Inertial Cavitation Bubble Analysis Software User Guide

Aditya Bhatnagar

July 2020

1 Introduction

This user guide is an attempt to familiarize the user/reader with the software developed for the purpose of Inertial Cavitation Bubble Analysis as well as provide a deeper understanding of the algorithms and methodologies implemented behind the software. This user guide will go over in detail how to maneuver the visual interface of the software in the next section and the algorithms and procedures involved in the mask generation and analysis for each frame in subsequent sections.

The visual front-end interface for the software is developed in MatLab's App Designer and is segmented into four separate "blocks" or "panels." These four panels are:

- The Control Panel - the main source of manipulating settings relevant to mask analysis and input/output
- The Mask Overlay Preview Panel - used for viewing a preview of the bubble masks
- The Main Viewer Panel - used for viewing various data about individual frames
- The Bubble Analysis Panel - used for viewing the results of bubble analysis and any Fourier Series analysis results

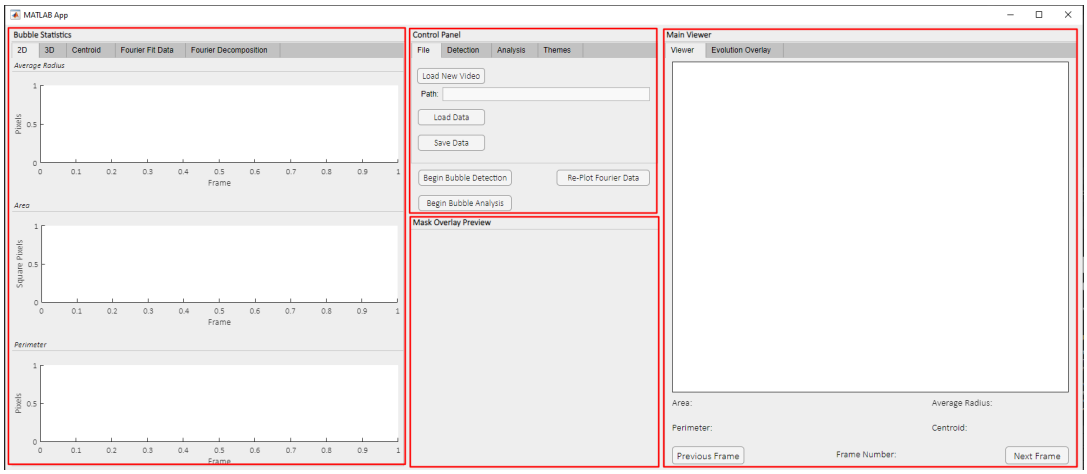

Figure 1: The four main panels of the interface

The bubble mask generation is done through a series of algorithms designed to target and isolate objects in the image based on various characteristics such as edges, color, size, location, and average pixel intensity. Two masks are initially generated, one generated by thresholding the gray-scale image and one by applying a Sobel edge operator to the entire image. These two masks are then compared through extensive logic to decide which combination of masks to use.

The bubble analysis relies completely on the generated masks as multiple pieces of information are generated through the use of the *regionprops* MatLab function, such as the mask centroid, area, and perimeter. The user has the option to fit a Fourier Series to the X and Y points on the perimeter of the bubble with multiple settings for the Fourier fit being available in the Control Panel's Analysis tab, such as: minimum arc length between perimeter points, a static or dynamic number of terms, the (maximum) number of terms (maximum only if the dynamic number of terms option is selected), and plotting options.

2 Getting Familiar with the Interface

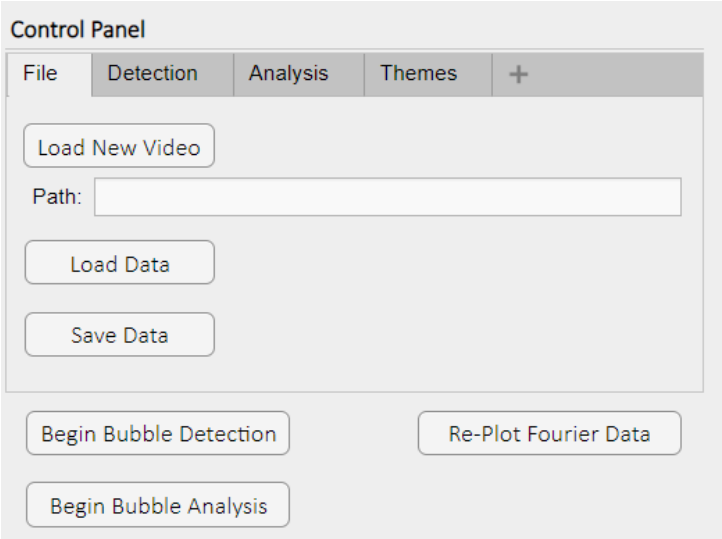

Figure 2: An image of the control panel layout on startup

2.1 Control Panel

The "Control Panel" will be the main location where users will interact with the software. Within the panel, there are multiple tabs that contain controls and settings for different parts of the bubble analysis, which include:

- File Tab — used for basic video input/output
- Detection Tab — contains multiple settings and toggles relevant to mask generation, frame preprocessing, and bubble detection
- Analysis Tab — used for control settings relevant to the analysis of the bubble through the image mask
- Camera Settings Tab — used for converting pixels to microns and frames to seconds
- Theme Tab — does not contain settings relevant to bubble analysis, can change the color theme of the program

The control panel at this time also contains four buttons that initiate different functions within the program:

- Begin Bubble Detection Button — this button triggers the generation of image masks for each frame in the video using a predefined algorithm (the algorithm will be covered later)
- Begin Bubble Analysis Button — this button triggers the analysis of previously generated masks (the algorithm will be covered later)
- Re-Plot Fourier Data Button — if the user decides to fit a Fourier Series to the mask, this button refreshes the plots generated
- Export To Excel Button — if the user wishes too, all of the current analysis will be stored in an excel table for later viewing and further excel analysis

2.1.1 File Tab

The File tab is used for loading new videos for analysis or for loading/saving data that has previously been analyzed. Loading a new video for analysis can be done by clicking the "Load New Video" button at the top of the tab group. This opens up a dialog box in which the user can choose an .avi file to load into the software. Once the video has been loaded in and all frames processed, the "Path" text box populates with the file path to the video so that the user may verify the correct video

is loaded. If an incorrect video has been loaded, the user may simply click the "Load New Video" button again to clear the old video from memory and load a new one. It is important to note that at this time, only the processing of .avi file types is supported.

Underneath the button to load new videos and the text box containing the file path, the user will find two buttons labeled "Load Data" and "Save Data". The "Load Data" button is able to load data from a previously analyzed video and plot all of the results in the current program instance. The "Save Data" button is similar in that the user should be able to save all data from an analysis session to a single file once the analysis is complete. This file may later be opened using the "Load Data" button so that the user does not have to reanalyze the same video. The data is stored in MATLAB's .mat format. These two buttons have similar text boxes to the "Load New Video" button's "Path" text box as the text boxes update with the full path of the file when the button is selected.

2.1.2 Detection Tab

The Detection tab is used for controlling and tuning various settings relevant for frame preprocessing, bubble detection, and mask generation. Within the tab the user will find 5 different checkboxes:

- Ignore First Frame Check Box — This checkbox will ignore the first frame of the video during bubble detection and automatically add it to the list of frames to ignore for bubble analysis.
- Multibubble Analysis Check Box – This checkbox will inform the application to expect multiple bubble per frame, this feature is under development currently.
- Run Detection Both Ways Check Box — Run the detection algorithm forwards in time and backwards in time and compare the resulting masks. If disabled the analysis only runs forwards.
- Frame Ignore Warning Check Box — Allows the user to enable/disable alerts when selecting a frame to ignore or use during bubble analysis.
- Preprocess Frames Check Box — Allows the user to toggle frame preproceesing before bubble detection algorithms begin.

Underneath the Preprocess Frames Check Box the user will find a button group that determines the frame preprocessing method and strength. The user may either sharpen or soften the image in each frame with imsharpen or imguassfilt, respectively. The numeric edit field under the button group determines the strength of the filter. A higher number in the edit field correlates to a stronger filteration of the image.

2.1.3 Analysis Tab

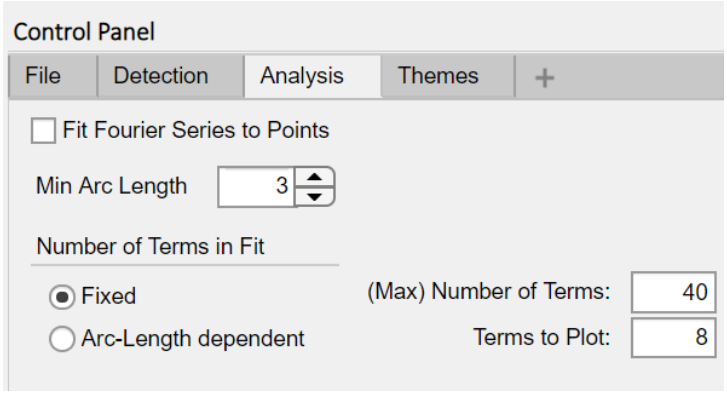

Figure 3: A screen capture of the analysis panel on program start up

The Analysis tab is where a large majority of the settings that will be used for analysis reside. The tab offers the user the option to fit a Fourier Series to the perimeter points on the image mask. There are subsequent settings for the Fourier fit, such as the minimum pixel arc length between points, whether a fixed or dynamic number of terms should be used in the fit, the number of terms to plot — or the maximum number of terms if the user opts for a dynamic number of terms — and how many terms should be plotted after the analysis has been completed.

2.1.4 Camera Settings Tab

The Camera Settings Tab contains only two edit fields, a button group, and an update button. The edit fields contain default values 0 but the user may input the camera/video specific conversions and click update to generate a set of plot data that is in the converted units. The button group allows the user to toggle between the standard Pixel/Frame units and the Micron/Seconds unit. While during the analysis the program will plot with whichever units are selected at the time, the user may change the selection afterwards and click update to refresh the plots.

2.2 Mask Overlay Preview Panel

This panel is used to display an overlay of the generated image mask on the original frame once the Bubble Detection Algorithm has completed. The purpose of this component is to inform the program to ignore certain frames during analysis due to the mask for those frames being insufficient or incorrect. Adding and removing a frame from the list of frames to ignore can be done by left-clicking on the desired frame. It is understandable that for some frames, the small preview may be insufficient and as such the user may right click on the image to be previewed and selected the menu option to open the image in a separate, much larger window. The ability to zoom in on the image even further is under development.

2.3 Main Viewer Panel

The main viewer panel consists of two tabs: the "Viewer" tab, and the "Evolution Overlay" tab. Both tabs' purpose is to display information about the completed analysis to the user.

2.3.1 Viewer Tab

The "Viewer" tab shows an overlay of the following information as calculated by the analysis algorithm onto the original frame:

- Generated mask boundaries
- Centroid
- Perimeter tracking points
- Points used for Fourier fit (if option is selected before analysis)
- Resulting Fourier fit (if option is selected before analysis)

Below the plot showing the overlay of the information in the "Viewer" tab, the user can find additional information about the current frame in view such as:

- Mask area (square pixels)
- Mask Perimeter (pixels)
- Average Radius (pixels)
- Centroid coordinates (pixels)
- Current frame number

Changing the frame number and pressing enter on the keyboard will automatically jump to the entered frame if it is within the range of [1, Total Number of Frames in the Video].

2.3.2 Evolution Overlay Tab

The "Evolution Overlay" tab shows an overlay of each of the mask perimeters from every frame with a color map ranging from red to blue. Red perimeters are indicative of bubble perimeters earlier in the video and transitions to purple and then blue signify perimeters later in the video.

2.4 Bubble Statistics Panel

The "Bubble Statistics Panel" is where the majority of the results from the analysis reside. It contains five tabs that each contain different information from the analysis. The five tabs are:

- 2D
- 3D
- Centroid
- Fourier Fit
- Fourier Decomposition

2.4.1 Non-Optional Analysis Tabs

These tabs produce data every time the "Begin Analysis" button is pushed.

- The "2D" tab contains contains three plots: an Average Radius plot, an Area plot, and a Perimeter plot, each of which depict their namesake over the course of the video. From these plots, the user can easily see how the various quantities of the mask change over time.
- The "3D" tab contains two plots: a bubble Surface Area and Volume plot. While these plots currently reside at zero for the entirety of the video, the process of using axi-symmetry assumptions for calculations of these quantities is under development.
- The "Centroid" tab consists of one plot that depicts the centroid positions for each frame in an X-Y plane. These points also utilize the same color map as the "Evolution Overlay" plot as red points are near the beginning of the video and blue points are towards the end.

2.4.2 Optional Analysis Tabs

These tabs display data about the Fourier fit and as such only display data when the option to "Fit Fourier Series To Points" is selected before bubble analysis begins.

- The "Fourier Fit" tab contains a multi y axis plot that depicts the Normalized Fourier Term Amplitude for term N on the left axis, where N is the term number and the radius of the bubble as calculated by the Fourier Series on the right axis. Both these quantities are plotted against time on the x axis. The number of N terms to display is the value previously set in the "Analysis" tab of the "Control Panel". Each Normalized Fourier Term Amplitude plot is part of a larger color map that depicts which term that plot is representing with purple being N = 2 and yellow being N = the previously set value.
- The "Fourier Decomposition" tab contains various options for decomposing the bubble into its representative terms:
  - A numeric edit field to enter the frame to decompose
  - A numeric edit field to enter the number of terms to decompose to
  - How to sort the resulting plots. "Descending" orders the plots by highest to lowest magnitude of the normalized FT amplitude, and "Ascending" orders the plots lowest to highest magnitude.
  - An option to maintain the same color map as the Normalized Fourier Term Amplitude plot. This is particularly useful when the number of terms to decompose to is the same as the N number of terms that have been plotted previously.

  The resulting plots will be displayed below the settings panel and the user is able to scroll to the right to view the other plots.

3 Bubble Mask Generation

The mask generation for each frame is based on two visual cues: color and edges. One mask is generated based on color and one mask is generated based on the edges found in the image and both masks are compared to determine the final mask for each frame. Despite using more than one method to generate the mask, this method is not infallible as extenuating circumstances can fool the algorithm. These circumstances are the reason for the user needing to select frames to ignore during the analysis as the results would not be usable.

3.1 Mask Generation Algorithms

Algorithm 3.1.1: Color-Based Image Mask Generation:

Step 0:
Read in a matrix representing the gray-scale image

Step 1:
Apply a Gaussian filter with a standard deviation of 1 and 3x3 kernel to the matrix with *imgaussfilt*

Step 2:
Generate a pixel value threshold based on Otsu's Method with *graythresh*

Step 3:
Binarize the image based based on the previously generated threshold value multiplied by 1.25, so that any values below the threshold are set to 0 and values above are set to 1

Step 4:
Flip black and white in the image with *imcomplement* so that dark areas (where the bubble may be are now marked with logical true

Step 5:
Since the dark vignette areas are now marked with true as well, use *imclearborder* to set the vignette areas to false (0)

Step 6:
Clean up the resulting image with *bwmorph* to remove stray pixels and bridge small gaps between pixels

Step 7:
Remove objects unlikely to be the bubble based on the Remove Outliers Algorithm

Step 8:
Find the number of connected components in the image with *bwconncomp*

Step 9:
If the there is still more than one connected component, attempt to isolate the object most likely to be the bubble with the Isolate Object Algorithm

Step 10:
Fill any holes in the mask with *imfill* and return the mask

Algorithm 3.1.2: Edge-Based Image Mask Generation:

Step 0:
Read in a matrix representing the gray-scale image

Step 1:
Generate an edge threshold using the Sobel operator and *edge*

Step 2:
Multiply the threshold by 0.01 and binarize the image based on a Sobel operator and the new threshold to generate "false edges" representative of areas with high noise using the *edge* command

Step 3:
Dilate all the lines in the resulting logical mask with 3 pixel long 0 and 90 degree line-shaped structuring elements using *imdilate*

Step 4:
Use *imcomplement* to flip true and false areas in the mask so that areas of low noise (possible bubble locations) are now marked as true

Step 5:
Since the low noise vignette areas are now marked with true as well, use *imclearborder* to set the vignette areas to false (0)

Step 6:
Fill any holes in the image with *imfill*

Step 7:
Remove objects unlikely to be the bubble based on the Remove Outliers Algorithm

Step 8:
Find the number of connected components in the image with *bwconncomp*

Step 9:
If the there is still more than one connected component, attempt to isolate the object most likely to be the bubble with the Isolate Object Algorithm and return the resulting mask

3.2 Mask Clean-Up and Object Isolation Algorithms

Algorithm 3.2.1: Remove Outliers:

Step 0:
Read in a matrix representing the gray-scale image

Step 1:
Calculate the number of connected components with *bwconncomp* and the centroid of those components with *regionprops*

Step 2:
Using the information from *bwconncomp*, get rid of objects smaller than or equal to 50 square pixels and objects larger than half the image size

Step 3:
Get rid of any objects with a centroid farther than 200 pixels from the center of the image and return the cleaned-up mask

Algorithm 3.2.2: Isolate Object:

Step 0:
Read in a matrix representing the gray-scale image

Step 1:
Calculate the number of connected components with *bwconncomp* and the centroid and circularity of those components with *regionprops*

Step 2:
Using the information from *bwconncomp* get the index of the largest object in the cell array

Step 3:
Get the index of the object closest to the center of the object in the previous frame using the cell array returned by *regionprops*

Step 4:
Get the index of the object that covers the area with the lowest pixel average on the original image

Step 5:
Calculate the "significance" value for each object using the following formula:
$\text{Significance} = \text{Distance from Center} \div Area \times AverageValueofPixelsCovered$
And get the index for the object with the lowest "significance" value, therefore the most likely object to be the bubble

Step 6:
If three or more of the indices match, that object is the one to keep. If less than three indices match, use the most circular object as calculated by *regionprops*

Step 7:
Get rid of the other objects and return the mask

3.3 Mask Comparison Logic

Logic Block 3.3.1: Basic Mask Comparison Logic Psuedocode:

```
If one of the masks is empty
    The final mask is the mask that is not empty
Otherwise if both masks contain something
    If both masks contain an area that is similar and size and have centroids close to each
    other
        The final mask is a logical sum of the two masks
    Otherwise if the Edge-Based mask is larger and both centroids are close to each other
        If there is no overlapping region between the two masks
            Create a new mask from the logical addition of the two masks
            Isolate the object most likely to be the bubble as determined by the Isolate
             Object Algorithm
            The final mask is the mask returned by the Isolate Object Algorithm
        Otherwise if there is an overlapping region between the two masks
            The final mask is the overlapping region
    Otherwise if the Color-Based mask is larger and both centroids are close to each other
        If one of the objects has a larger circularity than the object in the other mask (as
         determined by regionprops
            The final mask is the outcome of the Extended Mask Comparison Logic
        Otherwise if the objects in both masks are similar in circularity
            The final mask is the overlapping region between the two masks
    Otherwise if both masks are close in size but not close in centroid location
        Use the mask with the object that is closer to the center of the image
    If all else fails
        Use the more circular mask
```

Logic Block 3.3.2: Extended Mask Comparison Logic Psuedocode:

```
Calculate the circularity value of the mask with the higher circularity (the masks are
passed into the function as a higher circularity mask (HCM) and lower circularity mask
(LCM)).

Calculate the number of objects and the circularity for those objects in a mask created from
where either mask is true (Either Mask)

Calculate the number of objects and the circularity for those objects in a mask created from
where both masks are true (Both Mask)

If the number of objects in the Either mask is greater than one

    Isolate the most likely object based on the Isolate Object Algorithm in the Either mask

    Calculate the circularity of the object in the new Either mask

If the number of objects in the Both mask is equal to 1
    If the Either mask has a higher circularity than the HCM mask

        The final mask is the Either mask

    Otherwise if the Both mask has a higher circurlarity than the HCM mask

        The final mask is the Both mask

    Otherwise if none of the above conditions are met

        The final mask is the HCM mask

Otherwise if the number of objects in the Both mask is equal to 0

    If the Either mask has a higher circularity than the HCM mask

        The final mask is the Either mask

    Otherwise if none of the above conditions are met

        The final mask is the HCM mask
```

3.4 Mask Generation Function Psuedocode

```
function Output Matrix = Generate Mask(Input Matrix)
    Get the size of the input matrix
    Create an output matrix of the same size
    For 1 to the Number of Frames in the Video
        Index the input matrix to get a target frame
        Generate a mask based on color for the target frame
        If it's not empty, calculate its centroid and circularity
        Generate a mask based on edges for the target frame
        If it's not empty, calculate its centroid and circularity
        Compare the masks using the Basic Mask Comparison Logic
        Check the final mask one last time for extra objects, remove them if present using the Isolate Object Algorithm
        Assign the final mask to its proper location in the Output Matrix
    End
end
```

4 Bubble Mask Analysis

The analysis of the resulting masks is done with a mixture of custom and in-built functions to MATLAB, the most useful of these functions being *regionprops* to calculate and generate basic information about the 2 dimensional mask, such as:

- Centroid
- Area
- Perimeter

These three quantities, along with the average radius of the mask as calculated by the 50 perimeter "tracking points", are standard in the analysis and cannot be toggle on or off. However, through options in the "Control Panel," there are certain analysis options the user can select should they be desired. Namely, the option to fit a Fourier Series to the points on the perimeter of the mask, and analyze the resulting fit. The following options are things the user may alter in the Fourier Fit of the points:

- The minimum arc length between points on the perimeter that the Fourier Series will be fit to
- The number of (maximum) terms in the fit — The user may select that the number of terms in the fit be dependent on the number of points in the fit, in this case the value entered in this field will be used as the absolute maximum number of terms in the fit.
- The number of terms to plot in the Fourier Fit tab in the "Bubble Statistics" panel

After the analysis has been completed the user is able to see most of the results of the analysis in the "Bubble Statistics" panel. The various tabs are there to assist the user in navigating the results. The 2D tab contains the quantities calculated in the standard analysis of the bubble, the 3D tab will contain information about Surface Area and Volume, based on axi-symmetry calculations (this feature is under development). The Centroid tab shows a plot of all the X-Y coordinates the centroid of the bubble takes over time with a red-blue color map applied to indicate the variation over time. The Fourier Fit tab shows the Normalized Fourier Term Amplitude over time on one axis against the radius of the bubble on the other axis. The Normalized FT plots have a different color map applied to them which depicts the term number. In the Fourier Decomposition panel, the user is able to decompose the Fourier Fit into its respective terms for a specific frame. If the number of terms to decompose to is the same as the number of terms that have been plotted in the first place, it's useful to have the Maintain Colormap checkbox enabled to easily identify which plots correspond to which term despite the fact that a legend entry also provides this data.

While there aren't as many bespoke algorithms for the analysis as for the mask generation, one important algorithm that is used is Perimeter Points algorithm. The Perimeter Points algorithm is used twice in the analysis for each frame. The first time is to generate 50 "tracking points" for each frame which are used to calculate the average radius of the mask. The second time is to generate the points to be used for fitting the Fourier Series to the mask, where the number of points is dependent on the user-set minimum arc length between the points. This algorithm takes in the logical mask, centroid (as calculated by *regionprops*), and number of tracking points (N) to generate as input and returns an Nx2 matrix with evenly angularly spaced tracking points on the perimeter, where each row corresponds to an (X, Y) perimeter point. Due to the precision required in the fit, this algorithm is designed to generate points at a sub-pixel level through the use of linear interpolation.

4.1 General Mask Analysis Algorithms

Algorithm 4.1.1: Perimeter Points Algorithm:

Step 0:
Input the logical mask, (X, Y) centroid, and number of tracking points to return

Step 1:
Thin the logical mask to just the perimeter using *bwperim* and *bwmorph* in sequence

Step 2:
Calculate the angular step size by dividing 2π by the number of tracking points and create the vector of angles to look along for a tracking point

Step 3:
Using *ndgrid*, create X and Y index matrices

Step 4:
Using the X and Y index matrices, use *atan2* to create a matrix of radian values for each element in the matrix, centered about the centroid of the object in the mask. Anywhere the matrix of radian values is less than 0, add 2π so that the range of the matrix radian values is between 0 and 2π

Step 5:
Repeat the following steps until the number of iterations matches the number of elements in the vector of angles to look along

Step 6:
Index the vector of angles to get a angular direction to look in (look direction) and set up a "working radian" matrix for that angle by subtracting the look direction from each element in the radian values matrix, setting any elements not corresponding to a one in the logical mask matrix to NaN

Step 7:
Find the upper and lower bounds of the values closest to the desired angle. The upper bound will be the minimum positive value in the matrix and the lower bound will either be the largest negative value or the largest positive value (if the angle is small enough not to cause the working radian matrix to go below 0)

Step 8:
Find the indices of the elements corresponding to the upper and lower bound values if more than one element index is returned for either of the bound values, use the index of the element that is farthest away from the centroid of the object in the mask

Step 9:
Find the distance (radius) from the center of those elements (pixels) to the bubble centroid using the distance formula

Step 10:
Get a weighted average radius for the look direction using the following formula:

$$UpperBoundRadiusWeight = 1 - \left|\frac{UpperBound}{UpperBound - LowerBound}\right|$$

$$LowerBoundRadiusWeight = 1 - \left|\frac{LowerBound}{UpperBound - LowerBound}\right|$$

$$WeightedRadius = UpperRadiusWeight \times UpperRadius + LowerRadiusWeight \times LowerRadius$$

Step 11:
Get the (X, Y) values of the perimeter point by multiplying the weighted radius by the cosine and sine of the look direction angle

4.2 Fourier Series Fit Algorithms

Algorithm 4.2.1: Fourier Series Fit Algorithm:

Step 0:
Read in a matrix of perimeter points, the minimum arc length between points, the maximum number of terms in the fit equation, and a boolean on whether or not to use an adaptive number of terms in the fit equation

Step 1:
If an adaptive number of terms are to be used, use the Adaptive Number of Terms Algorithm to determine the number of terms to use

Step 2:
Create the function files to fit the X and Y points to based on the number of terms to use in the fit equation

Step 3:
Use *fittype* to create a fit type using the generated function files

Step 4:
Use *fit* and the generated fittypes to fit the X and Y points to their respective equations and return the *cfit* objects containing values for all the coefficients in the fit

Step 5:
Use *feval* to generate a series of points for plotting the fit and return these vectors

Step 6:
Delete the generated function files

Algorithm 4.2.2: Adaptive Number of Terms Algorithm:

Step 0:
Read in a value for the maximum number of terms

Step 1:
Do a preliminary fit to determine the radius of the bubble. X values are fit to the equation $a0 + a1cos(t)$ and Y values are fit to the equation $b0 + b1sin(t)$. The radius is calculated as $\sqrt{a1^2 + b1^2}$

Step 2:
The adaptive number of terms value is calculated with: $NumberofTerms = \frac{r \times \pi}{MinimumArcLength}$ where the Minimum Arc Length value is the value set by the user for the minimum arc length between perimeter points to fit the Fourier Series to

Step 3:
If the calculated number of terms is less than the maximum number of terms, use the calculated number of terms value, otherwise use the maximum number of terms value

4.3 Mask Analysis Function Pseudocode

function Output Struct = Analyze Bubble(Masks, Min Arc Length, Do Fit, Max Number Terms, Adaptive Terms, Frames to Ignore)

Get the number of frames by looking at the size of the mask matrix
Define the output struct with the fields: Centroid, Tracking Points, Area, Perimeter, Perimeter Points (plotting purposes), Average Radius, Surface Area, Volume, Fourier Fit Points, X Points Fourier Fit (cfit), Y Points Fourier Fit (cfit), X Fourier Series Points (plotting purposes), Y Fourier Series Points (plotting purposes)
For 1 to the Number of Masks in the Matrix
If the current index is present in the vector of frames to ignore then set all values in the struct for that frame to NaN
Otherwise, get the target mask, use *regionprops* to calculate Centroid, Perimeter, and Area, use the Perimeter Points Algorithm with an input of 50 to get the tracking points, and calculate the Average Radius by averaging the distances from the centroid to each tracking point. Then if the user wishes to fit a Fourier Series to the points, use the Fourier Series Fit Algorithm to fit the equation to the points. Write all this information to the struct and iterate the index.
End

end