1-1-2013

3D Grain Segmentation in Superalloy Images Using Multichannel Edge-Weighted Centroidal Voronoi Tessellation Based Methods

Yu Cao
University of South Carolina

3D Grain Segmentation in Superalloy Images Using Multichannel Edge-Weighted Centroidal Voronoi Tessellation Based Methods

by

Yu Cao

Bachelor of Science
Northeastern University, China 2003

Master of Science
Northeastern University, China 2007

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2013

Accepted by:

Song Wang, Major Professor

Lili Ju, Co-Advisor and External Examiner

Michael Huhns, Committee Member

Manton Matthews, Committee Member

Yan Tong, Committee Member

Lacy K. Ford, Vice Provost and Dean of Graduate Studies

DEDICATION

To my beloved parents and my wife who give me all their love and have always been there for me through all the hard times.

ACKNOWLEDGMENTS

I would like to thank many people who had helped me in my Ph.D.'s study and dissertation, including Dr. Song Wang and Dr. Lili Ju, as my major professor and co-advisor, respectively, for offering valuable academic guidance and sincere career advice throughout my entire Ph.D. life; Dr. Michael Huhns, Dr. Manton Matthews and Dr. Yan Tong, for serving as my committee members and providing very constructive suggestions and comments on some critical problems.

I would like to thank Professor Marc D. Graef from the Carnegie Mellon University for providing the superalloy image dataset used in this dissertation. I would like to thank Qin Zou and Chengzhang Qu for constructing the ground-truth of the testing images.

I would like to thank my fellow lab-mates for providing a positive and stimulating environment in which not only I learned academic knowledge but also I harvested pleasure life experiences. I am especially grateful to Dr. Pahal Dalal, Dr. Andrew Temlyakov, Dr. Zhiqi Zhang, Jarrell Waggoner, Dhaval Salvi, Yuewei Lin, Dazhou Guo, Jun Zhou, Youjie Zhou, Xiaochuan Fan, Kang Zheng. Ping Liu and Shizhong Han.

I would like to thank Professor Jeffrey M. Siskind from the Purdue University and Professor Sven Dickinson from the University of Toronto for their inspirational comments and suggestions in my Ph.D. study.

I would like to thank my wife, Jiang Wu, for her moral countenance, love and understanding.

The last but not the least, I would like to thank my parents for giving me persistent supports throughout my life. Especially thank their supports to me on the way of pursuing academic path.

ABSTRACT

Accurate grain segmentation on 3D superalloy images is very important in materials science and engineering. From grain segmentation, we can derive the underlying superalloy grains' micro-structures, based on which many important physical, mechanical and chemical properties of the superalloy samples can be evaluated. However, grain segmentation is usually a very challenging problem since: 1) even a small 3D superalloy sample may contain hundreds of grains; 2) carbides and noises may degrade the imaging quality; and 3) the intensity within a grain may not be homogeneous. In addition, the same grain may present different appearances, i.e. intensities, under different microscope settings. In practice, a 3D superalloy image may contain multichannel information where each channel corresponds to a specific microscope setting. In this work, we develop a *Multichannel Edge-Weighted Centroidal Voronoi Tessellation* (MCEWCVT) algorithm to effectively and robustly segment the superalloy grains in 3D multichannel superalloy images. MCEWCVT performs segmentation by minimizing an energy function which encodes both the multichannel voxel-intensity similarity within each cluster in the intensity domain and the smoothness of segmentation in the 3D image domain. Based on MCEWCVT, we further develop a *Constrained Multichannel Edge-Weighted Centroidal Voronoi Tessellation* (CMEWCVT) algorithm which can take manual segmentation on a small number of selected 2D slices as constraints from the problem domain, and incorporate them into the energy minimization process to further improve the segmentation accuracy. We quantitatively evaluate the MCEWCVT and the CMEWCVT algorithms on an authentic Ni-based dataset and two synthesized datasets against ground-truth segmentation. The qualitative and quantitative comparisons among the MCEWCVT, the CMEWCVT and 18 existing

image segmentation algorithms on the authentic dataset demonstrate the effectiveness and robustness of the MCEWCVT and the CMEWCVT algorithms. In addition, the experiments on two synthesized datasets indicate that the optimal algorithm parameters found in the testing on the authentic dataset can be used on other superalloy datasets which have similar size and number of grains.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

In material industry, it usually takes scientists and engineers several years to design and develop a new kind of superalloy with specified properties. This process repeats the circle of generating superalloy samples and analyzing their underlying micro-structures. Like many other materials, superalloy is composed of a large number of grains, while the size, shape and interrelations of these grains usually determine the physical, mechanical and chemical properties of the superalloy sample, e.g. lightness, hardness, stiffness, electrical conductivity, fluid permeability and thermostability [44, 51]. These grains can be imaged by a microscope on serial-sectioned slices, as shown in Figure 0.1.

In modern industry, different applications may require different physical, mechanical and chemical properties of the related superalloy materials. For example, in turbine developments, the desirable superalloy materials should be high temperature tolerant (as illustrated in Figure 0.2). In addition, a steam turbine may require superalloy materials to be resistant to corrosion and oxidation. For an aircraft turbine, it further prefers the superalloy materials to be as light as possible. In submarine constructions, the involved superalloy materials are required not only to be strong in order to maintain the intact boat structure under extreme water pressure, but also to be low-magnetic in order to avoid magnet-oriented torpedo attack in battles (for example, the Russian Alpha class submarine as shown in Figure 0.3). On the contrary, high performance generators/transformers require wires made up of superalloy materials with high magnetic conductivity. Again, as mentioned above, all these properties are related to superalloy's micro-structures. Thus in material development, one inevitable step is to identify the micro-structures of superalloy samples. In terms of computer vision and image processing techniques, identifying the micro-structures is to

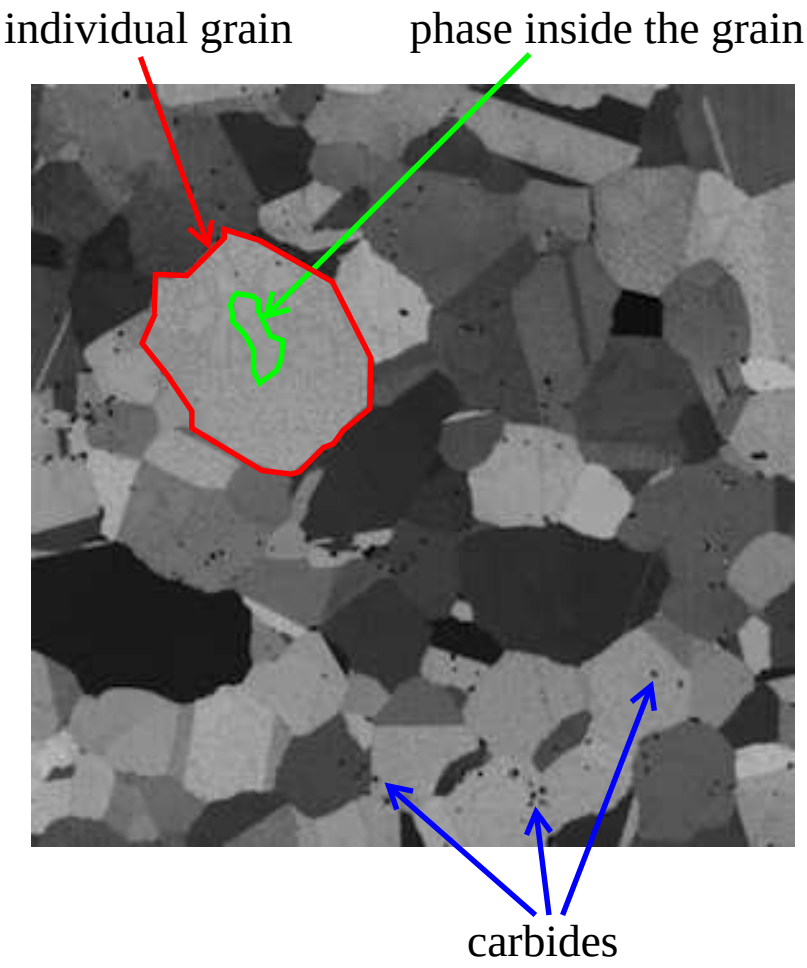


Figure 0.1 A superalloy image on one serial-sectioned slice.

pursue the segmentations of grains. Figure 0.1 provides an illustration of a 2D slice of a 3D superalloy sample. As can be seen from Figure 0.1, each cell (with similar intensities) is a grain, and those small black dots are carbides.

Currently, in the materials science community, a common practice to extract all the grains in a superalloy sample is to first manually segment each 2D slice and then reconstruct the 3D grains by corresponding segments between each two neighboring 2D slices. Given the large number of grains and slices, manual segmentation is very tedious, time consuming and prone to error.

The intensive labor burden will become even heavier when considering a special imaging phenomenon, i.e., many superalloy images contains multichannel information and each

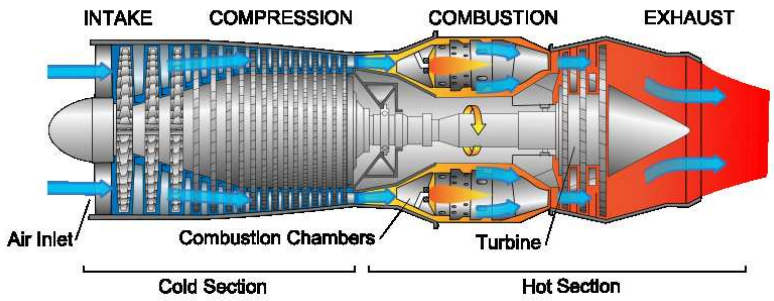


Figure 0.2 A jet engine illustration. The turbine in the red area is made of high temperature tolerant superalloys. (https://en.wikipedia.org/ wiki /File:Jet_engine.svg)

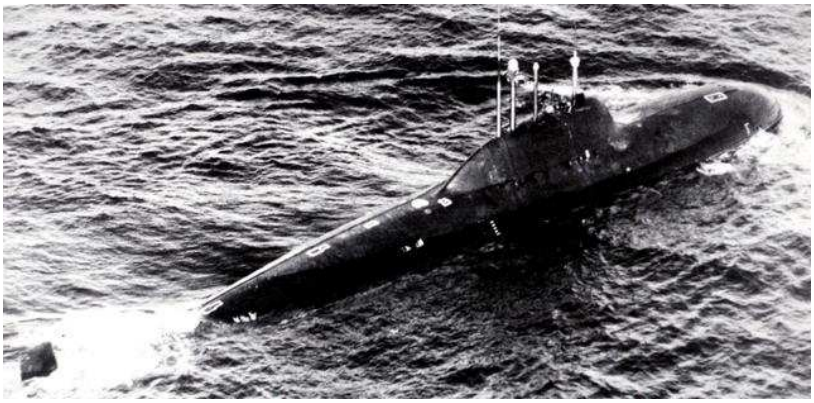


Figure 0.3 Russian Alpha-class submarine with double Ti-based superalloy hull. It is the first submarine whose whole shell is made of Ti-based superalloy. (http://en.wikipedia.org/wiki/File:Alfa_class_submarine_2.jpg)

channel corresponds to a specific microscope setting. This is called multichannel imaging. In principle, multichannel imaging provides more information to identify the boundary between adjacent grains since two adjacent grains may show similar intensities in one channel but different intensities in another channel. Figure 0.4 shows a 4-channel image of the same superalloy slice. We can see that adjacent grains g_1 and g_2 can be better separated in channels (a,b,d) than in channel (c). However, adjacent grains g_1 and g_3 can be better separated in (c) than in (a,b,d). In other words, the imaging patterns and intensity contrast patterns are not consistent for each pair of grains across all the channels. Consequently, this increases

the burden of manual segmentation since an annotator has to exam multiple images for just segmenting one 2D slice. For example, for the 170-slice 4-channel Ni-based superalloy dataset used in our later experiments, three annotators have to work full time for about two weeks to finish the manual segmentation. Therefore, automatic or semi-automatic image segmentation algorithms with limited human interactions are highly desirable for segmenting 3D superalloy images. However, grain segmentation from superalloy images is usually a very challenging problem since: 1) the grains to be segmented is of large quantity – even a small 3D superalloy sample may contain hundreds of grains; 2) carbides and noises may degrade the imaging quality; 3) the intensity within a grain may not be homogeneous, as shown in Figure 0.5; and 4) the segmentation is desired to conduct on 3D volume. Additionally, the developed algorithms should be able to leverage the multichannel information for more accurate grain segmentation.

In principle, many conventional 2D image segmentation methods can be used to segment the 2D image slice automatically [13, 37, 40, 45, 22, 9, 31, 30, 6, 36, 7] or semi-automatically [14, 25]. In [12], a stochastic segmentation algorithm following the "expectation maximization" / "maximization of the posterior marginals" (EM/MPM) principle is developed for segmenting 2D Ni-based grain images. However, these 2D segmentation methods do not consider the grain-structure continuity between neighboring slices. This not only affects the segmentation accuracy, but also brings difficulties in reconstructing the underlying 3D grain structures. 3D volume segmentation methods can address this limitation. However, some 3D segmentation methods, such as N-D graph cuts [10], are of high computational and memory complexity given the large number of grains. Some 3D methods, such as 3D random walks [25], requires manually specified seeds for each grain. Some other 3D methods, such as 3D level set methods [54, 39, 35, 30] and isosurface methods [50] usually handle the segmentation on a small number of disjoint structures very well. Statistical voxel clustering methods, such as K-means, have difficulties in considering the spatial information of the voxels and may produce many undesirable frag-

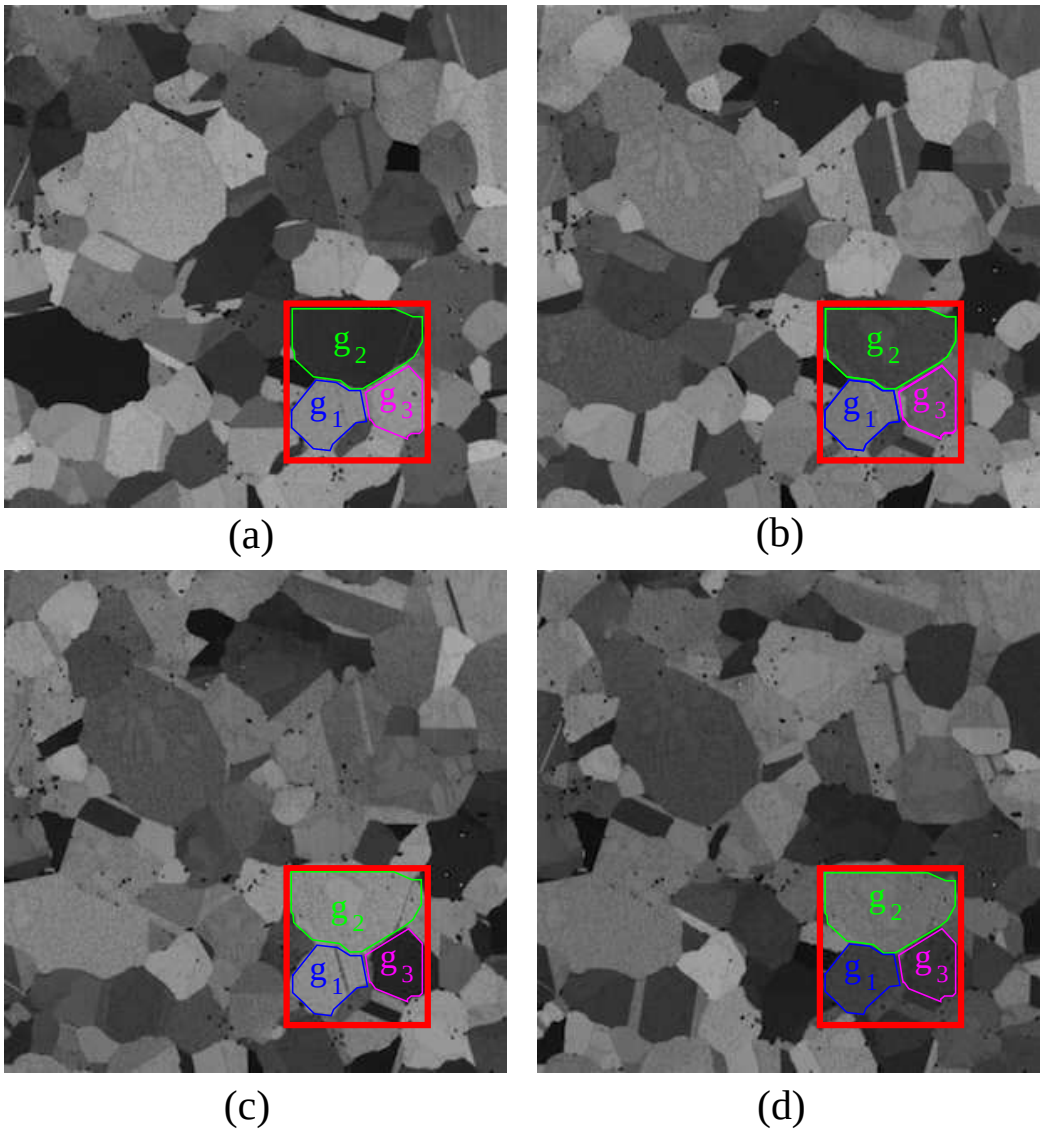


Figure 0.4 One slice of a superalloy sample with four image channels (4 different electronic microscope settings). (a) 4000_Series. (b) 5000_Series. (c) 6000_Series. (d) 7000_Series.

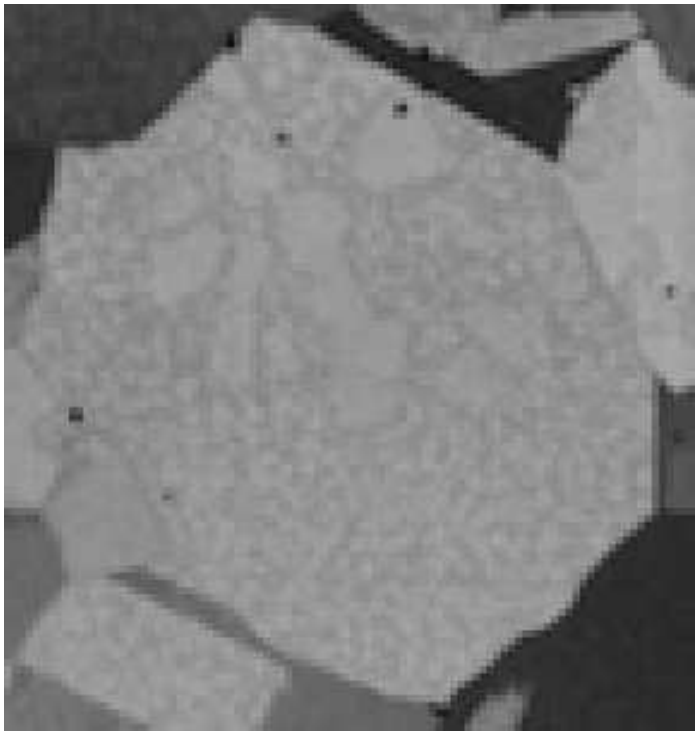


Figure 0.5 Inhomogeneous intensities inside a grain.

ments. These limitations prevent their applications to grain segmentation in 3D superalloy images. Additionally, many of these 2D and 3D methods have difficulties in leveraging the multichannel information for improving segmentation accuracy.

In this work, we first develop a new voxel clustering algorithm, called *Multichannel Edge-Weighted Centroidal Voronoi Tessellation* (MCEWCVT) for 3D multichannel superalloy image segmentation. In this algorithm, 3D grain segmentation is achieved by minimizing an energy function that consists of a multichannel clustering energy term defined in voxel intensity space and an edge smoothness energy term defined in 3D image space. As a result, the MCEWCVT can fully take the advantage of the multichannel information, and produce compact segments with smooth segmentation boundaries. Based on MCEWCVT, we further develop a *Constrained Multichannel Edge-Weighted Centroidal Voronoi Tessellation* (CMEWCVT) algorithm which can take manual segmentation on a small set of selected 2D slices as constraints, and incorporate them in the energy minimization process

to improve grain segmentation accuracy. In the experiments, we use the MCEWCVT and the CMEWCVT algorithms to segment a 4-channel 170-slice Ni-based 3D superalloy image, and compare these two algorithms with 18 existing 2D and 3D image segmentation methods. The comparisons demonstrate the effectiveness and robustness of the proposed MCEWCVT and CMEWCVT algorithms. We further apply MCEWCVT on two synthesized superalloy datasets, the experimental results on these two synthesized datasets indicate that the optimal algorithm parameters found in the testing on the authentic dataset can be used on other superalloy datasets which have similar size and number of grains.

The remainder of this work is organized as follows. In Chapter 1, we first introduce several existing 2D/3D segmentation methods, then we elaborate K-means, centroidal Voronoi tessellation and some other related materials. In Chapter 2, we defined the basic centroidal Voronoi tessellation model for the superalloy image segmentation problem, and then derive a new clustering energy suitable for multichannel superalloy image segmentation, followed by introducing the MCEWCVT algorithm. In Chapter 3, we develop the CMEWCVT algorithm to incorporate the manual segmentation on selected slices. In Chapter 4, we report the experimental results, together with qualitative and quantitative comparisons to 18 existing segmentation methods on an authentic IN100 Ni-based superalloy dataset. In Chapter 5, we report the experimental results on two synthesized datasets. In Chapter 6, we discussed three possible directions to which the proposed MCEWCVT and CMEWCVT algorithms can be further extended. Concludes are given at the end.

CHAPTER 1

BACKGROUND

In this chapter, we will introduce some related materials to the MCEWCVT and the CMEWCVT algorithms. These related materials include existing 2D/3D image segmentation methods; the well-known clustering algorithm – K-means and its Lloyd implementation; and the centrodial Voronoi tessellation (CVT) algorithm.

1.1 EXISTING IMAGE SEGMENTATION METHODS

Image segmentation is an important topic in both computer vision and image processing areas. In the past several decades, many 2D and 3D image segmentation methods have been developed and achieved remarkable qualitative and quantitative segmentation results on many imaging modalities. We will briefly elaborate several of them in the following.

1. **Mean Shift:** The original mean shift procedure was first proposed in 1975 by Fukunaga and Hostetler [24]. The goal of the Mean Shift algorithm is to identify the modes of a given density distribution. The idea of the mean shift algorithm can be described by the following mathematical formulation:

   Given an initial guess x and a kernel function $\mathcal{K}(\cdot)$, the weighted mean of the density in the kernel window determined by $\mathcal{K}(\cdot)$ is

   $$m(x) = \frac{\sum_{x_i \in N(x)} x_i \mathcal{K}(x_i - x)}{\sum_{x_i \in N(x)} \mathcal{K}(x_i - x)},$$

   where $N(x)$ is the neighborhood centered at x. Then the mean shift algorithm will update x to $m(x)$, and repeat the above procedure until $m(x)$ converges.

In the mean shift segmentation method [13], the authors first apply the mean shift algorithm on each pixel in the image, then for each pixel the mean shift algorithm would converge to a mode. Then the mode together with the spatial location of the corresponding pixel are combined to form a spatio-range element. These spatio-range elements are further merged into a certain number of clusters according to some predefined thresholds. Then the pixel's segment label is assigned as the cluster index.

2. **Statistical Region Merging:** The statistical region merging [40] method is built upon an assumption of *homogeneity*, i.e., inside any statistical region and given any color channel (R, G, B channel), the statistical pixels have the same expectation for this color channel, and the expectations of adjacent statistical regions are different for at least one color channel in R, G, B.

   Based on this assumption, a mathematical formulated region merging rule is derived, which requires the merged region must satisfy the above mentioned homogeneity assumption.

3. **Normalized Cuts:** Give a graph $G = (V, E)$, the normalized cuts [45] method is going to partition the graph into two disjoint sets, A, B, where $A \cup B = V$, $A \cap B = \emptyset$. An image segmentation can be achieved by recursively applying this graph bi-partition. The dissimilarity between A and B, also called as *cut*, is defined as

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v),$$

   where $w(u, v)$ are the weight for the edge connecting the node u and v. Then the normalized cuts algorithm formulates the graph partition problem as minimizing the following cost function:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)},$$

where $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$ is the total connections from nodes in A to all nodes in the graph, and $assoc(B, V)$ is similarly defined.

The terms $assoc(A, V)$ and $assoc(B, V)$ are the sum of weights of all edges that touch A and B respectively, and these two terms can also be reviewed as volumes of the partitions A and B respectively. By minimizing the normalized cuts cost function, the obtained segmentation results tend to have relative similar size of segments and suppress small segmentation fragments.

4. **EM/MPM:** Recently in [12], the authors propose a method called EM/MPM for 2D Ni-based grain segmentation problem. This method combines a region merging segmentation algorithm called the "stabilized inverse diffusion equation" (SIDE) and a stochastic segmentation method called the "expectation-maximization / maximization of the posterior marginals" (EM/MPM). In particular, the SIDE is a family of semi-discrete evolution equations which can stably capture the shape edge and suppress noises as well. The EM/MPM algorithm uses a Markov random field model built upon the pixel class label, and alternatively approximates the observed image model parameters and the pixel class labels during the optimization process.

   Specifically in [12], the SIDE algorithm is used to segment the grain boundaries, while the EM/MPM algorithm is used to classify two phases of the materials within each grain.

5. **Random Walks:** An image can be modeled as a graph, in which each pixel can be viewed as a node and the adjacency between pixels can be viewed as edges between nodes in the graph. The weights of the edges reflect the similarity between neighboring pixels.

   Given user-defined labels for a set of pixels/voxels in the image as seeds, the Random Walks algorithm [25] will release a random walker on the image graph for each

unlabeled pixel/voxel, and will assign a label for this pixel/voxel as the label of the seed which the random walker has the highest probability to reach.

1.2 K-MEANS AND CENTROIDAL VORONOI TESSELLATION

The invention of K-means method can be chased back to 1957. A brief introduction on the history of K-means is given in the following (quoted from http://en.wikipedia.org/wiki/K-means_clustering#cite_note-2):

The term "K-means" was first used by James MacQueen in 1967 [34], while the standard algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation. And it wasn't published outside Bell labs until 1982 [32]. In 1965, E. W. Forgy published essentially the same method, which is why it is sometimes referred to as Lloyd-Forgy [23].

As a data clustering method, K-means targets on partitioning a certain number observations into K clusters, where each observation belongs to the cluster with the closet mean value. The following mathematical formulation briefly describes the idea of K-means.

Given a set of observations $(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n)$, where each observation is a d-dimensional vector, K-means targets at partitioning the n observations into K clusters $S = \{S_1, S_2, \ldots, S_K\}$ ($K \leq n$), and in the meanwhile minimizing the following intra-cluster summation of Euclidean distance squares:

$$\arg\min_{S} \sum_{k=1}^{K} \sum_{\vec{x}_i \in S_k} ||\vec{x}_i - \vec{\mu}_k||_2^2,$$

where $\vec{\mu}_k$ is the mean value of the observations in partition S_k.

The above minimization problem can be solved by the Lloyd algorithm. Generally, the Lloyd algorithm achieves clustering by iteratively alternating between two steps: 1) Assignment Step; and 2) Updating Step. The assignment step assigns each observation to the cluster with the closest cluster center, while the updating step calculates the new cluster centers as the mean values of the observations in the corresponding cluster. The following mathematical formulation briefly describes the Lloyd Algorithm:

1. Assignment step:

$$S_k^{(t)} = \{\vec{x}_i \mid ||\vec{x}_i - \vec{\mu}_k^{(t)}|| \leq ||\vec{x}_i - \vec{\mu}_j^{(t)}||, \forall j \neq k\}$$

2. Updating step:

$$\vec{\mu}_k^{(t+1)} = \frac{\sum_{\vec{x}_i \in S_k^{(t)}} \vec{x}_i}{|S_k^{(t)}|}$$

The Lloyd algorithm provides a heuristic solution for the original NP-hard K-means problem. Here the heuristic solution refers to local optimum of the clustering, where the clustering results may not be unique with respect to different initializations. Figure 1.1 shows this phenomenon.

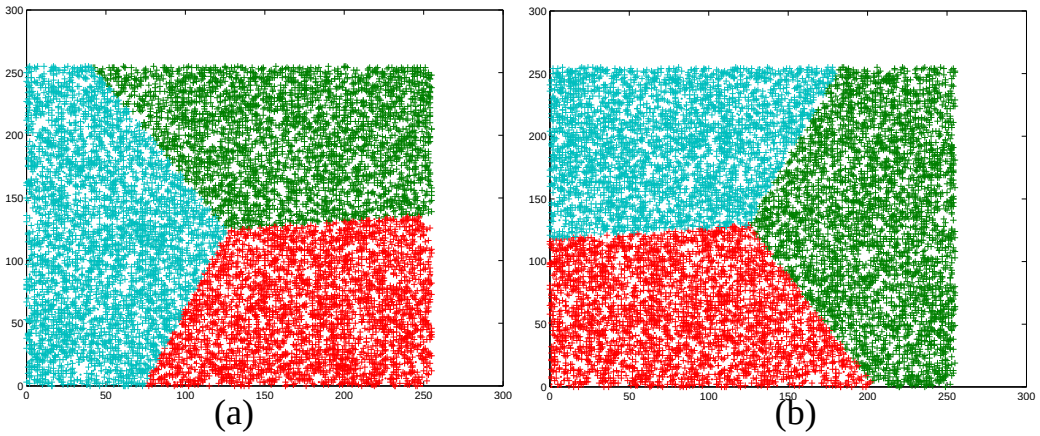


Figure 1.1 Different K-means clustering results with respect to different initializations. The clustering data are 2D integer two-tuples in range [0,255]×[0,255]. These data are clustered into three clusters. The examples are created and visualized using VLFeat [49].

Usually, one can use Forgy method (see [5]) or Random Partition method (see [26]) as the initialization method for K-means. Specifically, Forgy method randomly chooses a certain number of observations as the mean of clusters. While the Random Partition method randomly assigns cluster labels for each observation, and then calculate the mean of each cluster.

Regarding the complexity of the K-means (Lloyd implementation), the problem can be solved in time $O(n^{dK+1} \log n)$, given d as the dimension of observation, K as the number of clusters and n as the number of observations.

In 2003, Charles Elkan proposed an algorithm which uses the triangle inequality to accelerate K-means. The proposed algorithm can avoid unnecessary distance calculations by applying the triangle inequality in two different ways and by keeping track of lower and upper bounds for distances from observations to cluster centers [20].

Centroidal Voronoi Tessellation (CVT) [17] is a general case of the above described K-means method. The CVT is defined based on the concept of Voronoi tessellation (VT), which is further elaborated in the following.

A Voronoi tessellation (VT) is a way of partitioning a space into a number of regions. All points in a partitioned region are closer to a seed (also called generator) inside this region than any other seeds in other partitioned regions. Usually, the measurement for the term "closer" can be defined as distances, such as Euclidean distance, Manhattan distance. The following mathematical formulation further describes the idea of VT.

Let $\mathbb{U}$ be a set of observations, where $\mathbb{U} = \{\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_n\}$ and a distance measure $dist$ is defined on the space $\mathbb{U}$. Let K be a set of indexes and $\vec{w}_k, k = 1, \ldots, K$ are a set of generators. The Voronoi tessellation is then defined as a set of partitions $V_k, k = 1, \ldots, K$, where each partition is defined as

$$V_k = \{\vec{u}_i \in \mathbb{U} \mid dist(\vec{u}_i, \vec{w}_k) \leq dist(\vec{u}_i, \vec{w}_j),\ \forall j \neq k\}$$

A centroidal Voronoi tessellation (CVT) is a special kind of Voronoi tessellation. A Voronoi tessellation is a centroidal Voronoi tessellation when the generator is the center of mass of the voronoi tessellation, namely

$$\vec{w}_k = \frac{\sum_{\vec{u}_i \in V_k} \rho_i \vec{u}_i}{\sum_{\vec{u}_i \in V_k} \rho_i},$$

where ρ_i is a predefined density for $\vec{u}_i$.

The centroidal Voronoi tessllation is a widely used technique for many applications, such as data compression in image processing, optimal quadrature and optimal clustering, etc.

Many algorithms can be used to construct a CVT, including the Lloyd algorithm, probabilistic approaches, descent of gradient methods and Newton-like methods, etc.

In summary, we have introduced some related materials regarding the image segmentation techniques, K-means and centroidal Voronoi tessellation background. In the following chapters, we will introduce how to develop the MCEWCVT and the CMEWCVT algorithms.

CHAPTER 2

MULTICHANNEL EDGE-WEIGHTED CENTROIDAL VORONOI TESSELLATION ALGORITHM

A 3D superalloy image can be denoted as a function u defined on a domain $\Omega \subseteq \mathbb{R}^3$ where the values of u represent the gray intensity of the voxels. Since the voxels can be indexed by integer triplets, u is a discrete function defined over a set of points with integer coordinates, i.e., the point $(x, y, z) = (i, j, k)$, where (i, j, k) is an integer triplet that ranges over the volume domain. Thus, the domain of u for a superalloy volume is an index set $D = \{(i, j, k) \mid i = 1, \ldots, I,\ j = 1, \ldots, J,\ k = 1, \ldots, K\}$ for some positive integers I, J and K.

2.1 CENTROIDAL VORONOI TESSELLATION MODEL FOR SUPERALLOY IMAGES SEGMENTATION

Let $\mathbb{U} = \{u(i, j, k)\}_{(i,j,k)\in D}$ denote the set of intensity values of the 3D superalloy image and $\mathcal{W} = \{w_l\}_{l=1}^{L}$ denote a set of typical intensity levels. The *Voronoi* region V_l ($l = 1, 2, \ldots, L$) in $\mathbb{U}$ corresponding to w_l is defined by

$$\begin{aligned} V_l = &\{u(i, j, k) \in \mathbb{U} \mid |u(i, j, k) - w_l| \leq \\ &|u(i, j, k) - w_m|,\ l = 1, 2, \ldots, L,\ m = 1, 2, \ldots, L,\ l \neq m\}, \end{aligned} \tag{2.1}$$

where $|\cdot|$ could be pre-defined metric measurements such as the Euclidean distance. The set $\mathcal{V} = \{V_l\}_{l=1}^{L}$ is called a *Voronoi tessellation* or *Voronoi clustering* [41, 17, 19] of the set $\mathbb{U}$. The set of chosen intensities $\mathcal{W} = \{w_l\}_{l=1}^{L}$ are referred to as the *Voronoi generators*.

Since we have $V_p \bigcap V_q = \emptyset$ if $p \neq q$ and $\mathbb{U} = \bigcup_{l=1}^{L} V_l$, the Voronoi tessellation $\mathcal{V}$ can be viewed as a special partition of $\mathbb{U}$ in physical space.

Let ρ be a pre-defined density function defined on D. Given a partition of $\mathbb{U}$, denoted by $\{U_l\}_{l=1}^{L}$, the *centroid* (i.e., *center of mass* or *cluster mean*) of each cell U_l, with respect to the density ρ, is defined to be the intensity $w_l^* \in U_l$ which minimizes

$$\min_{w \in U_l} \sum_{u(i,j,k) \in U_l} \rho(i,j,k) |u(i,j,k) - w|^2. \tag{2.2}$$

Note that this centroid definition is slightly different from that used in [53] with the newly introduced density function. For an arbitrary Voronoi tessellation $(\{w_l\}_{l=1}^{L}; \{V_l\}_{l=1}^{L})$ of $\mathbb{U}$, it is often not the case that $w_l = w_l^*$ for $l = 1, 2, \ldots, L$, where $\{w^*\}_{l=1}^{L}$ are the corresponding centroids of $\{V_l\}_{l=1}^{L}$. If the generators of the Voronoi regions $\{V_l\}_{l=1}^{L}$ of $\mathbb{U}$ coincide with their corresponding centroids, i.e.,

$$w_l = w_l^*, \quad \text{for} \quad l = 1, 2, \ldots, L,$$

then we call the Voronoi tessellation $\{V_l\}_{l=1}^{L}$ a centroidal Voronoi tessellation (CVT) [17] of $\mathbb{U}$ and refer to $\{w_l\}_{l=1}^{L}$ as the corresponding CVT generators.

The construction of CVTs can be achieved by an “energy" minimization process [17]. Generally, for any set of generators $\mathcal{W} = \{w_l\}_{l=1}^{L}$ and any partition $\mathcal{U} = \{U_l\}_{l=1}^{L}$ of $\mathbb{U}$, the classic *clustering energy* of $(\mathcal{W}; \mathcal{U})$ can be defined as:

$$E_C(\mathcal{W}; \mathcal{U}) = \sum_{l=1}^{L} \sum_{u(i,j,k) \in U_l} \rho(i,j,k) |u(i,j,k) - w_l|^2. \tag{2.3}$$

Suppose we have determined the clusters $\{U_l\}_{l=1}^{L}$ for a given 3D superalloy image represented by $u(i,j,k)$ for $(i,j,k) \in D$. Then a segmentation in physical space of the image can be naturally produced as $\mathcal{D} = \{D_l\}_{l=1}^{L}$, where

$$D_l = \{(i,j,k) \mid u(i,j,k) \in U_l\}.$$

Note that L is not the number of grains in the underlying 3D superalloy image – each D_l may contain multiple disjoint 3D segments which may represent different grains. In other

words, non-adjacent grains may have identical or similar intensities and be clustered into the same D_l. In practice, we usually choose L to be much smaller than the expected number of grains in the image when using Voronoi tessellation algorithms.

Consequently, the classic clustering energy Equation (2.3) can be rewritten in physical segmentation terminology as

$$E_C(\mathcal{W};\mathcal{D}) = \sum_{l=1}^{L} \sum_{(i,j,k)\in D_l} \rho(i,j,k)|u(i,j,k) - w_l|^2. \tag{2.4}$$

It is well known from [17] that $(\mathcal{W};\mathcal{D})$ is a minimizer of $E_C(\mathcal{W};\mathcal{D})$ only if $(\mathcal{W};\mathcal{D})$ forms a CVT of D. Several algorithms of minimizing the above energy function can be found in [27, 46, 47].

2.2 Multichannel Clustering Energy

Let N denote the number of channels, i.e., we have N images,

$$u^1(i,j,k), u^2(i,j,k), \cdots, u^N(i,j,k),$$

of the same superalloy sample photographed under different microscope settings. Then we can rewrite $\mathbb{U}$ and $\mathcal{W}$ in the vector form

$$\mathbb{U} = \{\vec{u}(i,j,k) = (u^1(i,j,k), u^2(i,j,k), \cdots, u^N(i,j,k))^T \in \mathbb{R}^N\}_{(i,j,k)\in D}$$

and

$$\mathcal{W} = \{\vec{w}_l = (w_l^1, w_l^2, \cdots, w_l^N)^T \in \mathbb{R}^N\}_{l=1}^{L},$$

respectively.

Note that one grain may be visually separable in one channel but has the close intensity value with its adjacent grains in other channels. Thus, we need to choose proper measurement of the distance from $\vec{u}(i,j,k)$ to $\vec{w}_l$ in order to capture the grains with intensities that are distinct with its adjacent grains only in some (but not all) of the channels. In this work, we take the ∞-norm as the distance measurement, which is defined as

$$\|\vec{x}\|_\infty = \max(|x^1|, |x^2|, \ldots, |x^N|)$$

where $\vec{x} = (x^1, x^2, \ldots, x^N) \in \mathbb{R}^N$.

This way, we can rewrite the classic clustering energy Equation (2.4) as

$$E_C(\mathcal{W}; \mathcal{D}) = \sum_{l=1}^{L} \sum_{(i,j,k)\in D_l} \rho(i,j,k) \|\vec{u}(i,j,k) - \vec{w}_l\|_\infty^2. \tag{2.5}$$

Note that Equation (2.5) will reduce to Equation (2.4) when $N = 1$.

2.3 Edge Energy

Besides the multichannel clustering energy term, we also would like to further utilize an edge related energy term which enforces the continuity and smoothness on edges (faces) of the 3D superalloy grains/segments. In this work, we define the edge related energy term for a given clustering $\mathcal{D}$ of the physical space similar to that used in [53].

For each voxel $(i,j,k) \in D$, denote by $\mathbb{N}_\omega(i,j,k)$ a local neighborhood, which could be a $2\omega \times 2\omega \times 2\omega$ cube centered at (i,j,k) or a sphere centered at (i,j,k) with radius ω. We then define a local characteristic function $\chi_{(i,j,k)} : \mathbb{N}_\omega(i,j,k) \to \{0,1\}$ as

$$\chi_{(i,j,k)}(i',j',k') = \begin{cases} 1 & \text{if } \pi_{\vec{u}}(i',j',k') \neq \pi_{\vec{u}}(i,j,k)\ , \\ 0 & \text{otherwise,} \end{cases}$$

where $\pi_{\vec{u}}(i,j,k) : D \to \{1,\ldots,L\}$ tells which cluster $\vec{u}(i,j,k)$ belongs to. The edge energy term can be defined as

$$E_L(\mathcal{D}) = \sum_{(i,j,k)\in D} \sum_{(i',j',k')\in \mathbb{N}_\omega(i,j,k)} \chi_{(i,j,k)}(i',j',k'). \tag{2.6}$$

Generalizing the analysis for 2D cases in [53], it is not too hard to demonstrate that E_L is proportional to $\omega^4 A$ in the asymptotic sense where A is the area of the boundaries (surfaces in the 3D space) between the 3D segments. Note that the edge energy has nothing to do with the generators $\mathcal{W}$.

2.4 TOTAL ENERGY

In order to enforce the clear detection of grain boundaries, we take the density function ρ as

$$\rho(i,j,k) = 1 + |\nabla \vec{u}(i,j,k)|, \tag{2.7}$$

where $\nabla \vec{u}(i,j,k)$ is defined as

$$\vec{u}(i,j,k) = \sqrt{\frac{d_i}{2}^2 + \frac{d_j}{2}^2 + \frac{d_k}{2}^2} \tag{2.8}$$

with

$$\begin{aligned} d_i &= ||\vec{u}(i+1,j,k) - \vec{u}(i-1,j,k)||_\infty, \quad i \neq 1 \text{ and } i \neq I \\ d_j &= ||\vec{u}(i,j+1,k) - \vec{u}(i,j-1,k)||_\infty, \quad j \neq 1 \text{ and } j \neq J \\ d_k &= ||\vec{u}(i,j,k+1) - \vec{u}(i,j,k-1)||_\infty, \quad k \neq 1 \text{ and } k \neq K \end{aligned} \tag{2.9}$$

for the non-boundary cases, and

$$\begin{aligned} d_i &= 2||\vec{u}(2,j,k) - \vec{u}(1,j,k)||_\infty \text{ or } d_i = 2||\vec{u}(I,j,k) - \vec{u}(I-1,j,k)||_\infty \\ d_j &= 2||\vec{u}(i,2,k) - \vec{u}(i,1,k)||_\infty \text{ or } d_j = 2||\vec{u}(i,J,k) - \vec{u}(i,J-1,k)||_\infty \\ d_k &= 2||\vec{u}(i,j,2) - \vec{u}(i,j,1)||_\infty \text{ or } d_k = 2||\vec{u}(i,j,K) - \vec{u}(i,j,K-1)||_\infty \end{aligned} \tag{2.10}$$

for the boundary cases.

By combining Equation (2.5), Equation (2.6) and Equation (2.7), we can define the total energy for our model, i,e., the *multichannel edge-weighted clustering energy*, as:

$$\begin{aligned} E(\mathcal{W};\mathcal{D}) &= E_C(\mathcal{W};\mathcal{D}) + \lambda E_L(\mathcal{D}) \\ &= \sum_{l=1}^{L} \sum_{(i,j,k)\in D_l} (1 + |\nabla \vec{u}(i,j,k)|) \|\vec{u}(i,j,k) - \vec{w}_l\|_\infty^2 \\ &\quad + \lambda \sum_{(i,j,k)\in D} \sum_{(i',j',k')\in \mathbb{N}_\omega(i,j,k)} \chi_{(i,j,k)}(i',j',k'), \end{aligned} \tag{2.11}$$

where λ is a weight parameter to control the balance between E_C and E_L.

Basically, the total multichannel edge-weighted energy minimization is achieved through iteratively transferring voxels from one cluster to another. Specifically, in each iteration, we transfer a voxel to one specific cluster to decrease the total energy as much as possible.

Let us rewrite Equation (2.11) as

$$\begin{aligned} E(\mathcal{W};\mathcal{D}) = & \sum_{(i',j',k')\in D\setminus(i,j,k)} \rho(i',j',k')\|\vec{u}(i',j',k') - \vec{w}_{\pi_{\vec{u}}(i',j',k')}\|_\infty^2 \\ & + \rho(i,j,k)\|\vec{u}(i,j,k) - \vec{w}_{\pi_{\vec{u}}(i,j,k)}\|_\infty^2 \\ & + \sum_{(i',j',k')\in D\setminus(i,j,k)} \epsilon_{\mathcal{L}}(i',j',k') \\ & + \epsilon_{\mathcal{L}}(i,j,k) \end{aligned} \tag{2.12}$$

where

$$\epsilon_{\mathcal{L}}(i,j,k) = \lambda \sum_{(i',j',k')\in\mathbb{N}_\omega(i,j,k)} \chi_{(i,j,k)}(i',j',k').$$

Now let's consider the variation of the total energy when transferring a voxel (i,j,k) from its current cluster D_l to another cluster D_m. In Equation (2.12), the first term on the right side has no change. The change of the second term is

$$\rho(i,j,k)(\|\vec{u}(i,j,k) - \vec{w}_m\|_\infty^2 - \|\vec{u}(i,j,k) - \vec{w}_l\|_\infty^2). \tag{2.13}$$

Denote $n_k(i,j,k)$ the number of voxels within $(D_k \bigcap \mathbb{N}_w(i,j,k))\setminus(i,j,k)$. Based on the analysis from [53], we know that the changes in the third term and fourth term after transferring are both equal to

$$\lambda n_l(i,j,k) - \lambda n_m(i,j,k) = \lambda[n_l(i,j,k) - n_m(i,j,k)]. \tag{2.14}$$

Summarizing Equation (2.13) and Equation (2.14), we have the overall variation of the total energy by transferring voxel (i,j,k) from cluster D_l to D_m as

$$\begin{aligned} & \rho(i,j,k)\Big(\|\vec{u}(i,j,k) - \vec{w}_m\|_\infty^2 - \|\vec{u}(i,j,k) - \vec{w}_l\|_\infty^2\Big) + \\ & 2\lambda(n_l(i,j,k) - n_m(i,j,k)), \end{aligned}$$

which can be rewritten as

$$\begin{aligned}&[\rho(i,j,k)\|\vec{u}(i,j,k)-\vec{w}_m\|_\infty^2-2\lambda n_m(i,j,k)]\\&\quad-[\rho(i,j,k)\|\vec{u}(i,j,k)-\vec{w}_l\|_\infty^2-2\lambda n_l(i,j,k)].\end{aligned}\tag{2.15}$$

Thus we define the *multichannel edge-weighted distance* from a voxel (i,j,k) to a generator $\vec{w}_l$ to be

$$\begin{aligned}&dist(\vec{u}(i,j,k),\vec{w}_l)\\&=\sqrt{\rho(i,j,k)\|\vec{u}(i,j,k)-\vec{w}_l\|_\infty^2+2\lambda\tilde{n}_l(i,j,k)}\\&=\sqrt{(1+|\nabla\vec{u}(i,j,k)|)\|\vec{u}(i,j,k)-\vec{w}_l\|_\infty^2+2\lambda\tilde{n}_l(i,j,k)}\end{aligned}\tag{2.16}$$

where

$$\tilde{n}_l(i,j,k)=|\mathbb{N}_w(i,j,k)|-n_l(i,j,k)-1$$

is the number of voxels in

$$\mathbb{N}_w(i,j,k)\backslash(D_l\cup(i,j,k)).$$

We can notice that the term $|\mathbb{N}_w(i,j,k)|$ is constant given a fixed w (i.e., the size of neighborhood for the edge energy term), and we always have

$$|\mathbb{N}_w(i,j,k)|\geq n_l(i,j,k)+1,$$

which guarantees that

$$\rho(i,j,k)\|\vec{u}(i,j,k)-\vec{w}_l\|_\infty^2+2\lambda\tilde{n}_l(i,j,k)\geq 0.$$

In conclusion, moving a voxel to the cluster of a generator to which it has the shortest edge-weighted distance defined by Equation (2.16) will decrease the total clustering energy $E(\mathcal{W};\mathcal{D})$ the most.

2.6 MULTICHANNEL EDGE-WEIGHTED VORONOI REGIONS

Given a set of multichannel generators $\mathcal{W} = \{\vec{w}_l\}_{l=1}^L$, we can define the multichannel edge-weighted Voronoi region $\tilde{\mathcal{D}} = \{\tilde{D}_l\}_{l=1}^L$ in the physical volume space D as

$$\begin{aligned}\tilde{D}_l =& \{(i,j,k) \in D \mid dist(\vec{u}(i,j,k), \vec{w}_l) \\ & \leq dist(\vec{u}(i,j,k), \vec{w}_m), m = 1, \ldots, L, l = 1, \ldots, L, l \neq m\}.\end{aligned} \tag{2.17}$$

From Equation (2.15) and Equation (2.16), it is easy to find that when $\mathcal{W}$ is fixed, the multichannel edge-weighted Voronoi tessellation $\tilde{\mathcal{D}} = \{\tilde{D}_l\}_{l=1}^L$ associated with $\mathcal{W}$ corresponds to the minimizer of the multichannel edge-weighted energy $E(\mathcal{W}; \mathcal{D})$, i.e.,

$$\tilde{\mathcal{D}} = \arg\min_{\mathcal{D}} E(\mathcal{W}; \mathcal{D}).$$

Then we define the *multichannel edge-weighted Voronoi tessellation energy* for a given set of generators $\mathcal{W} = \{w_l\}_{l=1}^L$ to be

$$E_{MCEWVT}(\mathcal{W}) = E(\mathcal{W}; \tilde{\mathcal{D}}). \tag{2.18}$$

Algorithm 1 can be used to efficiently construct the multichannel edge-weighed Voronoi regions for a given set of generators.

Algorithm 1 : $\{\tilde{D}_l\}_{l=1}^L$ = MCEWVT($\vec{u}$,$\{\vec{w}_l\}_{l=1}^L$,$\{D_l\}_{l=1}^L$)

1: INPUT: A set of N images determined by $\vec{u}$, a set of generators $\{\vec{w}_l\}_{l=1}^L$ and an initial partition $\{D_l\}_{l=1}^L$ of the physical space D (can be arbitrarily chosen).
2: START:
3: **for all** voxels $(i,j,k) \in D$ **do**
4: a) calculate the multichannel edge-weighted distances from the voxel (i,j,k) to all generators $\{\vec{w}_l\}_{l=1}^L$.
5: b) transfer the voxel (i,j,k) from its current cluster to the cluster whose generator has the shortest multichannel edge-weighted distance to it.
6: **end for**
7: If no voxel in the loop is transferred, return $\{\tilde{D}_l\}_{l=1}^L = \{D_l\}_{l=1}^L$ and exit; otherwise, go to 3.

2.7 THE MCEWCVT MODEL AND ITS IMPLEMENTATION

In order to define the MCEWCVT model, we need to further determine the centroids of a given set of partition $\tilde{\mathcal{D}} = \{\tilde{D}_l\}_{l=1}^L$ of D, i.e., find $\mathcal{W} = \{\vec{w}_l^*\}_{l=1}^L$ such that

$$\vec{w}_l^* = \arg\min_{\vec{w}} \sum_{(i,j,k)\in\tilde{D}_l} \rho(i,j,k)\|\vec{u}(i,j,k) - \vec{w}\|_\infty^2 \tag{2.19}$$

for $l = 1, 2, \cdots, L$. Since we use the $\infty-$norm, it is hard to find an analytical solution for $\vec{w}_l$. Usually, the $\vec{w}_l$ defined through the above minimization process could be solved numerically. For example, the Powell method [42] could be used to effectively calculate $\vec{w}_l$ approximately although there is no derivative information available. For an example implementation of Powell method in C, one can refer to [43].

Definition (MCEWCVT) *For a given multichannel edge-weighted Voronoi tessellation* $\left(\{\vec{w}_l\}_{l=1}^L; \{\tilde{D}_l\}_{l=1}^L\right)$ *of* D*, we call it a multichannel edge-weighted centroidal Voronoi tessellation (MCEWCVT) of* D *if the generators* $\{\vec{w}_l\}_{l=1}^L$ *are also the corresponding centroids of the associated multichannel edge-weighted Voronoi regions* $\{\tilde{D}_l\}_{l=1}^L$*, i.e.,*

$$\vec{w}_l = \vec{w}_l^*, \quad l = 1, 2, \cdots, L.$$

Based on the CVT principle [17], we know that $(\mathcal{W}; \tilde{\mathcal{D}})$ is a minimizer of $E(\mathcal{W}; \tilde{\mathcal{D}})$ only if $(\mathcal{W}; \tilde{\mathcal{D}})$ forms a MCEWCVT of D. We propose Algorithm 2 for constructing the MCEWCVTs. As discussed in [53] for the EWCVT construction algorithms, some improvements of Algorithm 2-MCEWCVT can be obtained by using narrow-banded implementation and better initialization processes. We also notice that the energy $E_{MCEWVT}(\mathcal{W})$ keeps decreasing along the iterations in this algorithm.

2.8 COMPARING MCEWCVT WITH THE LLOYD ALGORITHM

The Lloyd algorithm provides a popular implementation to CVT/K-means. The proposed MCEWCVT algorithm and the Lloyd algorithm share similar algorithmic structures.

Algorithm 2 : $(\{\vec{w}_l\}_{l=1}^L, \{\tilde{D}_l\}_{l=1}^L)$ = MCEWCVT($\vec{u}$,L)

1: INPUT: A set of N images determined by $\vec{u}$ and an integer L.
2: START:
3: Arbitrarily choose an initial partition $\{\tilde{D}_l\}_{l=1}^L$ of the physical space D or using clustering methods.
4: For each cluster $\tilde{D}_l, l = 1, \ldots, L$, calculate its cluster centroid $\vec{w}_l^*$.
5: Take $\{\vec{w}_l^*\}_{l=1}^L$ as the generators, calculate an updated partitioning $\{\tilde{D}_l'\}_{l=1}^L$ by using Algorithm 1.
6: If $\{\tilde{D}_l'\}_{l=1}^L$ and $\{\tilde{D}_l\}_{l=1}^L$ are the same, return $(\{\vec{w}_l\}_{l=1}^L; \{\tilde{D}_l\}_{l=1}^L)$ and exit; otherwise, set $\tilde{D}_l = \tilde{D}_l'$ for $l = 1, \ldots, L$ and go to 4.

Specifically, both of them are iterative optimization algorithms where each iteration consists of two steps: 1) assignment step that assigns each data point to the cluster with the closest center; and 2) updating step that updates cluster centers to be the centroids of the data points in each cluster. In addition, both algorithms share a similar convergence property. According to [16, 21], as the clustering energy decreases monotonically, a clustering algorithm must be weakly convergent. Thus, the proposed MCEWCVT algorithm is weakly convergent given its minimization process. In Chapter 4, we will show that the clustering energy of MCEWCVT decreases in the optimization during the segmentation process of superalloy data.

However, there is a major difference between the proposed MCEWCVT algorithm and the Lloyd algorithm. The Lloyd algorithm handles the energy where the L_2 norm is used to measure the distance from a data point to a cluster center. In this way, the Lloyd algorithm updates the cluster centers using mean values of the data in each cluster. In MCEWCVT, we include an edge-smoothness term in the clustering energy and use a multichannel edge-weighted distance based on L_∞ norm. As a result, the MCEWCVT algorithm theoretically needs to use Powell method rather than mean values to update new cluster centers.

2.9 PUBLICATIONS

- A Multichannel Edge-Weighted Centroidal Voronoi Tessellation Algorithm for 3D Superalloy Image Segmentation

Yu Cao, Lili Ju, Qin Zou, Chengzhang Qu and Song Wang

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Spring, CO, 2011

- 3D Superalloy Grain Segmentation Using A Multichannel Edge-Weighted Centroidal Voronoi Tessellation Algorithm

  Yu Cao, Lili Ju, Youjie Zhou and Song Wang

  IEEE Transactions on Image Processing (TIP), Accepted, 2013

CHAPTER 3

CONSTRAINED MULTICHANNEL EDGE-WEIGHTED CENTROIDAL VORONOI TESSELLATION ALGORITHM

3.1 CONSTRAINED SUPERALLOY SEGMENTATION PROBLEM

MCEWCVT discussed in the previous chapter is a fully automatic image segmentation algorithm, which does not utilize prior knowledge from materials science domain. One possible way of using prior knowledge from materials science domain is to introduce convenient human interactions to guide the automatic segmentation.

For the 3D multichannel superalloy grain segmentation problem, we can manually segment a small set of selected 2D slices to help segmenting the other slices in the volume. More specifically, we can still use the MCEWCVT model to segment the entire 3D volume, but subject to additional constraints. For the classic *centroid Voronoi tessellation* (CVT) model (similar to K-means since it only involves the minimization of the intensity clustering energy), a variety of constraints has been proposed to be added into the energy function and the consequent minimization process [11, 33, 18, 52]. In [18], a constrained CVT was proposed for polynomial interpolation and numerical integration on surface. In [11], constraints were incorporated into the classic CVT model to avoid the empty cluster problem. A spatial constraint was used in [33] for hierarchical K-means clustering, which is used for image segmentation. In [52], some constraints from the problem domain were utilized, which lead to a dramatic performance boosting in road lane detection from GPS data. In this chapter, we develop a new algorithm to incorporate the constraints of pre-specified segmentation on selected slices into MCEWCVT.

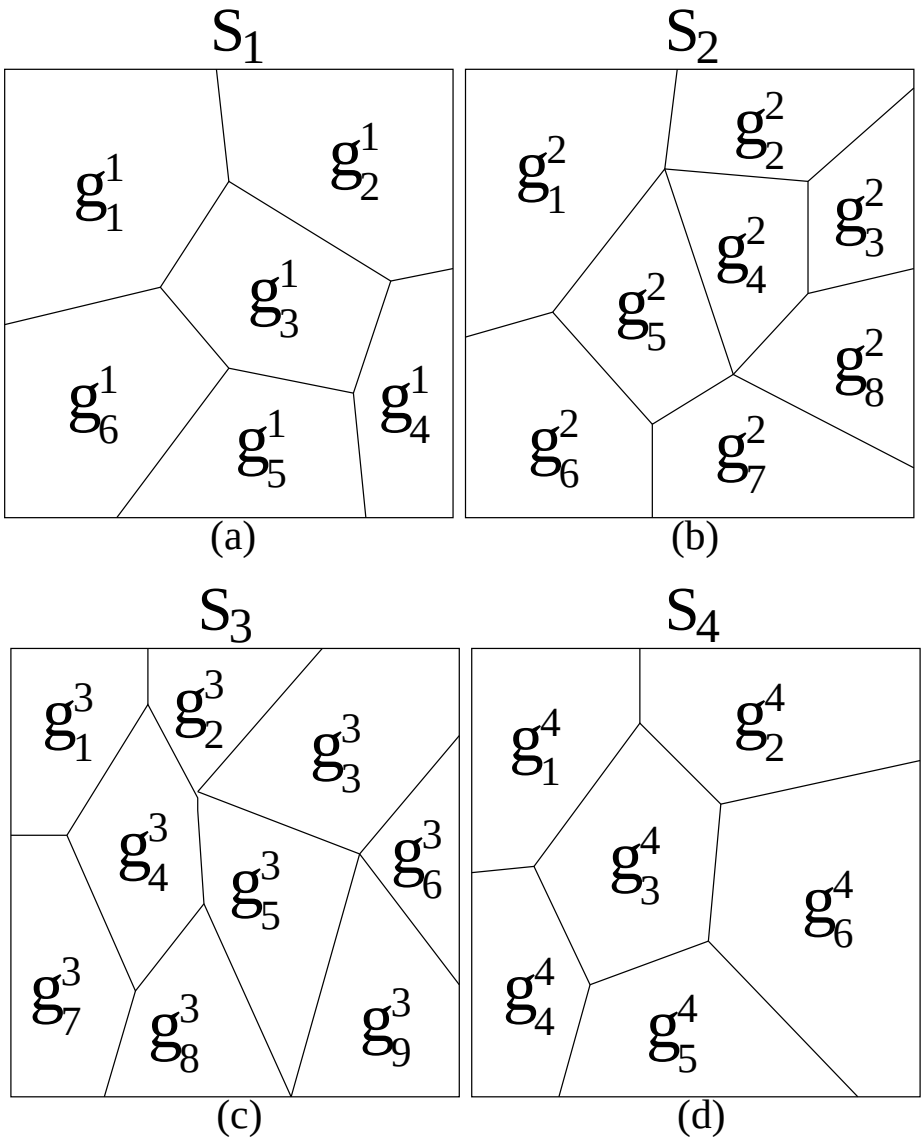


Figure 3.1 An illustration of human segmentation on some selected image slices.

As illustrated in Figure 3.1, 2D human annotated segmentations $\mathcal{S} = \{\mathcal{S}_m\}_{m=1}^{M}$ on M constraint superalloy image slices (the segmentations are annotated by considering all the channels) can be written as $\mathcal{S}_m = \bigcup_{n=1}^{N_m} g_n^m$ where g_n^m denotes the n-th 2D grain region on the m-th constraint superalloy image slice, and N_m is the number of grains on the m-th constraint superalloy image slice according to the annotated segmentation. It satisfies the

following equation

$$g_n^m \cap g_{n'}^{m'} = \emptyset, \quad \text{if } n \neq n'. \tag{3.1}$$

Thus the constraints over the tessellation (or say grouped clusters) $\mathcal{D}$ can be mathematically defined as:

$$\begin{aligned} \mathcal{C}_1(\mathcal{D}) &= \{\Psi(p) = \Psi(q), \ \ \forall\, p, q \in g_n^m\} \\ \mathcal{C}_2(\mathcal{D}) &= \{\Psi(p) \neq \Psi(q), \ \ \forall\, p \in g_n^m, q \in g_{n'}^m, \\ &\qquad g_n^m \text{ and } g_{n'}^m \text{ are adjacent grains}\} \end{aligned} \tag{3.2}$$

where $\Psi(\cdot)$ denotes the generic form of the clustering function which provides a cluster label for a given voxel. As noted in Section 2.1, for the voxels in the non-adjacent grains, they can bear same or different cluster labels and we do not impose any constraints.

We can define the 3D multichannel superalloy image segmentation problem under such human annotation constraints to be a constrained energy minimization problem in the form of

$$(\tilde{\mathcal{W}}; \tilde{\mathcal{D}}) = \arg\min_{(\mathcal{W};\mathcal{D})} E(\mathcal{W}; \mathcal{D}) \text{ , s.t. } \mathcal{C} = \{\mathcal{C}_1(\mathcal{D}) \ , \ \mathcal{C}_2(\mathcal{D})\}. \tag{3.3}$$

We would like to point out that the above defined constraints only take effect on the voxels within the same constraint 2D superalloy slice, but it is expected that those constraints will propagate their effects to other non-constraint slices during the energy minimization process.

In the following, we will develop a *constrained multichannel edge-weighted centroidal Voronoi tessellation* (CMEWCVT) algorithm to solve the above constrained minimization problem. In the algorithm we first enforce the initial clusters to satisfy constraint $\mathcal{C}$. And then the constraint is imposed on the whole energy minimization process.

3.2 Determine An Initial Configuration Satisfying Constraints

Denote the average multichannel intensities of the grains in $\mathcal{S}$ to be

$$\mathbb{U}^c = \{\vec{u}_1^1, \vec{u}_2^1, \cdots, \vec{u}_{N_1}^1, \vec{u}_1^2, \vec{u}_2^2, \cdots, \vec{u}_{N_2}^2, \cdots, \vec{u}_1^M, \cdots, \vec{u}_{N_M}^M\}. \tag{3.4}$$

We first run the K-means on $\mathbb{U}^c$ with a small cluster number using a random initialization. If there exist adjacent grains on the same constraint image slices g_n^m and $g_{n'}^m$ whose average intensities $\vec{u}_n^m$ and $\vec{u}_{n'}^m$ are clustered into a same cluster, we add a new cluster whose generator is either $\vec{u}_n^m$ or $\vec{u}_{n'}^m$ and use the new set of generators to run the K-means on $\mathbb{U}^c$ again. We repeat this process until we obtain a set of clusters which can group $\mathbb{U}^c$ in a way that no adjacent grains on the same constraint image slice belong to the same cluster. We then define the centers of these clusters as the initial generators for the CMEWCVT. See Algorithm 3 for the description of the whole procedure.

Algorithm 3 : ($\mathcal{W}^c$=$\{\vec{w}_l^c\}_{l=1}^L$, $\tilde{\mathcal{D}}^{\mathcal{S}}$=$\{\tilde{D}_l^S\}_{l=1}^L$) = ConstrainedGenerators($\mathbb{U}^c$, $\mathcal{S}$, L_0)

1: INPUT: Human annotated segmentation on certain superalloy image slices $\mathcal{S}$. The average intensities $\mathbb{U}^c$ of grain regions in $\mathcal{S}$. Initial cluster number guess L_0.
2: START:
3: $L = L_0$ and randomly initialized L cluster generators $\mathcal{W} = \{\vec{w}_l\}_{l=1}^L$.
4: Run the classic K-means on $\mathbb{U}^c$ with $\mathcal{W}$ to obtain a clustering of $\mathbb{U}^c$.
5: $\tilde{\mathcal{D}}^{\mathcal{S}} \leftarrow \{\tilde{D}_l^s\}_{l=1}^L$: If $\vec{u}_n^m$ belongs to the cluster with generator $\vec{w}_l$, then all voxels in g_n^m are in $\tilde{D}_l^s$.
6: **if** there exist g_n^m and $g_{n'}^m$, which belongs to the same cluster, for adjacent grains g_n^m and $g_{n'}^m$ **then**
7: $L = L + 1$
8: $\vec{w}_L = \vec{u}_{n'}^m$
9: $\mathcal{W} \leftarrow \{\mathcal{W}, \vec{w}_L\}$
10: Go to 4.
11: **else**
12: $\mathcal{W}^c = \mathcal{W}$
13: Return ($\mathcal{W}^c$, $\tilde{\mathcal{D}}^{\mathcal{S}}$).
14: **end if**

3.3 Constrained Multichannel Edge-Weighted Voronoi Tessellation

In MCEWCVT, the clustering energy is minimized by iteratively transferring each voxel from its current cluster to a cluster to which it has the shortest edge-weighted distance defined by Equation (2.16). In the new CMEWCVT model, we need to constrain the transferring of the voxels between clusters. Specifically, we will force the voxels in the

constraint slices to remain in their initial cluster, and for the voxels in the non-constraint slices, they are allowed to be transferred to new clusters.

Given a set of *constraint generators* $\mathcal{W}^c = \{\vec{w}_l^c\}_{l=1}^L$ (i.e., the output of Algorithm 3) and the corresponding constraints $\mathcal{S}$, we define the *constrained multichannel edge-weighted Voronoi tessellation* (CMEWVT), $\tilde{\mathcal{D}} = \{\tilde{D}_l^c\}_{l=1}^L$ in the physical volume space D as

$$\begin{aligned}\tilde{D}_l^c =&\{(i,j,k) \in D\backslash\mathcal{S} \mid dist(\vec{u}(i,j,k), \vec{w}_l^c) \leq \\ &dist(\vec{u}(i,j,k), \vec{w}_m^c),\ m = 1, \cdots, L\} \ \cup\ \tilde{D}_l^{\mathcal{S}},\end{aligned} \tag{3.5}$$

where (i,j,k) refers to the voxel on the non-constraint image slices. $\tilde{\mathcal{D}}^{\mathcal{S}}$ is given by Algorithm 3. Thus, we can define the constrained multichannel edge-weighted clustering energy as

$$\begin{aligned}E(\mathcal{W}^c;\mathcal{D};\mathcal{S}) &= E_C(\mathcal{W}^c;\mathcal{D};\mathcal{S}) + \lambda E_L(\mathcal{D}) \\ &= \sum_{l=1}^{L} \sum_{(i,j,k)\in\tilde{D}_l^c} (1 + |\nabla\vec{u}(i,j,k)|)\|\vec{u}(i,j,k) - \vec{w}_l^c\|_\infty^2 \\ &\quad + \lambda \sum_{(i,j,k)\in D} \sum_{(i',j',k')\in\mathbb{N}_\omega(i,j,k)} \chi_{(i,j,k)}(i',j',k').\end{aligned} \tag{3.6}$$

From Equation (2.16), it is also easy to find that when $\mathcal{W}^c$ and $\mathcal{S}$ are fixed, the constrained multichannel edge-weighted Voronoi tessellation $\tilde{\mathcal{D}} = \{\tilde{D}_l^c\}_{l=1}^L$ associated with $\mathcal{W}^c$ and $\mathcal{S}$ corresponds to the minimizer of the constrained multichannel edge-weighted clustering energy $E(\mathcal{W}^c;\mathcal{D};\mathcal{S})$, i.e.,

$$\tilde{\mathcal{D}} = \arg\min_{\mathcal{D}} E(\mathcal{W}^c;\mathcal{D};\mathcal{S}).$$

Then we define the *constrained multichannel edge-weighted Voronoi tessellation energy* for a given set of constrained generators $\mathcal{W}^c = \{\vec{w}_l^c\}_{l=1}^L$ to be

$$E_{CMEWVT}(\mathcal{W}^c, \mathcal{S}) = E(\mathcal{W}^c;\tilde{\mathcal{D}};\mathcal{S}). \tag{3.7}$$

Algorithm 4 can be used to effectively construct the constrained multichannel edge-weighed Voronoi tessellation for a given set of constrained generators.

Algorithm 4 : $\{\tilde{D}_l^c\}_{l=1}^L$ = CMEWVT($\vec{u}$,$\{\vec{w}_l^c\}_{l=1}^L$,$\{D_l^c\}_{l=1}^L$, $\mathcal{S}$)

1: INPUT: A 3D N-channel image determined by $\vec{u}$, a set of constrained generators $\{\vec{w}_l^c\}_{l=1}^L$ and an initial constrained partition $\{D_l^c\}_{l=1}^L$ of the physical space D. Human annotated segmentations constraints $\mathcal{S}$.
2: START:
3: **for all** voxels $(i,j,k) \in D$ **do**
4: **if** voxel $(i,j,k) \notin \mathcal{S}$ **then**
5: a) calculate the multichannel edge-weighted distances from the voxel (i,j,k) to all constrained generators $\{\vec{w}_l^c\}_{l=1}^L$.
6: b) transfer the voxel (i,j,k) from its current cluster to the cluster whose generator has the shortest multichannel edge-weighted distance to it.
7: **end if**
8: **end for**
9: If no voxel in the loop is transferred, return $\{\tilde{D}_l^c\}_{l=1}^L = \{D_l^c\}_{l=1}^L$ and exit; otherwise, go to 3.

3.4 The CMEWCVT Model and Its Construction

In order to define the CMEWCVT model, we need to further determine the centroids of a given set of partitions $\tilde{\mathcal{D}} = \{\tilde{D}_l^c\}_{l=1}^L$ of D, i.e., find $\mathcal{W}^c = \{\vec{w}_l^{c*}\}_{l=1}^L$ such that

$$\vec{w}_l^{c*} = \arg\min_{\vec{w}_l^c} \sum_{(i,j,k)\in\tilde{D}_l^c} \rho(i,j,k)\|\vec{u}(i,j,k) - \vec{w}_l^c\|_\infty^2 \tag{3.8}$$

for $l = 1, 2, \cdots, L$. Since we use the $\infty-$norm, it is hard to find an analytical solution for $\vec{w}_l^c$. Usually, the $\vec{w}_l^c$ defined through the above minimization process could be solved numerically. Again, the Powell method could be used to effectively calculate $\vec{w}_l^c$ approximately although there is no derivative information available.

Definition (CMEWCVT) *For a given constrained multichannel edge-weighted Voronoi tessellation* $\left(\{\vec{w}_l^c\}_{l=1}^L; \{\tilde{D}_l^c\}_{l=1}^L; \mathcal{S}\right)$ *of* D*, we call it a constrained multichannel edge-weighted centroidal Voronoi tessellation (CMEWCVT) of* D *if the generators* $\{\vec{w}_l^c\}_{l=1}^L$ *are also the corresponding centroids of the associated constrained multichannel edge-weighted Voronoi regions* $\{\tilde{D}_l^c\}_{l=1}^L$*, i.e.,*

$$\vec{w}_l^c = \vec{w}_l^{c*}, \quad l = 1, 2, \cdots, L.$$

Again based on the CVT principle, we know that $(\mathcal{W}^c; \tilde{\mathcal{D}}; \mathcal{S})$ is a minimizer of $E(\mathcal{W}^c; \tilde{\mathcal{D}}; \mathcal{S})$ only if $(\mathcal{W}^c; \tilde{\mathcal{D}}; \mathcal{S})$ forms a CMEWCVT of D. We propose Algorithm 5 for the construc-

tion of the CMEWCVTs. As discussed in [53] about the EWCVT construction algorithms, some improvements of Algorithm 5-CMEWCVT can be obtained by using narrow-banded implementation. We also note that the energy $E_{CMEWVT}(\mathcal{W}^c, \mathcal{S})$ keeps decreasing along the iterations in this algorithm.

Algorithm 5 : $(\{\vec{w}_l^c\}_{l=1}^L, \{\tilde{D}_l^c\}_{l=1}^L)$ = CMEWCVT($\vec{u}$,L_0,$\mathcal{S}$)

1: INPUT: A 3D N-channel images determined by $\vec{u}$ and initial cluster number guess L_0. Human annotated segmentation on certain superalloy image slices $\mathcal{S}$.
2: START:
3: Construct $\mathbb{U}^c$ using $\mathcal{S}$.
4: $(\{\vec{w}_l^c\}_{l=1}^L, \{\tilde{D}_l^{\mathcal{S}}\}_{l=1}^L)$=ConstrainedGenerators$(\mathbb{U}^c, \mathcal{S}, L_0)$.
5: Construct $\{D_l^c\}_{l=1}^L$ based on $\{\tilde{D}_l^{\mathcal{S}}\}_{l=1}^L$ and Voronoi tessellation on $D\backslash\mathcal{S}$ associated with $\{\vec{w}_l^c\}_{l=1}^L$ under the Euclidean distance in intensity space.
6: Construct constrained multichannel centroidal Voronoi tessellation region $\{\tilde{D}_l^c\}_{l=1}^L$=CMEWVT($\vec{u}$,$\{\vec{w}_l^c\}_{l=1}^L$,$\{D_l^c\}_{l=1}^L$, $\mathcal{S}$).
7: For each constrained cluster $\tilde{D}_l^c, l = 1, \ldots, L$, calculate its cluster centroid $\{\vec{w}_l^{c*}\}_{l=1}^L$.
8: Take $\{\vec{w}_l^{c*}\}_{l=1}^L$ as the generators, determine the corresponding constrained multichannel edge-weighted Voronoi clustering $\{\hat{D}_l^c\}_{l=1}^L$ by using Algorithm 2-CMEWVT, i.e., $\{\hat{D}_l^c\}_{l=1}^L$=CMEWVT($\vec{u}$,$\{\vec{w}_l^{c*}\}_{l=1}^L$,$\{\tilde{D}_l^c\}_{l=1}^L$, $\mathcal{S}$).
9: If $\{\hat{D}_l^c\}_{l=1}^L$ and $\{\tilde{D}_l^c\}_{l=1}^L$ are the same, $\{\vec{w}_l^c\}_{l=1}^L = \{\vec{w}_l^{c*}\}_{l=1}^L$ return $(\{\vec{w}_l^c\}_{l=1}^L; \{\tilde{D}_l^c\}_{l=1}^L)$ and exit; otherwise, set $\tilde{D}_l^c = \hat{D}_l^c$ for $l = 1, \ldots, L$ and go to step 7.

3.5 PUBLICATIONS

- Grain Segmentation of 3D Superalloy Images Using Multichannel EWCVT under Human Annotation Constraints

  Yu Cao, Lili Ju and Song Wang

  European Conference on Computer Vision (ECCV), Firenze, Italy, 2012

CHAPTER 4

EXPERIMENTS ON THE IN100 DATASET

4.1 TESTING DATASET

In this chapter, we test the proposed algorithms on an authentic Ni-based 3D superalloy image dataset. The dataset consists of 4 channels of superalloy slice images, which are taken under different electronic microscope. Each slice was photographed as new facets appearing by keeping abrading the up-front facet of the superalloy sample. As illustrated in Figure 4.1, the size of each 2D slice is 671×671 and the number of slices in each channel is 170. The resolution within a slice is $0.2\mu m$ and resolution between slices is $1\mu m$. Figure 4.2 shows sequentially sectioned images of 4-channels. From the figure, we can found that grains are evoluting from the top row to the bottom row. One can notice that the resolution within 2D slices are 5 times higher than that between adjacent image slices. This fact may not affect 2D segmentation/edge-detection performance, but will affect 3D segmentation performance, such as the proposed algorithms, where the measurement of

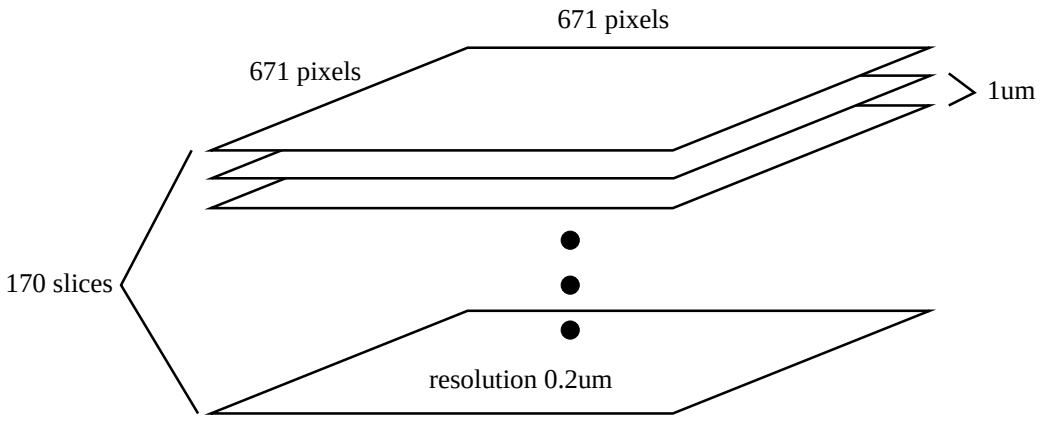


Figure 4.1 An illustration of the testing superalloy volume.

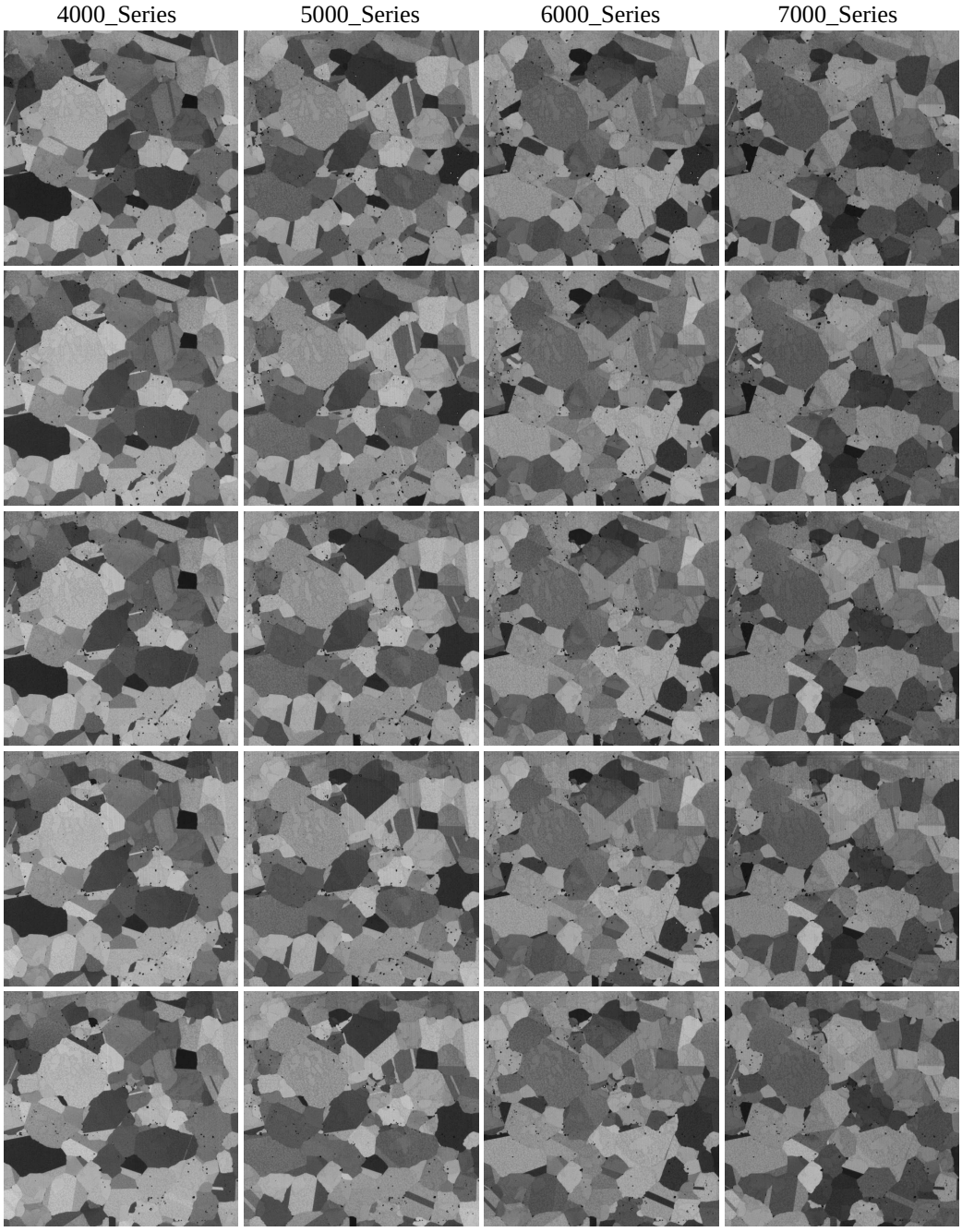


Figure 4.2 An illustration of sequentially sectioned images of 4-channels. Each row corresponds to one 2D slice.

boundary smoothness and segment compactness requires identical image resolutions along all three directions. To address this issue, we linearly interpolate the 3D superalloy image with 4 additional slices between each pair of consecutive slices in the original data and by this way, the interpolated data will contain $169 \times 5 + 1 = 846$ slices. The testing dataset also comes with the ground-truth segmentation created by manual segmentation on each 2D slice. The human annotator considers grain intensities variations from all 4 channels and characterizes unified boundaries of grains for all 4 channels using annotation tools provided in Berkeley Segmentation Benchmark [36]. Considering the computational issue, in the experiments, we downsize the interpolated image to size of $336 \times 336 \times 508$, and re-scale the segmentation results back to 671×671 which are further evaluated against the ground-truth.

4.2 Experiment Designs

In the experiments, we quantitatively evaluate a segmentation result by examining the coincidence between the detected boundaries and the ground-truth grain boundaries. Specifically, we calculate the F_1-measure (using the tool provided in [36]) which is the harmonic mean of precision and recall, i.e.

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}.$$

In the following, we evaluate the MCEWCVT and CMEWCVT algorithms under different parameter settings, and compare the performance with 18 well known 2D/3D segmentation and edge-detection methods quantitatively and qualitatively. For 2D methods, we directly apply them on non-interpolated 170 slices. For 3D methods, we apply them on the interpolated dataset and finally conduct quantitative evaluation only on the 170 original slices against ground-truth segmentation.

4.3 EVALUATION ON MCEWCVT

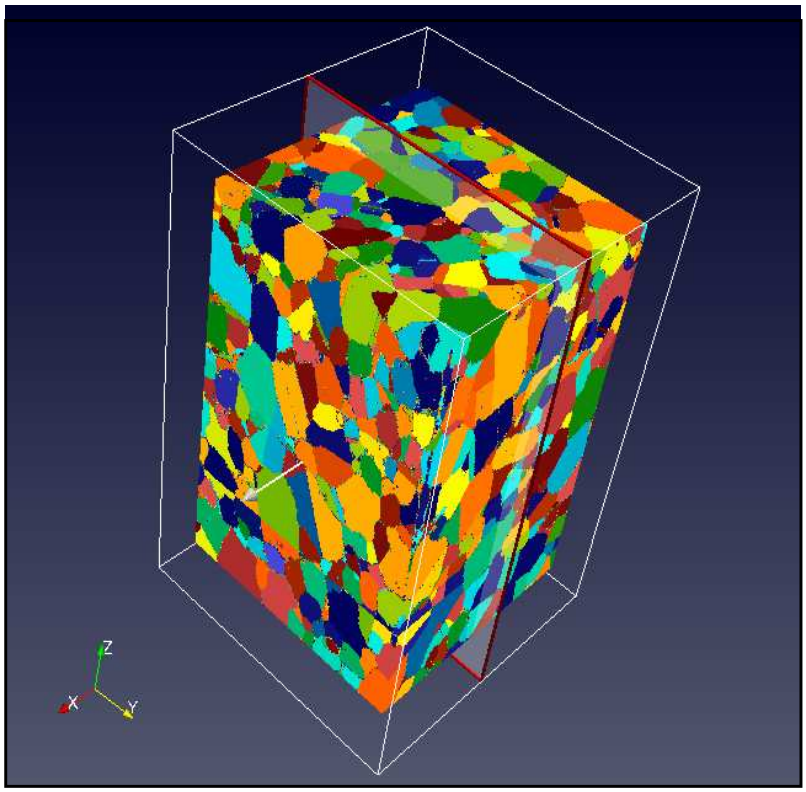
Theoretically, Algorithm 2-MCEWCVT stops when the energy function Equation (2.11) stay unchanged, and this may require a very long running time. A practical stop condition could be

$$\frac{|E^{(t+1)} - E^{(t)}|}{E^{(t)}} < \epsilon. \tag{4.1}$$

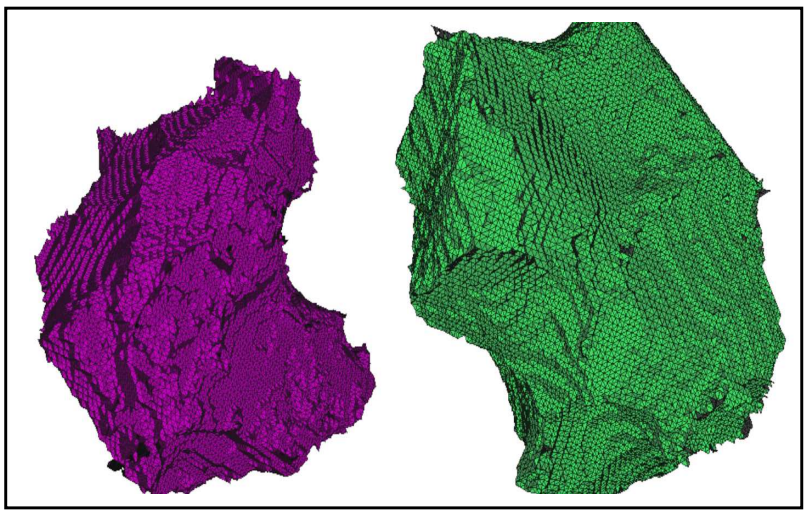
We can also set a maximum allowed iterations $maxIter$ to stop the algorithm. In our experiments, Algorithm 2-MCEWCVT stops when any one of these two stop conditions is triggered.

This way, there are totally 6 parameters/factors that can be tuned in MCEWCVT: L, the number as input for constructing initial clusters; λ, the weighting parameter which balances the intensity clustering energy term E_C and the edge smoothing energy term E_L; ω and $\mathbb{N}_\omega(i,j,k)$, which define the size and shape of a 3D neighborhood region (centered at each voxel (i,j,k)); ϵ and $maxIter$, the pre-defined threshold and maximum iterations as the stop conditions of MCEWCVT.

Enumerating all possible values and combinations of these six parameters/factors would be to some extent infeasible, thus we fix three parameters/factors ($\mathbb{N}_\omega(i,j,k)$, ϵ and $maxIter$), and test different values for the other three parameters (L, λ and ω). Specifically, we set $\epsilon = 10^{-4}$, $maxIter = 140$, and the shape of $\mathbb{N}_\omega(i,j,k)$ to be a sphere with radius ω. The different values tested for L, λ and ω are $L = \{15, 25, 35, 55, 100, 150, 200, 300, 500\}$, $\lambda = \{100, 200, 300, 400, 500\}$, $\omega = \{4, 6, 8\}$, respectively. Table 4.1 summarizes the segmentation accuracy (F_1-measure) under different $\{L, \lambda, \omega\}$. The best result is shaded. From the table, we can find that $L = 300$, $\lambda = 300$ and $\omega = 6$ lead to the highest segmentation accuracy among all tested values for L, λ and ω. The best segmentation result is visualized in Figure 4.3.



(a)



(b)

Figure 4.3 (a) Visualization of the MCEWCVT segmentation result using Paraview [4]. Clusters are represented in different colors. (b) Two segmented grains, visualized using MeshLab [2].

Table 4.1 Segmentation accuracy (F_1-measure) of MCEWCVT under different values of L, λ, and ω. The best result is shaded.

	$(L,\omega)=(15,4)$	$(L,\omega)=(15,6)$	$(L,\omega)=(15,8)$	$(L,\omega)=(25,4)$	$(L,\omega)=(25,6)$	$(L,\omega)=(25,8)$	$(L,\omega)=(35,4)$	$(L,\omega)=(35,6)$	$(L,\omega)=(35,8)$
$\lambda=100$	74.96%	86.81%	89.06%	75.90%	88.20%	90.46%	77.24%	89.27%	91.10%
$\lambda=200$	82.82%	89.34%	87.48%	84.27%	90.63%	89.32%	85.50%	91.30%	90.16%
$\lambda=300$	85.99%	89.19%	84.33%	87.29%	90.72%	86.86%	88.31%	91.34%	87.81%
$\lambda=400$	87.48%	88.63%	80.73%	88.75%	90.31%	83.73%	89.53%	90.98%	84.86%
$\lambda=500$	88.42%	87.83%	76.96%	89.55%	89.61%	80.34%	90.21%	90.44%	81.78%
	$(L,\omega)=(55,4)$	$(L,\omega)=(55,6)$	$(L,\omega)=(55,8)$	$(L,\omega)=(100,4)$	$(L,\omega)=(100,6)$	$(L,\omega)=(100,8)$	$(L,\omega)=(150,4)$	$(L,\omega)=(150,6)$	$(L,\omega)=(150,8)$
$\lambda=100$	77.33%	90.18%	92.05%	80.17%	91.28%	92.70%	81.63%	91.78%	92.95%
$\lambda=200$	86.22%	92.20%	91.22%	87.71%	92.75%	92.02%	88.59%	92.90%	92.22%
$\lambda=300$	89.03%	92.40%	89.09%	90.05%	92.97%	90.06%	90.64%	93.15%	90.37%
$\lambda=400$	90.32%	92.14%	86.35%	91.18%	92.79%	87.69%	91.61%	93.01%	88.05%
$\lambda=500$	91.10%	91.62%	83.56%	91.82%	92.40%	85.08%	92.15%	92.55%	85.72%
	$(L,\omega)=(200,4)$	$(L,\omega)=(200,6)$	$(L,\omega)=(200,8)$	$(L,\omega)=(300,4)$	$(L,\omega)=(300,6)$	$(L,\omega)=(300,8)$	$(L,\omega)=(500,4)$	$(L,\omega)=(500,6)$	$(L,\omega)=(500,8)$
$\lambda=100$	82.66%	92.02%	92.99%	84.06%	92.44%	93.42%	85.46%	92.46%	93.46%
$\lambda=200$	88.90%	93.08%	92.47%	89.58%	93.41%	92.74%	90.03%	93.34%	93.02%
$\lambda=300$	90.76%	93.26%	90.66%	91.13%	93.57%	90.79%	91.29%	93.53%	90.98%
$\lambda=400$	91.64%	93.13%	88.46%	91.88%	93.51%	88.66%	91.84%	93.44%	88.85%
$\lambda=500$	92.21%	92.73%	86.02%	92.23%	93.14%	86.47%	92.24%	93.15%	86.53%

Table 4.2 Segmentation accuracy (F_1-measure) with respect to 10 CVT random initializations, using parameters $\{L=300, \lambda=300, \omega=6\}$.

Initialization	#1	#2	#3	#4	#5
F_1-measure	93.5444%	93.5437%	93.5443%	93.5441%	93.5425%
Initialization	#6	#7	#8	#9	#10
F_1-measure	93.5413%	93.5425%	93.542%	93.5463%	93.5447%

4.4 Robustness of MCEWCVT to Random CVT Initializations

As mentioned above, we use CVT to initialize the clusters for the proposed MCEWCVT algorithm. Since CVT randomly chooses initial cluster centers, we would like to further investigate the robustness of the MCEWCVT to random CVT initializations. In this section, we choose parameters which lead to the best segmentation accuracy, i.e. $\{L=300, \lambda=300, \omega=6\}$, to evaluate the MCEWCVT algorithm based on 10 CVT initializations (all using $L=300$). The segmentation accuracies with respect to these 10 CVT initializations are presented in Table 4.2, from which we can see that the MCEWCVT algorithm is very robust to random CVT initializations.

4.5 Convergence Property of MCEWCVT

We previously mentioned in Section 2.8 that the proposed MCEWCVT algorithm is weakly convergent if its clustering energy decreases monotonically. In the experiments, we recorded the total clustering energies at each iterations. We found that as the MCEWCVT algorithm

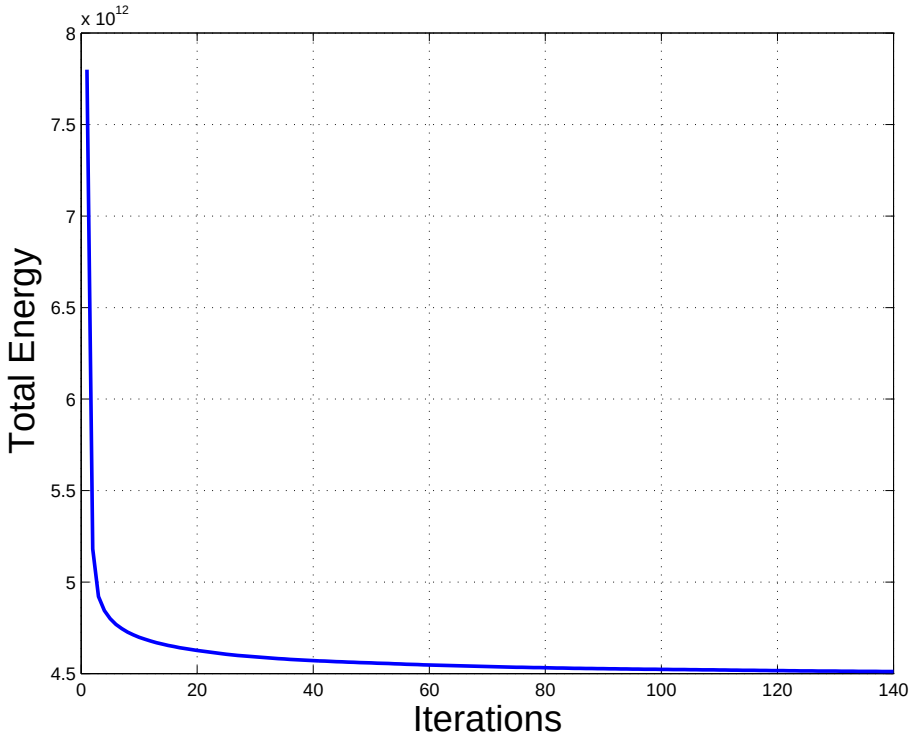


Figure 4.4 Energy decreasing as the MCEWCVT algorithm performs iterative minimization.

going through more iterations, the total clustering energy decreases as designed (see the definition of multichannel edge-weighted distance). This decreasing trend of clustering energy is illustrated in Figure 4.4. From the figure, we can notice an elbow-shape energy decreasing curve. The clustering energy decreases significantly at the beginning iterations and then mildly.

In addition, we found that the proposed CMEWCVT algorithm also shares the same weakly convergent property.

4.6 Parallel Computing Implementation

To evaluate the computing speed of the proposed MCEWCVT algorithm, we tested two computing models: 1) single CPU computing; and 2) parallel computing. For both cases, we test the MCEWCVT algorithm on a Linux cluster machine with 48 AMD Opteron(TM)

Processor 6234 at 2.4GHz, 256GB physical memories.

In the single CPU computing model, the proposed MCEWCVT algorithm takes about 72 hours (140 iterations) to segment all 170 slices. While this running time is long, it is still much more efficient than manual segmentation (2D segmentation on each slice followed by corresponding 2D segments across slices to construct a 3D segmentation), which usually takes weeks.

In the parallel computing model, we utilize OpenMP® [3] to speed up the computation based on CPU parallelism. Notice that in the Algorithm 1-MCEWVT, for each voxel (i, j, k), steps 4 and 5 first calculate the multichannel edge-weighted distances from this voxel to each generator, then transfer this voxel from it current cluster to the cluster whose generator has the shortest multichannel edge-weighted distance to it. The edge-weighted distance calculations of different voxels are totally independent, since the generators will only be updated after all the voxels have been enumerated. Thus based on OpenMP parallel implementation, the voxels in the testing image volume is divided into a certain number of blocks and processed by multiple processors. In the experiments, on the same cluster machine that used in the single CPU computing model, we utilize 40 processors and reduce the computation time to 2.5 hours. This new computation time indicates that the OpenMP parallel implementation for the proposed MCEWCVT algorithm can speed up the computation near-linearly.

In addition, we also apply the OpenMP based parallel implementation to the proposed CMEWCVT algorithm, and achieve a similar near-linear computational speed up.

4.7 Evaluation on CMEWCVT

In evaluating CMEWCVT, we use the same parameter settings of

$$\{L, \lambda, \omega, \mathbb{N}_\omega(i, j, k), \epsilon, maxIter\}$$

Table 4.3 Segmentation performance of CMEWCVT under different L, λ, ω, and $\mathcal{S}$. The top three performances are shaded in red, green, and blue.

	$(L,\omega)=(5,4)$	$(L,\omega)=(5,6)$	$(L,\omega)=(5,8)$	$(L,\omega)=(15,4)$	$(L,\omega)=(15,6)$	$(L,\omega)=(15,8)$	$(L,\omega)=(25,4)$	$(L,\omega)=(25,6)$	$(L,\omega)=(25,8)$
$(\lambda,M)=(100,3)$	79.60%/79.88%	90.69%/90.79%	90.30%/90.41%	78.91%/79.20%	89.92%/90.03%	90.21%/90.32%	79.61%/79.88%	90.23%/90.34%	90.68%/90.78%
$(\lambda,M)=(100,6)$	81.52%/82.25%	91.59%/91.87%	92.17%/92.42%	82.27%/82.96%	91.64%/91.92%	92.01%/92.27%	82.20%/82.90%	91.64%/91.92%	92.09%/92.34%
$(\lambda,M)=(100,9)$	83.22%/84.19%	92.14%/92.53%	92.66%/93.01%	82.70%/83.71%	91.88%/92.28%	92.59%/92.94%	82.78%/83.79%	91.69%/92.11%	92.51%/92.87%
$(\lambda,M)=(100,12)$	83.03%/84.34%	91.98%/92.52%	92.96%/93.41%	83.04%/84.36%	92.03%/92.56%	92.92%/93.37%	82.48%/83.84%	92.04%/92.57%	92.73%/93.20%
$(\lambda,M)=(100,15)$	83.33%/84.94%	92.36%/93.00%	93.06%/93.61%	83.65%/85.22%	92.51%/93.13%	78.66%/80.56%	83.25%/84.86%	92.44%/93.07%	93.09%/93.64%
$(\lambda,M)=(100,18)$	83.84%/85.70%	92.37%/93.13%	93.16%/93.81%	84.29%/86.09%	92.58%/93.32%	93.19%/93.84%	83.83%/85.69%	92.23%/93.00%	93.07%/93.73%
$(\lambda,M)=(100,21)$	83.91%/86.06%	92.44%/93.31%	81.78%/84.02%	84.35%/86.44%	92.62%/93.47%	93.22%/93.97%	84.19%/86.30%	92.32%/93.21%	93.18%/93.94%
$(\lambda,M)=(200,3)$	87.29%/87.45%	90.91%/91.01%	83.21%/83.41%	86.86%/87.02%	90.90%/91.00%	86.26%/86.42%	87.22%/87.38%	90.96%/91.05%	86.76%/86.92%
$(\lambda,M)=(200,6)$	88.45%/88.87%	92.64%/92.87%	89.01%/89.38%	89.00%/89.39%	92.69%/92.92%	89.15%/89.52%	89.16%/89.55%	92.58%/92.82%	89.34%/89.69%
$(\lambda,M)=(200,9)$	89.22%/89.79%	92.88%/93.22%	91.00%/91.44%	89.07%/89.66%	92.85%/93.19%	90.69%/91.15%	89.18%/89.76%	92.66%/93.01%	90.72%/91.18%
$(\lambda,M)=(200,12)$	89.17%/89.94%	93.15%/93.59%	91.51%/92.06%	89.23%/90.00%	93.18%/93.61%	91.48%/92.04%	88.89%/89.69%	92.95%/93.40%	91.54%/92.09%
$(\lambda,M)=(200,15)$	89.81%/90.71%	93.44%/93.96%	92.08%/92.72%	89.73%/90.64%	93.46%/93.98%	78.66%/80.57%	89.68%/90.60%	93.40%/93.92%	91.94%/92.59%
$(\lambda,M)=(200,18)$	89.82%/90.89%	93.44%/94.06%	92.14%/92.90%	90.14%/91.17%	93.58%/94.19%	92.30%/93.04%	89.67%/90.76%	93.30%/93.94%	92.30%/93.04%
$(\lambda,M)=(200,21)$	89.97%/91.20%	93.37%/94.11%	81.78%/84.03%	90.26%/91.45%	93.52%/94.24%	92.34%/93.20%	90.10%/91.31%	93.36%/94.09%	92.35%/93.20%
$(\lambda,M)=(300,3)$	89.45%/89.58%	88.78%/88.91%	73.98%/74.33%	89.29%/89.41%	89.92%/90.03%	79.68%/79.94%	89.61%/89.73%	90.33%/90.44%	79.98%/80.23%
$(\lambda,M)=(300,6)$	90.60%/90.92%	92.23%/92.48%	83.79%/84.37%	91.01%/91.32%	92.25%/92.50%	84.34%/84.90%	90.76%/91.08%	92.20%/92.45%	84.18%/84.75%
$(\lambda,M)=(300,9)$	91.09%/91.55%	92.82%/93.16%	87.75%/88.39%	90.87%/91.34%	92.66%/93.01%	87.40%/88.06%	90.80%/91.27%	92.79%/93.13%	87.17%/87.84%
$(\lambda,M)=(300,12)$	91.46%/92.04%	93.19%/93.62%	89.20%/89.94%	91.42%/92.00%	93.13%/93.57%	89.03%/89.77%	91.14%/91.75%	92.99%/93.43%	89.00%/89.75%
$(\lambda,M)=(300,15)$	91.67%/92.38%	93.35%/93.87%	89.95%/90.80%	91.73%/92.43%	93.44%/93.95%	78.66%/80.56%	91.73%/92.43%	93.31%/93.83%	89.76%/90.63%
$(\lambda,M)=(300,18)$	91.72%/92.56%	93.46%/94.08%	90.29%/91.26%	92.00%/92.80%	93.54%/94.15%	90.27%/91.24%	91.61%/92.46%	93.36%/93.98%	90.45%/91.41%
$(\lambda,M)=(300,21)$	91.64%/92.63%	93.59%/94.30%	81.78%/84.02%	91.97%/92.91%	93.48%/94.20%	90.40%/91.52%	91.68%/92.67%	93.38%/94.11%	90.41%/91.53%
$(\lambda,M)=(400,3)$	90.54%/90.65%	86.43%/86.59%	65.53%/66.01%	90.25%/90.36%	88.29%/88.42%	72.54%/72.91%	90.47%/90.57%	89.14%/89.26%	72.66%/73.02%
$(\lambda,M)=(400,6)$	91.71%/91.99%	91.07%/91.36%	78.93%/79.72%	91.89%/92.16%	91.33%/91.61%	79.15%/79.93%	91.73%/92.00%	91.21%/91.49%	78.96%/79.75%
$(\lambda,M)=(400,9)$	92.16%/92.55%	92.30%/92.67%	84.49%/85.33%	91.90%/92.30%	92.22%/92.59%	84.20%/85.06%	91.67%/92.09%	92.16%/92.54%	84.12%/84.98%
$(\lambda,M)=(400,12)$	92.03%/92.57%	92.68%/93.15%	86.69%/87.62%	92.30%/92.81%	92.51%/92.99%	86.43%/87.39%	92.11%/92.63%	92.58%/93.06%	86.33%/87.29%
$(\lambda,M)=(400,15)$	92.51%/93.13%	93.02%/93.58%	87.72%/88.78%	92.56%/93.18%	93.09%/93.63%	78.66%/80.56%	92.66%/93.27%	92.88%/93.45%	87.57%/88.65%
$(\lambda,M)=(400,18)$	92.46%/93.21%	93.06%/93.72%	88.28%/89.48%	92.69%/93.41%	93.21%/93.86%	88.34%/89.54%	92.39%/93.15%	93.03%/93.69%	88.35%/89.55%
$(\lambda,M)=(400,21)$	92.46%/93.34%	93.40%/94.13%	81.78%/84.02%	92.62%/93.47%	93.16%/93.91%	88.34%/89.74%	92.38%/93.27%	93.10%/93.86%	88.41%/89.80%
$(\lambda,M)=(500,3)$	90.96%/91.06%	83.78%/83.98%	59.60%/60.18%	90.56%/90.67%	86.19%/86.35%	66.84%/67.29%	90.81%/90.91%	87.18%/87.32%	66.79%/67.23%
$(\lambda,M)=(500,6)$	92.25%/92.51%	89.92%/90.25%	74.37%/75.36%	92.26%/92.51%	90.22%/90.55%	75.07%/76.02%	91.99%/92.25%	90.13%/90.46%	75.05%/76.01%
$(\lambda,M)=(500,9)$	92.55%/92.92%	91.60%/92.01%	81.30%/82.34%	92.45%/92.82%	91.43%/91.85%	81.00%/82.05%	92.22%/92.60%	91.26%/91.69%	81.17%/82.21%
$(\lambda,M)=(500,12)$	92.55%/93.04%	91.95%/92.47%	84.35%/85.47%	92.71%/93.19%	91.95%/92.47%	83.97%/85.13%	92.54%/93.04%	91.98%/92.50%	83.93%/85.09%
$(\lambda,M)=(500,15)$	92.96%/93.54%	92.56%/93.15%	85.52%/86.81%	92.99%/93.57%	92.44%/93.05%	78.66%/80.56%	93.13%/93.69%	92.34%/92.96%	85.33%/86.63%
$(\lambda,M)=(500,18)$	92.94%/93.63%	92.68%/93.38%	86.23%/87.68%	93.08%/93.75%	92.78%/93.47%	86.32%/87.76%	92.77%/93.47%	92.65%/93.36%	86.30%/87.74%
$(\lambda,M)=(500,21)$	92.83%/93.65%	92.96%/93.74%	81.79%/84.03%	93.04%/93.83%	92.67%/93.49%	86.33%/88.01%	92.84%/93.66%	92.69%/93.50%	86.42%/88.08%

as in the MCEWCVT evaluation. Besides, we uniformly selected M slices from the initial 170 slices as the constraint slices, i.e., taking their ground-truth manual segmentation as constraints. In our experiments, we try $M=\{3,6,9,12,15,18,21\}$, respectively.

Table 4.3 shows average F_1-measures under different parameter choices for $\{L,\lambda,\omega,M\}$. There are two F_1-measures shown for each parameter combination. The first measure, say Case I, only counts the $(170-M)$ non-constraint slices in calculating the average F_1-measure; while the second measure, say Case II, counts all 170 slices in calculating the average F_1-measure. From the results shown in this table, we can see that 1) more constraint slices usually lead to better performance; 2) The top three performances (backgrounded in red, green and blue, respectively) are achieved at $\lambda=\{200,300\}$ and $\omega=6$, which is consistent to the results in evaluating MCEWCVT; 3) We can achieve good performance by choosing $L=5$. The reason is that, the CMEWCVT starts with a small L and then increase L adaptively to interpret the manual segmentations on the constraint slices; 4) Even though the CMEWCVT quantitatively outperforms the MCEWCVT, the MCEWCVT has

Table 4.4 The number of clusters initialized by the CMEWCVT algorithm.

	$M = 3$	$M = 6$	$M = 9$	$M = 12$	$M = 15$	$M = 18$	$M = 21$
$L = 5$	89	163	212	223	277	304	319
$L = 15$	88	183	201	226	280	317	330
$L = 25$	84	179	202	221	262	299	326

a segmentation accuracy close to that of CMEWCVT. With this experiment results, we argue that CMEWCVT is still superior to the MCEWCVT algorithm. This is because the MCEWCVT high performance depends on parameter searchings on three parameters, i.e., L, ω, λ. In contrast, the CMEWCVT only needs a small initialization of L (e.g., $L = 5$), then the CMEWCVT will automatically determine a proper cluster number and the correspondence generator. This would reduce the number of parameters that need to be chosen in the real application.

Table 4.4 shows the number of clusters initialized by the CMEWCVT algorithm. Form Table 4.3, we can notice that the CMEWCVT algorithm achieves its best performance when using parameters $L = 5, M = 21$. The number of clusters initialized by the CMEWCVT algorithm on these two parameters is 319, which can be found in Table 4.4. Not surprisingly, this discovery is consistent with the best performance found in the MCEWCVT testing which is obtained based on 300 clusters.

Figure 4.5 shows the performance improvement on each slice when taking different number of slices as constraints. In this experiment, we fix $\{L = 5, \lambda = 300, \omega = 6\}$. We can see that, with more constraint slices, the performance on each non-constraint slice is generally increased. This further indicates that the proposed CMEWCVT algorithm takes the prior knowledge implied in the constraint-slice segmentation, and uses it to guide segmenting the other non-constraint slices. In this figure, almost all the slices have the F_1-measure around 0.8 or even higher except for the 40-th slice which has the F_1-measure around 0.7. This is due to the imaging quality problem as illustrated in Figure 4.6, from which we can see that the images from all four channels are polluted by many strong white

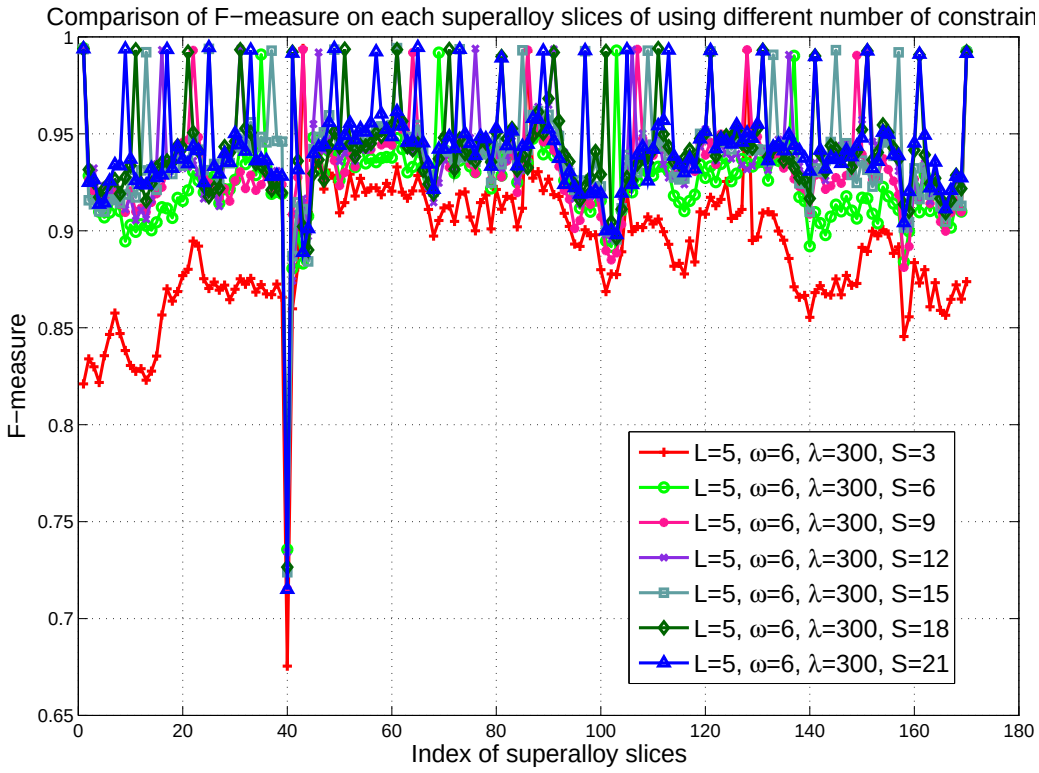


Figure 4.5 F_1-measure computed from each superalloy image slice when using different number of constraint slices.

stripes.

Figure 4.7 and Figure 4.8 show a 3D visualization the CMEWCVT segmentation using parameters $\{L = 5, \lambda = 300, \omega = 6, M = 21\}$ which corresponds to the best performance in Table 4.3.

4.8 Comparison to Existing 2D/3D Segmentation Methods

In order to further validate the proposed MCEWCVT and CMEWCVT algorithm, we compare it with 18 conventional 2D/3D image segmentation methods. Largely based on the code's availability, we choose eight 2D automatic segmentation methods, including 2D level set [30], mean shift [13], watershed [37], statistical region merging (SRM) [40], normalized cuts [45], efficient-graph based segmentation [22], topological watersheds [9], EM/MPM [12]; two semi-automatic 2D segmentation methods, including power water-

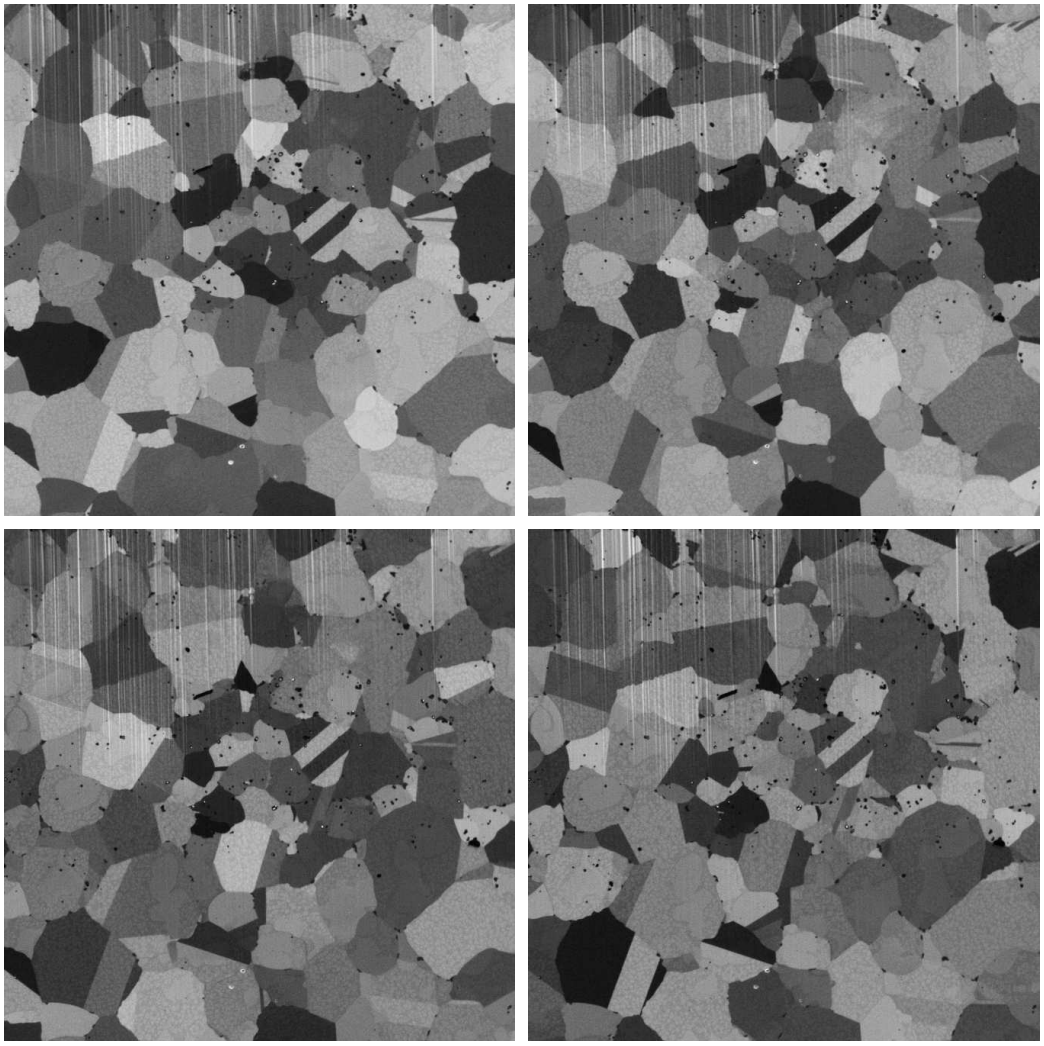


Figure 4.6 Superalloy image slice with strong imaging noises.

sheds [14], random walks [25]; five 2D soft edge-detection methods, including globalized probability of boundary (gpb) [6], Berkeley brightness/color/texture gradient detectors (pbCanny, pbCBTG and pbBGTG) [36], ultrametric contour maps (UCM) [7]; three 3D segmentation methods, including 3D watershed [30], 3D level set [37] and CVT/K-means (directly applied on 3D images as in the proposed methods). Specifically, for 2D methods, we only apply them onto 170 non-interpolated 2D slices. For 3D methods, we apply them to the interpolated 3D superalloy image and then project 3D segmentation boundaries onto 170 non-interpolated slices for quantitative evaluation. Note that gpb, pbCanny, pbCBTG

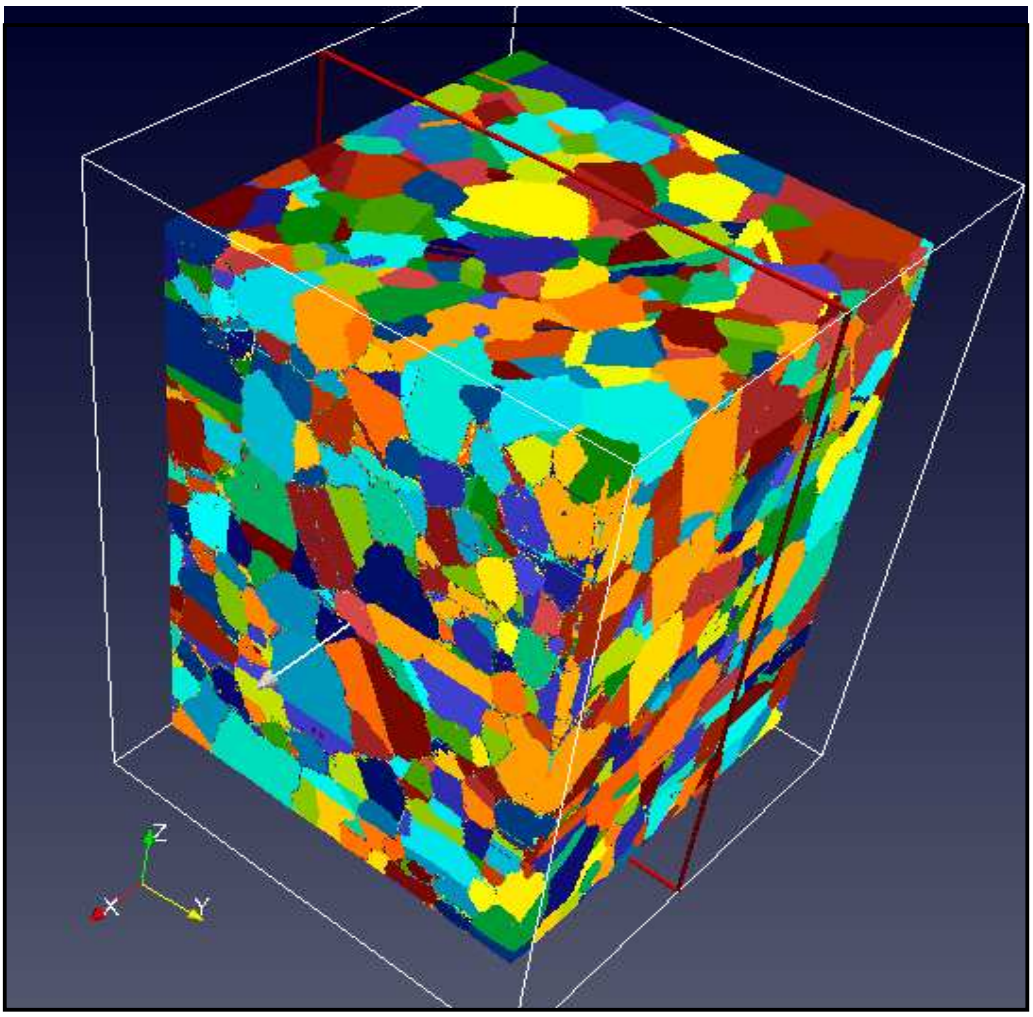


Figure 4.7 Visualization of the CMEWCVT segmentation result using Paraview [4]. Clusters are represented with different colors.

and pbBGTB are soft edge detection algorithms – they only detect disjoint edges and may not produce complete boundaries to partition an image into separate grains.

Note that most of these segmentation methods are not developed for multichannel imaging. While some of them can incorporate multi-dimensional color information, they usually use L_2 norm for defining the distance in the color space. However, for multichannel superalloy images (4 channels in our experiments), it is not clear that how to adapt the comparison methods on the multichannel images, and so as the later quantitative evaluation. In this work, we propose two set of schemes of adapting the comparison methods to the

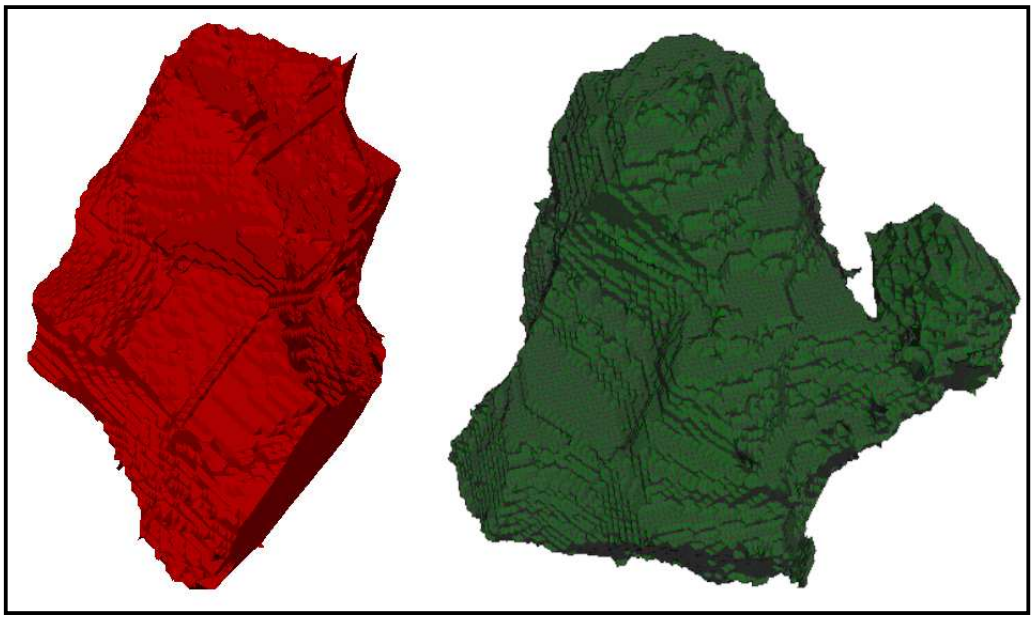


Figure 4.8 Two segmented grains, visualized using MeshLab [2].

multichannel images, i.e., *Scheme-I* and *Scheme-II*, which are elaborated in the following.

1. **Scheme-I:** We applied these comparison methods on each of the 4 channels independently, and then combine the segmentation results from all 4 channels to get a unified segmentation. Specifically, for soft edge detectors, for each pixel we take the maximum probability of boundary values over all 4 channels. For the other comparison methods, we combine the segmentations from all 4 channels using a logic OR operation. For semi-automatic comparison methods which need seeds, we give them special favors – a seed is put at the center of each grain extracted from ground-truth manual segmentation.

2. **Scheme-II:** We take the 4 image channels u_1, u_2, u_3 and u_4 and further create two synthesized channels as

$$u_5 = \frac{u_1 + u_2 + u_3 + u_4}{4}$$

and

$$u_6 = \sqrt{\frac{u_1^2 + u_2^2 + u_3^2 + u_4^2}{4}}.$$

Table 4.5 Segmentation accuracy (F_1-measure) of the CVT algorithm using different cluster number L. The best result is shaded.

$L=2$	$L=3$	$L=4$	$L=5$	$L=10$	$L=15$	$L=20$	$L=25$	$L=30$
60.36%	66.99%	65.62%	60.38%	60.25%	56.4%	53.74%	53.53%	52.96%
$L=40$	$L=50$	$L=60$	$L=100$	$L=150$	$L=200$	$L=300$	$L=500$	$L=800$
57.24%	56.02%	53.96%	54.54%	51.48%	49.04%	46.69%	42.28%	38.12%

For all the comparison methods except for CVT, we apply them to segment each of these six channels independently. In this way, for each comparison method, we obtain six different segmentation results on the testing 3D superalloy image. They are evaluated against the ground-truth segmentation independently, and the result with the best F_1-measure (obtained from one of these six channels) is finally selected as the segmentation result. We use this best F_1-measure as the segmentation accuracy for this comparison method.

In the following experiments, we will elaborate the quantitative and qualitative results for the above two evaluation schemes.

First of all, we evaluate the performance of the classical CVT/K-means algorithm since the proposed MCEWCVT can be treated as an extension of the CVT/K-means algorithm. The classical CVT algorithm does not consider the edge related energy term. In our experiment, we simply use it to do clustering in the multichannel intensity space with Euclidean distance. We try the cluster number of L in {2, 3, 4, 5, 10, 15, 20, 25, 30, 40, 50, 60, 100, 150, 200, 300, 500, 800}. From Table 4.5, we can see that as the cluster number increases, the segmentation accuracy first increases and then decreases. The best segmentation accuracy appears at $L = 3$ (shaded). In general, without the boundary-smoothness term, CVT is sensitive to image noise and this gets more severe when the cluster number is large.

We further evaluate the rest 17 comparison methods whose segmentation performances are shown (precision-recall curves for soft-edge detectors, precision-recall points for the rest) in Figure 4.9 (using Scheme-I, an enlarged version is shown in Figure 4.10) and Figure 4.11 (using Scheme-II). The parameters' selection for these comparison methods

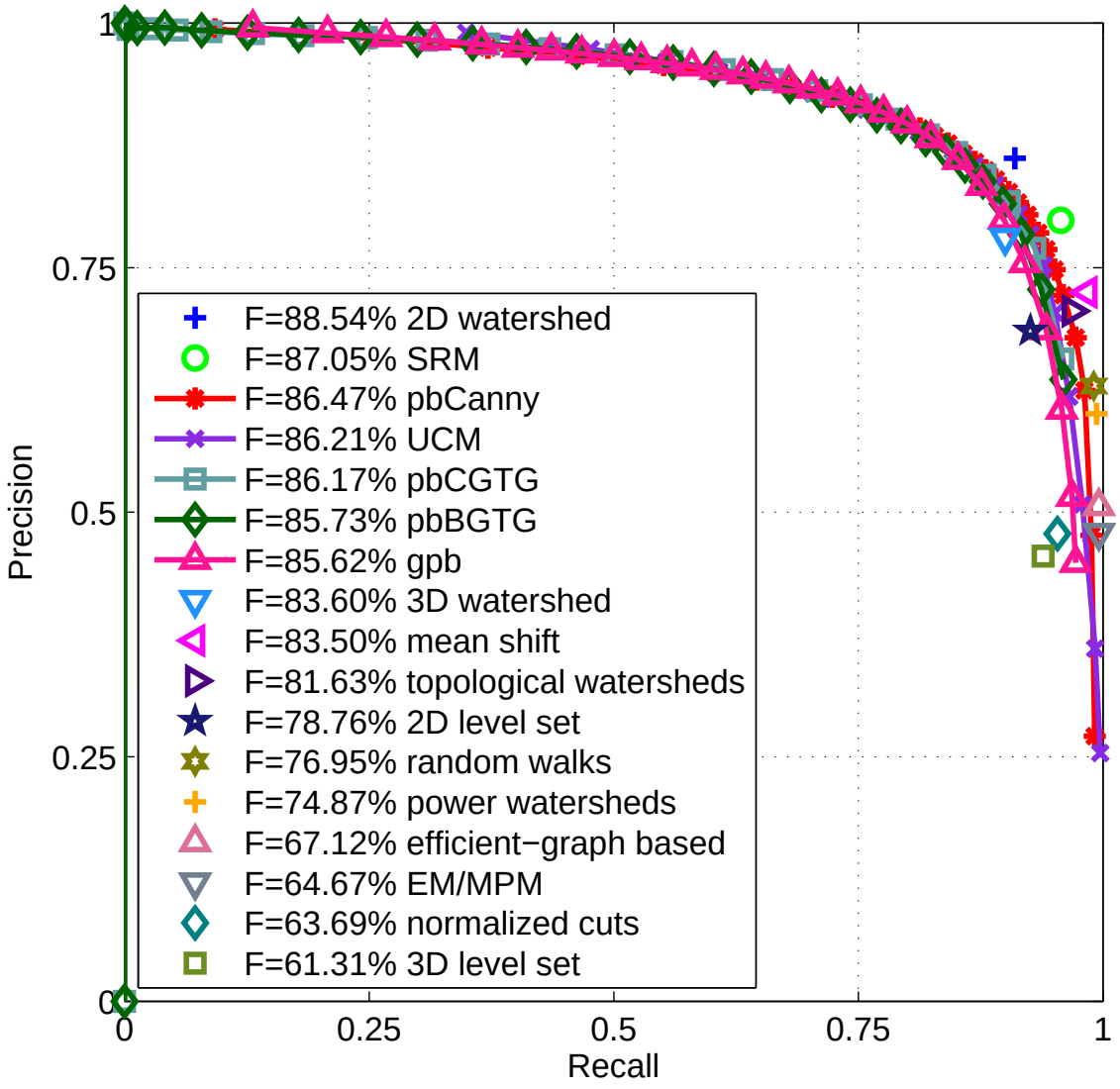


Figure 4.9 Quantitative evaluations on comparison methods using Scheme-I.

Table 4.6 Segmentation performance of the CMEWCVT, the MCEWCVT algorithms and the comparison methods (using Scheme-I). The F_1-measure of the CMEWCVT algorithm is based on using 21 constraint slices.

Methods	CMEWCVT	MCEWCVT	2D watershed	SRM	pbCanny
F_1-measure	$93.59\%/94.3\%$	93.57%	88.54%	87.05%	86.47%
Methods	UCM	pbCGTG	pbBGTG	gpb	3D watershed
F_1-measure	86.21%	86.17%	85.73%	85.62%	83.6%
Methods	mean shift	topological watersheds	2D level set	random walks	power watersheds
F_1-measure	83.5%	81.63%	78.76%	76.95%	74.87%
Methods	efficient-graph based	EM/MPM	normalized cuts	3D level set	CVT/K-means
F_1-measure	67.12%	64.67%	63.69%	61.31%	66.99%

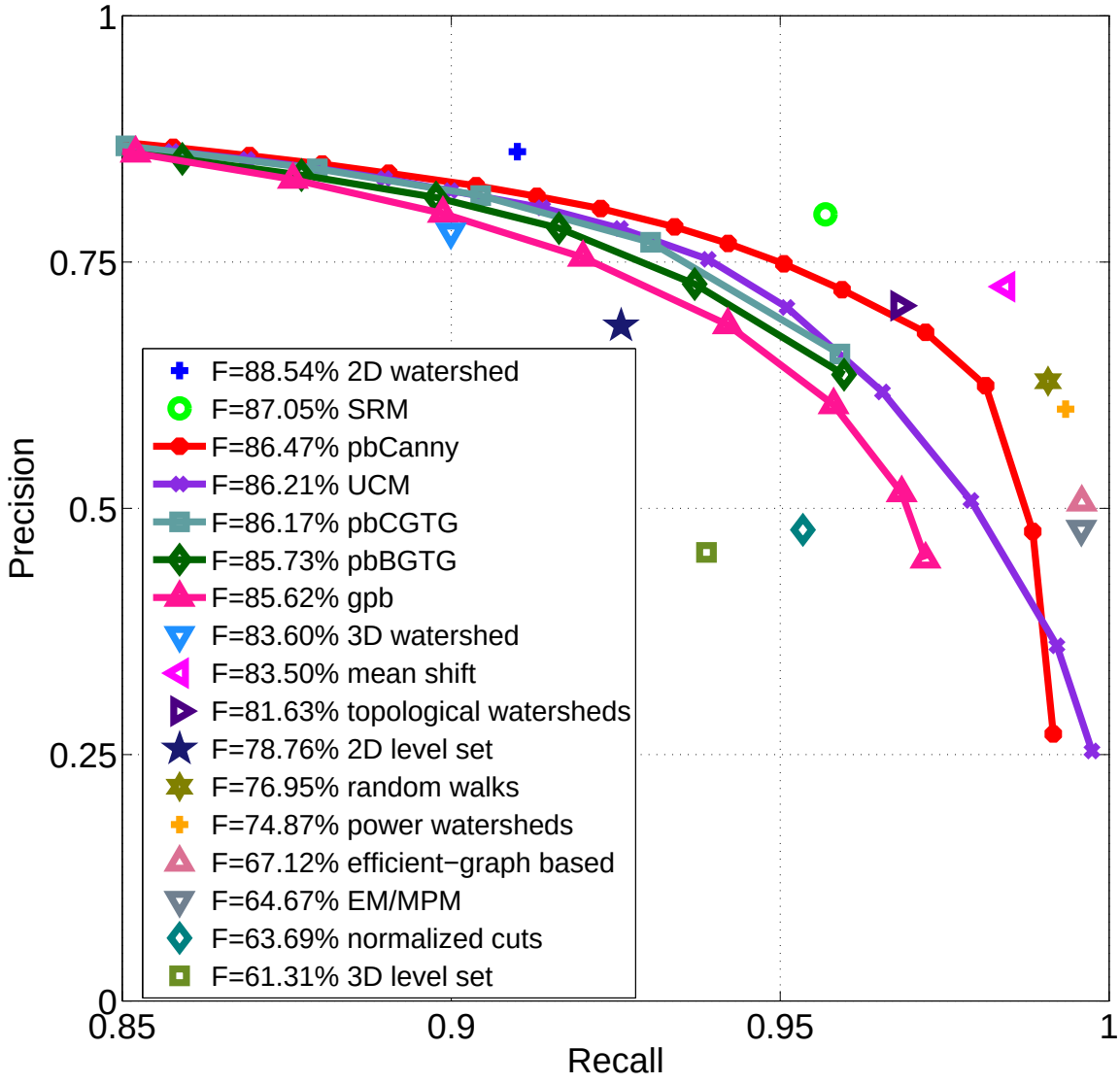


Figure 4.10 An enlarged version of quantitative evaluations on comparison methods using Scheme-I.

Table 4.7 Segmentation performance of the CMEWCVT, the MCEWCVT algorithms and the comparison methods (using Scheme-II). The F_1-measure of the CMEWCVT algorithm is based on using 21 constraint slices.

Methods	CMEWCVT	MCEWCVT	random walks	power watersheds	mean shift
F_1-measure	93.59%/94.3%	93.57%	88.87%	88.27%	86.32%
Methods	EM/MPM	pbCanny	UCM	pbCGTG	pbBGTG
F_1-measure	80.82%	80.54%	79.52%	79.51%	78.8%
Methods	topological watersheds	SRM	efficient-graph based	gpb	normalized cuts
F_1-measure	78.41%	76.36%	75.30%	73.42%	71.19%
Methods	3D watershed	2D watershed	CVT/K-means	2D level set	3D level set
F_1-measure	70.08%	68.94%	66.99%	66.74%	65.06%

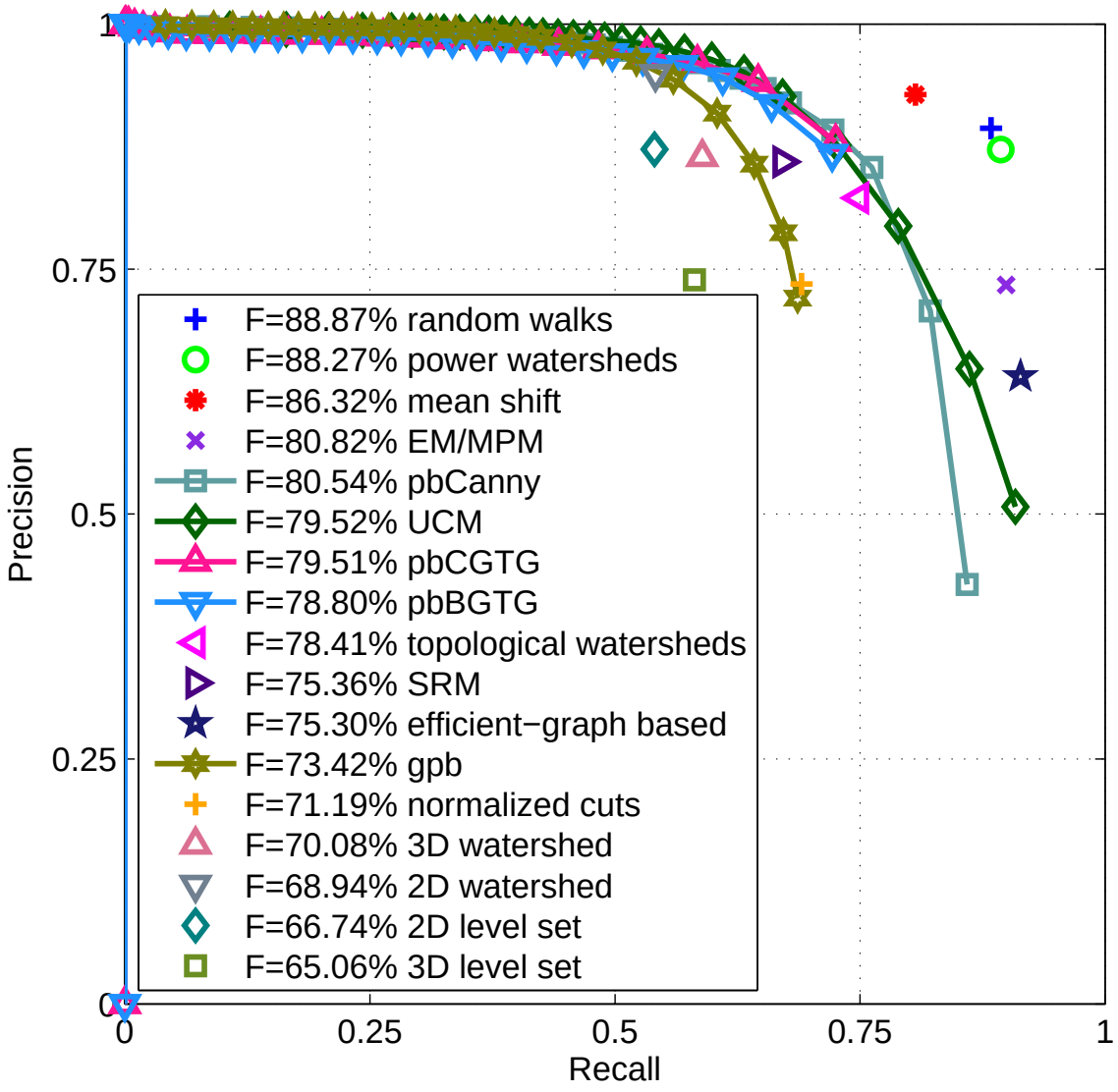


Figure 4.11 Quantitative evaluations on comparison methods using Scheme-II.

Table 4.8 Parameter settings for the 2D/3D comparison methods except CVT.

Methods	Parameter settings
random walks	2D seeds: center of the ground-truth grains; weighting parameter: 90
power watersheds	2D seeds: center of the ground-truth grains
mean shift	spatial bandwidth: 10; range bandwidth: 7; minimum region size: 30
EM/MPM	class: 15 (the maximum allowed in its GUI implementation); EM loops: 20; MPM loops: 20; Beta: 2; Min. variance: 20
pbCanny	derivative computation scale: 2; pb resolution: 100; multiplier for lower hysteresis: 0.33
ucm	oriented probability of boundary: calculated by gbp; output format: 'imageSize'
pbCGTG	radius: [0.01 0.02 0.02 0.02]; number of orientations: 8
pbBGTG	radius: [0.01 0.02]; number of orientations: 8
topological watersheds	filtration: 25
srm	segmentation number control parameter: 128
efficient-graph based	sigma: 0.5; K: 100; min: 20
gpb	resizing factor: 1.0
normalized cuts	number of segments: number of grains in the ground-truth
3D watershed	using 26-connectivity
2D watershed	using 8-connectivity
2D level set	2D seeds: center of the ground-truth grains; PDE: 'minimal variance'; initial level set circle radius: 5; iterations: 30
3D level set	$6,144$ seeds evenly distributed in 3D volume; PDE: 'minimal variance'; initial level set circle radius: 5; iterations: 30

are summarized in Table 4.8. We compile the best F_1-measures of the proposed algorithms and the comparison methods into Table 4.6 (using Scheme-I) and Table 4.7 (using Scheme-II), respectively (best F_1-measures under different parameter combinations for the MCEWCVT and the CMEWCVT algorithms, best F_1-measures with the best thresholding for soft-edge detector methods). From both Table 4.6 and Table 4.7, we can find that the proposed MCEWCVT and CMEWCVT algorithms outperform all the conventional 2D/3D comparison methods by a significant amount. The third best method, 2D watershed/random walks achieve performance at 88.54% and 88.87% for Scheme-I and Scheme-II, respectively. The soft-edge detection methods (e.g., pbCanny) achieves a reasonable performance, if a proper threshold to the edge probability can be selected. The semi-automatic methods (power watersheds and random walks) cannot achieve the best performance even with a set of ideal seeds. The EM/MPM is developed for 2D single-channel superalloy image segmentation and it produces much lower performance when segmenting a multichannel image. Table 4.9 further provides the p-values for the proposed MCEWCVT, CMEWCVT and the comparison methods (using Scheme-II). The small p-values in Table 4.9 indicate the statistical significance of the proposed MCEWCVT, CMEWCVT and the comparison methods.

Figure 4.12, Figure 4.13, Figure 4.14 and Figure 4.15 show qualitative comparisons between the segmentation results of the MCEWCVT, CMEWCVT algorithms and compar-

Table 4.9 p-values of the proposed MCEWCVT, CMEWCVT algorithms and the comparison methods (based on Scheme-II).

Methods	CMEWCVT	MCEWCVT	random walks	power watersheds	mean shift
p-value	0.23%	0.25%	0.34%	0.43%	0.2%
Methods	EM/MPM	pbCanny	UCM	pbCGTG	pbBGTG
p-value	1.05%	0.43%	0.42%	0.33%	0.36%
Methods	topological watersheds	SRM	efficient-graph based	gpb	normalized cuts
p-value	0.53%	0.36%	1.65%	0.35%	0.81%
Methods	3D watershed	2D watershed	CVT/K-means	2D level set	3D level set
p-value	0.3%	0.1%	1.59%	0.26%	0.67%

ison methods on a selected image slice (with 4 channels), where the comparison methods use Scheme-I. Furthermore, Figure 4.16, Figure 4.17, Figure 4.18 and Figure 4.19 show qualitative comparisons between the comparison methods using Scheme-II. Compared with the MCEWCVT algorithm, many small over-segmentation errors are suppressed in the proposed CMEWCVT. Meanwhile, the intensity differences between adjacent grains are better defined by human annotated segmentation. As a result, some under-segmentation errors in MCEWCVT are corrected in CMEWCVT. When compared with other conventional 2D/3D image segmentation methods, the proposed CMEWCVT algorithm achieves the best qualitative and quantitative results because 1) it considers both clustering energy term in the intensity space and the smoothing energy term in the image space; and 2) it can incorporate prior knowledge from human annotations.

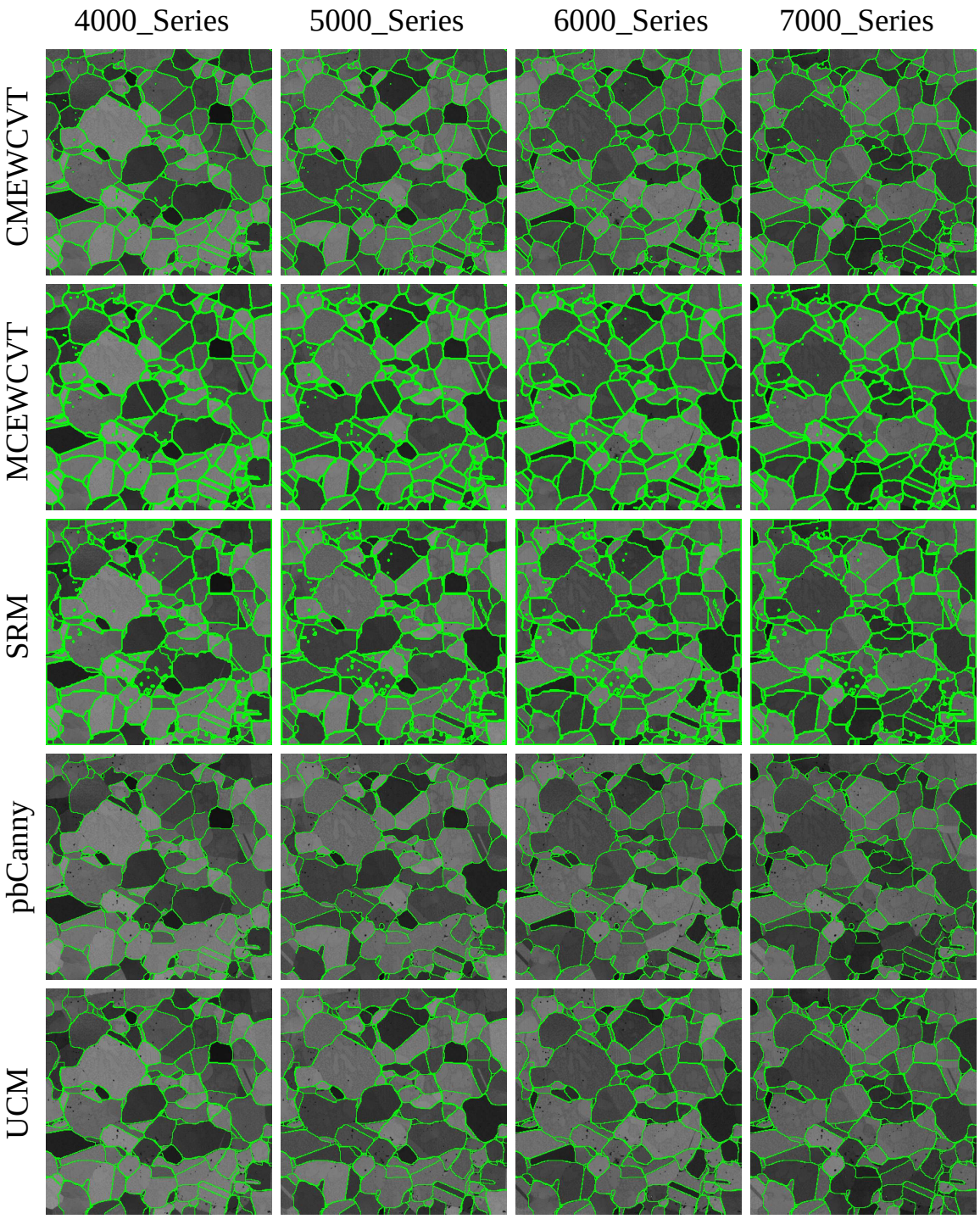


Figure 4.12 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-I. Methods: CMEWCVT, MCEWCVT, SRM, pbCanny and UCM.

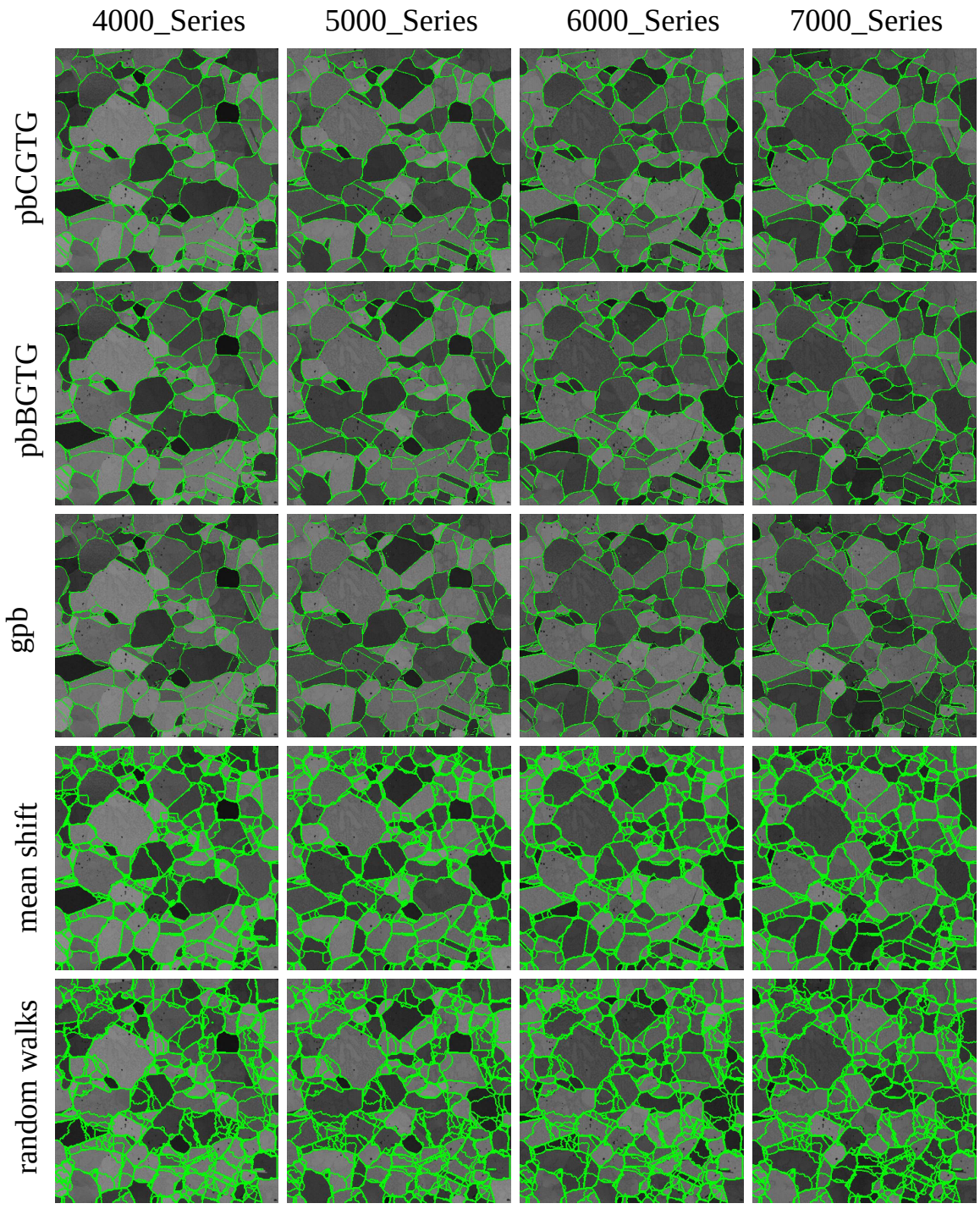


Figure 4.13 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-I. Methods: pbCGTG, pbBGTG, gpb, mean shift and random walks.

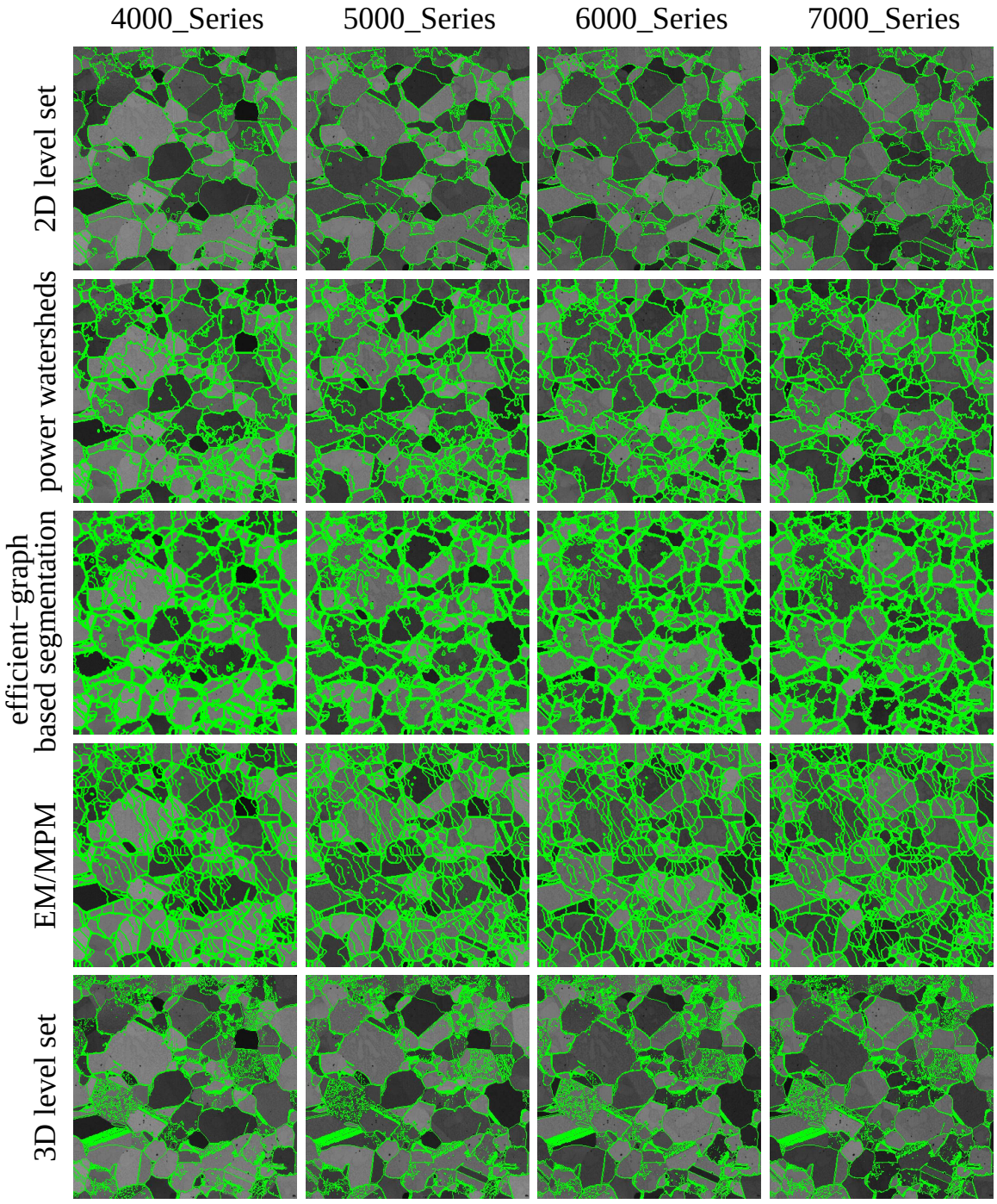


Figure 4.14 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-I. Methods: 2D levelset, power watersheds, efficient-graph based segmentation, EM/MPM and 3D level set.

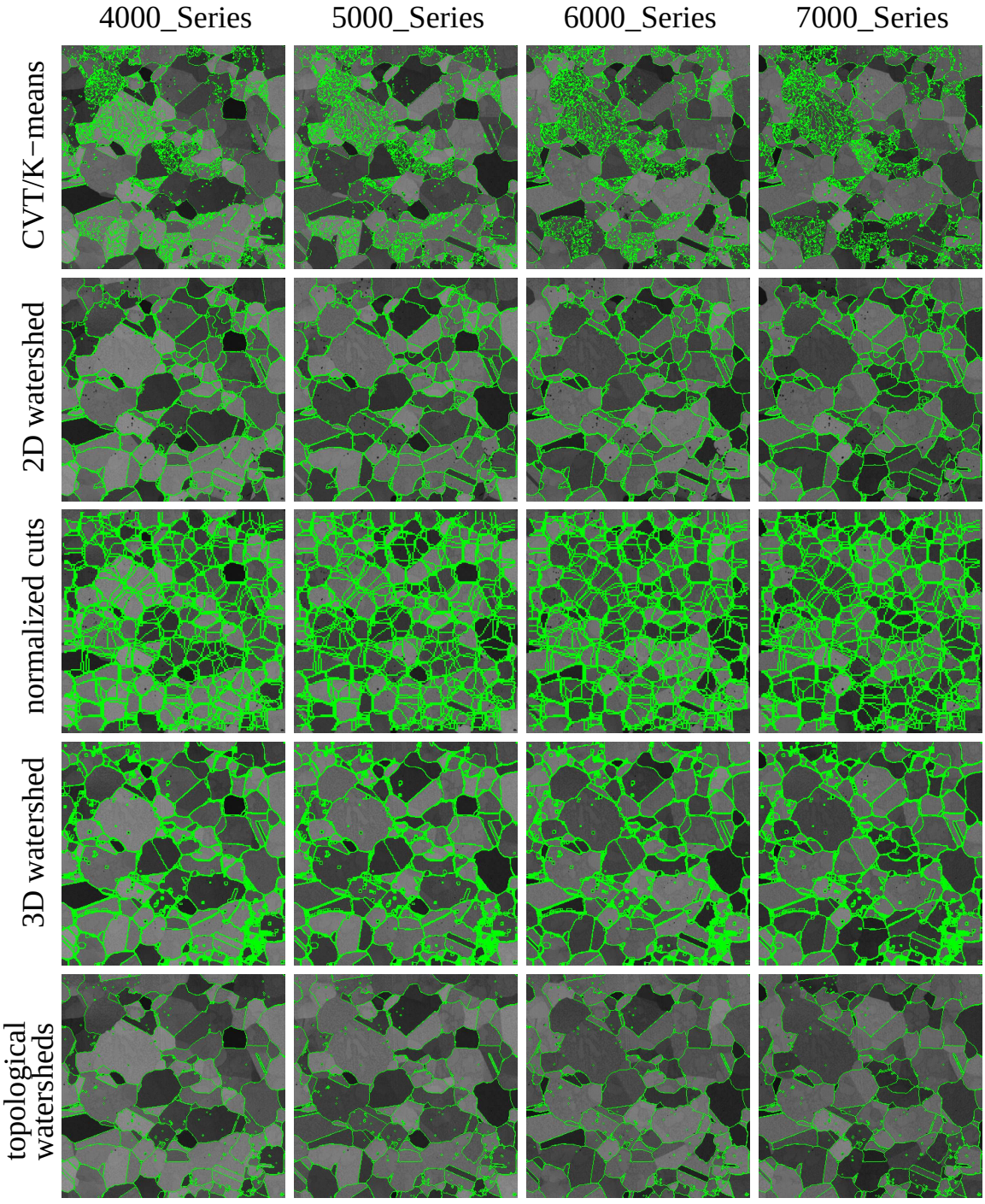


Figure 4.15 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-I. Methods: CVT/K-means, 2D watershed, normalized cuts, 3D watershed and topological watersheds.

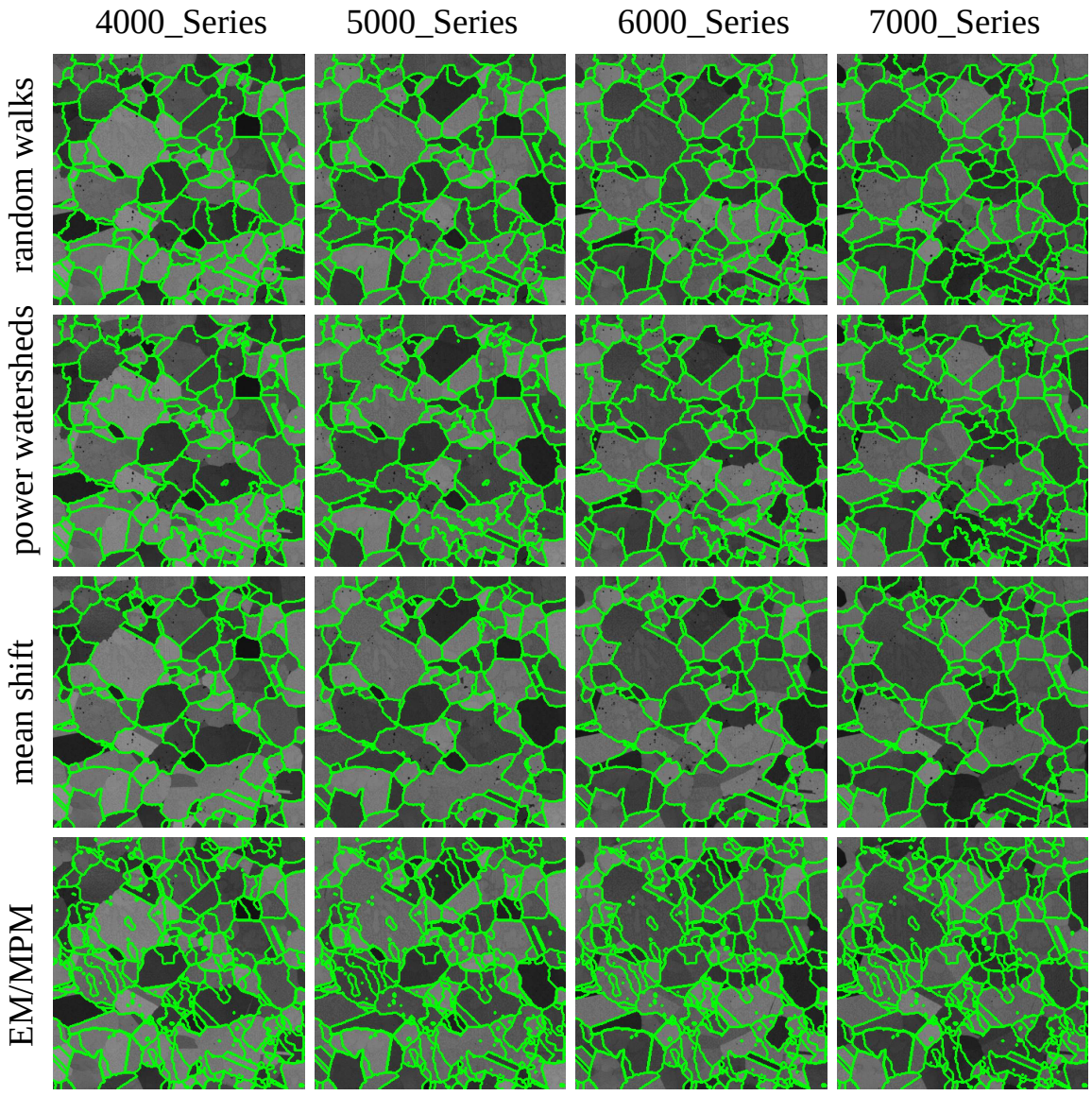


Figure 4.16 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-II. Methods: random walks, power watersheds, mean shift and EM/MPM.

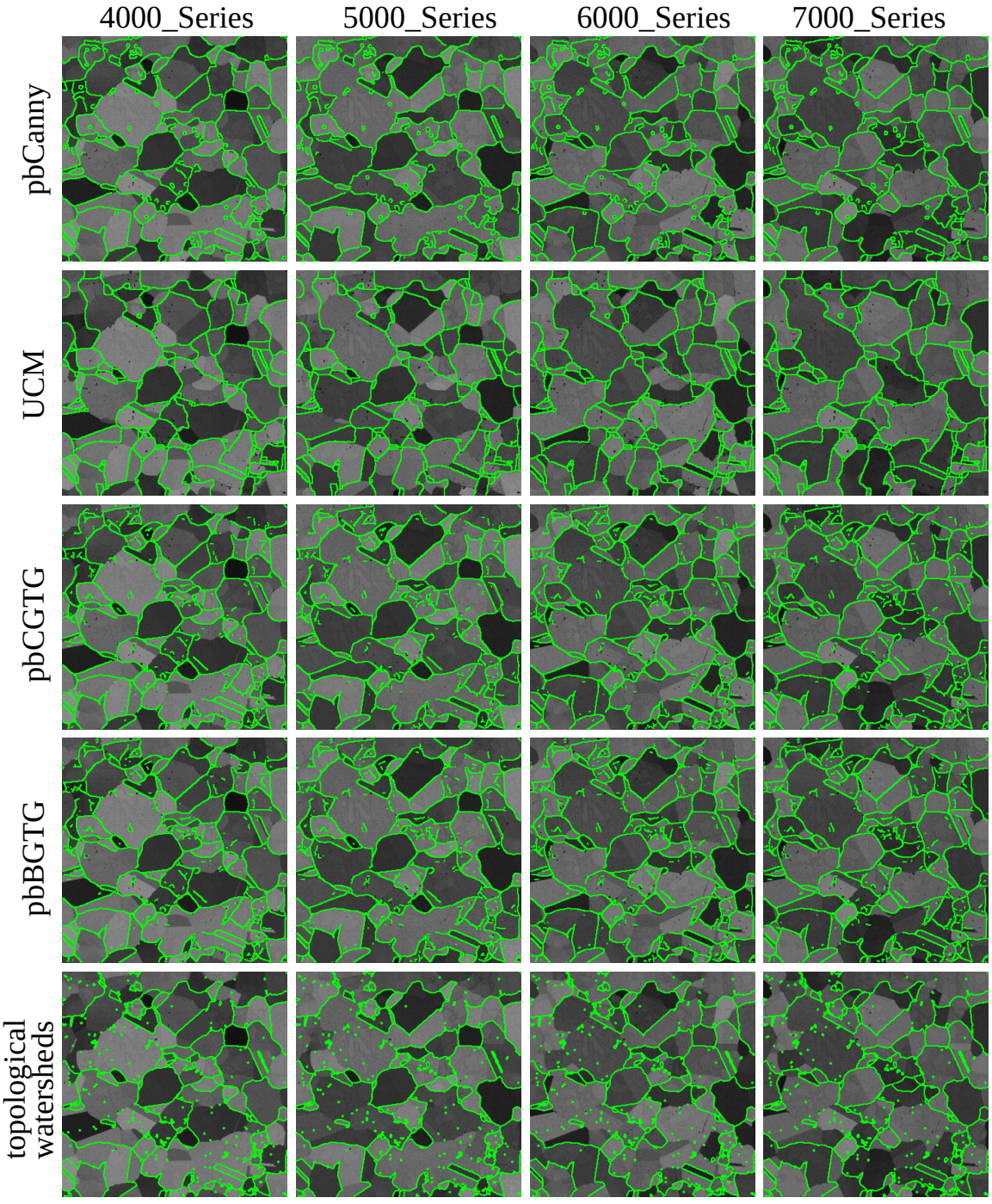


Figure 4.17 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-II. Methods: pbCanny, UCM, pbCGTG, pbBGTG and topological watersheds.

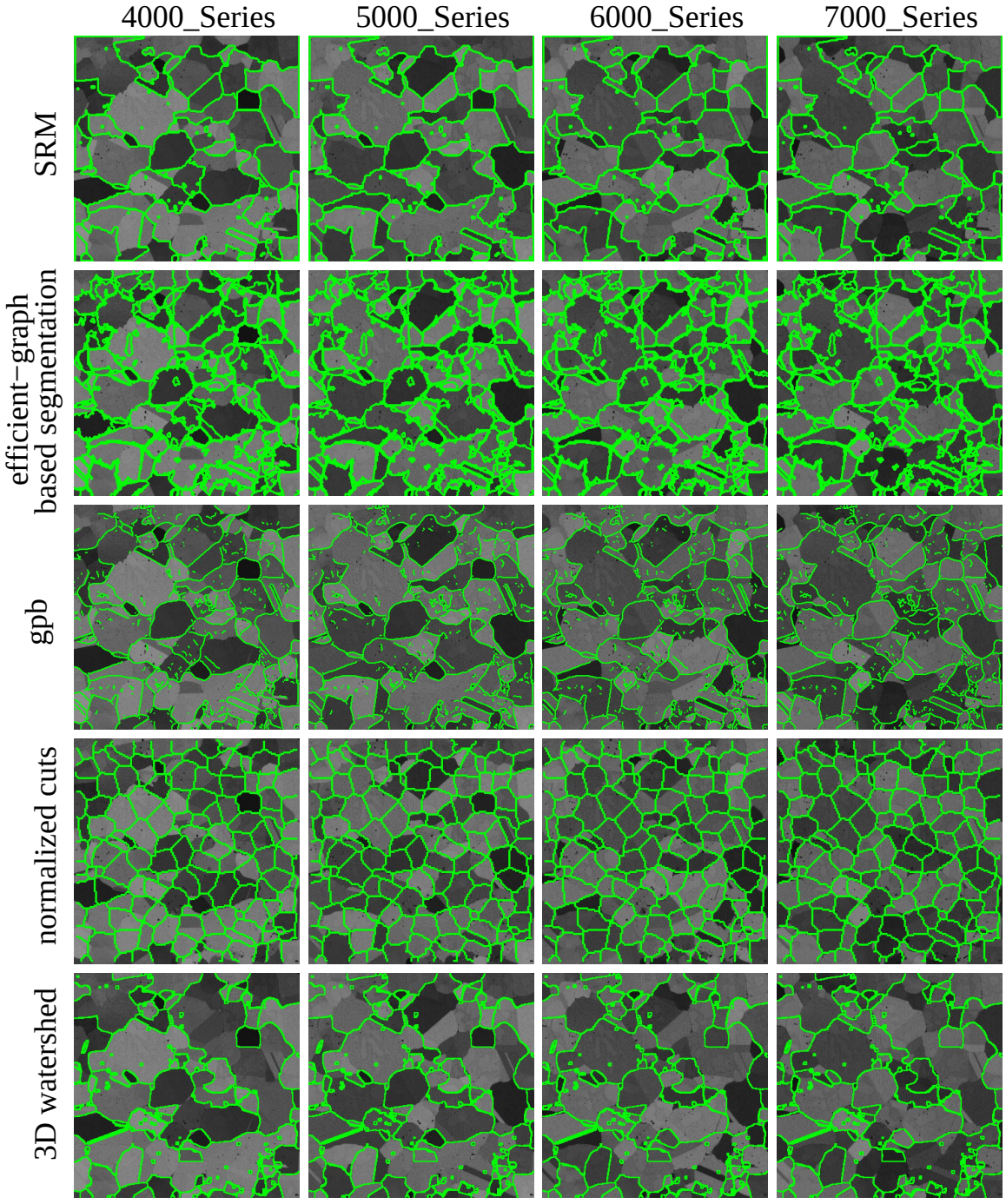


Figure 4.18 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-II. Methods: SRM, efficient-graph based segmentation, gpb, normalized cuts and 3D watershed.

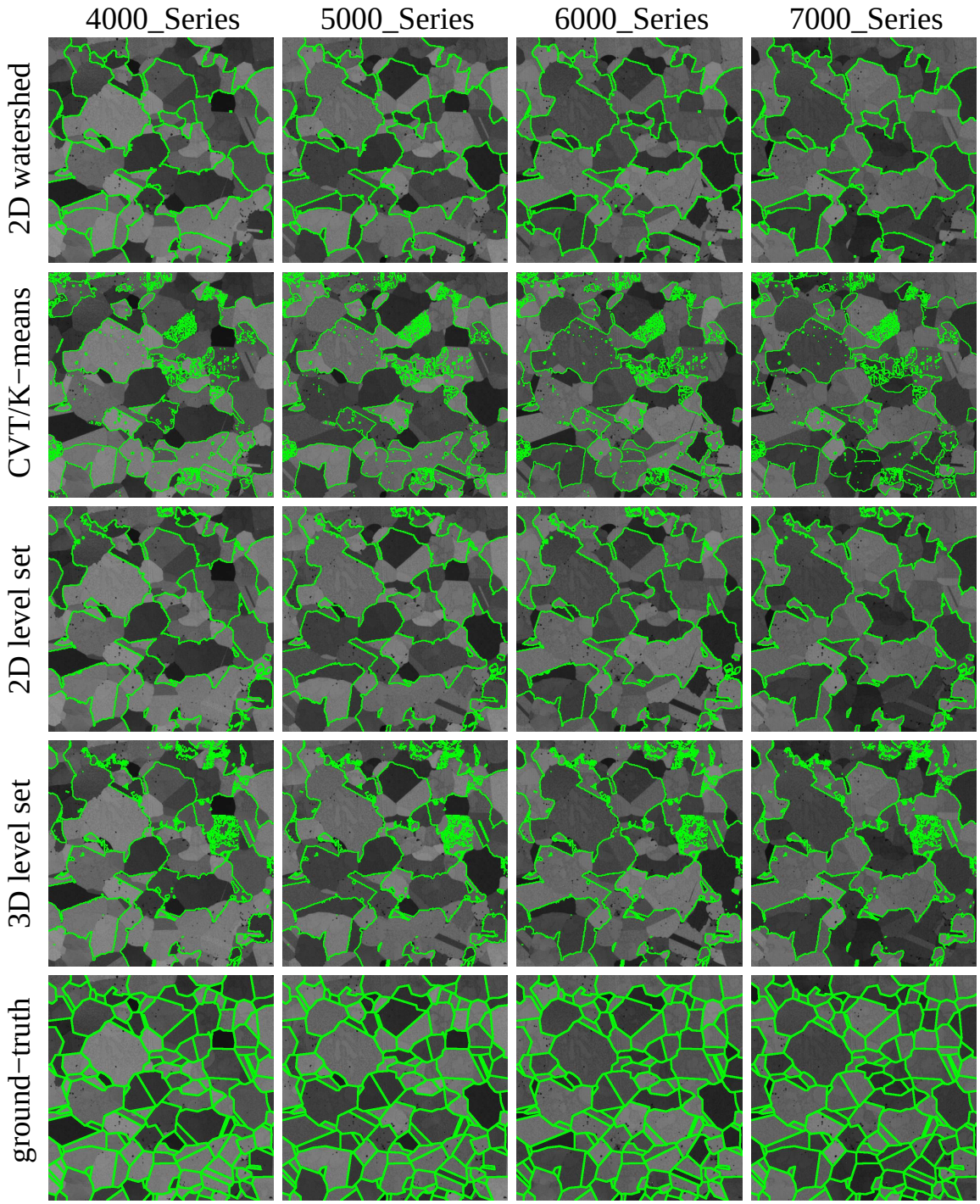


Figure 4.19 Qualitative comparisons of the segmentation results on one slice (4 channels), using Scheme-II. Methods: 2D watershed, CVT/K-means, 2D level set, 3D level set and the ground-truth segmentation.

CHAPTER 5

EXPERIMENTS ON SYNTHESIZED DATASETS

In the previous section, Table 4.1 provides the segmentation accuracy of the MCEWCVT algorithm when using different L, λ and ω. In practice, without the ground-truth segmentation, an important issue is how to select appropriate values for L, λ and ω to segment a superalloy image. Generally, the cluster number L is related to the number of grains, and the neighborhood ω is related to the grain size. In this chapter, we use synthesized superalloy images to justify that the optimal L, λ and ω values found from one dataset can be applied to other datasets which have similar grain numbers and sizes.

Given the high cost of collecting real data and the intensive labor in annotating the ground-truth segmentation in real data, synthesized data have been widely used in materials science for structural analysis. Recently, DREAM.3D [1] has become a popular tool for constructing synthesized superalloy data. In this paper, we use DREAM.3D to construct two 3D multichannel superalloy datasets: *Synthesized-I* and *Synthesized-II*. Each synthesized dataset is a $671 \times 671 \times 671$ volume with around $2,000$ grains and the grain sizes are sampled from a Log-Norm distribution derived from the grain sizes in IN100 dataset. We use the Log-Norm distribution as suggested in DREAM.3D [1] because the grain sizes in a superalloy sample can usually be described by such a distribution. Figure 5.1 shows the Log-Norm distribution (probability distribution function or PDF) with parameters $\{\mu = 6.96, \sigma = 1.62\}$ derived from the IN100 dataset. Furthermore, we set an intensity value for each 3D grain in each channel by independently sampling from the intensity distribution in the corresponding channel in IN100 data. Figure 5.2 shows the intensity distribution in each channel. Finally, we add zero mean, Gaussian white noise of

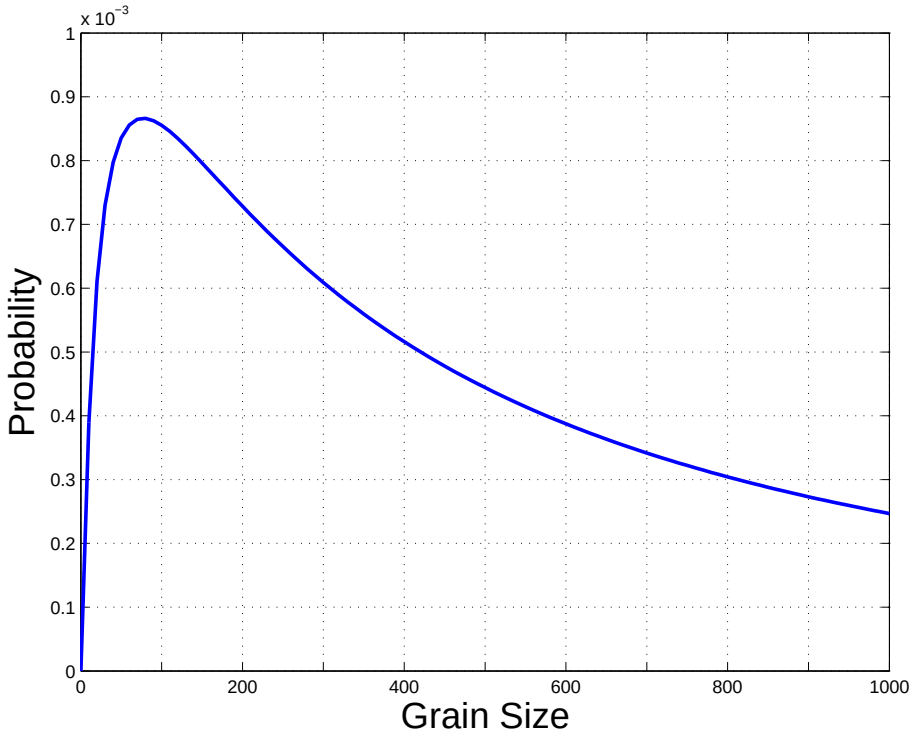


Figure 5.1 The Log-Norm distribution of the grain size derived from IN100 dataset.

random local variances (random numbers from a uniform distribution in [0,1]) to each pixel independently. In the experiment, this is achieved by directly using the MATLAB function 'imnoise'. Figure 5.3 shows one slice of the synthesized data in four channels. Similar as in the IN100 testing, we downsize the image volume size to $336 \times 336 \times 336$. The segmentation results on each 2D slices are re-scaled back to the original size (i.e. 671×671) which are further evaluated against the ground-truth.

We perform the proposed MCEWCVT algorithm on Synthesized-I and Synthesized-II datasets with parameters around $\{L = 300, \lambda = 300, \omega = 6\}$, which are the optimal parameters in segmenting IN100 data. Table 5.1 and Table 5.2 show the segmentation accuracy (F_1-measure) on Synthesized-I and Synthesized-II, respectively. From the results, we can see that the optimal parameters $\{L, \lambda, \omega\}$ for segmenting these two synthesized datasets are similar to those for segmenting IN100 data. This suggests that, in practice,

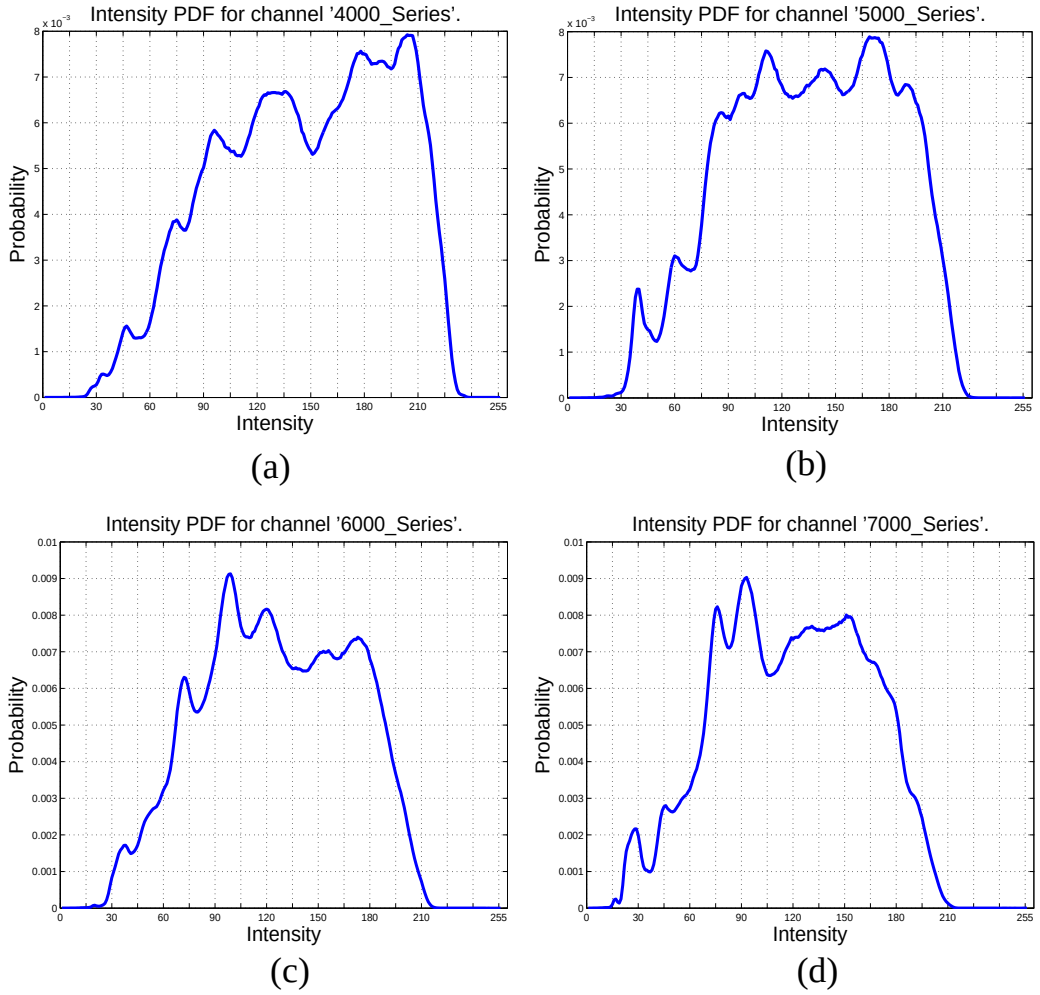


Figure 5.2 Intensity distribution in four channels: (a) 4000_Series; (b) 5000_Series; (c) 6000_Series; and (d) 7000_Series.

we may use the optimal parameters derived from one dataset to segment other superally datasets which have similar grain number and sizes. Note that, the segmentation accuracies on the synthesized data are higher than those on IN100 data. This is because the synthesized data do not contain complex imaging noise and sub-grain structures as shown in Figure 4.6.

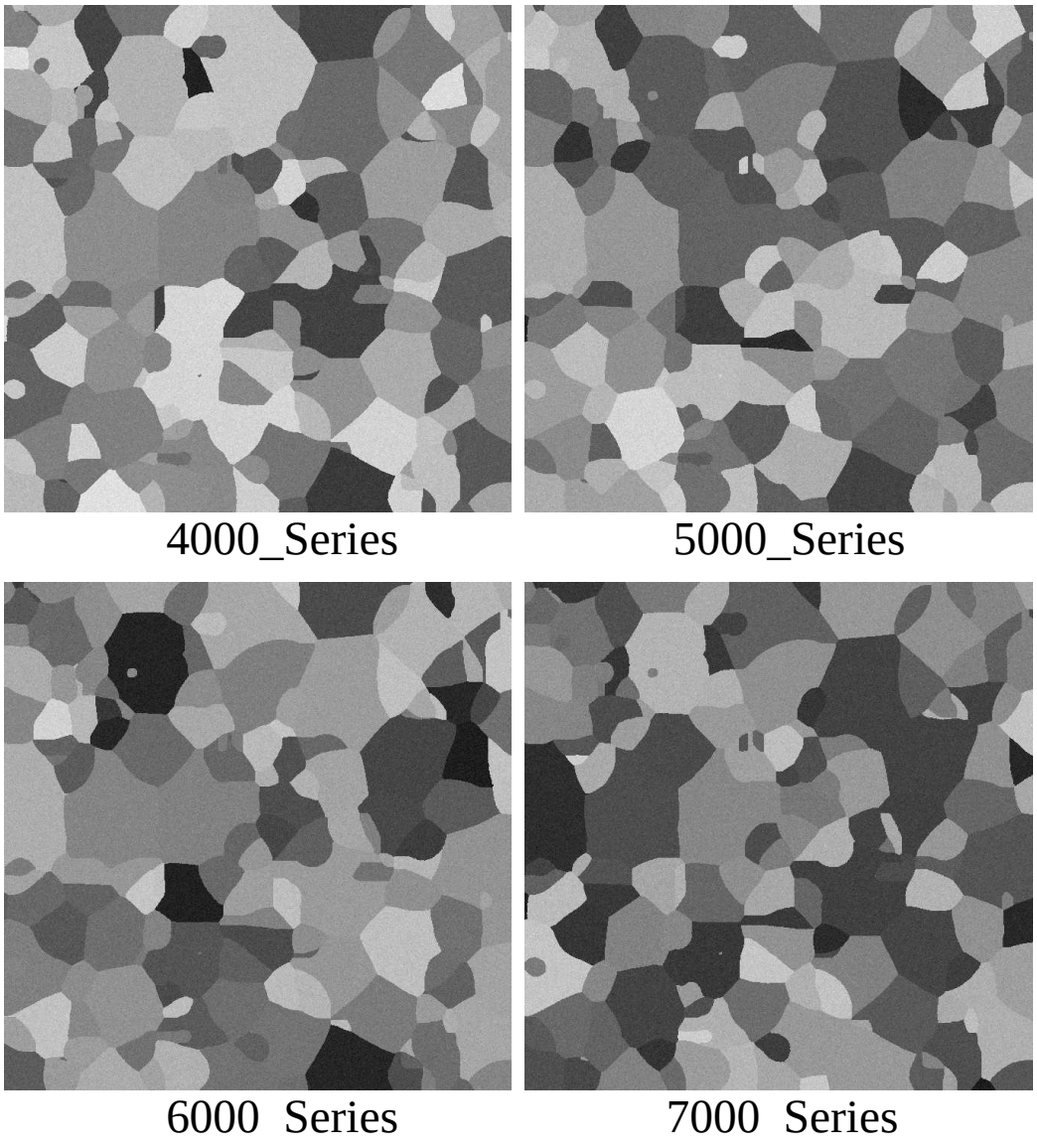


Figure 5.3 An illustration of synthesized multichannel superalloy images. Four channels are named as ‘4000_Series’, ‘5000_Series’, ‘6000_Series’ and ‘7000_Series’, respectively.

Table 5.1 Segmentation accuracy (F_1-measure) of the MCEWCVT algorithm on Synthesized-I. The best result is shaded.

	$(L,\omega)=(200,4)$	$(L,\omega)=(200,6)$	$(L,\omega)=(200,8)$
$\lambda=200$	88.68%	98.81%	98.32%
$\lambda=300$	96.16%	98.9%	97.65%
$\lambda=400$	98.27%	98.68%	96.96%
	$(L,\omega)=(300,4)$	$(L,\omega)=(300,6)$	$(L,\omega)=(300,8)$
$\lambda=200$	91.21%	98.91%	98.46%
$\lambda=300$	97.06%	99%	97.8%
$\lambda=400$	98.65%	98.8%	97.13%
	$(L,\omega)=(500,4)$	$(L,\omega)=(500,6)$	$(L,\omega)=(500,8)$
$\lambda=200$	93.84%	98.8%	98.6%
$\lambda=300$	97.93%	98.67%	97.95%
$\lambda=400$	98.3%	98.3%	97.1%

Table 5.2 Segmentation accuracy (F_1-measure) of the MCEWCVT algorithm on Synthesized-II. The best result is shaded.

	$(L,\omega)=(200,4)$	$(L,\omega)=(200,6)$	$(L,\omega)=(200,8)$
$\lambda=200$	88.7%	98.9%	98.3%
$\lambda=300$	96.2%	98.7%	97.7%
$\lambda=400$	98.2%	98.6%	97.1%
	$(L,\omega)=(300,4)$	$(L,\omega)=(300,6)$	$(L,\omega)=(300,8)$
$\lambda=200$	91.6%	98.8%	98.3%
$\lambda=300$	97.3%	99.1%	97.9%
$\lambda=400$	98.8%	98.7%	97.4%
	$(L,\omega)=(500,4)$	$(L,\omega)=(500,6)$	$(L,\omega)=(500,8)$
$\lambda=200$	94.2%	98.9%	98.5%
$\lambda=300$	98.1%	98.5%	97.6%
$\lambda=400$	98.8%	98.2%	97.3%

CHAPTER 6

FUTURE WORK

In this work, we developed two algorithms for the multichannel superalloy image segmentation, i.e., multichannel edge-weighted centroidal Voronoi tessellation (MCEWCVT), and constrained multichannel edge-weighted centroidal Voronoi tessellation (CMEWCVT). Both algorithms have the same clustering energy formulation, i.e., both consist of a multichannel intensity clustering term and an edge-smoothness term. The differences between these two algorithms lie in that the CMEWCVT algorithm utilize the human annotation as constraints to guide the initialization of the clusters and the subsequent voxels' transferring among different clusters. Both algorithms achieve reasonable good quantitative and qualitative results. In addition, by using human annotated segmentation as constraints, the CMEWCVT algorithm may have broader application since it doesn't require an elaborately selected number of clusters.

In the future, we can further extend the two proposed algorithms in following two directions: 1) accelerations for the energy minimization process at algorithm level; 2) incorporate topological structures as constraints for the energy minimization process; and 3) utilize the appearance information on each 2D slice with the super-resolution technique to construct a better interpolation for the 3D superalloy images.

As we mentioned before, the proposed MCEWCVT and CMEWCVT algorithms share a similar energy minimization framework as the Lloyd algorithm which alternates between the assignment step and the updating step. Notice that in the assignment step, each voxel will be transferred into the closest cluster (in term of the shortest edge-weighted distance from this voxel to the corresponding generator). And the transferrings of voxels are inde-

pendent of each other. Based on this fact, we utilize the OpenMP to achieve parallelism at implementation level , compared with the single CPU computing model, the parallel computing model speeds up ratio is approximately linear with respective to the number of processors.

Beside the above mentioned acceleration scheme, we may also consider the accelerations at algorithm level. Inspired by the work of Charles Elkan [20] which applies the triangle inequality to reduce the numbers of distance calculations, we may follow this idea to derive the lower bound and upper bound of the distance from a clustering data to a cluster center, where the lower bound is calculated based on cluster centers. However, since we use edge-weighted distance in the proposed algorithms, it is not trivial if the edge-weighted distance would follow the triangle inequality, or under what kind of conditions or approximations, the triangle inequality properties holds for the edge-weighted distance. The future extension on this direction still needs further investigation.

In this work, the proposed CMEWCVT algorithm utilizes the human annotated segmentation as constraints to initialize proper clusters, and guide the subsequent energy minimization process. These constraints can also be reviewed as prior knowledge from materials science domain in sense of these human segmentations give an implicit/soft definition of the similarity of multichannel intensities which might be different from the explicit/hard similarity definition based on certain mathematical distances.

In the future, we can utilize more prior knowledge from the materials science to further assist in solving the superalloy segmentation problem. For example, many theories of grain growth have been studied in [8, 48]. These theoretical rules are further implemented for simulating grain growth by methods including: Monte Carlo Potts method [15], Phase-field method [28] and Vertex method [38]. We can take the mathematical formulations of the grain growth theories from these methods as prior knowledge, and incorporate them into the future extension of the MCEWCVT and CMEWCVT algorithms.

In the process of superalloy slice collection, it is usually hard to enforce the resolutions

in the 2D slices to be identical to that between each 2D neighboring slices. In this work, we use a simple linear interpolation to handle this problem. In the future work, we can utilize appearance information on 2D slices (relative high resolution) and the super-resolution techniques to construct a better interpolation for 3D superalloy images [29].

CONCLUSION

In this work, we first introduce the superalloy image segmentation problem, then reviewed the current state-of-the-art 2D/3D segmentation methods and the techniques related to our work (e.g., centroidal Voronoi Tessellation, K-means, Lloyd algorithm, Elkan algorithm, i.e., accelerated K-means); then in Chapter 2, we develop a new algorithm, i.e., multichannel edge-weighted Voronoi tessellation (MCEWCVT) algorithm. Different from previous 2D/3D segmentation methods, the MCEWCVT algorithm is able to utilize the multichannel information and formulates the 3D superalloy image segmentation problem as a energy minimization problem. The energy function consists of a multichannel intensity clustering term and an edge smoothness term. By defining the multichannel edge-weighted distance, the energy minimization is achieved through iteratively transferring voxels to the closest (in term of the shortest multichannel edge-weighted distance) cluster, followed by updating the cluster centers.

In Chapter 3, we extend the MCEWCVT algorithm to a new semi-automatic segmentation algorithm, i.e., constrained multichannel edge-weighted Voronoi tessellation (CMEWCVT) algorithm. The CMEWCVT algorithm utilizes the human annotations on a small number of 2D slices as constraints. The constraints are used to create initial clusters and further guide the voxels' transferring process in the energy minimization process.

In Chapter 4, we first report the quantitative performances of the proposed MCEWCVT and CMEWCVT algorithms on a authentic IN100 4-channel Ni-based 3D superalloy dataset. The quantitative performances indicate that the MCEWCVT can achieve the best segmentation accuracy at 93.57% by enumerating on a set of parameters. As using more constraint 2D slices, CMEWCVT achieves better performance, and especially achieves the best seg-

mentation accuracy 94.5%. Another advantage of the CMEWCVT algorithm is that it does not need to enumerate on the cluster number in the real applications, since we only need to provide a small number for constructing a proper initial clusters. On the contrary, the MCEWCVT algorithm needs to enumerate on the cluster number.

In Chapter 4, we further compare the proposed MCEWCVT and CMEWCVT algorithms with 18 2D/3D segmentation/edge-detection methods. We set up two schemes that adapts the comparison methods to the multichannel images, and give additional favors for those methods which need seeds. The qualitative and quantitative comparison results indicate that the MCEWCVT and CMEWCVT algorithms outperform the 18 comparison methods.

In Chapter 4, we provide the segmentation accuracies enumerating on a set of parameters. We claim that the best parameters found in this testing can be transferred to other datasets which has similar number of grains and similar grain sizes. Thus, in Chapter 5, we further test the MCEWCVT algorithm on two synthesized datasets, i.e., Synthesized-I and Synthesized-II. These two synthesized datasets are constructed using DREAM.3D with multichannel intensity distributions, grain size distribution derived from IN100 dataset. Without having the complex image noises as in the IN100 dataset, the MCEWCVT algorithm achieves the best segmentation accuracies using the best parameters found in the IN100 testing.

In the future research on this topic, we expect to develop new extensions to incorporate more specific prior knowledge, such as intensity homogeneity definition, grain geometry and grain layout, to further improve the segmentation accuracy. Another possible extension could focus on speeding up the energy minimization process. Inspired by the Elkan implementation of the K-means algorithm, we may try to combine the multichannel information, the edge-energy term and the Elkan algorithm to achieve a faster implementation for the MCEWCVT and CMEWCVT algorithms.

BIBLIOGRAPHY

[1] *DREAM.3D: Digital representation environment for analyzing microstructure in 3D*, http://dream3d.bluequartz.net/.

[2] *Meshlab*, http://meshlab.sourceforge.net/, An Open Source, Portable, and Extensible System for The Processing and Editing of Unstructured 3D Triangular Meshes.

[3] *OpenMP: The OpenMP® API specification for parallel programming*, http://openmp.org/wp/.

[4] *Paraview: A parallel visualization application*, http://www.paraview.org/.

[5] M. R. Anderberg, *Cluster analysis for applications*, Academic Press, 1973.

[6] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, *Contour detection and hierarchical image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **33** (2011), 898–916.

[7] Pablo Arbeláez, *Boundary extraction in natural images using ultrametric contour maps*, In Proc. POCV, 2006.

[8] H. Atkinson, *Theories of normal grain growth in pure single phase systems*, Acta Metallurgica **36** (1988), 469–491.

[9] Gilles Bertrand, *On topological watersheds*, Journal of Mathematical Imaging and Vision **22** (2005), 217–230.

[10] Y. Boykov and G. Funka-Lea, *Graph cuts and efficient N-D image segmentation*, International Journal of Computer Vision **70** (2006), 109–131.

[11] P. Bradley, K. Bennett, and A. Demiriz, *Constrained k-means clustering*, 2000, Microsoft Technique Report, MSR-TR-2000-65, May.

[12] Hsiao-Chiang Chuang, Landis M. Huffman, Mary L. Comer, Jeff P. Simmons, and Ilya Pollak, *An automated segmentation for Nickel-based superalloy*, International Conference on Pattern Recognition, 2008, pp. 2280–2283.

[13] Dorin Comaniciu and Peter Meer, *Mean shift: A robust approach toward feature space analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002), 603–619.

[14] Camille Couprie, Laurent Najman, Leo Grady, and Hugues Talbot, *Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest*, IEEEInternational Conference on Computer Vision, 2009, pp. 731–738.

[15] Peter Streitenberger Dana ZÖllner, *Three-dimensional normal grain growth: Monte carlo potts model simulation and analytical mean field theory*, Scripta Materialia **54** (2006), 1697–1702.

[16] Q. Du and M. Emelianenko, *Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations*, SIAM Journal on Numerical Analysis **44** (2006), 102–119.

[17] Qiang Du, Vance Faber, and Max Gunzburger, *Centroidal Voronoi tessellations: applications and algorithms*, Society for Industrial and Applied Mathematics Review **41** (1999), 637–676.

[18] Qiang Du, Max Gunzburger, and Lili Ju, *Constrained centroidal Voronoi tessellations for surfaces*, Society for Industrial and Applied Mathematics, J. on Scientific Computing **24** (2003), 1488–1506.

[19] Qiang Du, Max Gunzburger, Lili Ju, and Xiaoqiang Wang, *Centroidal Voronoi tessellation algorithms for image compression, segmentation, and multichannel restoration*, J. Mathematical Imaging and Vision **24** (2006), 102–119.

[20] Charles Elkan, *Using the triangle inequality to accelerate k-means*, International Conference on Machine Learning, 2003, pp. 147–153.

[21] M. Emelianenko and A. Rand, *Nondegeneracy and weak global convergence of the Lloyd algorithm in R^d*, SIAM Journal on Numerical Analysis **46** (2008), 1423–1441.

[22] Pedro F. Felzenszwalb and Daniel P. Huttenlocher, *Efficient graph-based image segmentation*, International Journal of Computer Vision **59** (2004), 167–181.

[23] E. W. Forgy, *Cluster analysis of multivariate data: Efficiency versus interpretability of classifications*, Biometrics **21** (1965), 768–769.

[24] Keinosuke Fukunaga and Larry D. Hostetler, *The estimation of the gradient of a density function, with applications in pattern recognition*, IEEE Transactions on Information Theory **21** (1975), 32–40.

[25] Leo Grady, *Random walks for image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **28** (2006), 1768–1783.

[26] Greg Hamerly and Charles Elkan, *Alternatives to the k-means algorithm that find better clusterings*, International Conference on Information and Knowledge Management, 2002, pp. 600–607.

[27] J. A. Hartigan and M. A. Wong, *A k-means clustering algorithm*, J. the Royal Statistical Society. Series C **28** (1979), 100–108.

[28] C.E. Krill III and L. Chen, *Computer simulation of 3-d grain growth using a phase-field model*, Acta Materialia **50** (2002), 3059–3075.

[29] Yutaro Iwamoto, Xian-Hua Han, So Sasatani, Kazuki Taniguchi, Wei Xiong, and Yen-Wei Chen, *Super-resolution of mr volumetric images using sparse representation and self-similarity*, International Conference on Pattern Recognition, 2012, pp. 3758–3761.

[30] Gunnar L*ä*th*é*n, Jimmy Jonasson, and Magnus Borga, *Phase based level set segmentation of blood vessels*, International Conference on Pattern Recognition, 2008, pp. 1–4.

[31] Chunming Li, Chenyang Xu, Changfeng Gui, and Martin D. Fox, *Level set evolution without re-initialization: A new variational formulation*, IEEEComputer Vision and Pattern Recognition, 2005, pp. 430–436.

[32] Stuart Lloyd, *Least squares quantization in pcm*, IEEE Transactions on Information Theory **28** (1982), 129–137.

[33] Ming Luo, Yufei Ma, and Hongjiang Zhang, *A spatial constrained k-means approach to image segmentation*, Fourth IEEE Pacific-Rim Conference On Multimedia, 2003, pp. 738–742.

[34] James MacQueen, *Some methods for classification and analysis of multivariate observations*, Symposium on Math, Statistics, and Probability, 1967, pp. 281–297.

[35] Derek Magee, Andrew Bulpitt, and Elizabeth Berry, *Level set methods for the 3D segmentation of CT images of abdominal aortic aneurysms*, Medical Image Understanding and Analysis, 2001, pp. 141–144.

[36] David R. Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik, *A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics*, IEEEInternational Conference on Computer Vision, 2001, pp. 416–423.

[37] Fernand Meyer, *Topographic distance and watershed lines*, Signal Processing **38** (1994), 113–125.

[38] L.A. Barrales Mora, *2d vertex modeling for the simulation of grain growth and related phenomena*, Mathematics and Computers in Simulation **80** (2010), 1411–1427.

[39] Ken Museth, David E. Breen, Leonid Zhukov, and Ross T. Whitaker, *Level set segmentation from multiple non-uniform volume datasets*, IEEE Conf. Visualization, 2002, pp. 179–186.

[40] Richard Nock and Frank Nielsen, *Statistical region merging*, IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004), 1425–1458.

[41] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Dr Sung Nok Chiu, *Spatial tessellations: Concepts and applications of voronoi diagrams*, Wiley, 2000.

[42] Michael J. D. Powell, *An efficient method for finding the minimum of a function of several variables without calculating derivatives*, Computer Journal **7** (1964), 152–162.

[43] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical recipes in C: The art of scientific computing*, Cambridge University Press, 1992.

[44] Roger C. Reed, *The superalloys fundamentals and applications*, Cambridge Press, 2001.

[45] Jianbo Shi and Jitendra Malik, *Normalized cuts and image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (1997), 888–905.

[46] D. N. Sparks, *Euclidean cluster analysis*, J. the Royal Statistical Society. Series C (Applied Statistics) **22** (1973), 126–130.

[47] H. Späth, *Cluster dissection and analysis. theory, fortran programs, examples*, Prentice Hall, 1985.

[48] Carl V. Thompson, *Grain growth and evolution of other cellular structures*, Solid State Physics **55** (2001), 269–314.

[49] A. Vedaldi and B. Fulkerson, *VLFeat: An open and portable library of computer vision algorithms*, http://www.vlfeat.org/, 2008.

[50] Fabien Vivodtzev, Lars Linsen, Georges-Pierre Bonneau, Bernd Hamann, Kenneth I. Joy, and Bruno A. Olshausen, *Hierarchical isosurface segmentation based on discrete curvature*, Conf. Data Visualization, 2003, pp. 249–258.

[51] George F. Vander Voort, Francis J. Warmuth, and Samuel M. Purdy, *Metallography: Past, present, and future (75th anniversary vol.)*, Astm Special Technical Publication, 1993.

[52] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl, *Constrained k-means clustering with background knowledge*, International Conference on Machine Learning, 2001, pp. 577–584.

[53] Jie Wang, Lili Ju, and Xiaoqiang Wang, *An edge-weighted centroidal Voronoi tessellation model for image segmentation*, IEEE Transactions on Image Processing **18** (2009), 1844–1858.

[54] Ross T. Whitaker, *A level-set approach to 3D reconstruction from range data*, International Journal of Computer Vision **29** (1998), 203–231.