# Image Segmentation

Nathan D. Heavner Rachel Tutmaher James Folberth

## 1 Introduction

There are large amounts of data and images that are being generated today; sources include satellite imagery, medical imaging such as MRIs or CT scans, and social media. An automated process that can scan an image for significant objects or figures is greatly desired and is therefore a topic of much current research. Image segmentation algorithms involve reading in an image, converting the image to a matrix of pixel values, and grouping similar pixels together with the goal of identifying objects in single images and tracking the objects in the sequences of images. This will ideally allow for a greater number of images to be processed more quickly than is currently possible. However, this is a complicated task. It is difficult to design an algorithm that correctly identifies shapes and groups them accordingly. It is obviously undesirable to segment an object that should not be segmented, or on the other hand, to not "aggregate" an entire object properly.

The algorithm that we investigated for our project was specifically developed to track live cells imaged with bright field cell microscopy. Both the movement of the cell through space and its evolution through time (cell growth and division) were qualities that needed to be captured with an image segmentation routine. Other examples of image segmentation involve satellite imagery where roads, buildings, or even people may need to be identified. The sheer quantity of images and pictures that need to be analyzed and categorized for various reasons makes finding an accurate and fast automated image segmentation algorithm a necessity.

Typically, once the image to be analyzed has been converted to a matrix of pixel values, these pixel values are used to determine the "intensity" of each pixel. Then, neighboring pixels are grouped together based upon whether they have similar intensities. This process continues until "aggregates" are formed. An aggregate is a lump of neighboring pixel values that have been segmented together, the hope being that this aggregate represents an object in the original image. This process assumes objects in an image will have similar grayscale pixel values, which is most often the case. The size of the matrix with dimension being the number of vertical pixels by the number of horizontal pixels affects the speed and cost of the algorithm. There is a large class of algorithms that are being designed to process images and segment them. The method that we employ involves techniques from (Ruge-Stüben) Algebraic Multigrid methods (AMG); specifically it involves altering the "Segmentation by

Weighted Algorithm" (SWA) technique. This new method is favorable because initial testing by Inglis et al. claims that the algorithm's runtime is linear in the number of pixels!

We implemented this algorithm in C++, Julia, and Matlab and compared results. The results shown in this report are from the Julia and C++ code implementations. The main difference in the algorithm that is used in this project from other similar algorithms is that there is a "scale-invariant saliency measure." This measure decides when an aggregate represents an entire object and should not be further grouped with other pixels surrounding it. The prominence of an aggregate is the main identifier of individual objects within an image.

# 2 Algorithm Description and Implementation

Although, the algorithm used here is based on AMG and SWA, the final segmentation for this algorithm is determined by one image segmentation V-cycle, which is different from regular SWA where many V-cycles may be used. Also, a measure of "texture", obtained by varying the mean intensities at each level improves performance of cell segmentation, the main goal of the algorithm that we chose to implement. We also tested the algorithm on other objects. Significant fine-tuning in the segmentation results is made possible by the relatively large number of parameters that this algorithm calls for. The overall goal of this algorithm is to assign every pixel in the original image to an aggregate, therefore adequately segmenting a given image hopefully into segments that may be subsequently analyzed. In order to achieve this goal, the algorithm flows as follows.

First the finest level variables are defined. Among these are an intensity vector. This vector contains all of the intensity values from the original image (either column-wise or row-wise ordering) and the vector is rescaled so that the intensities range from 0 to 1. Additionally, a coupling matrix is created in the following manner:

$$A_{ij} = \begin{cases} e^{-\alpha(I_i - I_j)} & \text{if } i, j \text{ are neighbors} \\ 0 & \text{else.} \end{cases}$$

This transforms the problem of segmenting an image into the problem of segmenting a graph, where the nodes are the intensities of each pixel and the edge weights are given by the coupling matrix.

Next the graph is coarsened repeatedly until everything has been segmented. Blocks are checked for saliency at each level. A salient block is one which is "significantly" different from its surroundings within the matrix, meaning that the connections within the block are much stronger than the connections from any other neighbors. Once it has been determined that a block is "salient" it will not merge with any other blocks on coarser levels, however these blocks are propagated through all the levels in the down side of the V-cycle. The edge

weights which play a part in determining saliency with neighboring nodes are calculated via Galerkin coarsening.

Once the coarsest level is reached, the upward part of the V-cycle begins, the goal being to uniquely assign every finest-level pixel to one of the segments. Segments that are acquired on the coarsest level relate to overlapping blocks on the top level. Since a pixel may originally be classified as belonging to a certain block at the finest level and then later be reclassified at the coarsest level, it is vital that the salient blocks are retained throughout the coarse level progression so that that original relationship is maintained throughout the V-cycle. Once all segments have been found, nodes on the next finer level are assigned to segments on the coarser level according to weights in the interpolation matrix. This is repeated up to the original finest level. On each level, the overlap between segments is refined by the following threshold determining by a sharpening threshold. Nodes that belong for more than 85% to a segment are assigned 100% to that segment, while nodes that belong for less than 15% to a segment are fully decoupled from that segment. Once the algorithm reaches the finest level of the V-cycle, a post-processing step uniquely assigns each pixel to a segment.

# 3 Results

In this section, we demonstrate the efficacy of the segmentation algorithm. We first demonstrate in Figure 1 the improvement in segmentation accuracy obtained by taking multilevel intensity variance into account. The image in Figure 1 contains two segments which have different textures (variance within the segment) but identical average intensities. When we do not take multilevel variance into account, the algorithm detects only one segment. When we rescale the weighting matrix using the variance of each block, however, the two segments are successfully detected.

Regarding the checkered disk image, it should be noted that Inglis et al. says they use the row-sum method to determine the strong connections with using the AMG graph coarsening. We were unable to recreate their results using their parameters with the row-sum method, but if we used the more standard max-element method to determine the strong connections, their parameters and our implementation produced the desired results.

Next, we demonstrate the effect of making the saliency measure invariant of the segment scale by normalizing the similarity-weighted boundary length and area. With the usual saliency measure, it is much harder for the algorithm to detect the segments because the boundary length is much greater than the segment area compared to a simpler shape such as a circle. With the author's proposed scale invariant measure, the segment is easily detected.

The remaining figures demonstrate the performance of the algorithm with more complex images. The performance is quite good. In the cases where the segmentation is "incorrect," we can reasonably attribute this to human intelligence rather than algorithm performance.

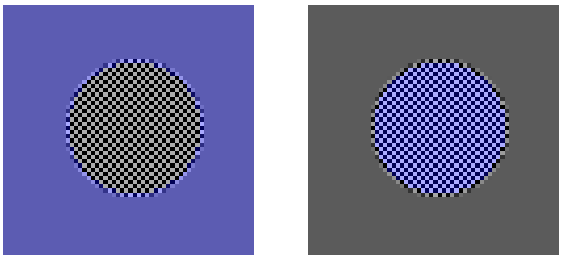


Figure 1: Without taking multilevel variance into account, the algorithm detected only one segment (the entire image). Rescaling the weighted graph using multilevel variance results in the segmentation shown above. Parameters used: $n = 60$, $\alpha = 10$, $\tilde{\alpha} = 10$, $\beta = 10$, $\theta = 0.1$, $\gamma = 0.1$, $d_1 = 0.15$, $\sigma = 5$, $\rho = 1$.

In Figure 3, for example, one segment includes the fruit from one pepper and the stem from another. From a purely visual perspective, these segments look very reasonable. It's only because humans know that the stem and fruit go together, and must be arranged in a particular orientation, that we would segment the image differently from the algorithm.

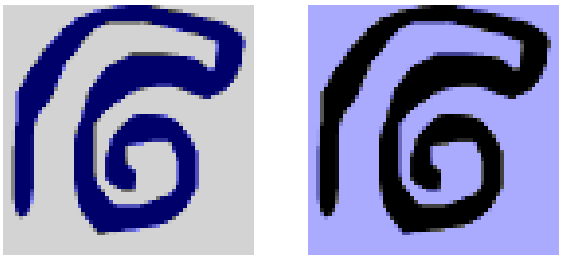
One weakness of the algorithm, mentioned by the authors, is the large number of parameters and difficulty in choosing them correctly. Currently, parameters are chosen by guessing and checking, which obviously presents a large obstacle to the algorithm's robustness and usefulness in industrial applications. There are, however, some general guidelines that can be used to improve the segmentation accuracy.

1. If the image has low contrast levels, increase the first-level intensity scaling factor $\alpha$. This amplifies the difference in intensity between two nodes in the weighted graph matrix.

2. If the image segments are likely to be bordered by very dark or bright regions (as is the case with bright field microscopy cell images), there is a chance that these borders will become salient segments themselves. To avoid this, decrease the coarse-level rescaling factor $\tilde{\alpha}$, since a large $\tilde{\alpha}$ causes nodes with high average intensity to become more disconnected from their neighbors.

3. If the algorithm detects too many segments, then several parameter changes may be beneficial. Decreasing $\theta$ yields stronger connections between nodes; decreasing $\gamma$ provides a stricter salience threshold; and increasing $\sigma$ prevents segment detection on fine levels. All these changes make it more difficult for nodes to be salient and result in fewer segments.

(a) Segmentation without scale invariant saliency measure.



(b) Segmentation with scale invariant saliency measure.

Figure 2: Parameters used: $n = 64$, $\alpha = 10$, $\tilde{\alpha} = 10$, $\beta = 10$, $\theta = 0.09$, $\gamma = 0.08$, $d_1 = 0.15$, $\sigma = 7$, $\rho = 1$.

# 4 References


1. R. Basri, A. Brandt, and E. Sharon, "Fast Multiscale Image Segmentation," The Weizmann Institute of Science, Rehov ot, Israel, 2000.

2. G. Sanders, T. Inglis, H. De Sterck, H. Djambazian, R. Sladek, S. Sundararajan, and T. Hudson, "Mulitlevel Space-Time Aggregation for Bright Field Cell Microscopy Segmentation and Tracking," *International Journal of Biomedical Imaging*, vol. 2010, article ID 582760, 2010.

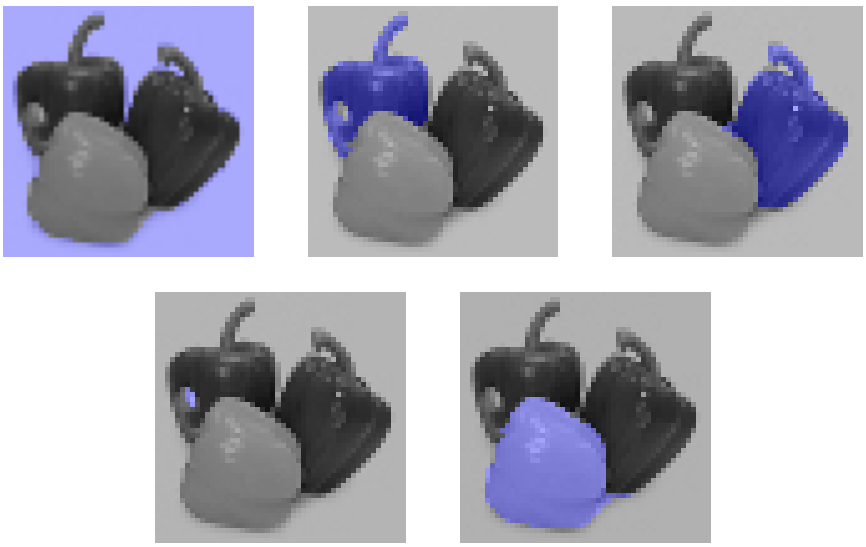


Figure 3: Parameters used: $n = 50$, $\alpha = 50$, $\tilde{\alpha} = 4$, $\beta = 10$, $\theta = 0.1$, $\gamma = 0.15$, $d_1 = 0.15$, $\sigma = 5$, $\rho = 1$.