**EMMPM Gui Instruction Manual**

EMMPM Gui Development Team

Prepared By: EMMPM Gui Development Team  
This document gives an overview of the proper use of the EMMPM Gui application

**TABLE OF CONTENTS**

**Section Page**

LIST OF FIGURES iii

LIST OF TABLES iv

1.0 Image Segmentation using THE EM/MPM GUI 1

1.1 Expectation Maximization/Maximization of Posterior Marginals 1

1.2 Loading the Example Image 1

1.3 Basic Input Parameters 2

1.4 Advanced Input Parameters 2

1.5 Running the Segmentation 3

1.6 Advanced Input Parameters 5

1.6.1 Simulated Annealing 6

1.6.2 Gradient or Edge Penalty 6

1.6.3 Curvature Penalty 6

1.6.4 User Initialization Areas 7

2.0 Miscellaneous Notes 10

2.1 Number of Classes 10

2.2 EM Loops 10

2.3 MPM Loops 11

2.4 Beta 11

2.5 Simulated Annealing 11

2.6 Edge Gradient Penalty 11

2.7 Beta E 11

2.8 Curvature Penalty 11

2.9 Beta C 11

2.10 R Max 11

2.11 Gamma 11

3.0 References 12

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS 8

**LIST OF FIGURES**

**Figure Page**

Figure 1. The EM/MPM Gui File Selection 1

Figure 2. The EM/MPM Gui Image Display Area 2

Figure 3. The EM/MPM Gui Basic Inputs 3

Figure 4 Segmentation Queue 3

Figure 5. Segmentation Queue Active 4

Figure 6. The EM/MPM Gui Output after Segmentation 4

Figure 7. Advanced Options 5

Figure 8. Enabling User Initialization Areas 6

Figure 9. User Initialization Areas Dialog 6

Figure 10. Detailed Callout of the User Interface 7

Figure 11. Initial Segmentation with 2 Classes allows parts of the matrix to be segmented as class 1 8

Figure 12. New segmentation with 3 classes results in more accurate results 9

**LIST OF TABLES**

**Table Page**

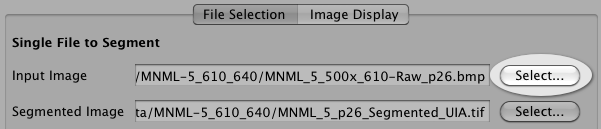
**No table of figures entries found.**  
In your document, select the words to include in the table of contents, and then on the Home tab, under Styles, click a heading style. Repeat for each heading that you want to include, and then insert the table of contents in your document. To manually create a table of contents, on the Document Elements tab, under Table of Contents, point to a style and then click the down arrow button. Click one of the styles under Manual Table of Contents, and then type the entries manually.

# Image Segmentation using THE EM/MPM GUI

## Expectation Maximization/Maximization of Posterior Marginals

One current research grade algorithm is the Expectation Maximization/ Maximization of Posterior Marginals (EM/MPM). A number of papers ([1], [2], [3]) have been written on this technique and the explanations are better left to the developers. The basic premise is to use a mixture of Gaussian distributions to model the physical system. By knowing these various curves one can attempt to predict which pixel of an image should belong to which class designation. The basic EM/MPM algorithm is a nested loop structure with the outer loop being the EM loop and the inner loop being the MPM loop. Several input parameters control how the image is segmented and in the most basic version include the "Beta" and "Gamma" terms. In advanced versions of the EM/MPM algorithm more parameters are available to the user. The EIM project selected this algorithm to be implemented in a more user friendly software package in combination with the enhancements made by other team members during the contracting period. This chapter will walk the new user through segmenting an example image starting with basic controls and moving to using the newer advanced options now available.

## Loading the Example Image

 Figure . The EM/MPM Gui File Selection

Click the "Select" button (See figure 3.1) to display the "Open Input Image" file dialog box. After selecting the example image "MNML\_5\_500x\_610-Raw\_p26.bmp" from the examples directory click the "Open" button from the dialog and the image will be loaded into the EM/MPM Gui. Click on the "Image Display" tab to view the input image (Figure 3.2). Within the "Image Display" tab there are two major sections that include the display of the selected input image and the display of the histogram of the input image. If the default values for the X and Y axis of the histogram are not appropriate to view the detail of the histogram the user can adjust these values using the text input fields found above the histogram plot.

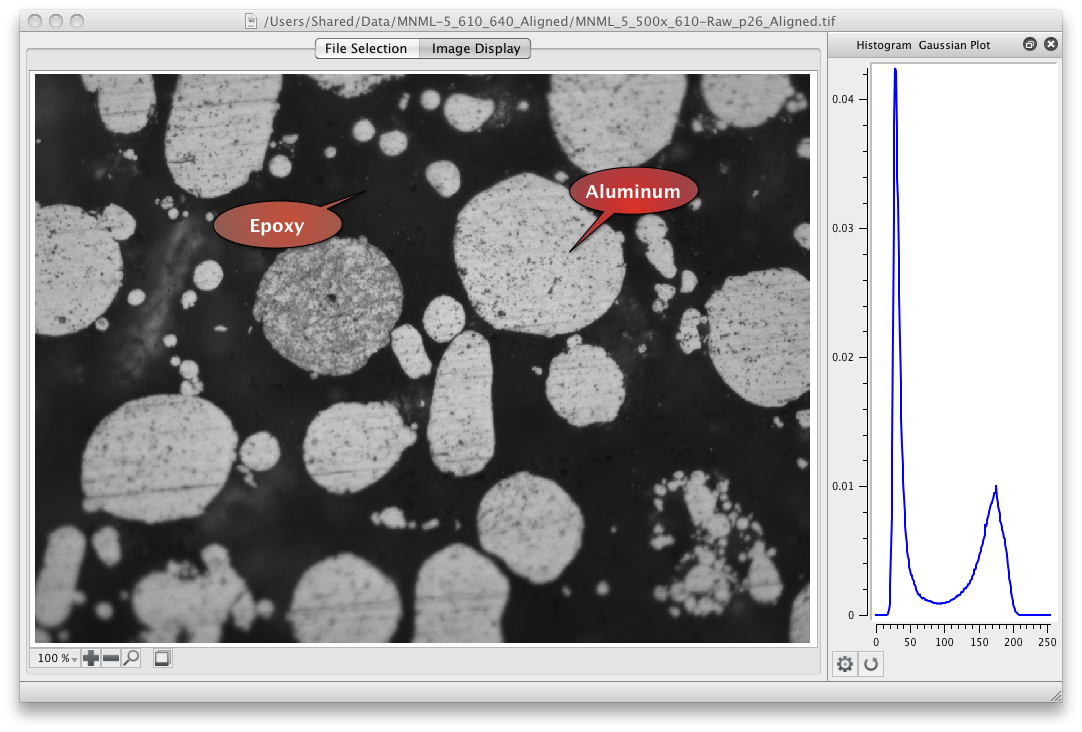


Figure . The EM/MPM Gui Image Display Area

## Basic Input Parameters

The basic inputs to the EM/MPM algorithm are the number of classes, the number of EM loops, the number of MPM loops, a Beta value and a Gamma value. Taking each one of the parameters individually we have the following explanations:

* **Classes:** This is the expected number of phases that can be segmented from the input image. In our example we use "2" because there is a darker epoxy based matrix surrounding a lighter Aluminum second phase.
* **EM Loops:** This controls the number of Expectation Maximization Loops that are performed. The EM Loops control how well the mixture of Gaussians model will fit the input image histogram.
* **MPM Loops:** This controls the number of MPM loops that will be performed. The MPM loops control how well refined the final segmentation will be.
* **Beta:** The Beta term controls how rough or smooth boundaries between each of the segmented phases will be. Another description is "... the spatial interaction parameter and is used to control how likely for two neighboring class labels to disagree."
* **Min. Variance:** This is the minimum value of the variance (standard deviation squared) the will be allowed by the EM/MPM algorithm. The default is 20. Some segmentation runs may require this to be set higher or lower.

## Advanced Input Parameters

An additional checkbox “Use Manual Mu/Sigma/Gray Scale Values” allows the user to enter the initial Mu (Mean) and Sigma (Variance) values for each class. As the user enters this information the histogram plot will update with green curves representing the Mu/Sigma values. It should be noted that the curves displayed on the image histogram curve are only an approximation. The user should also make sure that the gray scale values are unique for each class. When these are filled out to the users satisfaction then the Segmentation can be started.

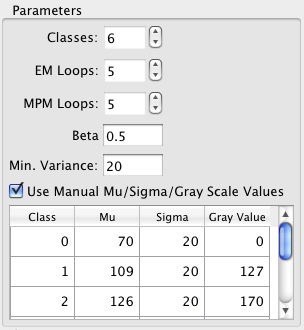


Figure . The EM/MPM Gui Basic Inputs

## Running the Segmentation

Now that we have selected our initial input values we can go ahead and run the segmentation. By clicking on the "Segment" button located below the inputs the image segmentation will start.

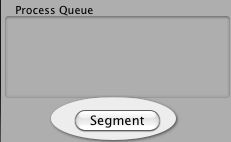


Figure Segmentation Queue

During the segmentation the mixture of Gaussians will be updated on the same plot as the histogram as well as the evolving segmentation. The final segmentation is displayed in the Image Display Area. The red colored plot lines are the individual Gaussian distributions that the EM/MPM generated while the black line is the combined Gaussian distributions. In this case both the combined Gaussian and the individual Gaussian distributions are very close to the same values and so end up laying on top of each other. The separation is more evident as more classes are used to segment the image.

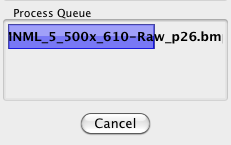


Figure . Segmentation Queue Active

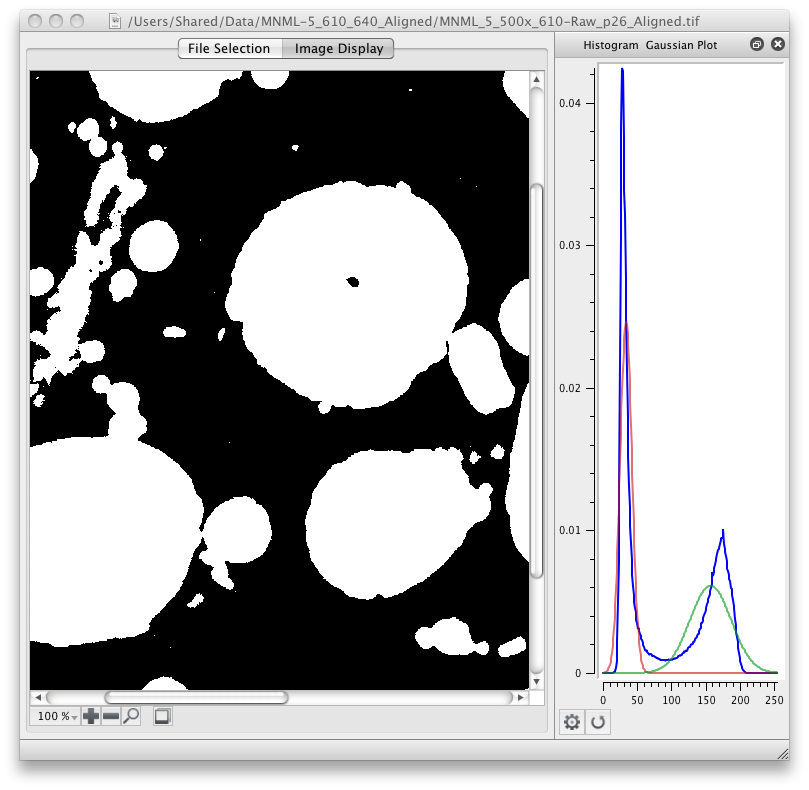


Figure . The EM/MPM Gui Output after Segmentation

## Advanced Input Parameters

Several advanced options are allowed in the user interface as seen in the figure. Each of these are described in practical terms in the following sections.

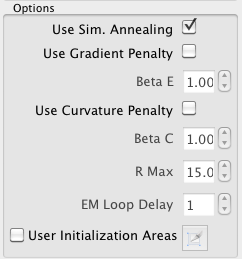


Figure . Advanced Options

### Simulated Annealing

According to [2] the following description of the "Simulated Annealing" is given. *The EM/MPM/SA algorithm was developed to form a preliminary segmentation with a low value of 'Beta', to classify textured regions, which is then progressively refined with larger and larger values of 'Beta' to sharpen the boundaries. In practice, a low value of 'Beta' is equivalent to choosing a high temperature, T , in equation (1). A sufficient number of MPM cycles are run to provide a steady-state classification of pixels. Pixels that have a high probability of being classified as a single region tend to be stable, while those in which the probability of belonging to one of two or more different classes is uncertain tend to set up a dynamic equilibrium classification that changes as the MPM algorithm is executed. The pixels with high probability of classification are frozen in that class and the computation is not continued. Classification of the others continues at a lower temperature. The temperature is gradually lowered until pixel accuracy of classification of the boundaries is achieved.*

Ferb:Users:mjackson:Workspace:EIM-Guides:EmMpmGui:Images:equation_1.png

### Gradient or Edge Penalty

This option attempts to apply a penalty function for large gray value gradients that occur at edges. More information about this penalty function can be found in [4].

* **Beta E:** is a weight to control how much the edge cost information will be dominant in the classification process. Setting this value to Zero will have the same effect as disabling the option.

### Curvature Penalty

This option applies a Curvature Penalty function in an attempt to smooth out boundaries between classes. The paper presented in [4] is the best place for an in depth explanation of the implementation of the curvature penalty function.

* **Beta C:** A user-specified weight to control how much the curvature term dominates the classification process. Performing a morphological filtering using more than one structuring element incorporates the curvature penalty. Rather than using a single circular element, eight circular structuring elements with different radii are used. Setting this value to Zero will have the same effect as disabling the option.
* **R Max:** The maximum radius in pixels is specified by this parameter. 15 pixels seems to be a good starting point but note the larger this value is the slower the algorithm will run so the user should take into consideration any time constraints needed when running the program.
* **EM Loop Delay**: This parameter is the number of EM loops to delay before applying this penalty function.

### User Initialization Areas

This option allows the user to designate areas of the image as having the approximate same mean and standard deviation of the final segmentation. This can be useful due to the algorithm used to pick the initial mean and standard deviation values. In some cases the model computed by the EM/MPM will not converge accurately to the image histogram.



Figure . Enabling User Initialization Areas

By designating similar areas for each phase the initial estimates for the mean and standard deviation become much more accurate. This should allow for a more accurate segmentation and also allow the user to need less EM loops, which will reduce the time needed to run the EM/MPM algorithm. To enable this option one needs to check the "User Initialization Areas" check box.

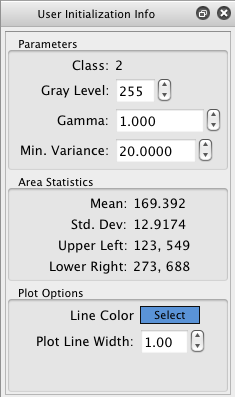


Figure . User Initialization Areas Dialog

To designate areas click on the "User Initialization Area Lasso Tool" and then draw out an area in the image by click dragging the outline of the area you want. If you do not get it exactly correct you can always resize and move the area after this operation. When you release the mouse button a dialog box will appear where you can set the initial values for that particular class. Because output images are saved as a <b>gray scale</b> image the setting of the Gray Level value is important at this point. While the defaults will attempt to spread out the values of the gray levels for each class sometimes this will create an image that is difficult for the human vision to discern differences between the various classes. The user should always take care to check this value for each class that they designate. Zero (0) means pure black and 255 means pure white. For a 2-class system the typical values that are used are 0 and 255. Setting the gray level to other values also has other implications that are discussed later on. By clicking/moving/resizing an individual area the details of that area will be displayed in the "User Initialization Info" palette. If the palette is **not** shown the user can use the **Window** menu to make it visible.

Once the user has designated areas that represent each class the segmentation can then be run again to see if this has produced results that better match the intended segmentation. In this case the segmentation produces results where parts of the matrix are segmented as the light phase as shown in figure 1.12 where the left side of the image is the resulting segmentation of the right side of the image.

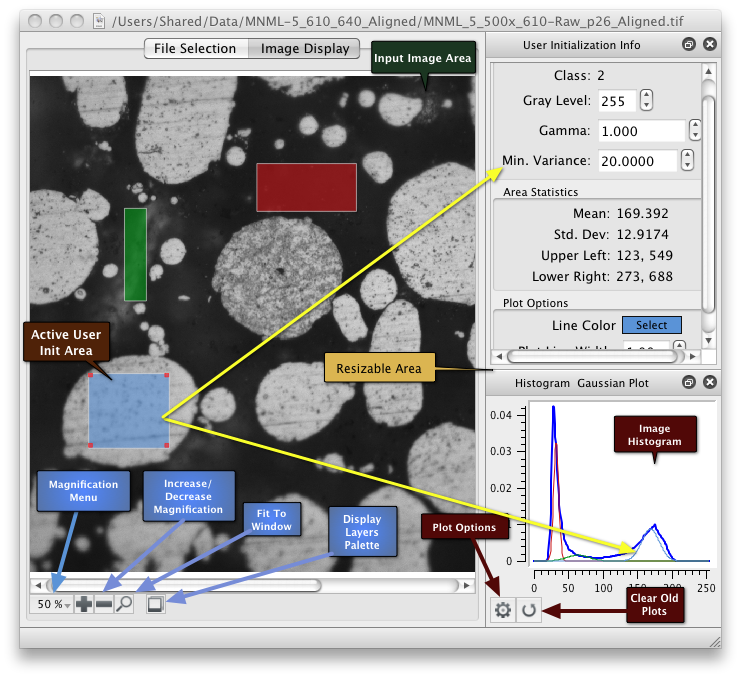


Figure . Detailed Callout of the User Interface

Allowing the EM/MPM algorithm to use more classes to perform the segmentation then joining the classes into a single gray scale value when the final image is written can sometimes alleviate this type of segmentation error. This can be exactly achieved through adding an additional user initialization area but setting the output gray scale value to the same value (Zero in our example) as the matrix. Running the segmentation with all other values the same results in a segmentation that can be considered more accurate to the original input image.

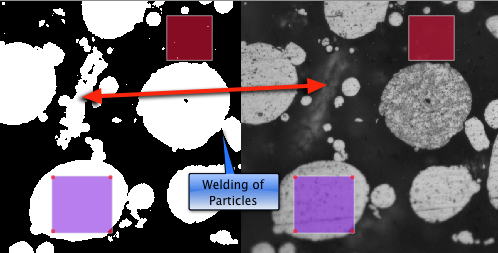


Figure . Initial Segmentation with 2 Classes allows parts of the matrix to be segmented as class 1

Also note in the image that areas where the particles come close to each other but still have some matrix (class 0) between them get "welded" together in the first segmentation. In the second segmentation with the addition of another class the welding or joining of the particles does not happen. The last change from the first segmentation was the adjustment of the "Beta" value from 0.5 to 3.0. This allows some of the smaller holes to fill back in with the proper phase.

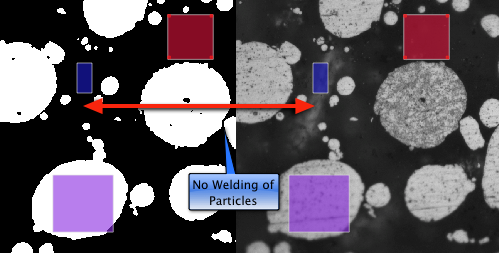


Figure . New segmentation with 3 classes results in more accurate results

The ability to directly edit values in the "User Initialization Info" such as the Gray Value, Display Color and Gamma value allows the user some flexibility on how the segmented image is saved. The **Gamma** value has the effect that if a larger value is used for one class versus another then the EM/MPM will tend to assign the pixel to the class that has the larger gamma value instead of each class having equal probability as in the standard EM/MPM algorithm. Editing the gray scale value can have interesting consequences in that the user can effectively segment with more than the final number of classes and have those classes automatically combined when the image is written by simply assigning the same gray scale value for the classes that the user want to all contribute to the same final class in segmentation. For instance if the user wants a final binary segmentation with gray levels of zero and 255 but designates 4 total classes to use, 2 for the matrix and 2 for the phase, the user can set the two matrix classes to have a gray scale of 0 and the 2 phase classes to have a gray scale of 255. Even though the algorithm is using 4 classes the final image written to disk will only have 2 gray scale values: 0 and 255.

Another idea is to save the actual EM/MPM label map by setting the gray scale values to the value of the class. This will effectively save the EM/MPM label map but the resulting image will appear all black in common image viewers. The user would need to apply a custom color table to the image in their favorite image editor/viewer in order to visually enhance the image to the point where the human eye can see the difference between the segmented regions.

# Miscellaneous Notes

## Number of Classes

Use more classes if you need more Gaussians to fit odd shaped peaks. In this sample, there are 2 peaks in the histogram and the method successfully fits two Gaussian peaks within it that gives a good segmentation.

## EM Loops

Quality of optimization of the Gaussian fit to the histogram. Too few gives bad fit to histogram.

## MPM Loops

Controls the degree of convergence of the segmentation. Too few gives a speckled image.

## Beta

Typically, without the morphology regularization, beta of 0.5 gives jagged edges of precipitates. Too low of a value leads to jaggy boundaries whereas too high a value leads to welding of particles.

## Simulated Annealing

Simulated annealing raises the Beta value as the segmentation goes on. On samples with really fine channels, it tends to weld things together because of capillarity (Beta is the pairwise interaction between pixels. Really large values of Beta simulate real large interfacial energies. Regions in contact tend to grow together to remove interfacial energy). Not using simulated annealing can lead to speckled images. Using it can lead to particle welding.

## Edge Gradient Penalty

The gradient penalty detects systematic drops in intensity and allows you to split the particles apart, even though the intensity in the channel doesn't ever drop to what you would put as a threshold. Not using a gradient penalty can lead to intensity dips being smoothed out. Using too high a gradient penalty can cause individual pixels to be segmented because of Poisson noise that changes their intensity vis a vis neighboring pixels.

## Beta E

Segmentation is very sensitive to Beta E values. On some samples, you have a lot of Poisson noise, and it's not useful to raise Beta E above maybe 2 or 3. If you do, you just get sand, because it splits the high brightness pixels from their neighbors. Too low a Beta E can lead to jaggy boundaries.

## Curvature Penalty

There are two ways of smoothing a curve: raising Beta E and applying a curvature penalty. Raising Beta E makes particles consolidate, Curvature penalties do not.

## Beta C

You may be able to get smoother boundaries by increasing this value, but morphological filtering does not have prior knowledge in it. If you get unrealistic segmentations with Beta C, they will show more about the actual structuring element used in the morphological filter than the material or prior knowledge. Too high a value of Beta C can lead to unrealistically shaped particles. Too low can lead to jaggy edges that you're tempted to smooth out with Beta E increases, which causes particles to weld.

## R Max

R Max is (roughly) the maximum number of pixels you can erode back from the boundary. You have to use a larger value of Beta C if you use a smaller value of R Max. Larger R Max values lead to longer segmentation times. Too low a value can lead to jaggy interfaces. Too high can lead to excessive wait times for segmentations.

## Gamma

Gamma is analogous to the chemical potential. Increasing Gamma leads to bigger particles because it favors growth of particles, at the expense of the matrix. We're off by a scaling value on this. The original papers on MPM concluded that Gamma had little influence on the final segmentation. But, magnitudes on the order of 100 lead to noticeable results. A high value of Gamma for a class causes pixels to leave that class, all other things being equal. Too low a value can lead to particles that are too small. Too large can lead to particles that are too large.

# References

[1] Mary L. Comer, The EM/MPM Algorithm for Segmentation of Textured Images: Analysis and Further Experimental Results, IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 9, NO. 10, OCTOBER 2000.

[2] J P Simmons, P Chuang, M Comer, J E Spowart, M D Uchic and M De Graef, Application and further development of advanced image processing algorithms for automated analysis of serial section image data, Modelling Simul. Mater. Sci. Eng. 17 (2009).

[3] David Doria, Expectation Maximization of Gaussian Mixture Models in VTK - Release 0.00, September 21, 2010, Insight Journal [http://hdl.handle.net/10380/3218]

[4] J. Dumke and M. Comer, "Automated segmentation of alloy microstructures inserial section images," Proceedings of SPIE, vol. 6498, p. 64980E, 2007