REU Report

Numerical Algorithms for the Automatic Processing of Image Data, Specifically Grain Growth

James Eckstein

May 2020

Contents

1 Introduction

1.1 Motivation

Most inorganic solids are made up of many grains or cystalites. These materials are known as polycrystalline materials and their characteristics can be affected by the structure of their grains. For example, the thermal and electrical conductivity of a polycrystalline material are positively correlated to the average size of a grain, so in situations where high conductivity is important, one may anneal their parts which reduces grain boundaries by melting grains together.

Consequently, understanding these properties is an important area of research in materials science. One of the the challenges accompanying this research is the expense associated with actually testing these properties. Thus there is substantial interest in creating a model which can accurately predict the evolution of grains over time.

In order to accurately do this, large amounts of statistical data about these grain structures must be processed. Needless to say that without a numerical algorithm to process these images, this can be quite an undertaking.

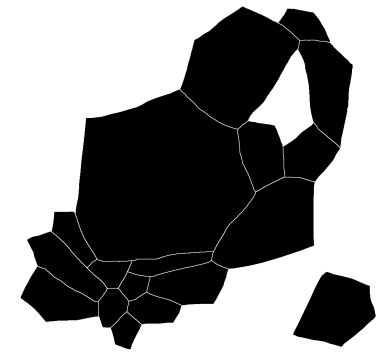

Figure 1: This is not the original data, but it is similar and shares many of the features of the given data

The focus of this project was to create a numerical algorithm for gathering data from images of grain boundaries. This is a continuation of a project started by Charlotte Blake[1] last year and the images were provided by Professor Barmak's laboratory at Columbia University. These images have been pre-processed to make them cleaner, and the code written by Charlotte Blake is substantial foundation on which to build. The ultimate goal is to create a utility that can be used by researchers as a tool to analyze and gather data from 2d images of grains and their boundaries.

1.2 Existing Work

This project aims to create an efficient algorithm which can be used by researchers to take an image or set of images similar to Figure 1 and gather data about each of the grains shown in the image. As stated earlier, this project was started by Charlotte Blake. Her algorithm was able to detect the area, perimeter, and number of sides of every grain[1].

The goal of this semester was to expand the capabilities of this algorithm to gather more data from these images and to improve its robustness.

2 Additions

2.1 Support for Islands

Initially, the algorithm could not detect grains that were detached from other grains. Grain 18 in Figure 2 is an example of a detached grain or island. The

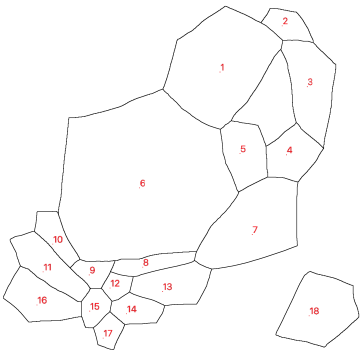

Figure 2: This is a numbered version of Figure 1

method implemented to address this takes the data from the pre-existing algorithm, which included the grain boundaries of all grains that bordered other grains, and removes all the detected grain boundaries from the image. This new image only contains the bodies that were not detected by the original algorithm, which could be islands and other smudges or image artifacts. An example of one such smudge or artifact can be seen in the top left corner of Figure 1. Generally these are just small imperfections in the image that are clearly non-intentional and should not be included in the outputted data. Because of image cleaning done earlier in the algorithm[1], these smudges and islands appear as single pixel loops.

By using matlab's **find**[2] function, the algorithm finds some random illuminated point in the image. Starting at this point, the algorithm collects the path of the boundary which it is connected to, afterwards subtracting it from the image. It repeats this process until it can no longer find an illuminated point in the image, meaning that all the bodies in the image have been indexed.

These collected paths are processed, first by making sure that each has a length that indicates that it is an island, not a smudge. This threshold is currently set at 100px, which is conservative, but in testing no smudge was found with a perimeter longer than that. A corner detection test is ran on the remaining paths, which finds the location of all the corners on each island grain boundary. The remaining island loops are sliced into segments which end at these corners, and then are fed into the array of already detected segments.

2.2 Correctly Matching Grains

The process of improving the statistical accuracy of the algorithm can be aided by comparing the outputted data with real results. Most of the images provided

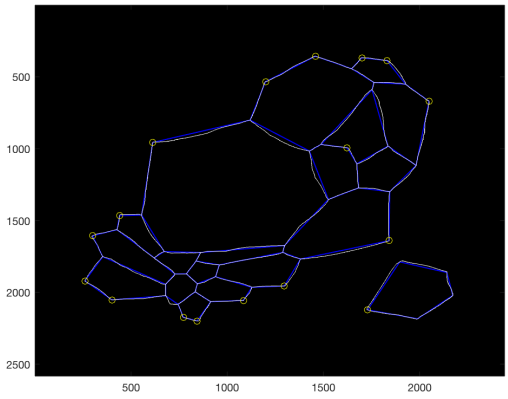

Figure 3: This image shows the output of the algorithm, the island grain (bottom left) is being detected

by Prof. Barmak's laboratory have a companion spreadsheet which contains the measured area, perimeter, and side count of each grain in said image and another image which shows how those grains were numbered. For the sake of clarity, grain data outputted from the algorithm will be labeled as detected data, while data read from the spreadsheet will be referred to as given data.

In order to correctly pair the detected grains with the given data, it is first necessary to convert the outputted perimeter and area measurements from pixels to μm and μm^2. This was done by getting the scale factor from the relevant spreadsheet and using the formulas illustrated in Blake's paper[1]. With the detected data scaled, every detected grain is matched to a grain in the spreadsheet using the perimeter, area, and side count measurements. To do this the percent difference between the perimeter and area of every detected grain and every given grain is found and if both are within a threshold, they are weighted by a formula and placed into a matrix. This matrix is populated such that each row contains the potential pairings of a specific detected grain and each column contains the potential pairing of a grain from the given data. Having a threshold reduces run time, and it is safe to assume that each detected-given grain pair will have an area difference of less that 1% and a perimeter difference of less that 10%.

Using this, every potential pairing for each detected grain is considered. Because every grain should only be paired with one other grain, if a detected grain has more than one given grain which passed the threshold, the matrix is used to average out the total difference caused by a given pairing. This is necessary because pairing grains by only looking at which pairs have the smallest percent difference produces bad results and looking at the overall impact of a pairing has proved much more accurate. This is done by slicing out the row and column of the grains in each potential pairing from the matrix and averaging all

the remaining values and the value of the pairing being tested. Then whichever pairing has the lowest overall average is determined to be the best fit.

This test is ran with all grains that share the same side count and then with all the remaining unpaired grains. In the end, every detected grain is matched to a grain in the given data.

The percent difference weighting formula is arbitrary and was found through testing. This testing was printing the index of a grain pair at its center on the provided numbered image. This allowed for certainty about which grains were matched properly.

Using these methods, the maximum perimeter and area differences of the detected and given grains is consistently about 0.1-0.6% and 6-7% respectively for the tested data.

2.3 Associating Segments to Grains

Associating segments and grains is done by looping over each grain and comparing it's vertices with the starting point, corners, and ending point of each segment, in that order. Doing this in order makes it easy to assign another parameter to each grain which maps the connection of its vertices. This parameter is an array which contains the end points of a grain's edges. If a grain's vertices match consecutive corners and endpoints in a segment, then it can be assumed that those vertices are connected. An edge containing those vertices is made and the segment is added to a list of the grain's associated segments.

2.4 Finding Vertex Angles

Using the edges found in the last section, finding the angle at each vertex of a grain is a matter of correctly linking each vertex and using an angle between vectors formula on each linked pair. Linking them means ordering the edges so the ending point of an edge matches the starting point of the next. Then the angle between each was found using the following formula:

$$\theta = \arctan(\frac{\|\mathbf{v} \times \mathbf{u}\|}{\mathbf{v} \cdot \mathbf{u}})$$

Where $\mathbf{v}$ and $\mathbf{u}$ are found by centering the coordinates of two linked edges at the point which they share. The results of this are believable, with the outer angle of each grain totalling to 2π, except those which are visibly non-convex.

3 Future Improvements

The improvement that comes most readily to mind is using the vertex angle calculations to improve the detected side count. Discrepancies in observed side count between the detected and given data tend to be a result of the detected data reporting more sides than are really there. This is a result of the assumption

that each branchpoint is a vertex connected to two sides. This generally works, but there are cases where a grain's flat edge meets the intersection two other grains, causing an over-counting of sides. This could be solved by removing all the sides that are nearly 180 degrees from each other.

References

[1] C. Blake. *Efficient Numerical Algorithms for Automatic Data Collection With Application to Materials Science*, Fall REU Report, December 2018 and Spring REU Report, April 2019

[2] The MathWorks, Inc. find documentation. `https://www.mathworks.com/help/matlab/ref/find.html`, 2020