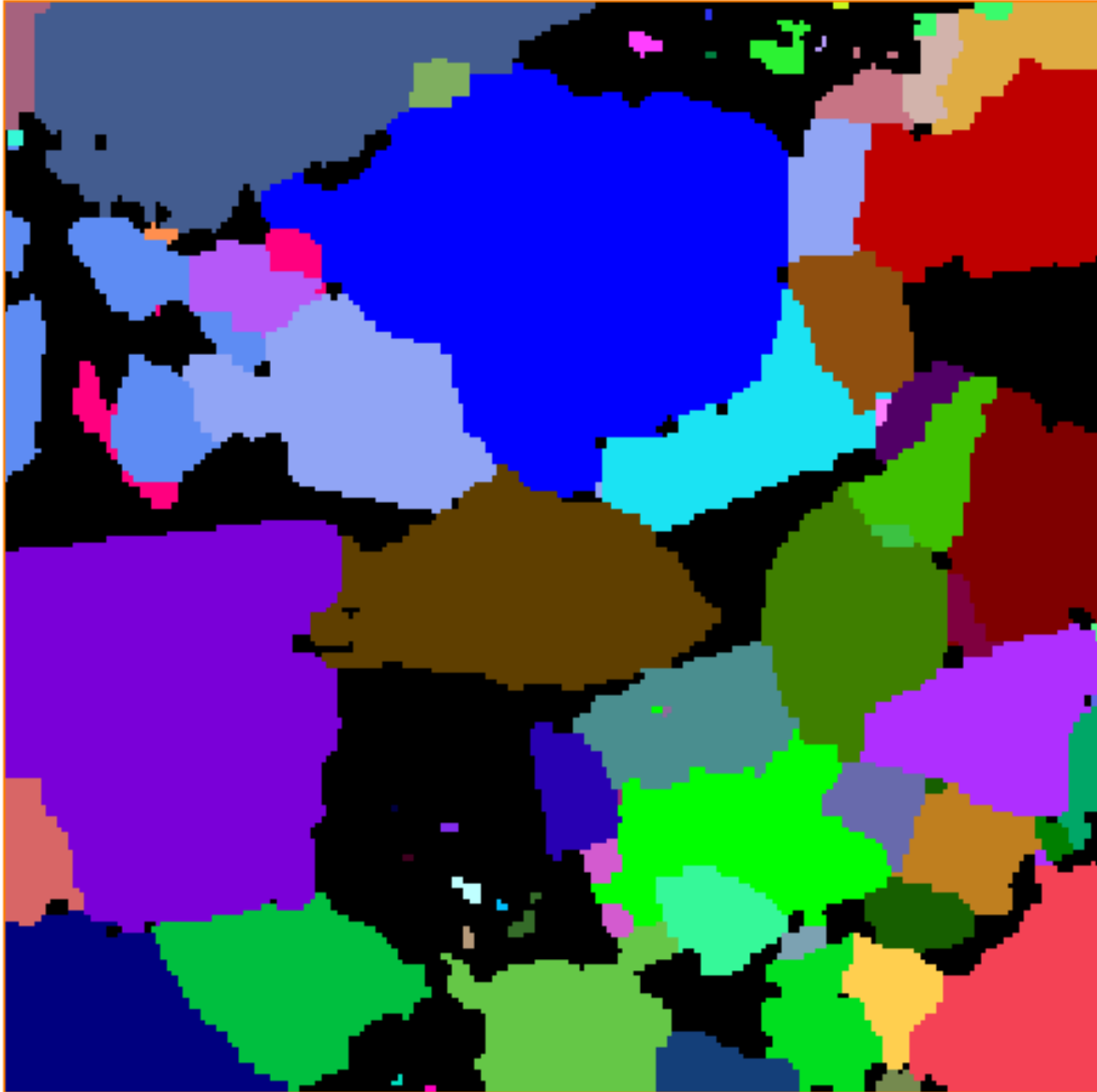


Figure 4.16. Partitioned quartz-feldspar image after 1 grain assembly iteration and, 1 voxel overwrite iteration based upon neighbor counting, 820 $\mu$ m per side, 200<sup>3</sup> voxels

Figure 4.17 shows the same cross section for the partition generated from three voxel overwrite and grain assembly iterations, and identifies a total of 1041 grains. The voxel cluster that was surrounded on all sides by the same grain has been removed, and there are significantly fewer unjustified jagged grain-grain interfaces. Although the grains still are still likely over partitioned, the total grain count has dropped by three grains.

Figure 4.18 results from ten voxel overwrite and grain assembly iterations, partitioning 1035 total grains. The increase in iterations decreases over partitioning by six total grains, and for the most part, further refines the interfaces between the grains. Many of the smaller grains that are likely a piece of a large grain shrink, when comparing their size changes from Figure 4.16 through Figure 4.18.

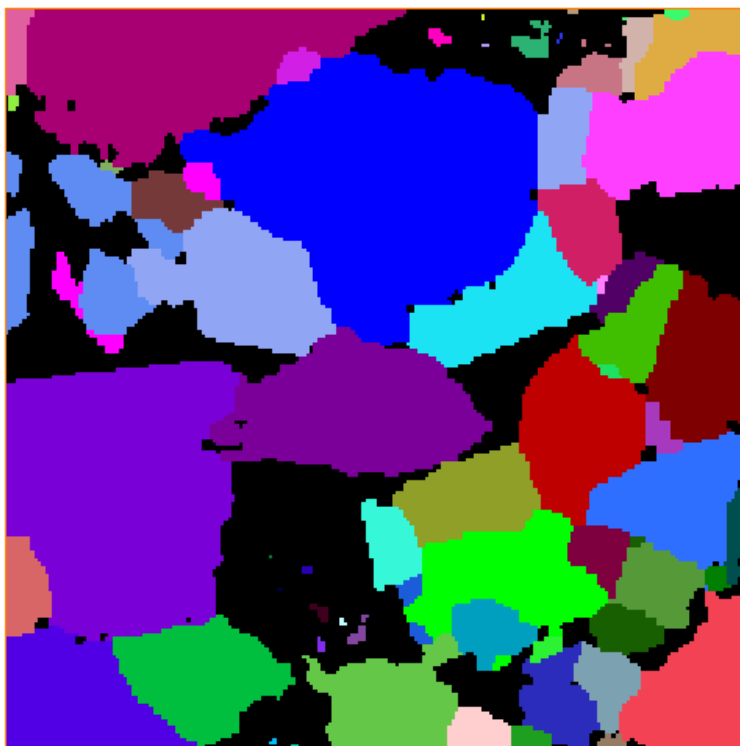


Figure 4.17. Partitioned quartz-feldspar image after 3 grain assembly iterations and 3 voxel overwrite iterations based upon neighbor counting, 820 $\mu$ m per side, 200<sup>3</sup> voxels

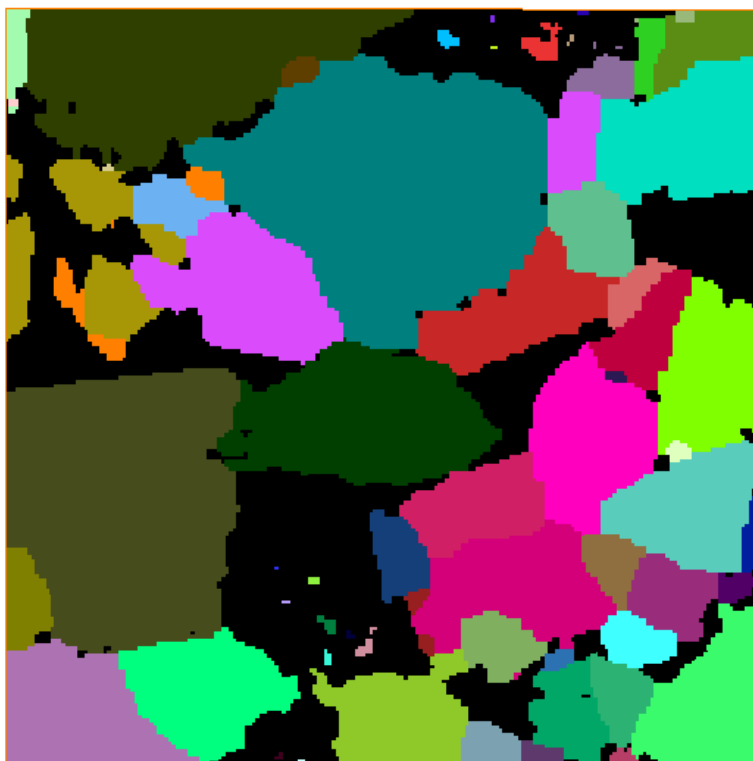


Figure 4.18. Partitioned quartz-feldspar image after 10 grain assembly iterations and 10 voxel overwrite iterations, 820 $\mu$ m per side, 200<sup>3</sup> voxels

Although the partition shown in Figure 4.18 is still not completely ideal, it must be stated that this is a fairly difficult image to process for several reasons. Primarily, there are very few fully represented (or interior) grains in this image. Whenever a grain touches the outer boundary, it is difficult for the program to properly partition, which makes sense because the grain is not presented in its entirety. Furthermore, this image has many small void spaces that, although possibly correct, could also be caused by an improper segmentation. Lastly, partitioning this image is fairly subjective, and what the author perceives as a correct partition may differ from what other individuals believe. Taking these factors into consideration, it is easier to objectively quantify the power of the program's additions and alterations by comparing Figures 4.16 through 4.18 to the original program's output. The images will be compared in detail in the following section.

A second, more objective image was also analyzed in a similar manner to observe the combined impacts of the iterative techniques. This Ottawa (F-75) sand has been analyzed with a previous version of VOX2GRAINS (Willson, 2012), finding that the grains, for the most part, are fairly similar in size to one another. As shown in the segmented image in Figure 4.19, the sample is contains many angular grains, which can be more difficult for the program to partition than simpler geometries like spheres. Due to the lack of consolidation, there is little room for argument about what does and does not constitute a whole grain, especially when viewing the sample in three dimension.

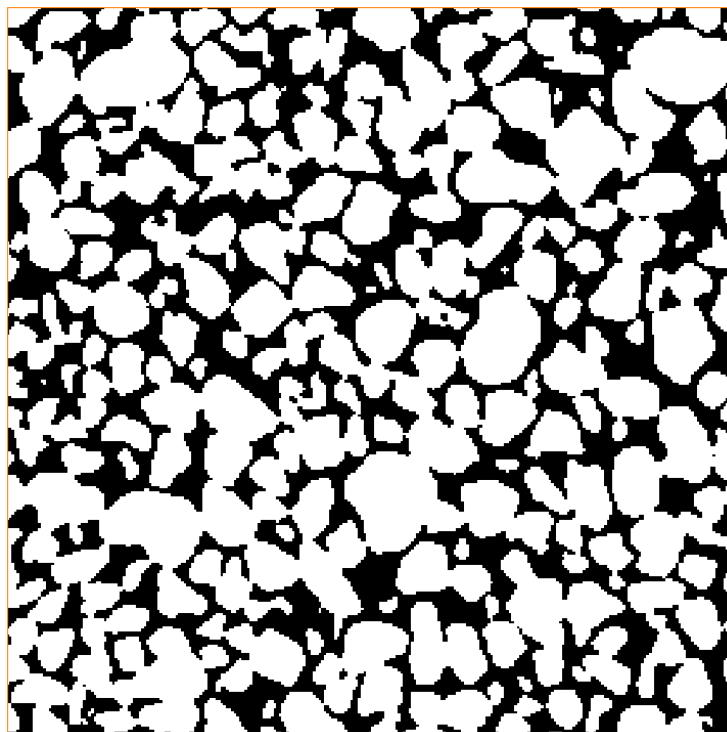


Figure 4.19. Ottawa sand segmented image, 3.025 mm per side,  $275^3$  voxels

Figure 4.20 shows a cross section through the Ottawa sand image generated by the updated VOX2GRAINS with one voxel overwrite iteration and one grain assembly iteration. The program finds many interfaces correctly, but there are a significant number of poor interfaces also, usually in cases where the grain appears to be over partitioned. This image resulted in 5040 total grains.

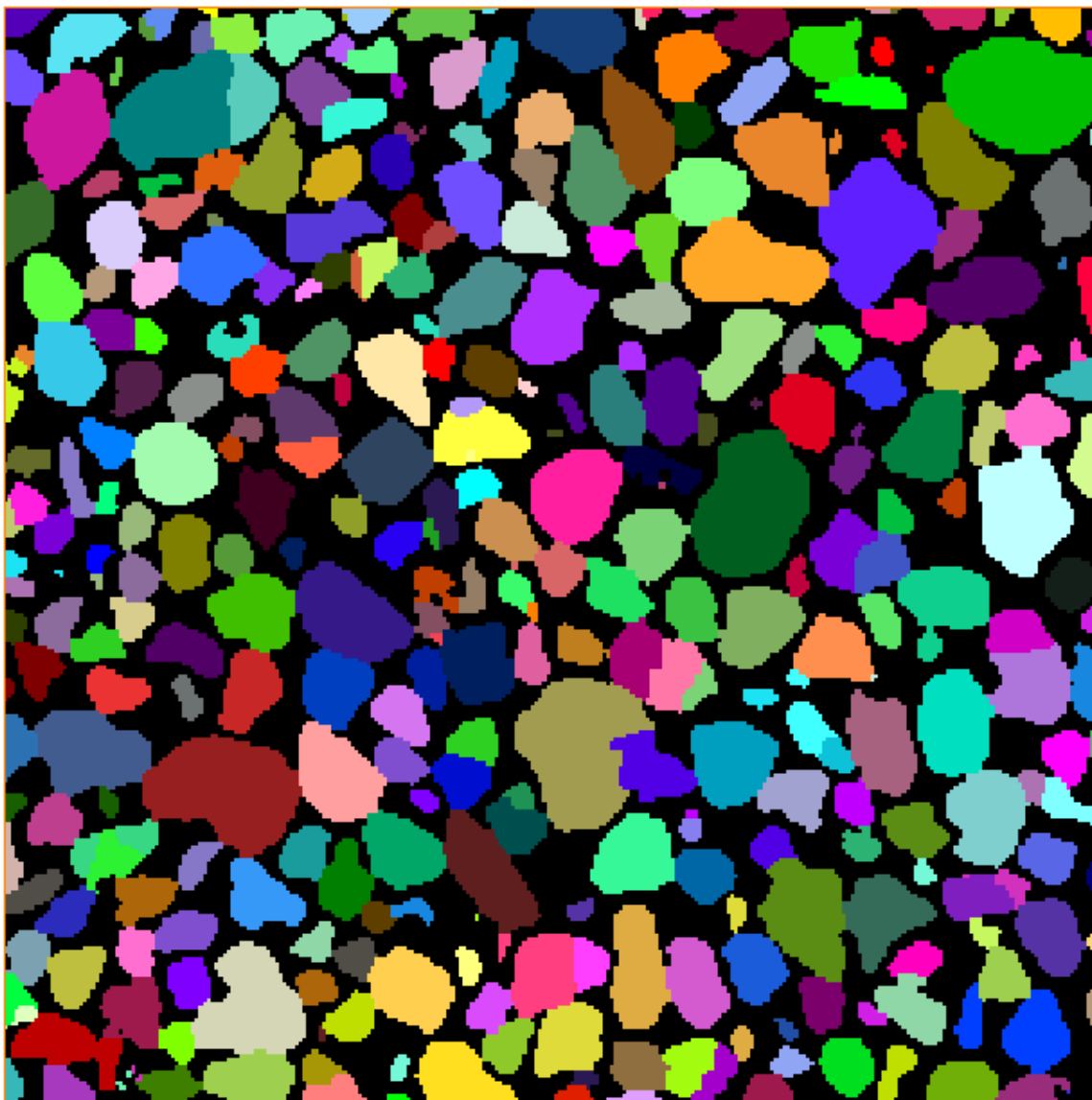


Figure 4.20. Partitioned Ottawa sand after 1 grain assembly iteration and 1 voxel overwrite iteration, 3.025 mm per side,  $275^3$  voxels

The original segmented image was then processed with three grain assembly iterations and three voxel overwrite iterations. Figure 4.21 provides a cross sectional slice through this image. In total, 5039 grains were identified. While over partitioning still exists in this image, the interfaces between the grains make more sense with the surrounding geometries. There are fewer slots and drawers, and the grain-grain contacts are more planar.

This trend continues when the image is process with ten voxel overwrite and grain assembly iterations, resulting in a total of 5033 identified grains. A cross section of this image is shown in Figure 4.22. The interfaces are close to ideal for almost all interior grains, with the exception being single grains that have been incorrectly partitioned into multiple grains.

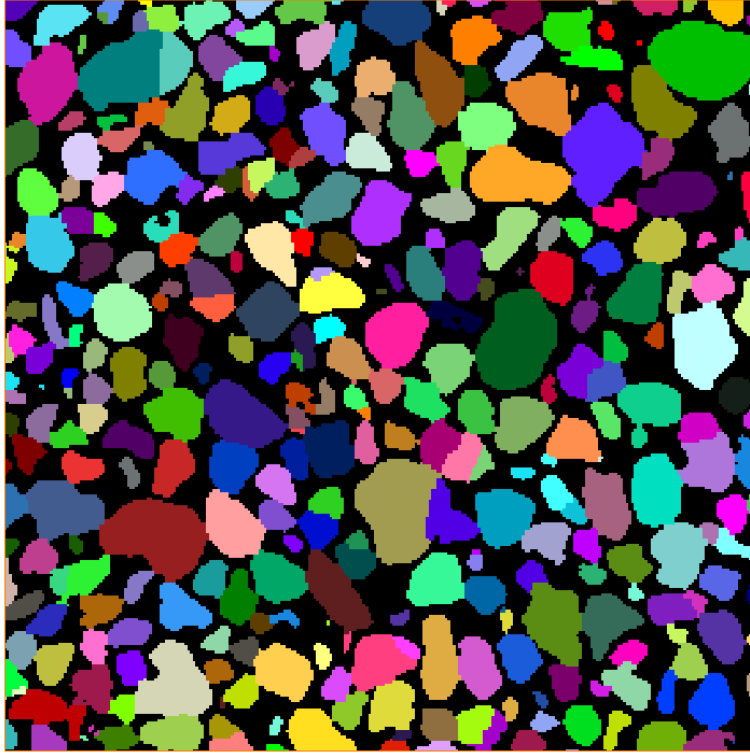


Figure 4.21. Partitioned Ottawa sand after 3 grain assembly iterations and 3 voxel overwrite iterations, 3.025 mm per side,  $275^3$  voxels

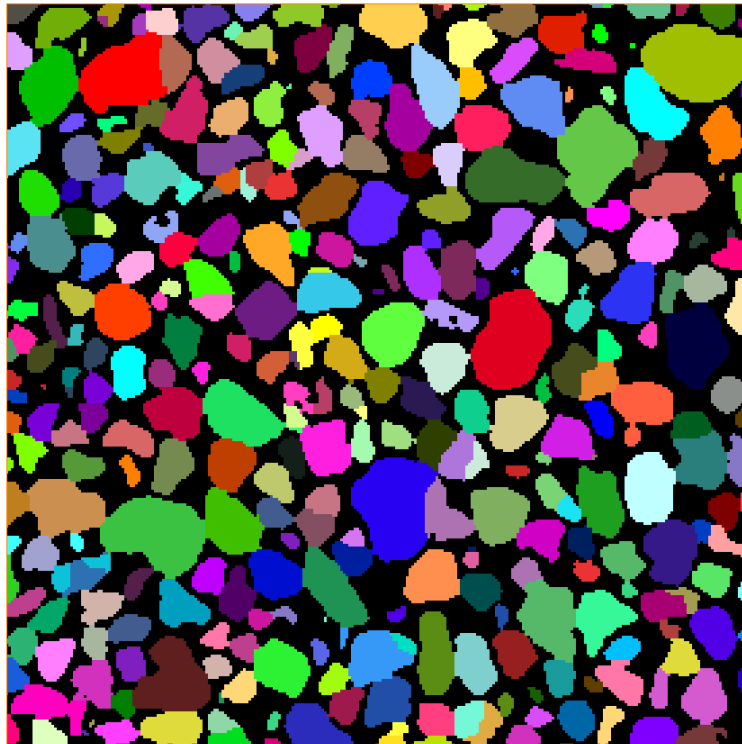


Figure 4.22. Partitioned Ottawa sand after 10 grain assembly iterations and 10 voxel overwrite iterations, 3.025 mm per side,  $275^3$  voxels

#### 4.1.7. VOX2GRAINS Before and After Comparison

The partition presented in Figures 4.18 and 4.22 includes all main body adjustments and additions implemented into VOX2GRAINS. Provided the segmented images, these are among the best partitions that the program is capable of generating without any refinement. In order to see the full impact that the additions and alterations have on generating the initial partition, Figures 4.18 and 4.22 are compared to the program's initial outputs for the same segmented images. Figure 4.23 is compared with Figure 4.18 and Figure 4.22 is compared with Figure 4.24.



Figure 4.23. Partitioned quartz-feldspar using original program, 820 $\mu$ m per side, 200<sup>3</sup> voxels

Comparing these images, we see that Figure 4.18 provides a much more believable partition. The slots and drawers have been almost completely removed, and the interfaces are far more consistent with the information available. The original program identified a total of 1467 grains, whereas the updated software identifies 1037, meaning that the algorithm reduces over partitioning significantly. In a fairly subjective image like this, defining the true number of grains can be very

subjective, so it is difficult to exactly quantify the degree of over partitioning. However, it can be stated objectively that the partition shown in Figure 4.18 is much more realistic than the partition shown in Figure 4.23.

It is significantly less subjective to identify the grains in Figures 4.22 and 4.24. Looking at Figure 4.24, we see that the program identified a number of interfaces fairly accurately initially, but there are many instances of unjustified slots and drawers, and many grains are over partitioned. A total of 5,847 grains were identified from the initial program, whereas 5,033 were identified with the updated program. Comparing these images, we see that the updated reduced over partitioning in many instances, and provided more realistic contacts for more grain-grain interfaces. Figure 4.22 still contains some over partitioned grains, but the improvement in overall image quality from the program's initial state is significant. The remaining, but reduced issue of over partitioning is addressed through post processing refinements.

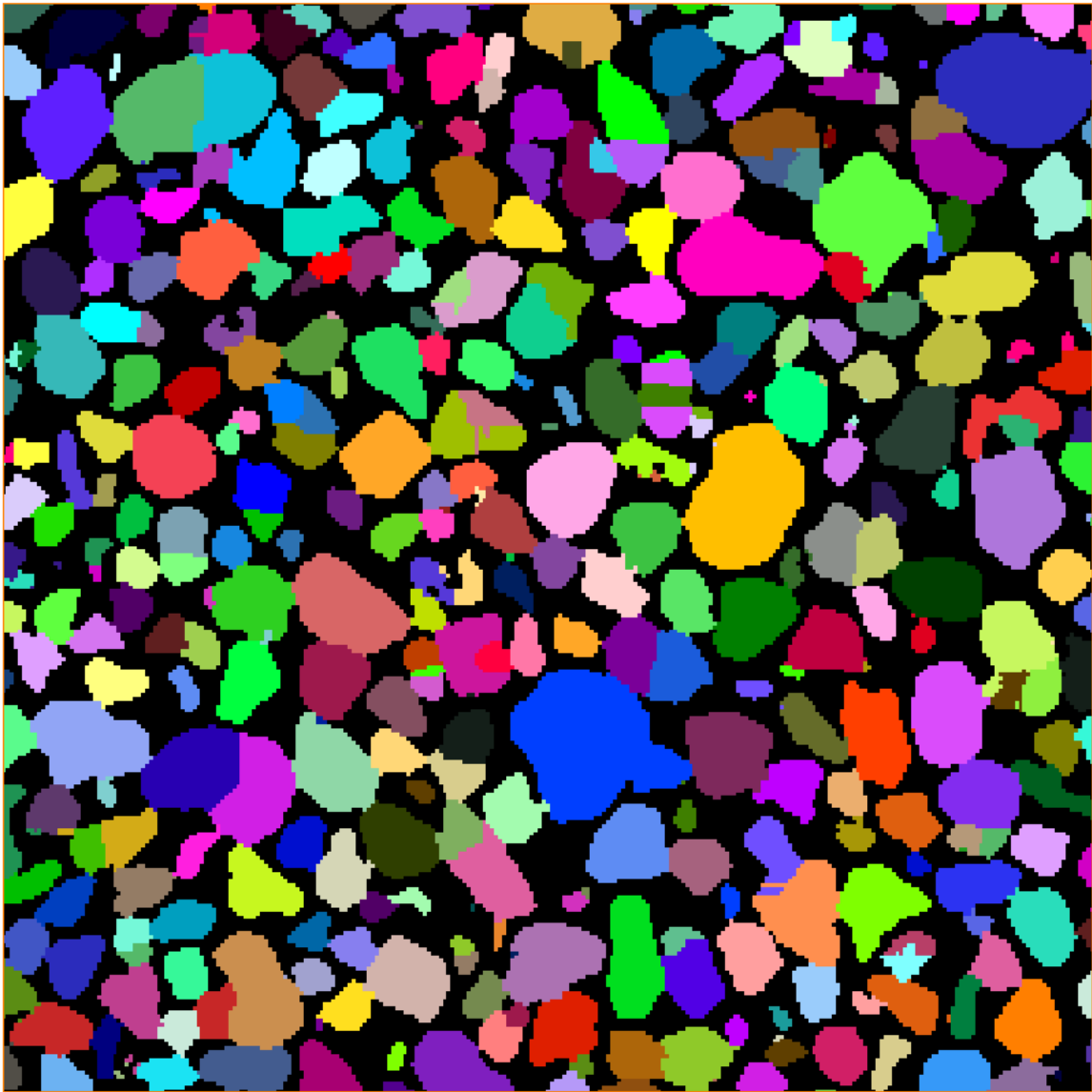


Figure 4.24. Partitioned Ottawa sand from original program, 3.025 mm per side,  $275^3$  voxels

## 4.2. Post Processing Refinements

Overall, the post-processing refinements added provided mixed results for the images tested. Automerge features were not found to be particularly effective, but machine learning and planar regression provide significant advantages for combating over partitioning and smoothing grain-grain interfaces.

### 4.2.1. Automerge

Altering the logic of automerge was not very effective in merging the over partitioned grains. Figures 4.25 and 4.26 show the resulting refinement when threshold values of 0.95 and 0.85 are applied, respectively. More aggressive values were also tested, and although the intermediate grains were merged under these more aggressive conditions, the vast majority of initially identified grains in the image were merged into one, and thus the partition was nonsensical. Overall, the results from this alteration were extremely poor.



Figure 4.25. Partitioned overlapping sphere pack with updated automerge logic and a contact area to surface area ratio of 0.95, 500 voxels per side with a nominal voxel width of 0.01 cm



Figure 4.26. Partitioned overlapping sphere pack with updated automerge logic and a contact area to surface area ratio of 0.85, 500 voxels per side with a nominal voxel width of 0.01 cm

While automatically overwriting the grains that have no exposed surface area is an added option, it still leaves many over partitioned grains in the resulting image, since any grain that has any exposed surface area is not considered for merging. A highly consolidated artificial sphere pack was tested. The goal was to provide something that is difficult for the program, so that there will be multiple instances of over partitioned grains. Figure 4.27 shows the VOX2GRAINS output with one grain assembly iteration and three grain overwrite iterations. This image (500 voxels along each side) contains 1985 total grains. Figure 4.28 shows the same image after automatically reassigning all grains with no exposed surface area. In this image, this process reduces the grain count to 1810. Many of the images that are fixed are very small, since the larger grains are more likely to touch the void space.

This feature provides a quick way of reassigning grains that were obviously over partitioned, though Figure 4.28 shows that several instances of over partitioned grains remain. This was not designed to identify and remediate all over partitioned grains, but to provide a robust way of quickly performing a pass to ensure that any obviously over partitioned grains are reassigned. VOX2GRAINS identifies more grains as consolidation and angularity of grain shape increase, so the number of merges this feature performs can vary widely between images of different rock types.

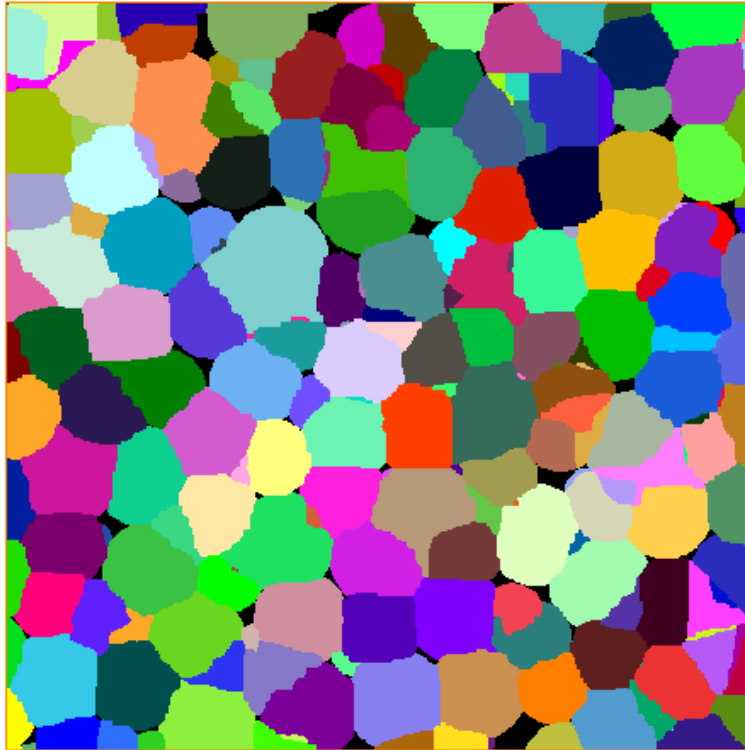


Figure 4.27. Partitioned artificial sphere pack with 1 grain assembly iteration and 3 voxel overwrite iterations, 500 voxels per side with a nominal voxel width of 0.01 cm

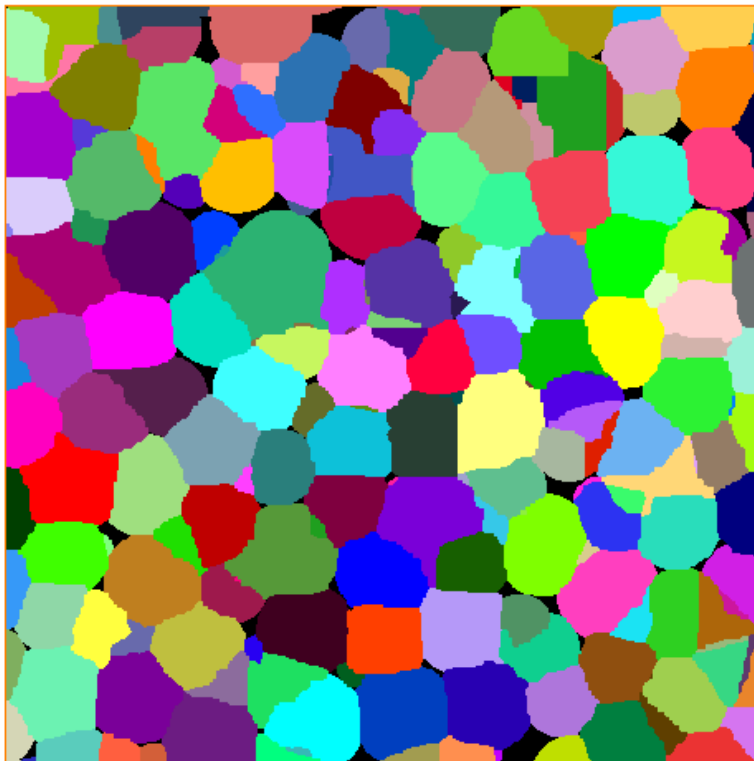


Figure 4.28. Partitioned artificial sphere pack after automatically overwriting all grains with no exposed surface area, 500 voxels per side with a nominal voxel width of 0.01 cm

#### 4.2.2. Planar Regression: Single Plane

This version of planar regression was first implemented with simple three sphere packs, and was found to be highly effective in these scenarios. Recall that in this version of planar regression, the program treated each interface between grain pairs as a single plane, regardless of whether or not there were multiple isolated contact clusters. Figure 4.29 (a) shows a three dimensional representation of the image presented in Figures 4.29 (b) and (c). Figure 4.29 (b) shows a slice through the image after planar regression without minimization has been applied and Figure 4.29 (c) shows the same slice after both planar regression and planar minimization. Though there is little difference between the two images, the minute adjustments made by the program from (b) to (c) provide a proof of concept for this method.

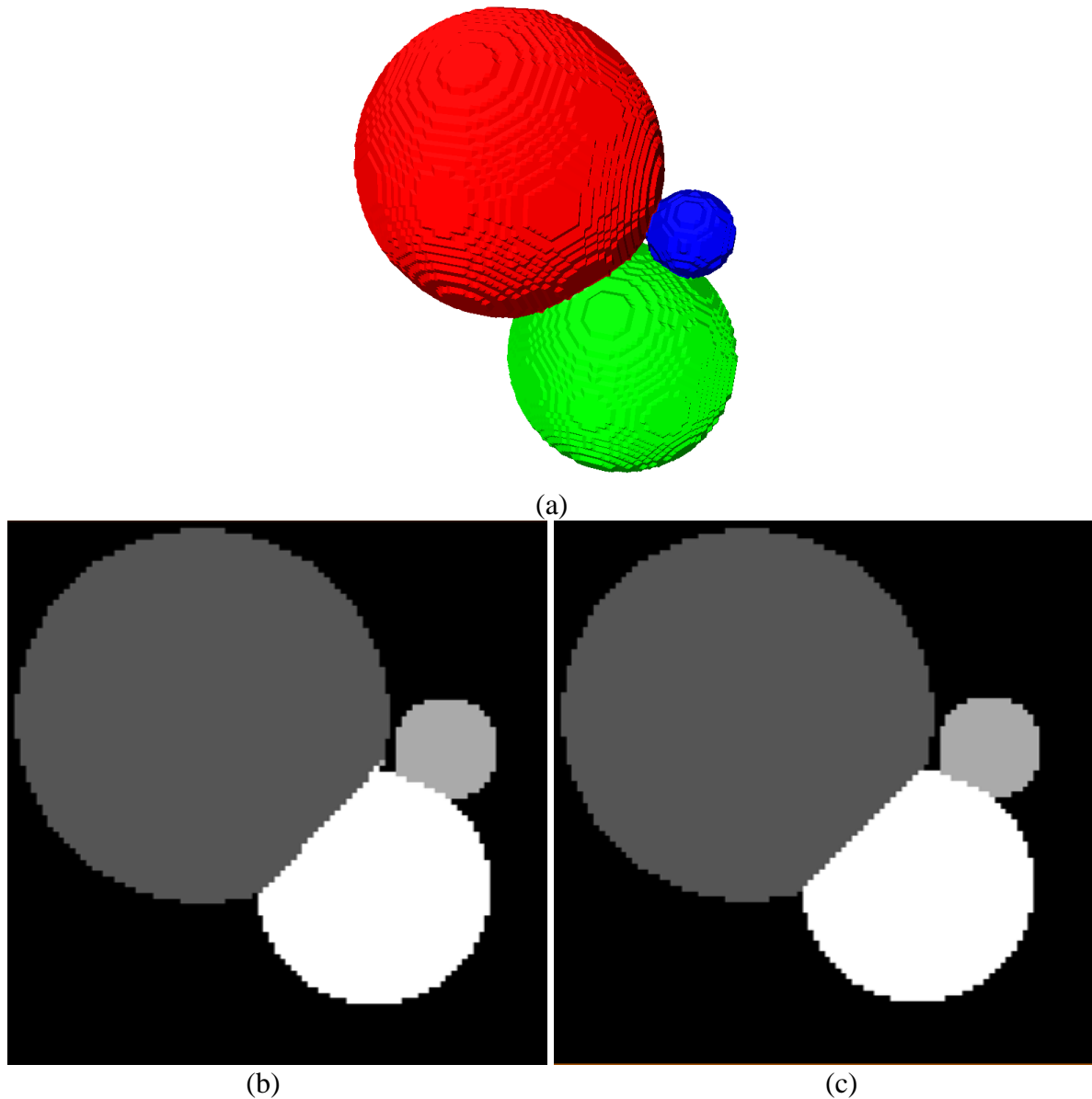


Figure 4.29. Simple three sphere pack tested for single contact plane planar regression, 100 voxels per side with a nominal voxel width of 0.01 cm

When tested on more populated sphere pack images with low degrees of consolidation, the program performed significantly worse. The same was true for more realistic rock images, Figures 4.30 (a) and (b) show examples of common errors the program would make when applied to sphere packs with varying contact orientations.

In Figure 4.30 (a), the algorithm assigned every voxel on one side of the calculated contact plane to the yellow grain, and every voxel on the other side of the plane to the red grain. Although this worked well for the grain pack shown in Figure 4.29, the algorithm fails in the scenario. This issue is compounded further as planar regression continues, as shown in Figure 4.30 (b). Although the red grain in Figure 4.30 (a) and (b) likely should not be a single grain, this image provides an understanding of the limitations of this methodology. These errors were very common with more complex orientations and geometries. Also, there were many instances with real rock images where two grains neighbor one another in two isolated clusters, often with two very different contact orientations. In these more realistic scenarios, the single plane planar regression algorithm for each grain pair is not sufficient.

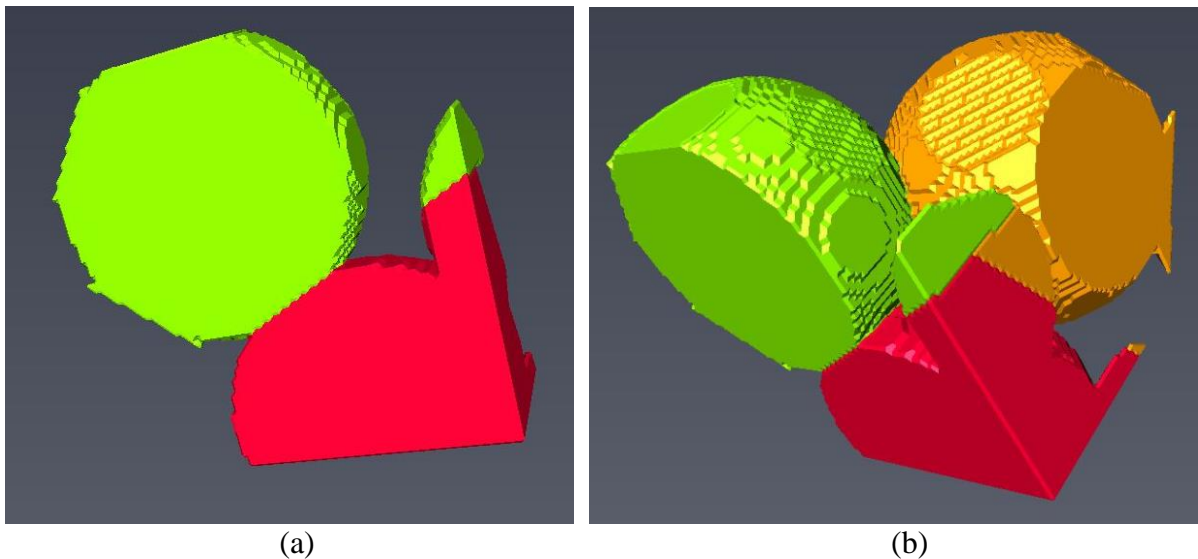
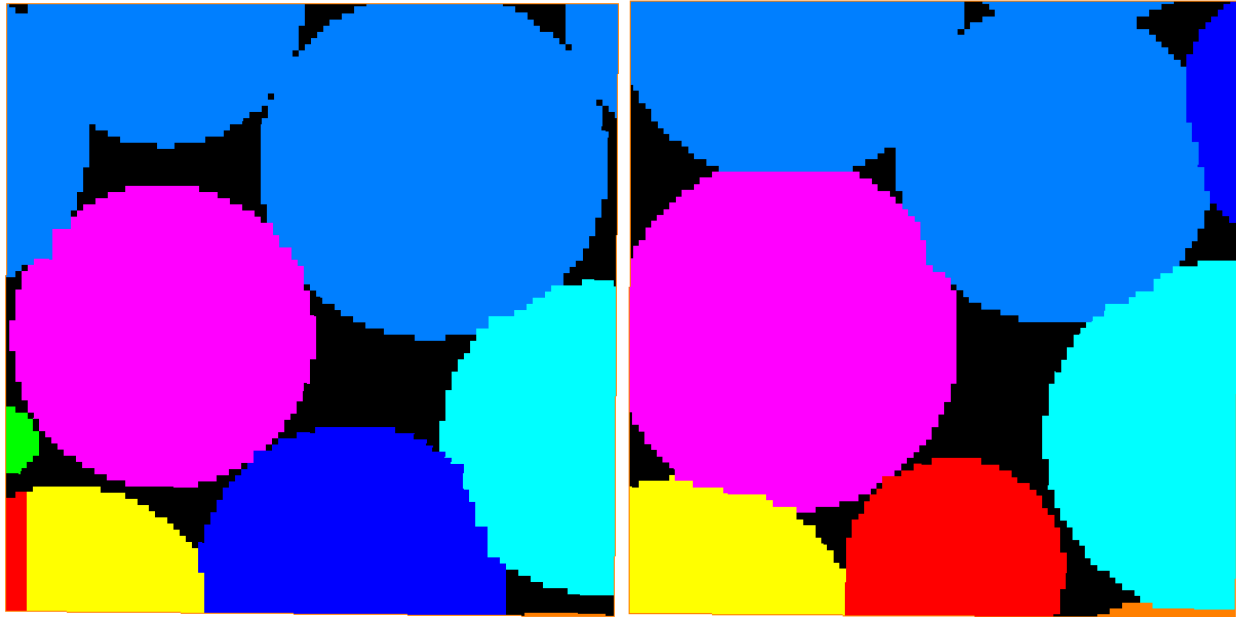


Figure 4.30. Sphere pack after single-plane planar regression was applied

#### 4.2.3. Planar Regression: Multiplane

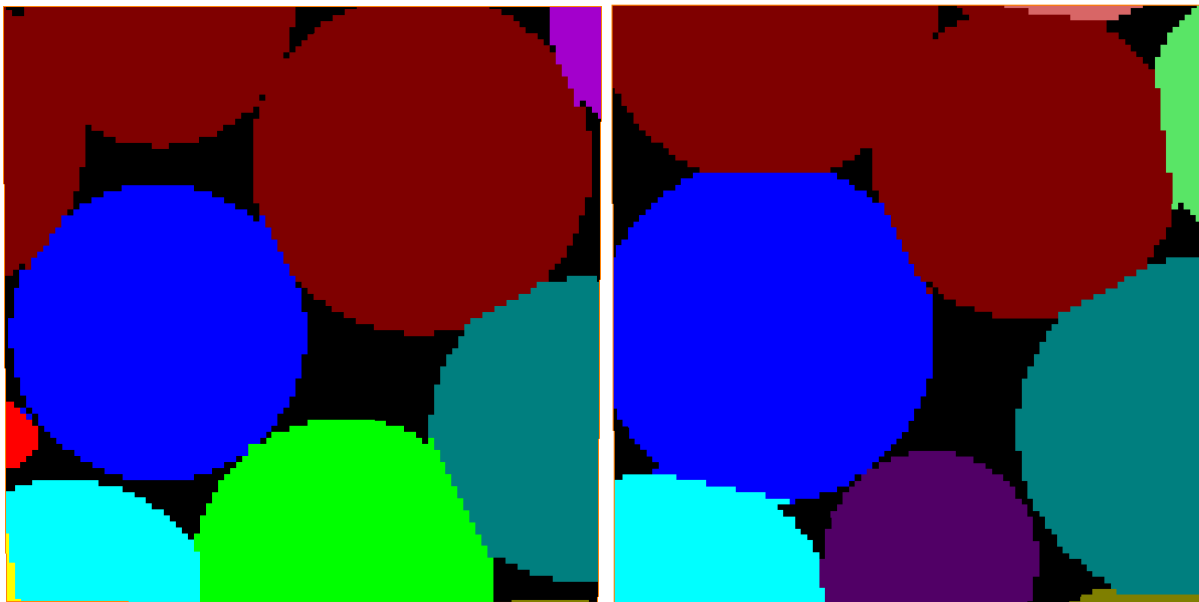
Similarly to single plane, multiplane planar regression was first tested on spheres. The images used for testing were manipulated to ensure multiple isolated interfaces between a single grain pair. Figure 4.31 (a) and (b) illustrate separate cross sections through one such example. Note that in these figures, planar regression has not yet been applied; these show the VOX2GRAINS output with no prior refinement. For the purpose of this study, spheres that, in reality, would constitute separate grains, were merged together, as shown by the three sphere pieces on the top of the image. Figure 4.32 (a) and (b) show the same cross sections after multiplane planar regression has been applied. Figure 4.33 displays the three dimensional image after planar regression is applied to each isolated contact cluster. The other spheres in this image were visually removed for illustrative purposes.



(a)

(b)

Figure 4.31. Sphere pack prior to multiplane planar regression with no minimization, 100 voxels per side with a nominal voxel width of 0.01 cm



(a)

(b)

Figure 4.32. Sphere pack after multiplane planar regression with no minimization, 100 voxels per side with a nominal voxel width of 0.01 cm

Here, planar regression was designed as a final step in refining the grain interfaces. For this reason, the images to best test planar regressions are images that have already been sufficiently cleaned of over partitioned grains. To illustrate this point, refer to Figure 4.34. Although the over partitioning has been significantly reduced when compared against the original program's output, there are still

visible occurrences. Note that after proving this methodology through sphere testing, the algorithm was adjusted to only apply planar regression interfaces to interior grain pairs, with neither grain touching the boundary.

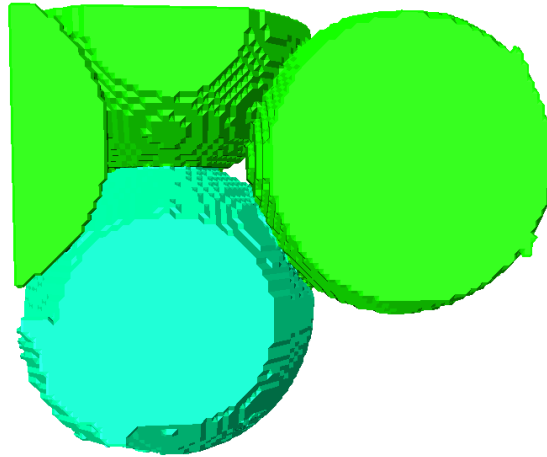


Figure 4.33. Grains with multiple isolated contact points after multiplane planar regression with no minimization is applied

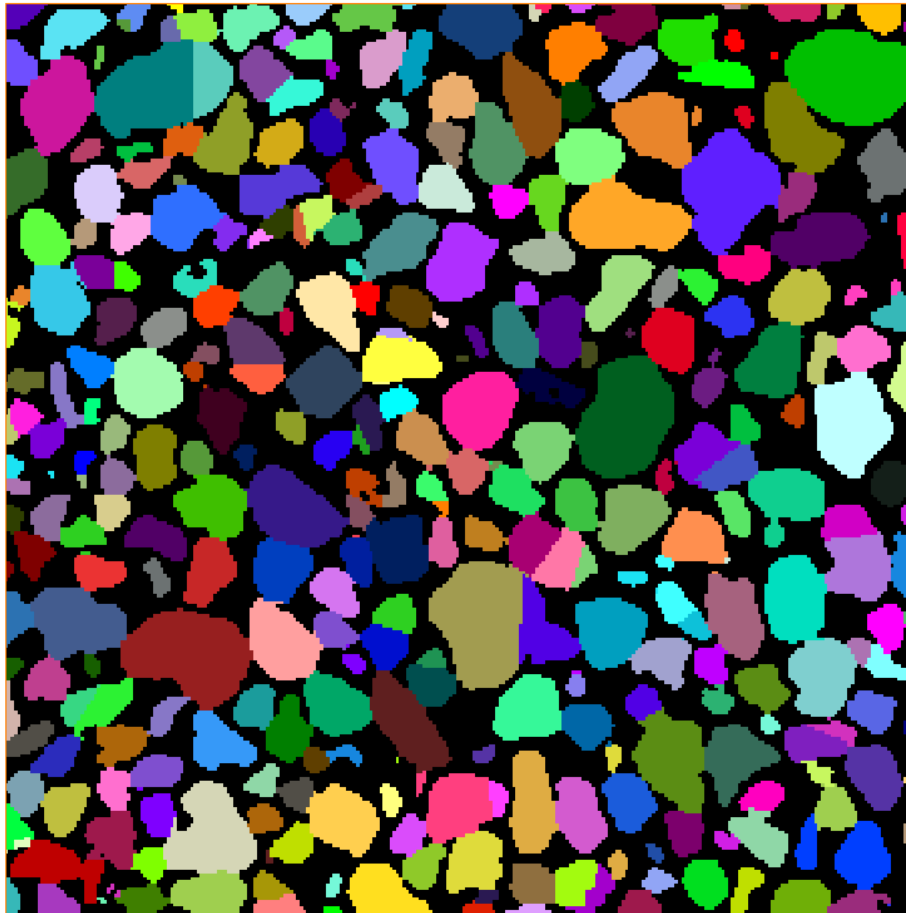


Figure 4.34. Partitioned Ottawa sand with 3 grain assembly iterations and three voxel overwrite iterations after planar regression with no minimization, 3.025 mm per side,  $275^3$  voxels

Although this figure shows that the program is working correctly, the results for the originally over partitioned grains are not ideal. If the image is first cleaned for all over partitioned grains, as shown in Figure 4.35, then the resulting planar partition is far more desirable, shown in Figure 4.36. Figure 4.36 was generated without minimizing the interface plane.

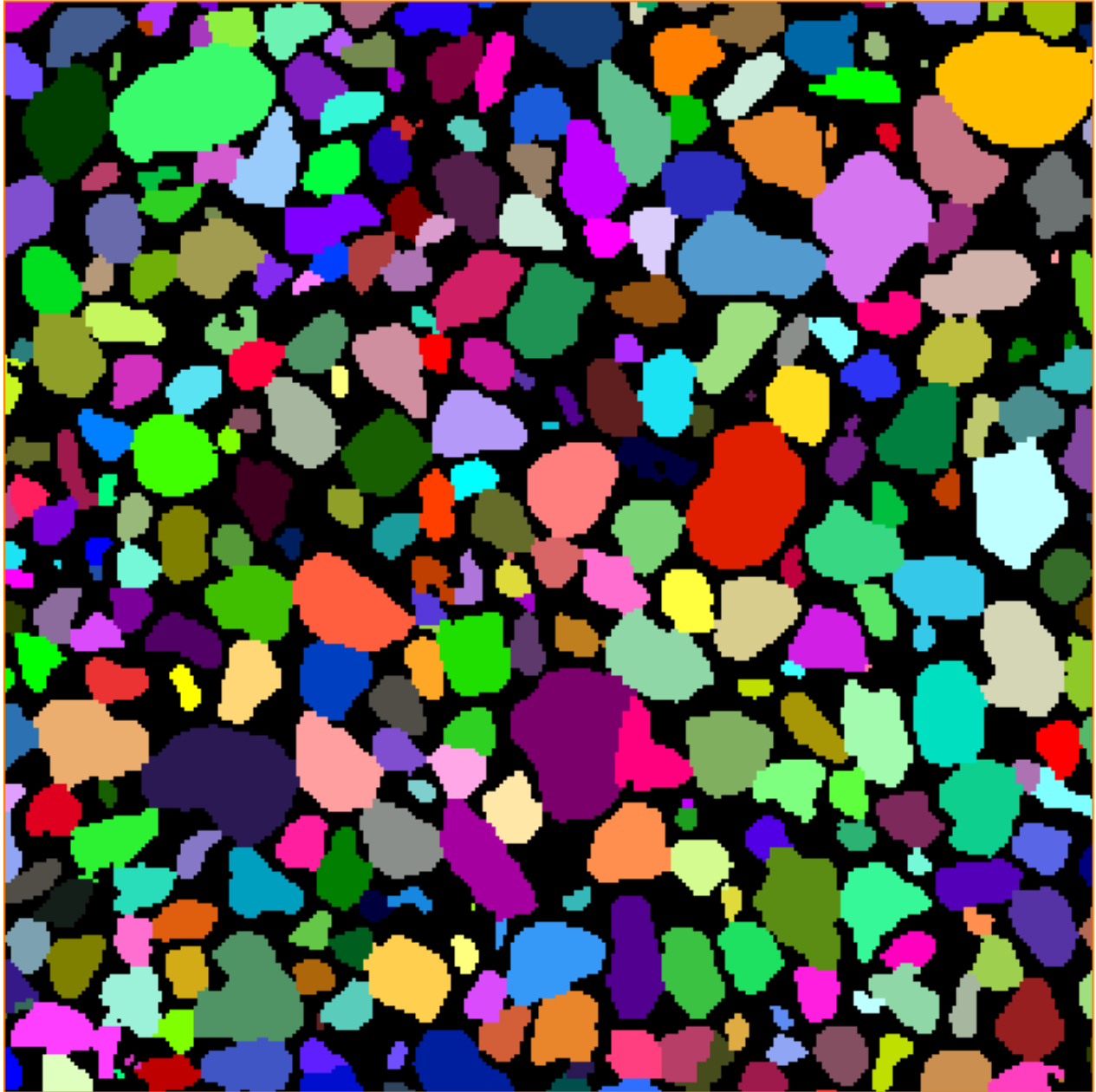


Figure 4.35. Partitioned Ottawa sand with 3 grain assembly iterations and three voxel overwrite iterations prior to planar regression; after manually refining over partitioned grains, 3.025 mm per side,  $275^3$  voxels

Within the program, the user has the ability to easily turn minimization on or off when applying planar regression to the grain contacts. Figure 4.37 shows the program output whenever slide

minimization is included. Minimization involves extending the plane outward, which causes the streaks that are visible in Figure 4.37. Although this is only occurs on a fraction of the total grain interfaces, it is extremely undesirable. These results show that the planar extension method needs to be further refined. The issue arises in finding the balance of extending the plane while ensuring that the extension does not extrude into other sections of the grain.

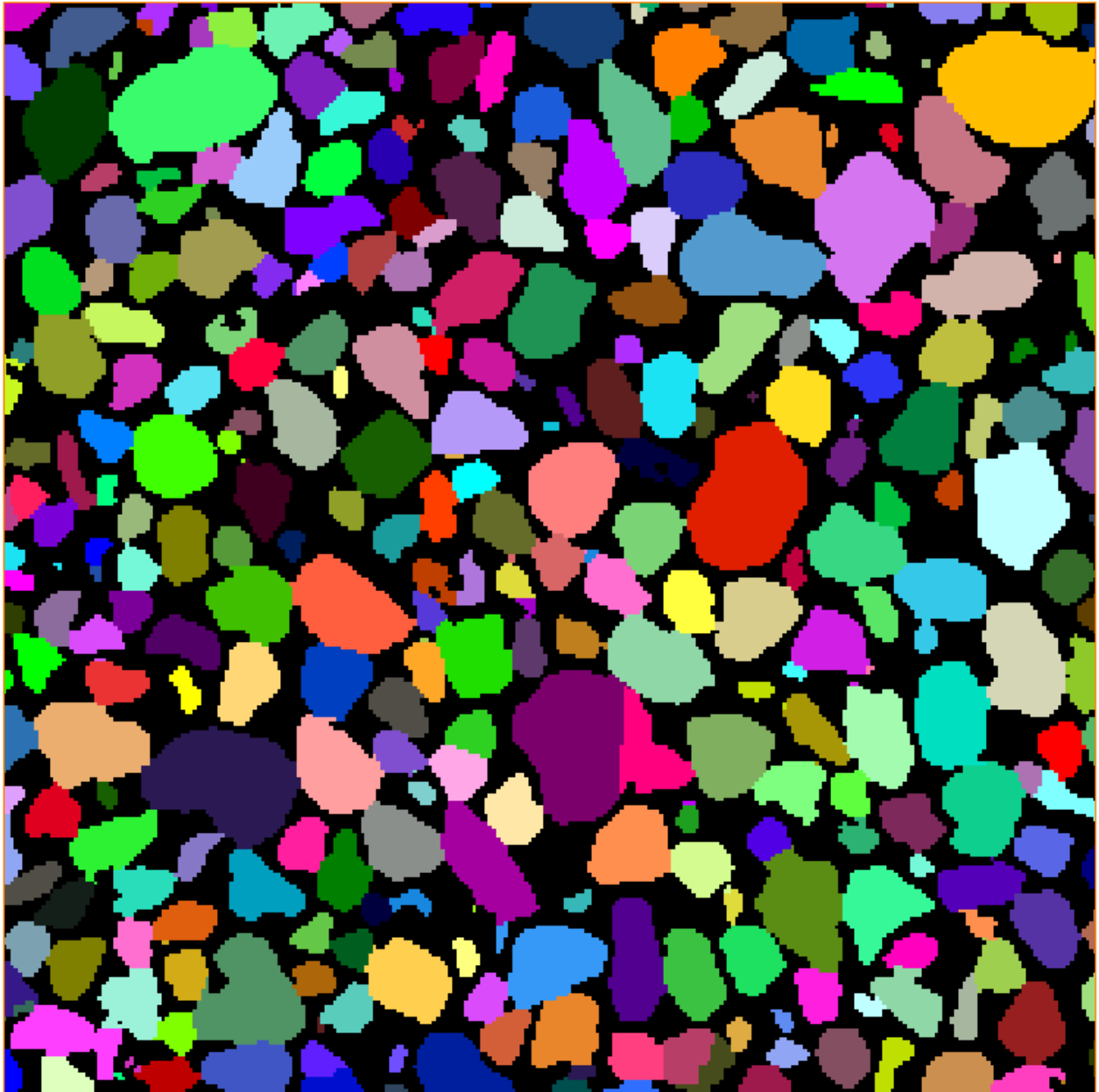


Figure 4.36. Partitioned and manually refined Ottawa sand after planar regression applied with no minimization, 3.025 mm per side,  $275^3$  voxels

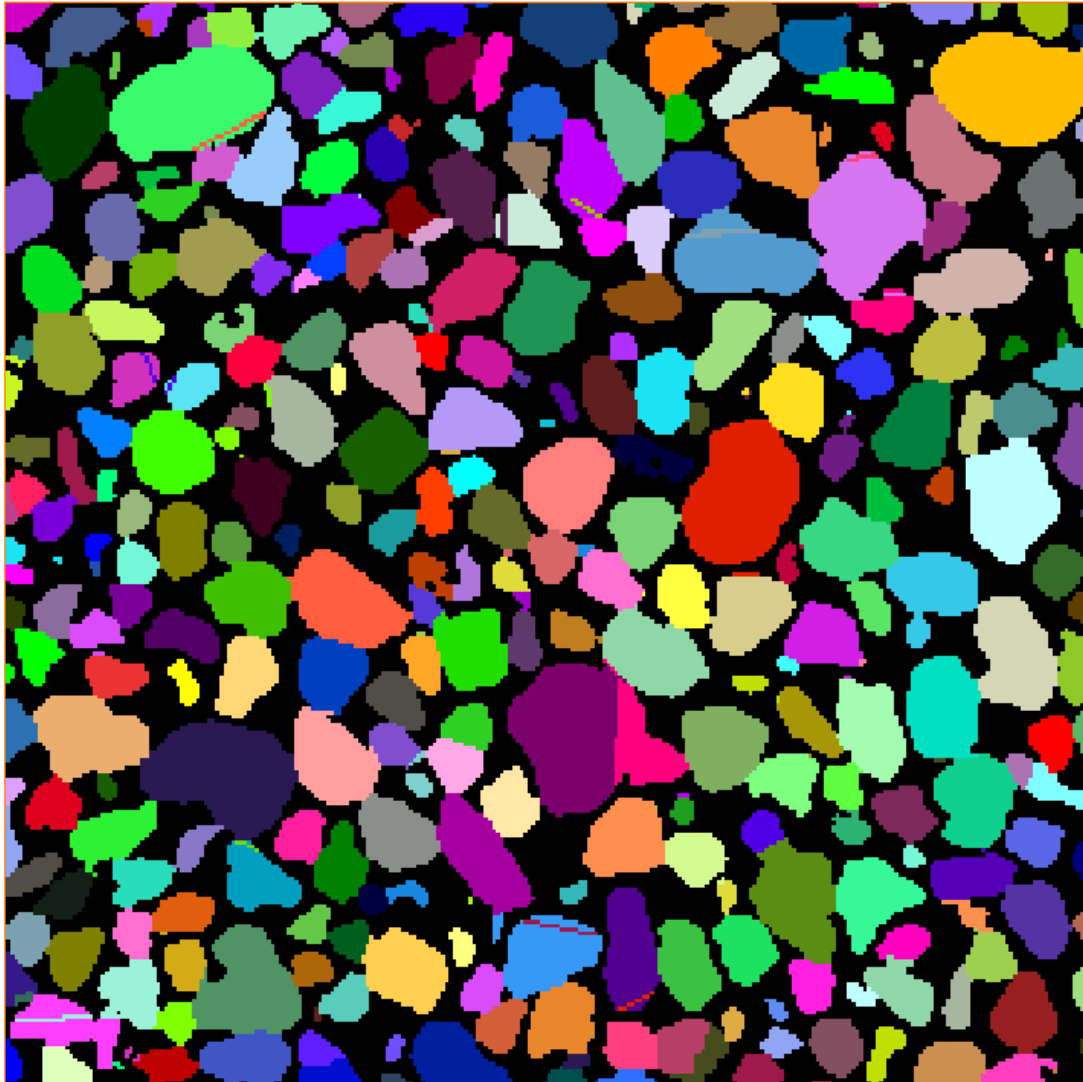


Figure 4.37. Partitioned and manually refined Ottawa sand after planar regression applied with minimization, 3.025 mm per side,  $275^3$  voxels

#### 4.2.4. Machine Learning

For the images tested, machine learning proved to be a viable method of efficiently and accurately merging grains that were initially over partitioned. The accuracies of the generated logistic regression equations are shown below in Table 4.1 for each corresponding Trial. Since the number of merge decisions in the training and cross validation sets was only about 20%, predictive accuracies are not the only output that should be examined. Consider an exaggerated example where only five percent of the data points should be merged. If the algorithm determines that all pairs should not be merged, then the algorithm is still operating at 95% accuracy. For this reason, it is also important to consider the number of false positives and false negatives. False positives refer to examples where the algorithm predicted that the pair should be merged, but the user specified that the pair should not be merged. Conversely, false negatives refer to examples in which the user specified that the example should be merged, but the algorithm determined that the pair should not be merged.

Table 4.1. Machine learning results for each trial

<b>Trial No.</b>	<b>Predictive Accuracy for CV Set</b>	<b>Incorrect Decisions</b>	<b>False Positives</b>	<b>False Negatives</b>
<b>Trial 1</b>	97.88%	15	11	4
<b>Trial 2</b>	98.16%	13	10	3
<b>Trial 3</b>	97.74%	16	6	10

Trial 2 generated the most accurate prediction equation. The values for this equation are shown in Table 4.2. Since this trial did not include feature scaling, the value of the theta parameters cannot be considered as weights or importance for the equation.

Table 4.2. Calculated theta values for Trial 2

	<b>Corresponding Statistic</b>	<b>Value</b>
	<b>Intercept</b>	1.5133
<b>theta 2</b>	<b>Surface Area G1</b>	-0.0344
<b>theta 3</b>	<b>Surface Area G2</b>	-0.0306
<b>theta 4</b>	<b>Volume G1</b>	-0.0218
<b>theta 5</b>	<b>Volume G2</b>	-0.0097
<b>theta 6</b>	<b>Inscribed Radius G1</b>	0.1956
<b>theta 7</b>	<b>Inscribed Radius G2</b>	-0.1024
<b>theta 8</b>	<b>Coord. Number G1</b>	0.1471
<b>theta 9</b>	<b>Coord. Number G2</b>	-0.2525
<b>theta 10</b>	<b>X Pos. G1</b>	-0.0512
<b>theta 11</b>	<b>Y Pos. G1</b>	0.2257
<b>theta 12</b>	<b>Z Pos. G1</b>	-0.2189
<b>theta 13</b>	<b>X Pos. G2</b>	0.0133
<b>theta 14</b>	<b>Y Pos. G2</b>	-0.2505
<b>theta 15</b>	<b>Z Pos. G2</b>	-0.0971
<b>theta 16</b>	<b>Contact Area</b>	0.4513
<b>theta 17</b>	<b>X Major Axis G1</b>	-0.2741
<b>theta 18</b>	<b>X Major Axis G2</b>	0.5148
<b>theta 19</b>	<b>Y Major Axis G1</b>	0.0987
<b>theta 20</b>	<b>Y Major Axis G2</b>	0.2609
<b>theta 21</b>	<b>Z Major Axis G1</b>	-0.1548
<b>theta 22</b>	<b>Z Major Axis G2</b>	-0.3228
<b>theta 23</b>	<b>Aspect Ratio G1</b>	-0.0374
<b>theta 24</b>	<b>Aspect Ratio G2</b>	0.6617
<b>theta 25</b>	<b>Contact Loc. X</b>	0.0144
<b>theta 26</b>	<b>Contact Loc Y</b>	-0.0029
<b>theta 27</b>	<b>Contact Loc Z</b>	0.1885
<b>theta 28</b>	<b>Contact Norm Vect X</b>	-0.1056
<b>theta 29</b>	<b>Contact Norm Vect Y</b>	0.8446
<b>theta 30</b>	<b>Contact Norm Vect Z</b>	-0.0522

The logistic regression equation was subsequently applied to all interior grains for the five images that data was taken from. Since the exterior grains are not fully presented, this logistic regression equation was not believed to be valid for these grains. For the first image, referred to here as Ottawa sand 1, VOX2GRAINS identified 5039 grains before any refinement. After the logistic regression equation was applied, a total of 3723 grains were identified. Figure 4.38 provides a cross section through the partitioned image before any refinement, and Figure 4.39 shows the same cross section after the image was refined using the logistic regression equation developed from Trial 2.

Comparing these two images, we see that the program correctly merged several pairs of grains that appear to be over partitioned, but it also appears to falsely merge some pairs that should be separate. Cross sections showing before and after the application of the Trial 2 logistic regression equation for the other four images used in data collection are shown in the Appendix.

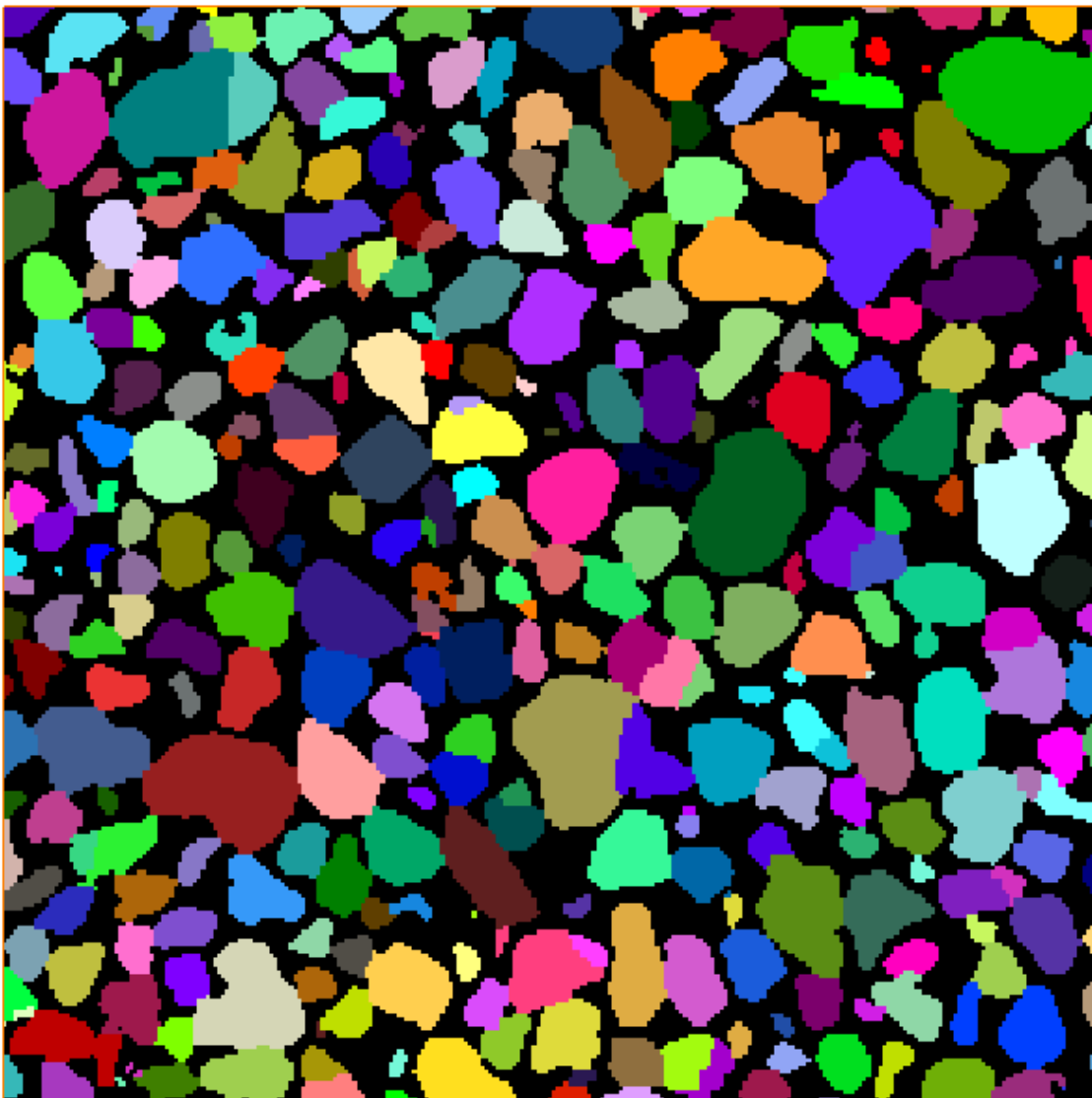


Figure 4.38. Ottawa sand 1 VOX2GRAINS updated output, 5039 grains, 3.025 mm per side,  $275^3$  voxels

While this feature is useful, the over merging is not desirable. The logistic regression equation developed from Trial 3 was also applied to each of the five images, since this trial resulted in a satisfactory accuracy while maintaining a lower number of false positives. Similarly, the cross sections for Ottawa sand 1 resulting from Trial 3 logistic regression equation is shown in Figures 4.40. For this image, the Trial 3 logistic regression equation identified a total of 3944 grains. When compared with the image in Figure 4.39, Figure 4.40 shows fewer over merged instances, while still correctly merging many of the initially over partitioned grains. The cross sections of the other four similar images after refined using the logistic regression equation derived from Trial 3 are also shown in the Appendix.

The weights of the equation and their corresponding spatial statistic are shown in Table 4.3. This trial included feature standardization. Normally, this would mean that the resulting equation weights could be interpreted as relative importance, assuming no multicollinearity. However, in this situation, many of the features likely contain similar information, and therefore multicollinearity likely impacts many of the variable magnitudes. For the trials presented, no effort was made to reduce multicollinearity, since multicollinearity has no impact on predictive power of the logistic regression equation.

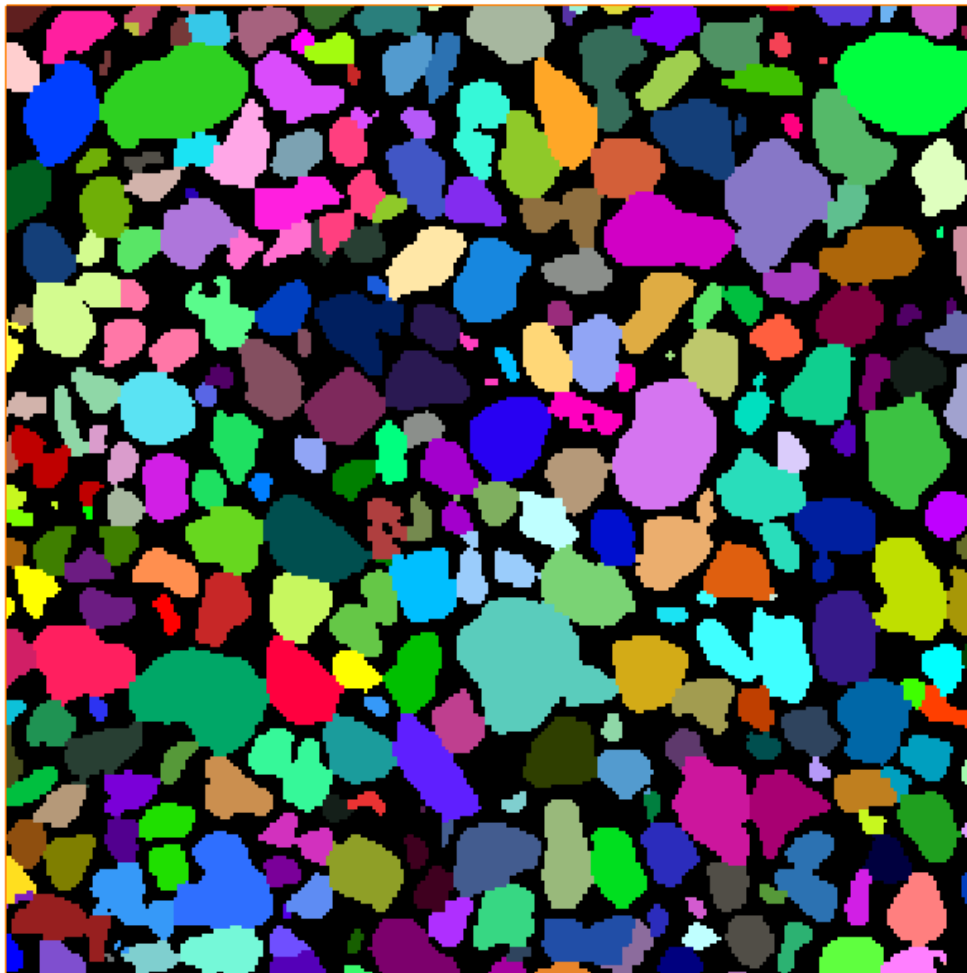


Figure 4.39. Ottawa sand 1 after logistic regression applied: trial 2 equation, 3723 grains, 3.025 mm per side,  $275^3$  voxels

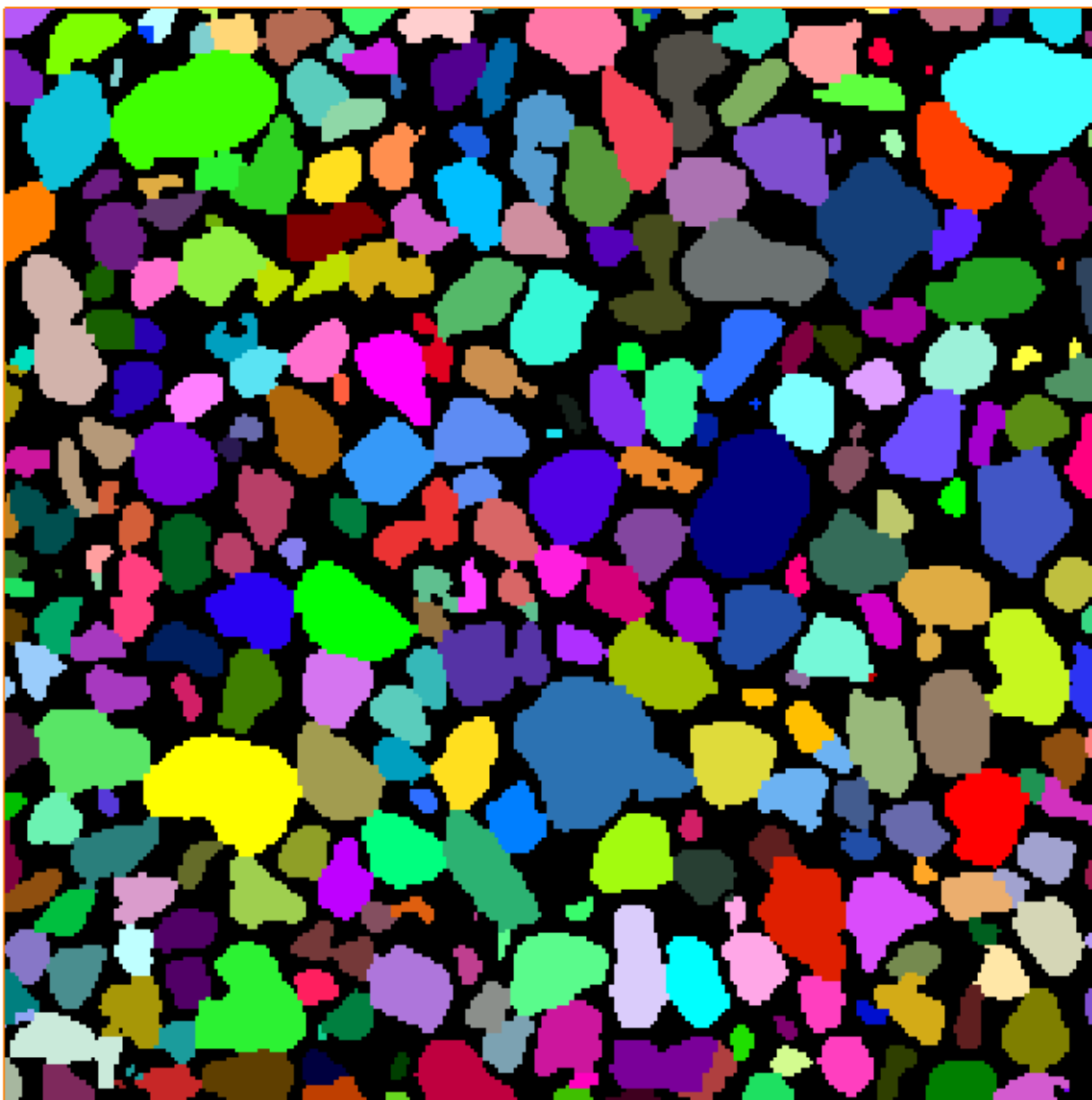


Figure 4.40. Ottawa sand 1 with trial 3 logistic regression applied, 3944 grains, 3.025 mm per side,  $275^3$  voxels

Comparing the images before and after the logistic regression equation has been applied, the images show the capability of distinguishing between over partitioned and correctly partitioned grains within a reasonable accuracy. Regarding machine learning, algorithms are only as good as the quality of the data, and a significant amount of time was spent generating the data used for the training and cross validation sets. While the equations have been useful for all five images tested here, these images are all of the same rock type. Different rock types can have very different characteristics and physical features, which ultimately can lead to very different partitions. Since the refinement equations are based solely on the spatial statistics, it follows that the equations

presented here would likely not be well suited for rocks with significantly different spatial characteristics.

Table 4.3. Logistic regression equation parameters from trial 3

	<b>Corresponding Statistic</b>	<b>Value</b>
	<b>Intercept</b>	-4.8476
<b>theta 2</b>	<b>Surface Area G1</b>	-1.2298
<b>theta 3</b>	<b>Surface Area G2</b>	-1.1896
<b>theta 4</b>	<b>Volume G1</b>	-0.4561
<b>theta 5</b>	<b>Volume G2</b>	-0.6566
<b>theta 6</b>	<b>Inscribed Radius G1</b>	-0.9082
<b>theta 7</b>	<b>Inscribed Radius G2</b>	-0.7760
<b>theta 8</b>	<b>Coord. Number G1</b>	-0.1244
<b>theta 9</b>	<b>Coord. Number G2</b>	-0.7111
<b>theta 10</b>	<b>X Pos. G1</b>	-0.2461
<b>theta 11</b>	<b>Y Pos. G1</b>	0.0441
<b>theta 12</b>	<b>Z Pos. G1</b>	0.8293
<b>theta 13</b>	<b>X Pos. G2</b>	-0.1450
<b>theta 14</b>	<b>Y Pos. G2</b>	-0.5933
<b>theta 15</b>	<b>Z Pos. G2</b>	0.7462
<b>theta 16</b>	<b>Contact Area</b>	4.7915
<b>theta 17</b>	<b>X Major Axis G1</b>	-0.2638
<b>theta 18</b>	<b>X Major Axis G2</b>	0.2625
<b>theta 19</b>	<b>Y Major Axis G1</b>	-0.1099
<b>theta 20</b>	<b>Y Major Axis G2</b>	0.1506
<b>theta 21</b>	<b>Z Major Axis G1</b>	-0.2106
<b>theta 22</b>	<b>Z Major Axis G2</b>	-0.1746
<b>theta 23</b>	<b>Aspect Ratio G1</b>	-0.7501
<b>theta 24</b>	<b>Aspect Ratio G2</b>	-0.2175
<b>theta 25</b>	<b>Contact Loc. X</b>	0.4648
<b>theta 26</b>	<b>Contact Loc Y</b>	0.1384
<b>theta 27</b>	<b>Contact Loc Z</b>	0.9463
<b>theta 28</b>	<b>Contact Norm Vect X</b>	0.0200
<b>theta 29</b>	<b>Contact Norm Vect Y</b>	0.4332
<b>theta 30</b>	<b>Contact Norm Vect Z</b>	-0.1616

## **CHAPTER 5. CONCLUSION AND RECOMMENDATIONS**

### **5.1. Conclusions**

Overall, many of the algorithms described in this thesis enhance the accuracy and reliability of automated grain partitioning for microtomographic segmented images through the computer program VOX2GRAINS. Specifically, regarding the VOX2GRAINS main body adjustment and alterations:

- The results show that an accurate distance map greatly improves partition quality and reduces the over partitioning inherent to programs that make use of the watershed transform.
- Adjusting the outer looping directions of the image during the granular assembly process was investigated and found to impact the image, but not to significantly improve the interfaces.
- Iterating granular assembly and reassigning voxels based upon neighbor counting is highly effective at reducing slots and drawers, resulting in cleaner grain-grain interfaces. Combined, these techniques are also proven to slightly reduce over partitioning even further.

Post processing refinement options extend the control that the user has when combating the remaining over partitioning due to the watershed transform. Specifically, the results showed:

- Automatic refinements have been developed to reassign obviously over partitioned grains with no exposed surface area. This method is insufficient for capturing all over partitions.
- Machine learning based upon particle and pair statistics such as contact area, surface areas, and aspect ratios, was found to be 98% accurate for determining whether or not two grains should or should not be merged together for several Ottawa (F-75) sand images. Although the applicability of this technology to other rock types is not known for certain, the results shown here were extremely satisfactory.
- Single plane planar regression was found to be effective for very simple geometries in packs with a small number of grains, but the image quality quickly deteriorates with increases in consolidation, shape irregularities, and grain populations.
- Multiplane planar regression is found to be well suited for simple and more complex geometries and consolidation states. While minimization of planar regression interfaces is implemented into the program, the results often yield undesirable outcomes.

### **5.2. Recommended Future Work**

- It is recommended that more rock images be processed and refined using machine learning. Although it is believed that different rock types will have different logistic regression equations, this should be verified. Since it can be timely to collect data and generate a decision equation, the verified equations for each rock type analyzed should be stored with a description in a library for quick refinement testing.
- Although multiplane planar regression without minimization has provided good results for the images tested, an accurate minimization algorithm would increase the capabilities of the program. The current approach breaks the minimization into translation and rotation. It is recommended to implement a minimization technique that combines these two

components into one to more efficiently find the true minimum. Furthermore, the planar extension method required for plane minimization should be further refined.

- For the testing in this thesis, the machine learning equation generation was carried out externally in MATLAB. A similar algorithm could be developed directly within VOX2GRAINS through the visualization software Avizo.
- Although a fairly rare occurrence for the images tested, there are scenarios in which the user may desire a single grain to be split into two. While manual refinement is an option, it seems possible that a particular rock partition could result in many such instances. As a preventative measure, it is recommended to investigate split refinements through machine learning just as shown here for merge decisions.
- VOX2GRAINS is currently not parallelized. Images of up to 600 cubed voxels have been successfully analyzed on the desktop version of the program, and images of up to 1000 cubed voxels have been analyzed on the HPC version. For larger images, processing time can take several hours. It is believed that by parallelizing the program, computation performance time can be significantly reduced.
- The partition qualities in this thesis were evaluated solely via visual inspection. To truly verify the results, it is recommended that a real rock image be tested and compared with a physical geologic interpretation of the same sample for more definitive conclusions on what constitutes individual grains. The logistic regression machine learning decisions should also be verified with physical geologic interpretations.

## REFERENCES

- Adams, R., and Bischof, L., 1994, Seeded Region Growing: IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 16, no. 6, p. 641-647.
- Arns, C. H., Bauguet, F., Limaye, A., Sakellariou, A., Senden, T. J., Sheppard, A. P., Sok, R. M., Pinczewski, V., Bakke, S., Berge, L. I., Øren, P.-E., and Knackstedt, M. A., 2005, Pore Scale Characterisation of Carbonates Using X-ray Microtomography, Society of Petroleum Engineers.
- Arns, C. H., Knackstedt, M. A., Pinczewski, M. V., and Lindquist, W. B., 2001, Accurate estimation of transport properties from microtomographic images: Geophysical Research Letters, v. 28, no. 17, p. 3361-3364.
- Arns, C. H., Madadi, M., Sheppard, A. P., and Knackstedt, M. A., 2007, Linear elastic properties of granular rocks derived from X-ray-CT Images, Society of Exploration Geophysicists.
- Aste, T., Saadatfar, M., Sakellariou, A., and Senden, T.J., 2004, Investigating the geometrical structure of disordered sphere packings: Physica A: Statistical Mechanics and its Applications, v. 339, no. 1-2, p. 16-23.
- Bakke, S., Oren, P.-E., 1997, 3-D Pore Scale Modeling of Sandstones and Flow Simulations in the Pore Networks, Society of Petroleum Engineers.
- Ballard, T., and Beare, S., 2013, Particle Size Analysis For Sand Control Applications, Society of Petroleum Engineers.
- Bauguet, F., Arns, C. H., Saadatfar, M., Turner, M., Sheppard, A. P., Sok, R., Pinczewski, V., and Knackstedt, M. A., 2005, Rock Typing and Petrophysical Property Estimation Via Direct Analysis on Microtomographic Images, Society of Core Analysts.
- Bernard, D., Gendron, D., Heintz, J.-M., Bordère, S., and Etourneau, J., 2005, First direct 3D visualisation of microstructure evolutions during sintering through X-ray computed microtomography: Acta Materialia, v. 53, no. 1, p. 121-128.
- Beucher, S., 2000, The Watershed Transform Applied To Image Segmentation: Scanning Microscopy.
- Beucher, S., and Lantuejoul, C., 1979, Use of Watersheds in Contour Detection: International Workshop on Image Processing: Real-time Edge and Motive Detection/Estimation.
- Blunt, M. J., Bijeljic, B., Dong, H., Gharbi, O., Iglauer, S., Mostaghimi, P., Paluszny, A., and Pentland, C., 2013, Pore-scale imaging and modeling: Adv. Water Resour., v. 51, p. 197-216.

- Breu, H., Gil, J., Kirkpatrick, D., and Werman, M., 1995, Linear time Euclidean distance transform algorithms: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 17, no. 6, p. 529-533.
- Clausnitzer, V., and Hopmans, J. W., 1999, Determination of phase-volume fractions from tomographic measurements in two-phase systems: *Advances in Water Resources*, v. 22, no. 6, p. 577-584.
- Coles, M. E., Hazlett, R. D., Muegge, E. L., Jones, K. W., Andrews, B., Dowd, B., Siddons, P., Peskin, A., Spanne, P. and Soll, W. E., 1996, *Developments in Synchrotron X-Ray Microtomography with Applications to Flow in Porous Media*, Society of Petroleum Engineers.
- Dalla, E., Hilpert, M., and Miller, C.T., Computation of the interfacial area for two-fluid porous medium systems: *Journal of Contaminant Hydrology*, v. 56, no. 1-2, p. 25-48.
- Denison, C., Carlson, W. D., and Ketcham, R. A., 1997, Three-dimensional quantitative textural analysis of metamorphic rocks using high-resolution computed X-ray tomography: Part I. Methods and techniques: *Journal of Metamorphic Geology*, v. 15, no. 1, p 29-44.
- Eberly, D., 2000, *Least Squares Fitting of Data: Geometric Tools*, LLC.
- Elliot, J. C., and Dover, S. D., 1981, X-ray microtomography: *Journal of Microscopy*, v. 126, no. 2, p. 211-213.
- Erdogan S. T., Nie, X., Stutzman, P. E., and Garboczi, E. J., 2010, Micrometer-scale 3-D shape characterization of eight cements: Particle shape and cement chemistry, and the effect of particle shape on laser diffraction particle size measurements: *Cement and Concrete Research*, v. 40, no. 5, p. 731-739.
- Erdogan S. T., Quiroga, P. N., Fowler, D. W., Saleh, H. A., Livingston, R. A., Garboczi, E. J., Ketcham, P. M., Hagedorn, J. G., and Satterfield, S. G., 2006, Three-dimensional shape analysis of coarse aggregates: New techniques for and preliminary results on several difference coarse aggregates and reference rocks: *Cement and Concrete Research*, v. 36, no. 9, p. 1619-1627.
- Golchert, D. J., Moreno, R., Ghadiri, M., Lister, J., and Williams, R., 2004, Application of X-ray microtomography to numerical simulations of agglomerate breakage by distinct element method: *Advanced Powder Technology*, v. 15, no. 4, p. 447-457.
- Hosterman, J. W., and Loferski, P. J., 1981, *Sample Preparation for X-Ray Diffraction Analysis and Clay Mineralogy of Devonian Shale from the Appalachian Basin*: United States Department of Energy.

- Iassonov, P., Gebrenegus, T., and Tuller, M., 2009, Segmentation of X-ray computed tomography images of porous materials: A crucial step for characterization and quantitative analysis of pore structures: *Water Resources Research*, v. 45, no. 9.
- Iassonov, P., and Tuller, M., 2010, Application of Segmentation for Correction of Intensity Bias in X-ray Computed Tomography Images: *Vadose Zone Journal*, v. 9, no. 1, p. 187–91.
- Ketcham, R.A., 2005, Computational methods for quantitative analysis of three-dimensional features in geologic specimens: *Geosphere*, v. 1, no. 1, p. 32-41.
- Kippax, P., 2005, Appraisal of the Laser Diffraction Particle-Sizing Technique: *Pharmaceutical Technology*, v. 88.
- Knackstedt, M. A., Arns, C. H., Limaye, A., Sakellariou, A., Senden, T. J., Sheppard, A. P., Sok, R. M., Pinczewski, W. V., and Bunn, G. F., 2004, Digital Core Laboratory: Properties of reservoir core derived from 3D images, Society of Petroleum Engineers.
- Knackstedt, M. A., Kelly, J., Saadatfar, M., Senden, T., and Sok, R., 2005, Rock fabric and texture from Digital Core Analysis: Society of Petrophysicists and Well-Log Analysts.
- Krumbein, W. C., and Sloss, L. L., 1951, *Stratigraphy and Sedimentation*: W. H. Freeman and Co., San Francisco.
- Lin, C.L., and Miller, J.D., 2005, 3D characterization and analysis of particle shape using X-ray microtomography (XMT): *Powder Technology*, v. 154, no. 1, p. 61-69.
- Lindblad, J., 2005, Surface area estimation of digitized 3D objects using weighted local configurations: *Image and Vision Computing*, v. 23, no. 2, p. 111-122.
- Lindquist, W. B., Lee, S.-M., Coker, D. A., Jones, K. W., and Spanne, P., 1996, Medial axis analysis of void structures in three-dimensional tomographic images of porous media: *Journal of Geophysical Research*, v. 101, no. B4, p. 8297-8310.
- Lindquist, W. B., and Venkatarangan, A., 1999, Investigating 3D geometry of porous media from high resolution images: *Physics and Chemistry of the Earth, Part A: Solid Earth and Geodesy*, v. 24, no. 7, p. 593–599.
- Mauer, C. R. Jr., Qi, R., and Raghavan, V., 2003, A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 25, no. 2, p. 265-270.
- Mills, G., 2016, Extracting Physically Realistic Pore Network Features from XCT Data and Investigating the Impact of Pore Topology on Flow in Porous Media, Doctoral Dissertation, Department of Petroleum Engineering, Louisiana State University.

- Ng, Andrew, Machine Learning, Coursera, September 2017, lecture videos, <[www.coursera.com/learn/machine-learning/](http://www.coursera.com/learn/machine-learning/)>.
- Oh, W., and Lindquist, W. B., 1999, Image Thresholding by Indicator Kriging: IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 21, no. 7, p. 590-602.
- Oren, P.-E., and Bakke, S., 2003, Reconstruction of Berea sandstone and pore-scale modelling of wettability effects: Journal of Petroleum Science and Engineering, v. 39, no. 3-4, p. 177-199.
- Proussevitch, A. A., and Sahagian, D. L., 1999, Recognition and separation of discrete objects within complex 3D voxelized structures: Computers and Geosciences, v. 27, no. 4, p. 441-454.
- Reed, A. H., Thompson, K. E., Briggs, K. B., and Willson, C. S., 2010, Physical Pore Properties and Grain Interactions of SAX04 Sands: IEEE Journal of Oceanic Engineering, v. 35, no. 3, p. 488-501.
- Reed, A. H., Thompson, K. E., Willson, Briggs, K. B., B., Richardson, M. D, and Hefner, B., 2005, Quantification of Sediment Properties from Pore Structure and Grain Contacts: A Microcomputed Tomography Analysis of SAX04 Sands: Naval Research Laboratory.
- Richard, P., Philippe, P., Barbe, F., Bourlés, S., Thibault, X., and Bideau, D., 2003, Analysis by x-ray microtomography of granular packing undergoing compaction: Physical Review E: Statistical, Nonlinear, and Soft Matter Physics, American Physical Society, v. 68.
- Saadatfar, M., Sheppard, A. P., and Knackstedt, M. A., 2006, Grain Partitioning and its Applications: Advances in X-ray Tomography for Geomaterials.
- Sakellariou, A., Sawkins, T. J., and Limaye, A., 2004, X-ray tomography for mesoscale physics applications: Physica A: Statistical Mechanics and its Applications, v. 339, no. 1-2, p. 152-158.
- Seidler, G. T., Martinez, G., Seeley, L. H., Kim, K. H., Behne, E. A., Zaranek, S., Chapman, B. D., Heald, S. M., and Brewe, D. L., 2000, Granule-by-granule Reconstruction of a sandpile from x-ray microtomography data: Physical Review: E, Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics, v. 62, no. 6 Pt B, p. 8175-8185.
- Sheppard, A.P., Arns, C.H., Sakellariou, A., Senden, T.J., Sok, R.M., Averdunk, H., Saadatfar, M., Limaye, A., and Knackstedt, M., 2006, Quantitative properties of complex porous materials calculated from x-ray  $\mu$ CT images: SPIE, v. 6318.
- Sheppard, A.P., Sok, R. M., and Averdunk, H., 2005, Improved Pore Network Extraction Methods: International Symposium of the Society of Core Analysts.

- Sheppard, A. P., Sok, R. M., and Averdunk, H., 2004, Techniques for image enhancement and segmentation of tomographic images of porous materials: *Physica A: Statistical Mechanics and its Applications*, v. 339, no. 1-2, p. 145-151.
- Sheppard, A.P., Sok, R.M., Averdunk, H., Robins, V. B., and Ghous, A., 2006, Analysis of rock microstructure using high-resolution x-ray tomography: *International Symposium of the Society of Core Analysts*.
- Sifakis, E., Grinias, I, and Tziritas, G., 2002, Video Segmentation Using Fast Marching and Region Growing Algorithms: *EURASIP Journal on Advances in Signal Processing*.
- Sifakis, E., and Tziritas, G., 2000, Moving object localisation using a multi-label fast marching algorithm: *Signal Processing: Image Communication*, v. 16, no. 10, p. 963-976.
- Sok, R. M., Knackstedt, M. A., Varslot, T., Ghous, A., Latham, S., and Sheppard, A. P., 2010, Pore Scale Characterization of Carbonates at Multiple Scales: Integration of Micro-CT, BSEM, and FIBSEM, *Society of Petrophysicists and Well-Log Analysts*.
- Tahmasebi, P., and Sahimi, M., 2012, Reconstruction of three-dimensional porous media using a single thin section: *Physical Review: E, Statistical, Nonlinear, and Soft Matter Physics*, v. 85, no. 6 Pt 2.
- Tanaka, S., Kato, Z., Uchida, Nozomu, and Uematsu, K., 2003, Direct observation of aggregates and agglomerates in alumina granules: *Powder Technology*, v. 129, no. 1-3, p. 153-155.
- ThermoFisher Scientific, 2017, PerGeos Software,  
< <https://www.fei.com/software/pergeos-for-oil-gas/>>.
- Thompson, K.E., 2007, Computing Particle Surface Areas and Contact Areas from Three-Dimensional Tomography Data of Particulate Materials: *Particle and Particle Systems Characterization*, v. 24, no. 6, p. 440-452.
- Thompson, K. E., Willson, C. S., White, C. D., Nyman, S., Bhattacharya, J. P., and Reed, A. H., 2008, Application of a new grain-based reconstruction algorithm to microtomography images for quantitative characterization and flow modeling, *Society of Petroleum Engineers*.
- Thompson, K.E., Willson, C.S., Zhang, W., 2006, Quantitative computer reconstruction of particulate materials from microtomography images: *Powder Technology*, v. 163, no. 3, p. 169-182.
- Vincent, L., and Soille, P., 1991, Watersheds in digital spaces: an efficient algorithm based on immersion simulations: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 13, no. 6, p. 583-598.

- Wildenschild, D., and Sheppard, A. P., 2012, X-ray imaging and analysis techniques for quantifying pore-scale structure and processes in subsurface porous medium systems: *Advances in Water Resources.*, v. 51, p. 217–246.
- Williams, R. A., and Jia, X., 2002, Tomographic Imaging of Particulate Systems: *Advanced Power Technology*, v. 14, no. 1, p. 1-16.
- Willson, C., Lu, N., and Likos, W. J., 2012, Quantification of Grain, Pore, and Fluid Microstructure of Unsaturated Sand from X-RAY Tomography Images: *Geotechnical Testing Journal*, v. 25, no. 6, p. 1-13.
- Zanjani, M. S., 2016, Impacts of Rock-Brine Interactions on Petrophysical Properties of a Sandstone Reservoir, Utilizing Geomechanical and Pore Network Modeling, Doctoral Dissertation, Department of Petroleum Engineering, Louisiana State University.

## APPENDIX. MACHINE LEARNING OUTPUTS

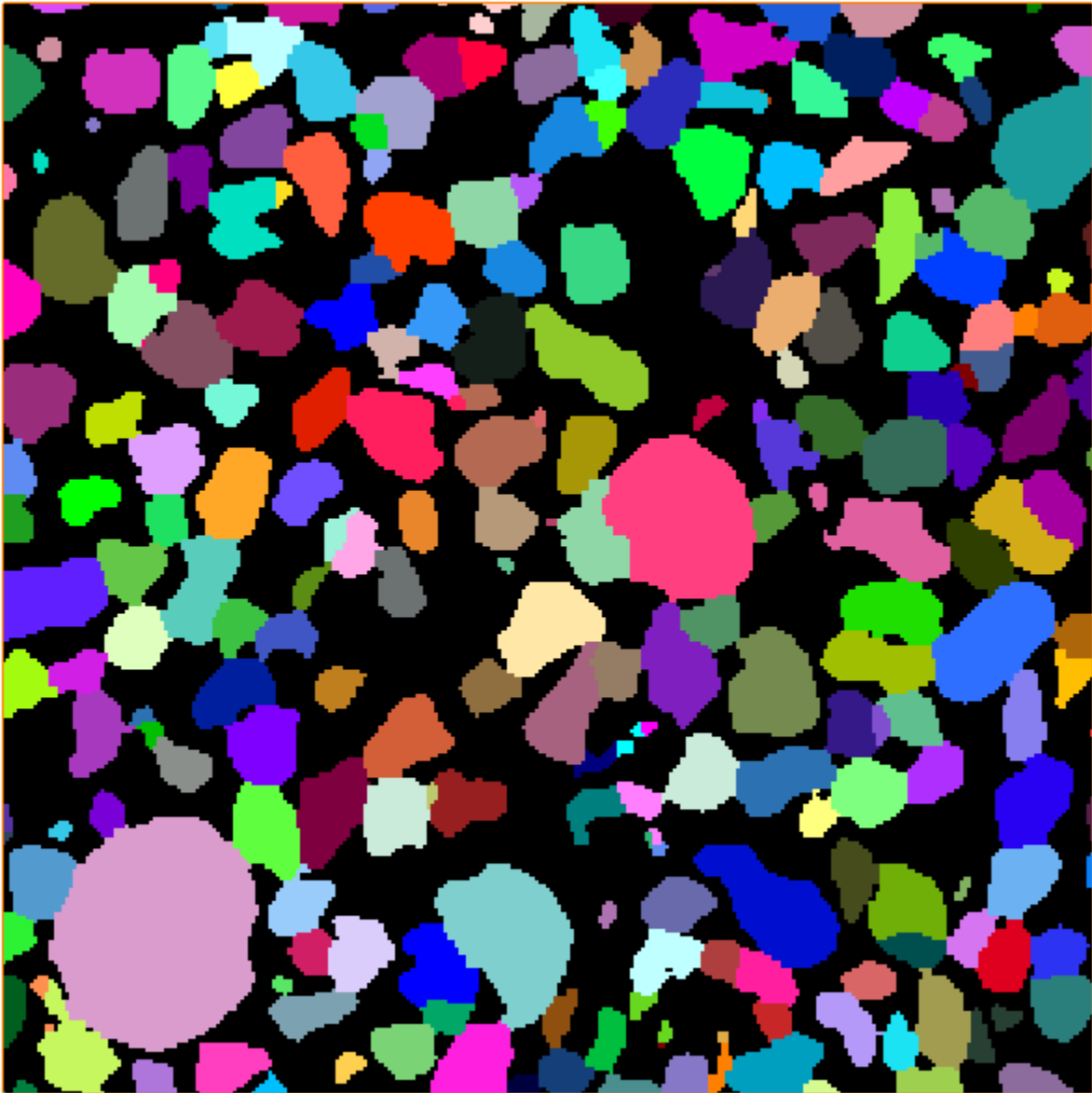


Figure A.1. Ottawa sand 3, VOX2GRAINS updated output, 3450 grains, 3.025 mm per side,  $275^3$  voxels

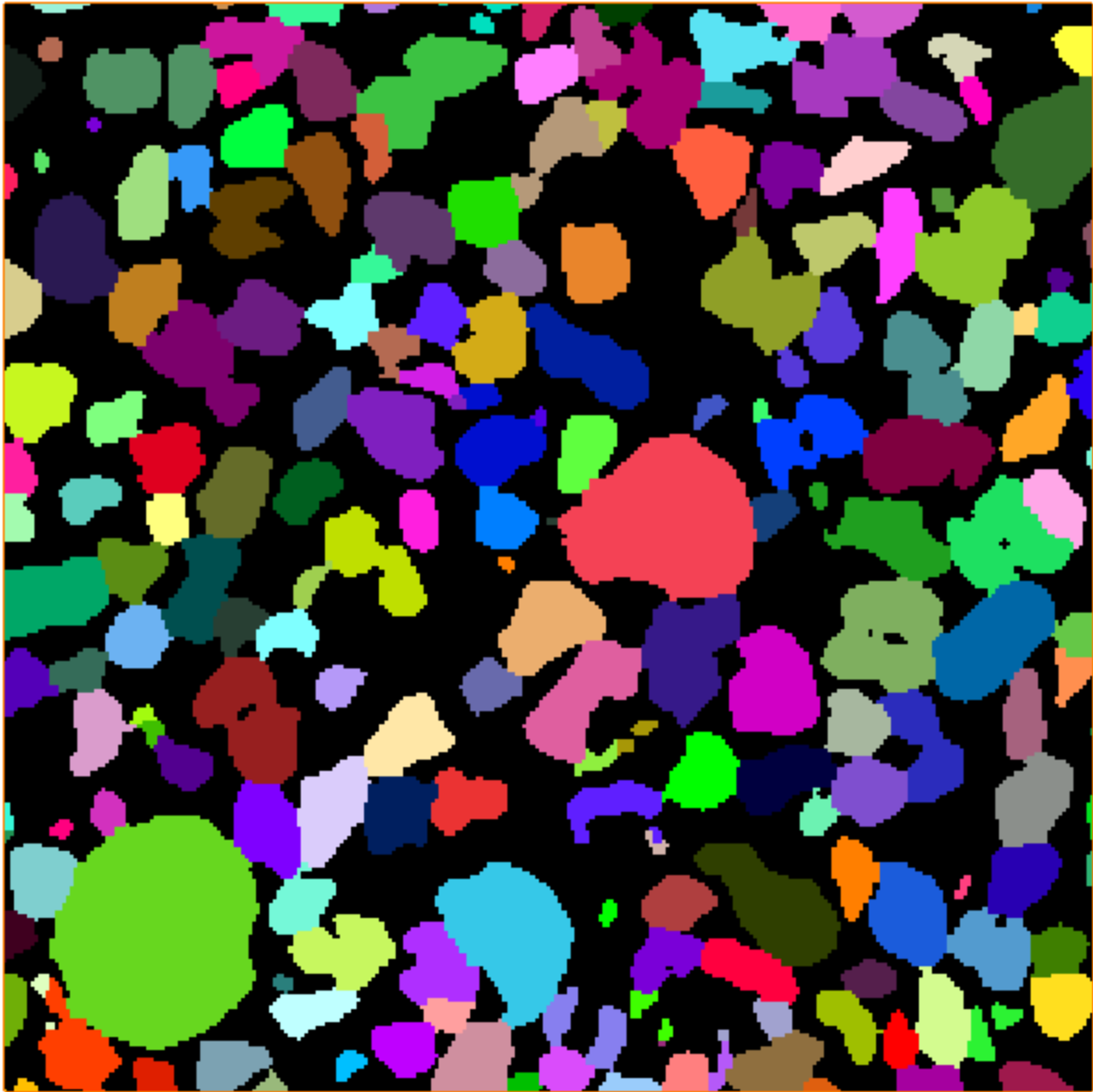


Figure A.2. Ottawa sand 3 after trial 2 logistic regression equation, 2584 grains, 3.025 mm per side,  $275^3$  voxels

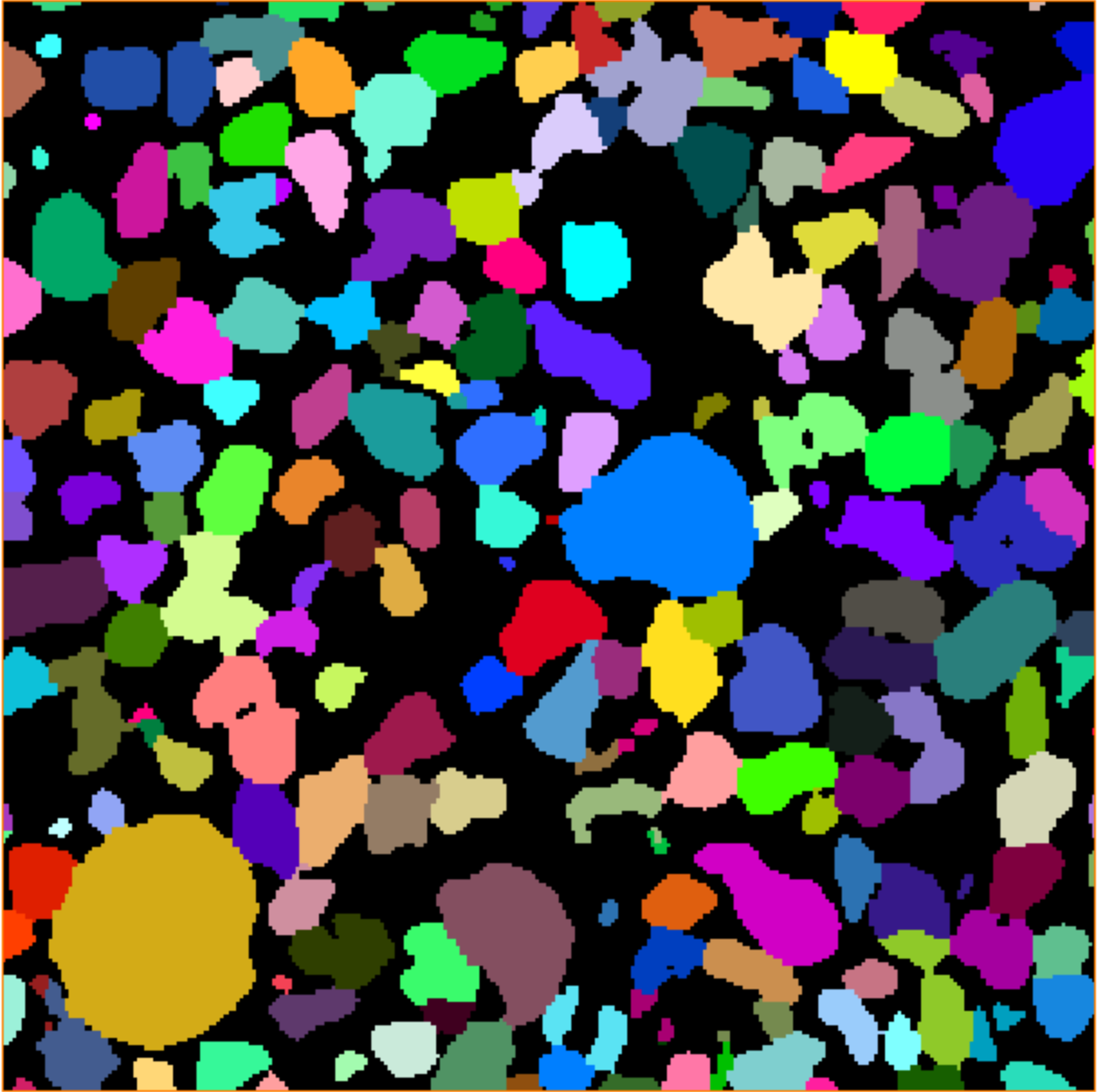


Figure A.3. Ottawa sand 3 after trial 3 logistic regression equation, 2661 grain, 3.025 mm per side,  $275^3$  voxels

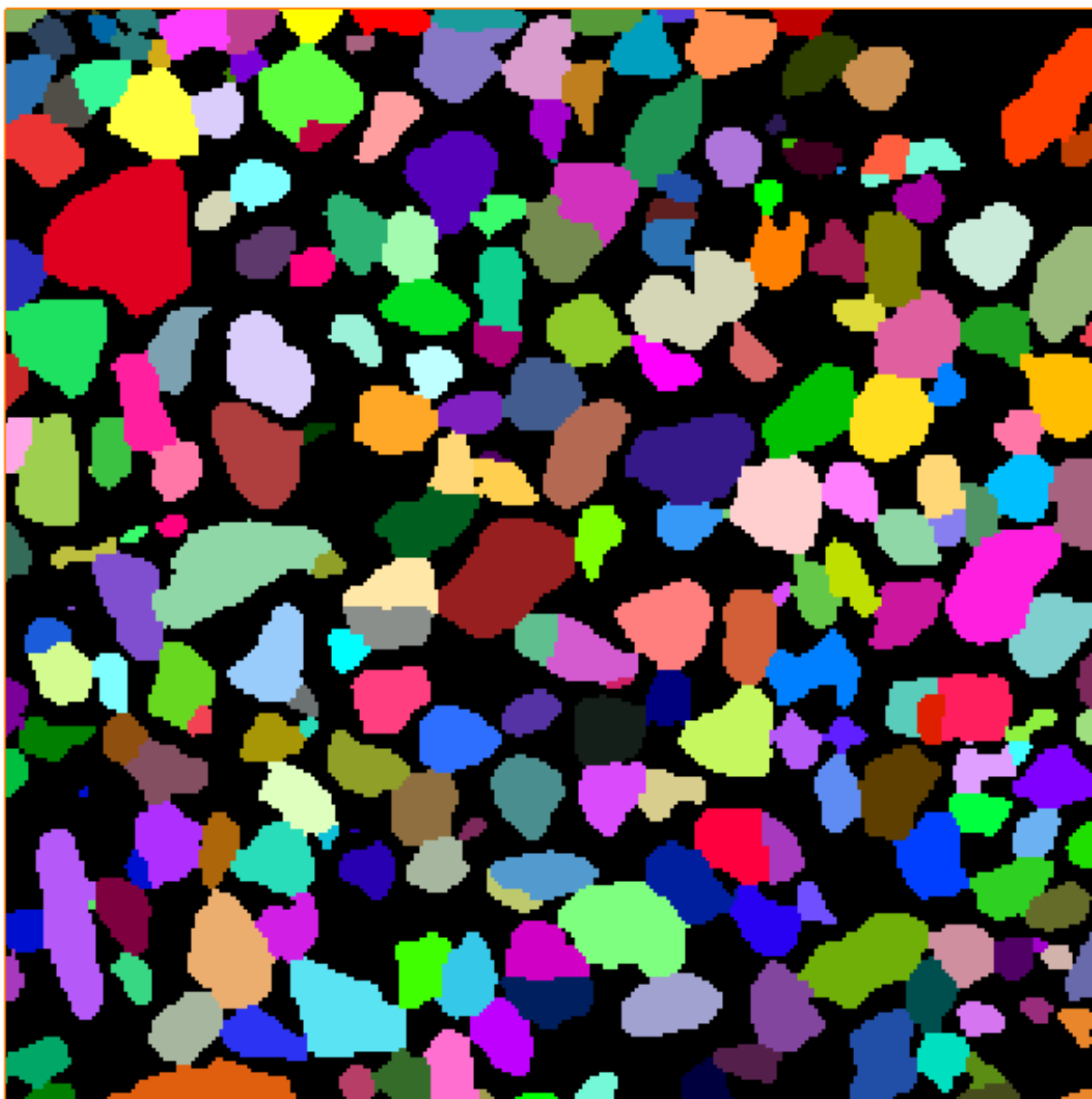


Figure A.4. Ottawa sand 5 VOX2GRAINS updated output, 3775 grains, 3.025 mm per side,  $275^3$  voxels

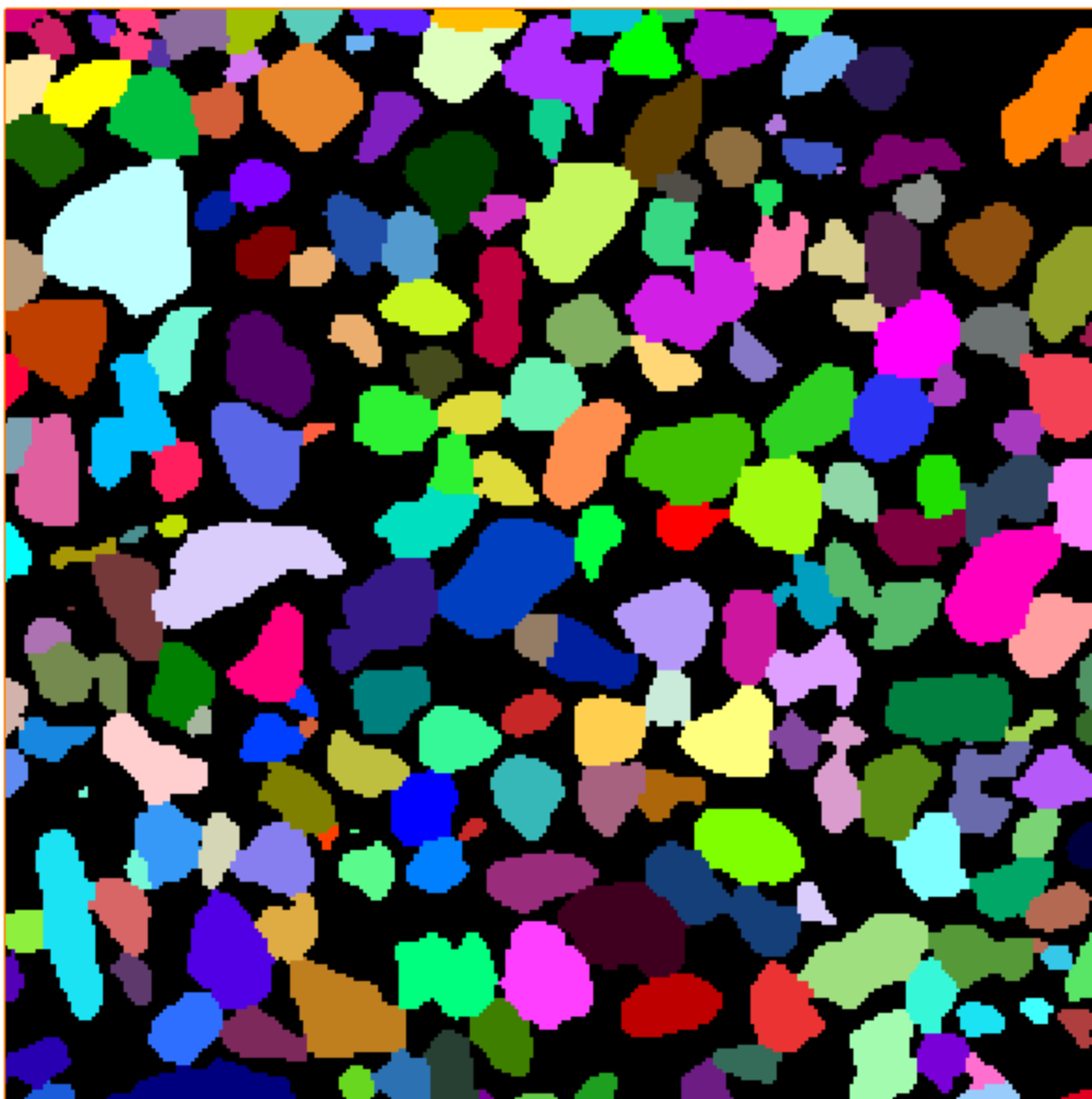


Figure A.5. Ottawa sand 5 after trial 2 logistic regression equations, 2792 grain, 3.025 mm per side,  $275^3$  voxels

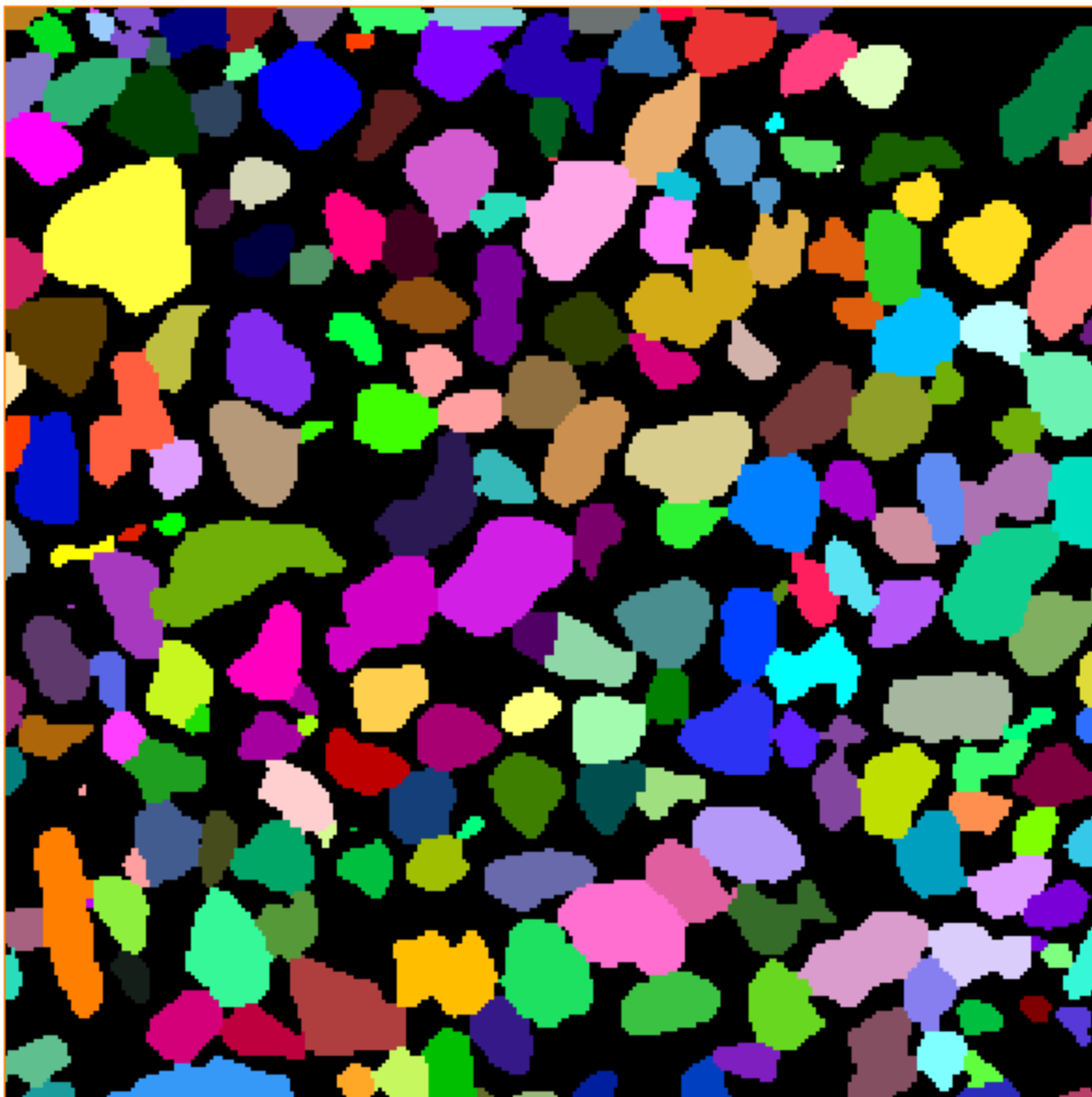


Figure A.6. Ottawa sand 5 after trial 3 logistic regression equation, 2892 grains, 3.025 mm per side,  $275^3$  voxels

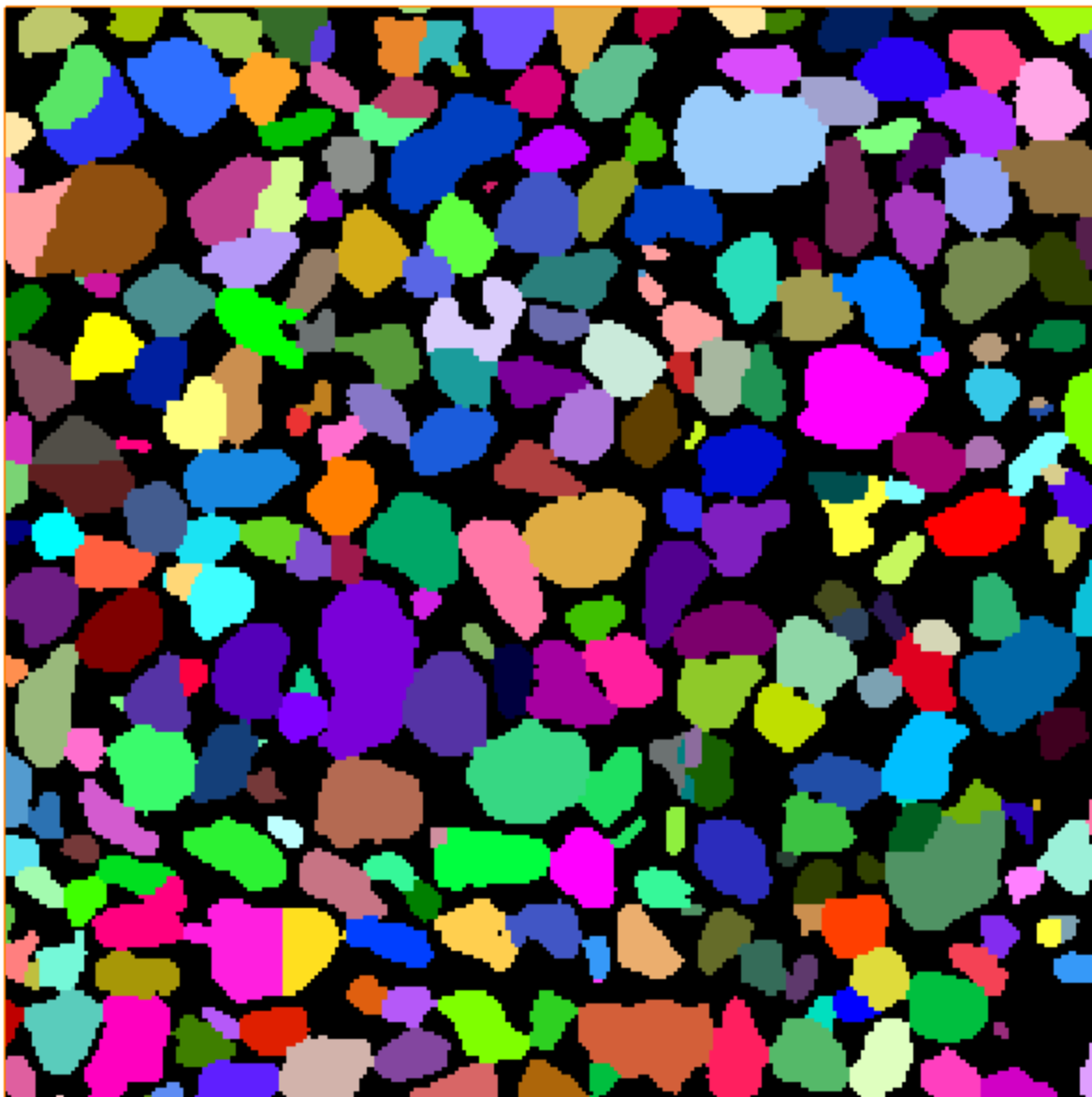


Figure A.7. Ottawa sand 7 VOX2GRAINS updated output, 4006 grains, 3.025 mm per side,  $275^3$  voxels

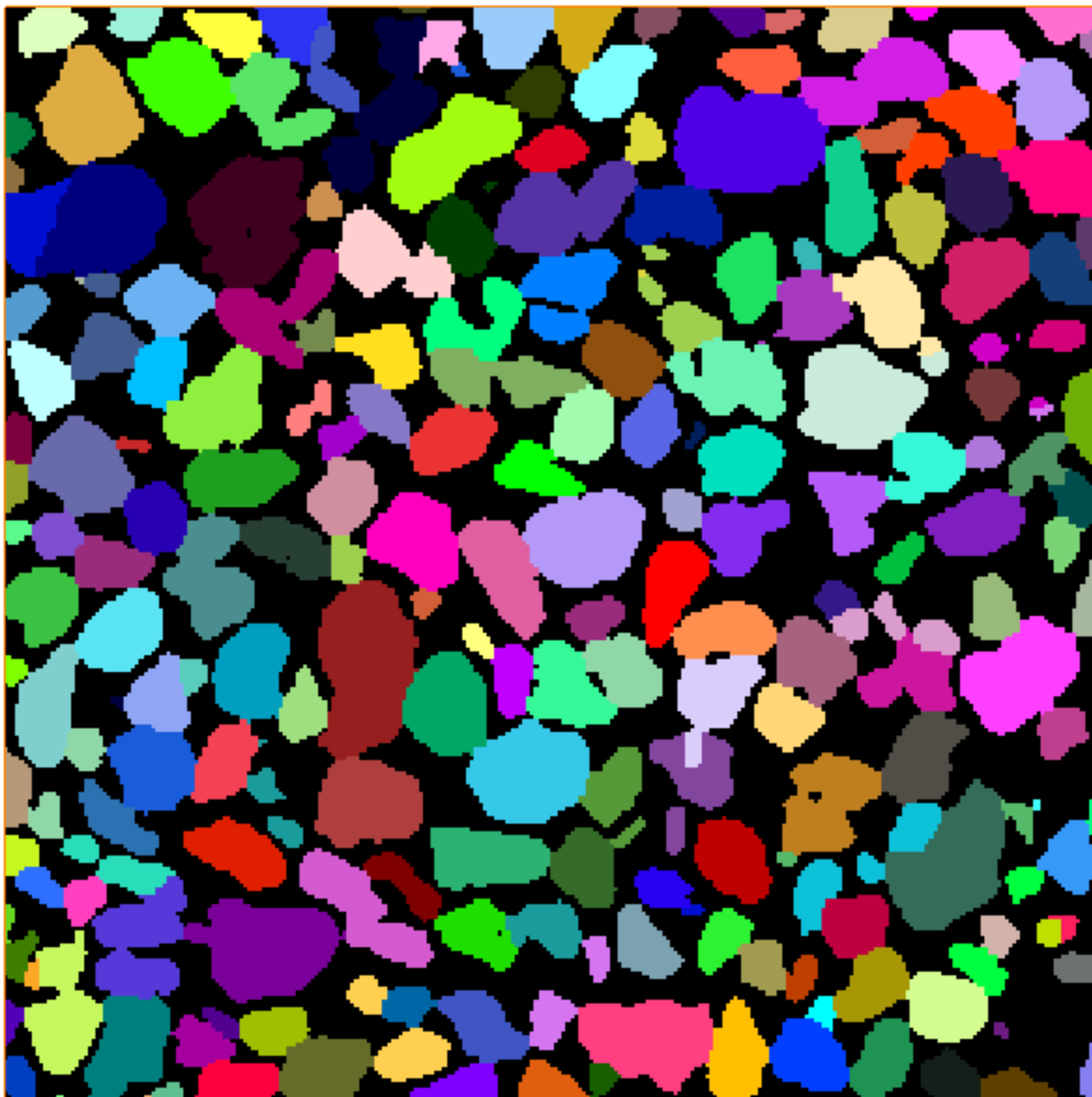


Figure A.8. Ottawa sand 7 after trial 2 logistic regression equation, 2993 grains, 3.025 mm per side,  $275^3$  voxels

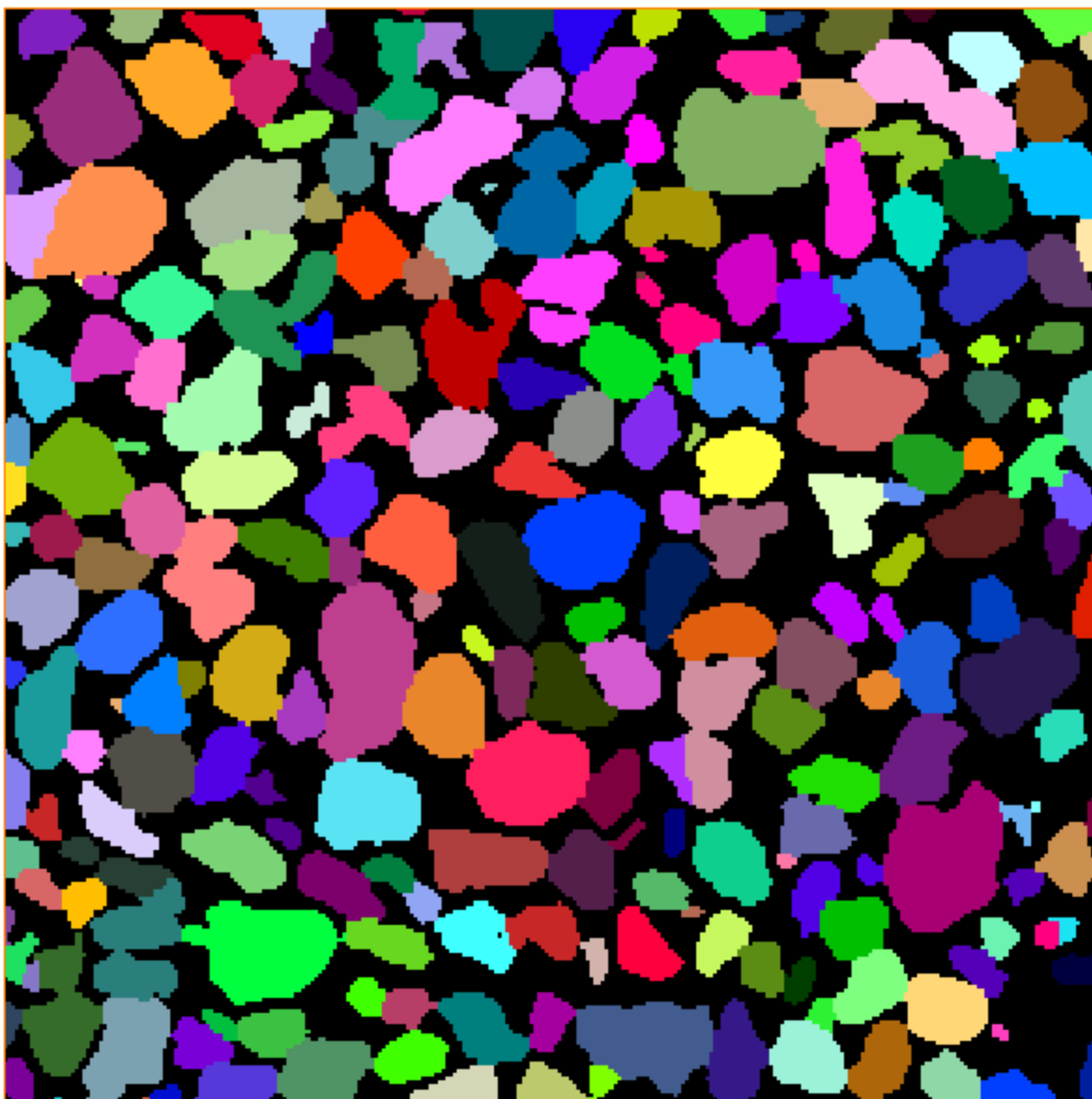


Figure A.9. Ottawa sand after trial 3 logistic regression output, 3108 grains, 3.025 mm per side,  $275^3$  voxels

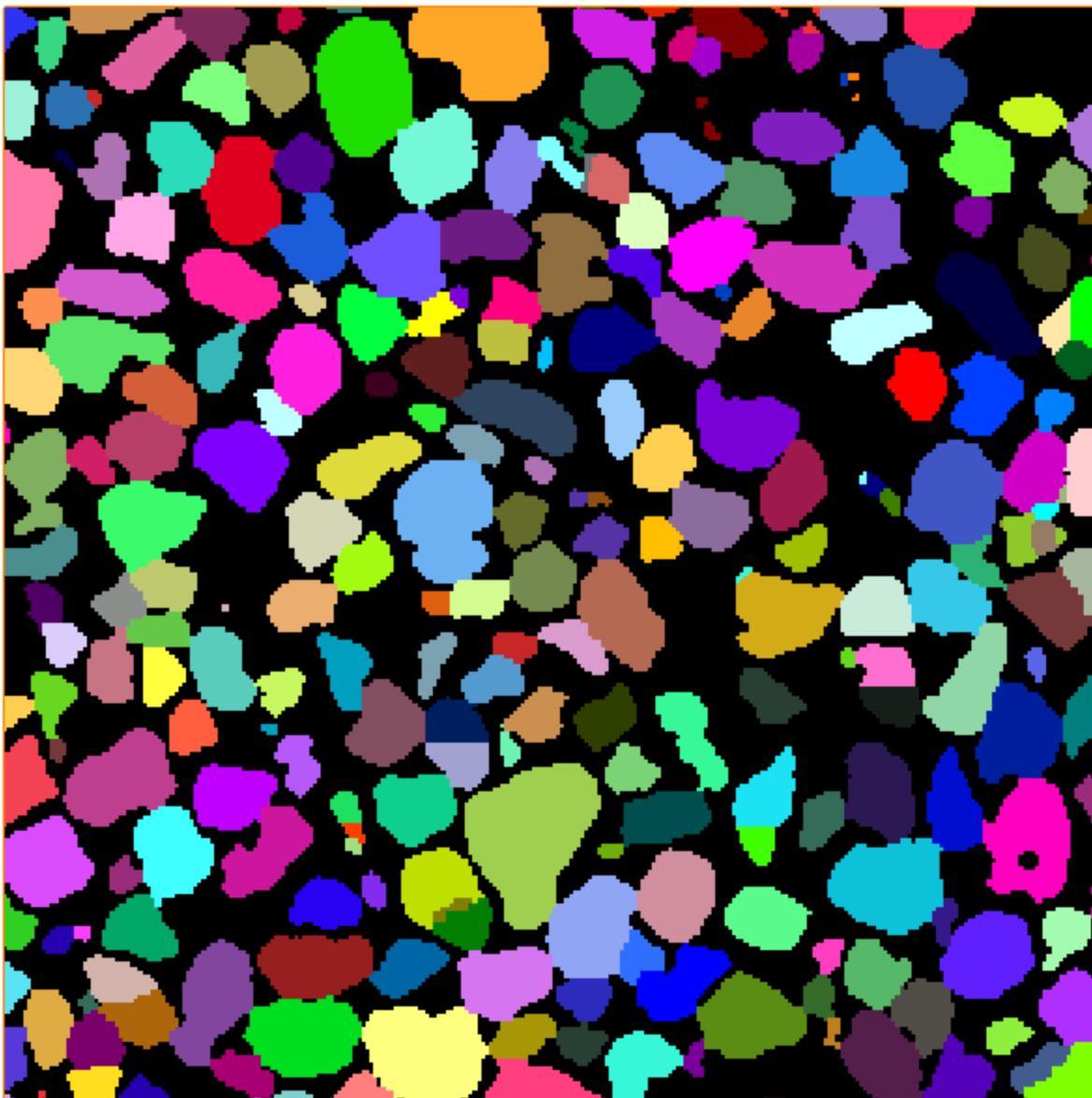


Figure A.10. Ottawa sand 10 VOX2GRAINS updated output, 3960 grains, 3.025 mm per side,  $275^3$  voxels

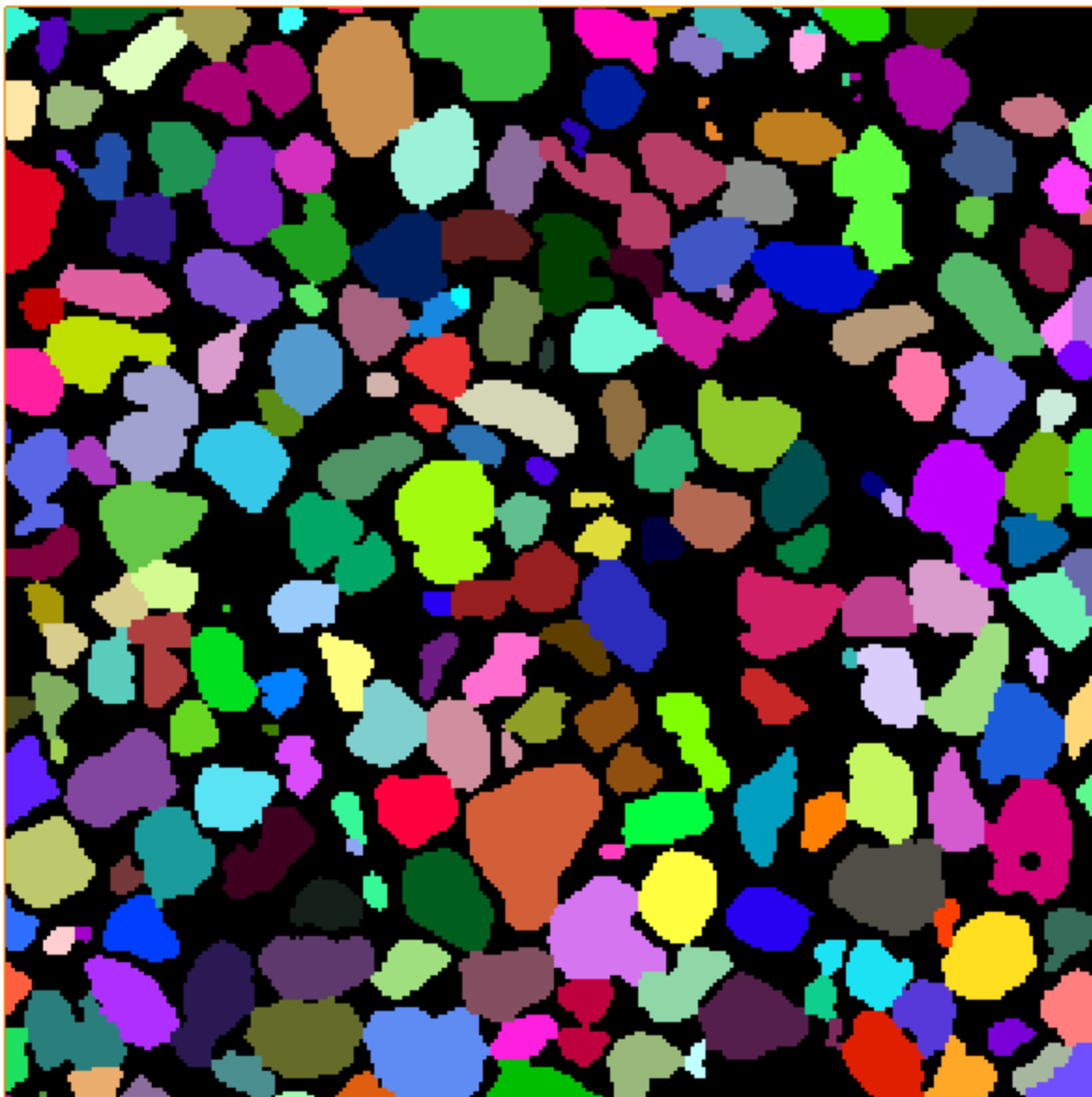


Figure A.11. Ottawa sand 10 after trial 2 logistic regression equation, 3045 grains, 3.025 mm per side,  $275^3$  voxels

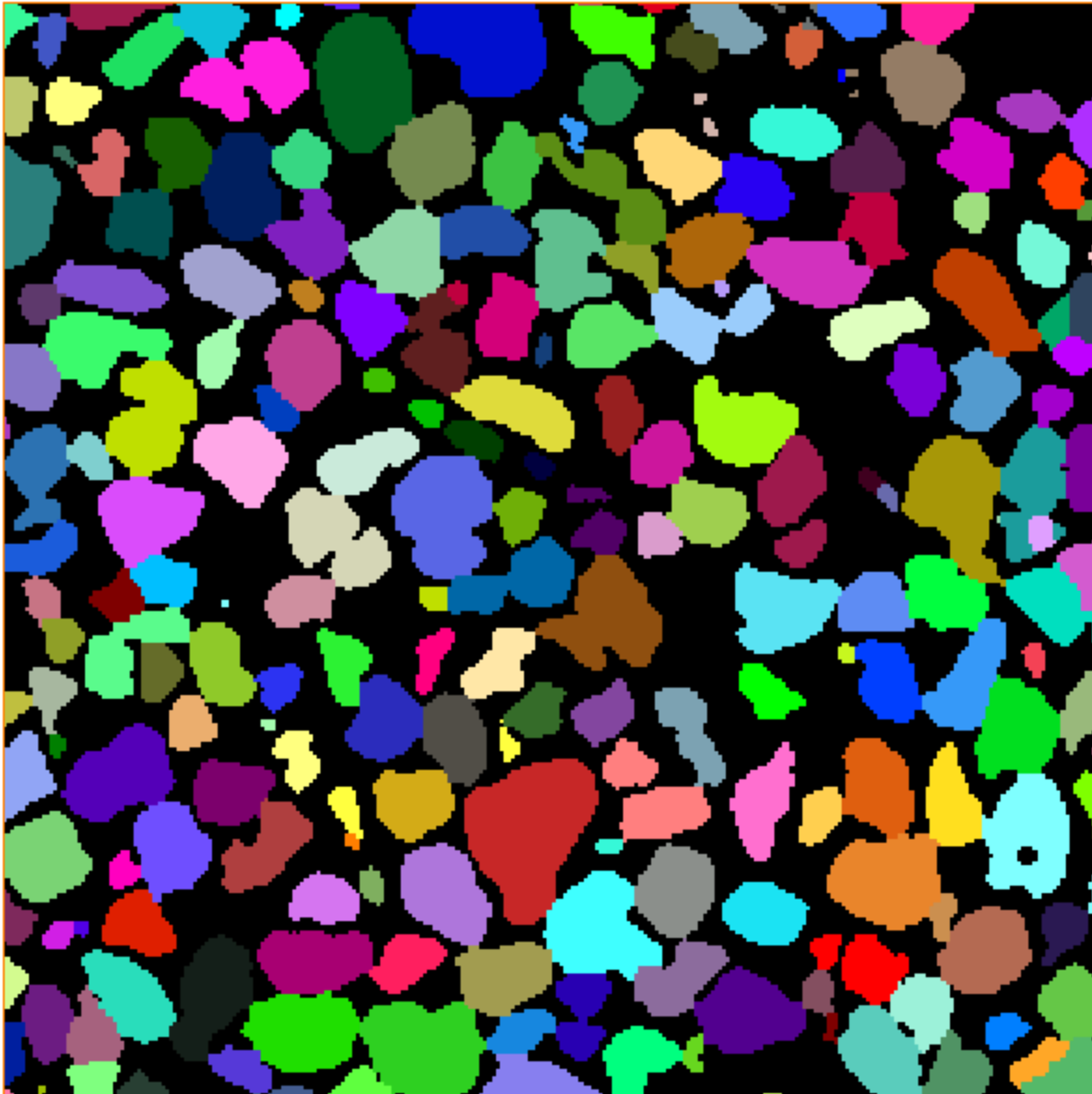


Figure A.12. Ottawa sand 10 after trial 3 logistic regression equations, 3149 grains, 3.025 mm per side,  $275^3$  voxels

