SOFTWARE FOR THE ANALYSIS OF 3D BIOLOGICAL DATA SETS

A Design Project Report

Presented to the Engineering Division of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering (Electrical and Computer)

by

Adam McCann

Project Advisor: Bruce Land

Degree Date : May 2011

EXECUTIVE SUMMARY

This project is the creation of a software interface and data processing toolbox tailored to the specific needs of a neurobiology research group. The project scope and requirements were set in collaboration between Cornell University's Electrical and Computer Engineering Department and Cornell's Department of Neurobiology and Behavior. The software created is called receptorToolsGUI and is run within MATLAB, utilizing its image processing, user interface, and 3D rendering functionalities.

Professor Ron Harris-Warrick's lab is investigating the cellular consequences of spinal cord injury (SCI). As part of their research, they have begun to analyze the SCI-induced changes in sensitivity to serotonin (5-HT) by identifying the presence of serotonin receptors. The serotonin receptors can be visualized in transverse image sections of a spinal cord through the use of a confocal microscope. The images are spaced closely together to create a three-dimensional representation of a short segment of the spinal cord. In characterizing the presence of the serotonin receptors, the researchers wanted to identify potential receptors in the noise environment imposed by the confocal microscope. Once potential receptors were identified, their volume and average intensity also needed to be determined. The lab also requested that the 3D data set be rendered in such a way that the visualization might help to characterize the data set.

Analysis of the spinal cord image sequences was previously done in 2D using ImageJ, an open source, Java-based image processing program developed at the National Institutes of Health. While this program is adequate in many aspects, it is not designed for the 12-bit images from the lab's confocal microscope and does not provide the user the necessary computational flexibility. The researchers needed a piece of software where the all the image processing subtleties would be known and they could define the core functionality.

At all design stages, the project technical requirements were of equal importance to the user interface usability for the neurobiology researchers. The software resulting from this project is a user-friendly packaging of the following tools that can operate on 8, 12, or 16 bit images:

- Image I/O with image sequence handling
- Serotonin receptor identification in 2D or 3D
  - Full data output to MATLAB workspace and/or Microsoft Excel
  - Annotated output images for corresponding receptor id data to image
  - Auto generates receptor characterizing plots
  - Allows auto or interactive user thresholding
- 3D rendering of data set
  - Colored coded volume visualization based on receptor size
- Conversion of 3D data set to usable 2D data set via max z projection
- Image Histogram
- Basic image arithmetic operations
- Detailed User's Manual with instructions for expanding software capability

A final draft of the software was finished in early April and was used in the receptor analysis of a paper submitted by an undergraduate researcher, Gabrielle Van Patten, as part of her research honors thesis. The software should see good use as the lab continues receptor research and was made to be easily expandable should they want to add more functionality.

Software for the Analysis of 3D Biological Data Sets

Adam McCann

ABSTRACT

This project is the creation of a software interface and data processing toolbox tailored to the specific needs of a neurobiology research group. The project scope and requirements were set in collaboration between Cornell University's Electrical and Computer Engineering Department and Cornell's Department of Neurobiology and Behavior. The data set and characterization process are essential to the research conclusions of the Harris-Warrick group in Cornell's Neurobiology Department. The data set is a sequence of greyscale 2D images taken from a confocal microscope at different depths. The process of characterizing the data set involves segmentation according to intensity and geometric shape within a unique noise environment imposed by the confocal microscope. Prior to my project's completion, the research group was only able to perform this analysis manually in 2D. The application of image processing techniques to this biological context provides value to the researcher by improving the precision and accuracy of measurements as well reducing the time and frustration encountered through manual data characterization.

1. INTRODUCTION

Professor Ron Harris-Warrick's lab is investigating the cellular consequences of spinal cord injury (SCI). As part of their research, they have begun to analyze the SCI-induced changes in sensitivity to serotonin (5-HT) by identifying the presence of serotonin receptors. The serotonin receptors can be visualized in transverse image sections of a spinal cord through the use of a confocal microscope. The images are spaced closely together to create a three-dimensional representation of a short segment of the spinal cord. In characterizing the presence of the serotonin receptors, the researchers wanted to identify potential receptors in the noise environment imposed by the confocal microscope. Once potential receptors were identified, their volume and average intensity also needed to be determined. The lab also requested that the 3D data set be rendered in such a way that the visualization might help to characterize the data set.

Analysis of the spinal cord image sequences was previously done in 2D using ImageJ, an open source, Java-based image processing program developed at the National Institutes of Health. While this program is adequate in many aspects, it is not designed for the 12-bit images from the lab's confocal microscope and does not provide the user the necessary computational flexibility. What the researchers needed most was a piece of software where the all the image processing subtleties would be known and they could define the core functionality.

The software was built is several stages with frequent input from the researchers that would later use it. At the start, much of the focus was on developing the image processing and denoising tools necessary to identify the serotonin receptors most accurately. Once the deconvolution solution to the confocal microscope image distortions was found, focus was switched to creating a basic I/O and interface structure so that multiple images could be opened and displayed without overwhelming the memory load on MATLAB. This basic infrastructure turned out to be a real challenge because the raw images are of significant size, 1024x1024 12 bit images. After setting up the infrastructure and means of passing the raw data from user-selected plots to user-selected tools, all the tools were added into a file menu system and connected to their respective source scripts. The last tool created was the 3D rendering of input data sets. Finally, a user's manual was written to facilitate the use of the program.

At all design stages, the project technical requirements were of equal importance to the user interface usability for the neurobiology researchers. Approaching the software design with this mentality led to an overall system architecture that is modular and easily expandable.

2. BACKGROUND

This section covers the background work in different fields that are being used in this project in creating the data set for which the software is designed for. A brief explanation of the neurobiological goals and methods for the project is provided as well as a discussion of confocal microscopes and the noise environment they induce.

2.1 Neurobiological Goals and Methods

Muscle paralysis after spinal cord injury is partly caused by a loss of brainstem-derived serotonin (5-HT), which normally maintains motoneuron excitability by regulating crucial persistent calcium currents[1]. It has been observed that one of the long term effects of the lost serotonin is that the motoneurons compensate by becoming more sensitive to serotonin to regain excitability. In an effort to better characterize the increase in motoneuron sensitivity, the Harris-Warrick lab is monitoring the long term changes in overall number, size, and intensity of 5-HT_{2C} receptor sites in healthy mice and mice with a spinal transection. An increase in receptor activity restores the calcium channels in motoneurons to enable the recovery of motor function in the absence of serotonin, but without regulation from the brain.

The subcellular distribution of neurotransmitter receptors is a basic feature underlying the functional properties of neurons[2]. The position of the receptor in the neuron affects their role in the transmission and integration of information by the neuron. The visualization of the localized receptors that are being studied in this context is biologically meaningful at a resolution of ~ 0.1 – 1μm. The technique that achieves this resolution is the combination of immunofluorescence and confocal microscopy[2]. Adding specific antibodies to the extracted spinal cord causes the 5-HT_{2C} receptors fluoresce under certain wavelengths of light. The confocal microscope uses a laser to excite the fluorescently labeled proteins.

2.1 Confocal Microscopes

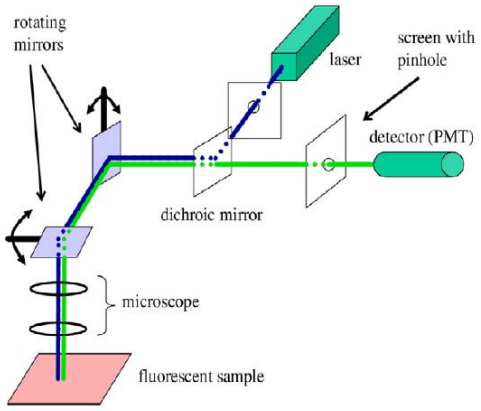


Figure 1: Schematic diagram of confocal microscope[3]

The confocal microscope used to collect the samples analyzed by this project was a laser scanning confocal microscope. In this method, a laser is reflected off a dichroic mirror (passes many colors of light, but reflects a small range of colors) then reflected off a set of moving mirrors before being directed as the fluorescent sample. The sample is excited and gives off light of a different frequency than the laser emits. This sample generated light is

redirected through the same moving mirrors as the laser and passes through the dichroic mirror. Before the fluoresced light reached the detector it passes through a pinhole placed in front of the detector (photo multiplier tube) so as to eliminate out-of-focus light. The light that is passed through the pinhole is measured by the detector as one pixel in the resulting images. The mirrors rotate to move the laser across the sample and generate the full image.

The resolution of the confocal microscope is limited by the diffraction of light. When the fluoresced light is passed through the circular pinhole aperture it diffracts in a pattern that is referred to as the Point Spread Function (PSF). For confocal microscopes, the point spread function is an Airy function in the lateral dimension, but is a Sinc function in the axial dimension (see Figure 2). In order to get the original, undistorted image back out, we must perform the deconvolution of the output image with the PSF seen in Figure 2. Deconvolution, or the inverse transfer function of a convolution, is extremely sensitive to the noise from the photo multiplier tube. Due to the noise, the deconvolution is typically performed statistically instead of inverse filtering. MATLAB has a few statistical deconvolution techniques, but I was not able to implement any in a way that was as successful as the method implemented in ImageJ. For this reason, deconvolution was not included as a tool in the resulting software. Instead, the instructions for using ImageJ's method are described in the User's Manual.

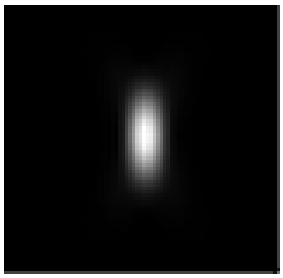


Figure 2: Confocal PSF

3. METHODS

This section covers the software's infrastructure and core functionalities. For operation details and instructions, the User's Manual is available in the Appendix.

3.1 Infrastructure

The key issues that the infrastructure had to address are the I/O setup, the way in which users will pass image information to functions, and the way in which images can be displayed to without slowing down MATLAB. It seemed that use of ImageJ is quite common with cell level image work, so I modeled my infrastructure and I/O after ImageJ.

The software is setup so that the I/O is different for opening 2D vs 3D data sets. If the user is using a 2D data set, they open the individual raw images, then open the process, and finally select the desired image from a drop list in the process interface. In this way, each 2D process will compile a list of open images upon creation of its interface. If the user is using a 3D data set, they open the process and that process will prompt them to open a set of input files. This works because no function in the software is for the manipulation of 3D data sets, i.e. all 3D tools operate on image sequences from source files and output data, not 3D data sets. Similarly, the user can save or display any 2D image in a variety of bit depths (8, 12, or 16), but cannot save or display image sequences.

When images are opened the user is prompted to give the bit depth for the image to be opened. The software then creates a figure that displays an 8-bit version of the image at half size. Displaying in this way makes the I/O quick, allows the user to display many images without slowing down MATLAB, and doesn't heavily degrade the quality of the displayed version of the image. In creating the figure for the image, the software stores the raw image data in the 'User Data' property of the figure and labels the 'Tag' property on the image in the format 'image#' where # starts at 0 for the first image and counts up for each subsequent image that is opened. When 2D processes are opened they scan the open figures for the 'Tag' that starts 'image'. Note that the 'Tag' is different than 'Name' which is displayed on the figure's menu bar. A figure's 'Tag' is only seen by background functions looking for it.

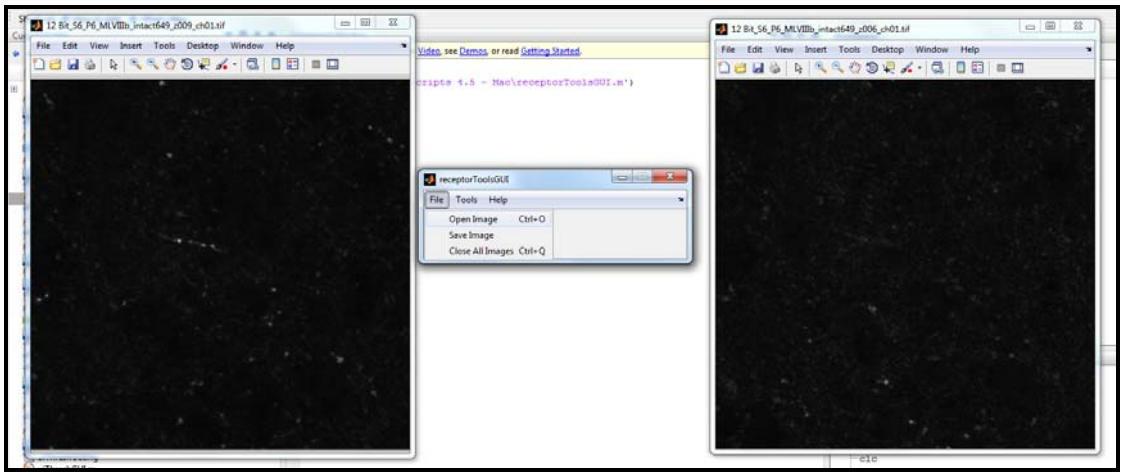


Figure 3: Open Images Screen Shot

3.2 Receptor Identification

Receptor identification in this software assumes that the image has already been deconvolved. The user selects the image to look in, then minimum sized blob that will be considered a receptor and finally the output options.

3.2.1 Two-Dimension Identification

Once the user has selected in the image and desired settings, the software will convert the input image to 8-bit and open a threshold interface. This interface allows the user to set a hardline number if they wish or to adjust the threshold interactively using the updating preview image and histogram to help select a desired level. Upon accepting the threshold, the software will convert the image to black and white (b/w). The b/w image is then operated on to restrict the size of the blobs to be those that are greater in radius than the user specified parameter.

The size restriction operation is a morphological open using a circular structuring element of radius equal to the minimum allowable radius. The morphological open is implemented by first applying a morphological erosion with the disc structuring element, then applying a morphological dilation with that same structuring element. A morphological erosion will iterate through the input image and check at each non-zero pixel to see if the structuring element centered at that pixel would "fit". "Fit" means that when the structuring element is centered on the current image pixel and overlaid, if there are any pixels that don't match in the overlay the current pixel is changed to zero. A morphological dilation will iterate through the input image and at overlay the centered structure element on the current pixel. The overall operation has the effect of slightly rounding the edges of all the blobs and removing those whose radius is smaller than the user specified value.

Following the size restriction, the image is still b/w and is passed to a connected component analysis. This will use a 4-connection criterion for labeling all the pixels in a connected blob with the blob number. This is done recursively starting from the top left corner of the image using the following algorithm:

```
setlabel(position, label)

      // make value pixel at current position equal to label

      // look at pixel to the right of current pixel
            // if right pixel is unlabeled and nonzero
                  // call setlabel(right of position,label)

      // look at pixel to the left of current pixel
            // if left pixel is unlabeled and nonzero
                  // call setlabel(left of position,label)

      // look at pixel below the current pixel
            // if pixel below is unlabeled and nonzero
                  // call setlabel(below position,label)

      // look at pixel above the current pixel
            // if pixel above is unlabeled and nonzero
                  // call setlabel(above position,label)
```

After the b/w, size restricted image has been component labeled a loop just collects the area and intensity values for the now segmented blobs. As an output option, the user can display the following plots:

- Histograms for receptor size and average intensity
- Cumulative histograms for size and average intensity
- Scatter plot of receptor size vs avg intensity

Additionally, the data may be exported to Microsoft Excel or to the MATLAB workspace for further analysis. This was particularly useful for the researchers in performing T-Tests on their accumulated data. The other display options are for viewing an image with the blob numbers shown or an RGB overlay that shows identified receptors in red overlaying the blue background image.

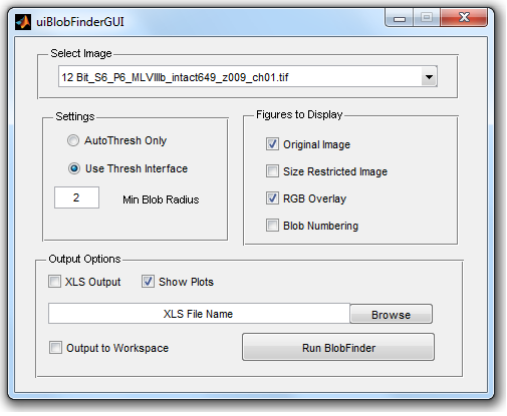


Figure 4: 2D Receptor ID Interface

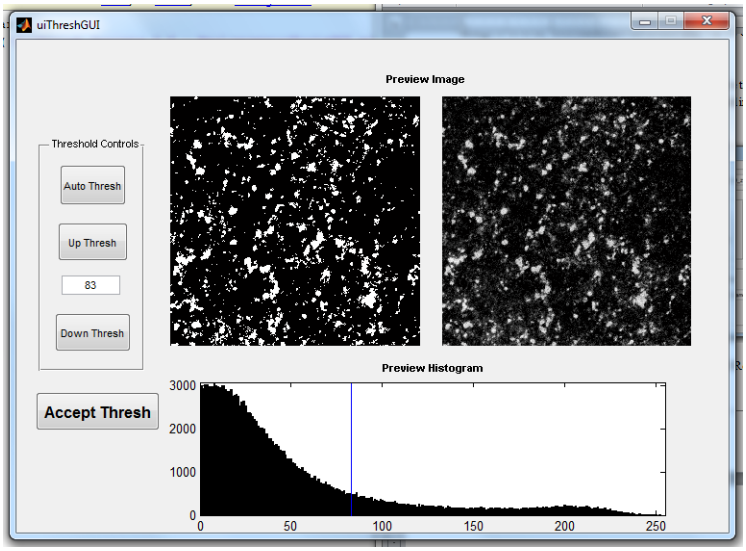


Figure 5: Threshold Selection Interface

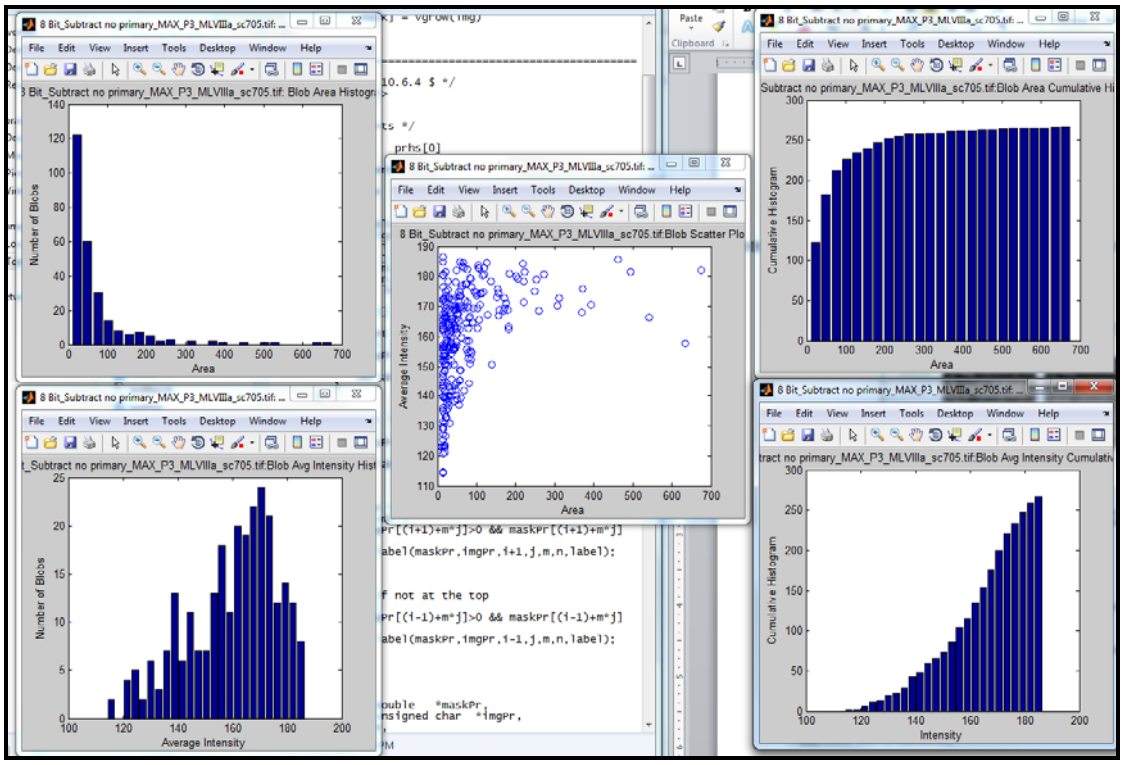


Figure 6: Output receptor data plots

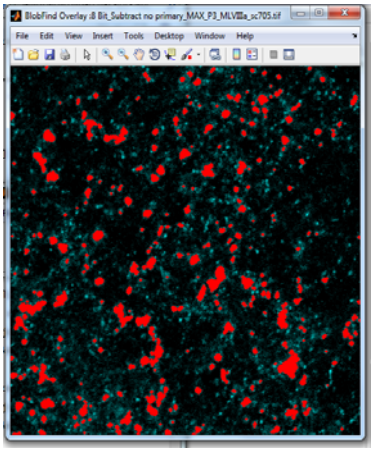


Figure 7: Output RGB Overlay

3.2.2 Three-Dimension Identification

For the 3D case, the user is prompted to select a group of images. These images are displayed as they are loaded in. Once loaded in, the user is given the same minimum size settings and is provided with a list of the images in the order they will stacked. As with the 2D case, the user is also give the option to display the same 5 data plots (with volume instead of area) and the option to export to Excel or MATLAB workspace. Additionally the user has a 3D render option.

Once the images and settings are confirmed the software will convert the data matrix to 8-bit and open a threshold interface. This interface is exactly the same as with 2D except that the preview image a max projection of the stacked image set onto one image. Upon accepting the threshold, the software will convert the data matrix to black and white (b/w). The b/w image is then operated on to restrict the size of the 3D blobs to be those that are greater in radius than the user specified parameter.

The size restriction operation is a morphological open, as in the 2D case, but using a ball structuring element of radius equal to the minimum allowable radius. The morphological operators extend to 3D in a very straightforward fashion. The overall operation has the effect of slightly rounding the edges of all the blobs and removing those whose radius is smaller than the user specified value.

Following the size restriction, the data matrix is still b/w and is passed to a connected component analysis. This will use a 6-connection criterion for labeling all the pixels in a connected blob with the blob number. This is done recursively starting from the top left corner of the data matrix using the same algorithm as in 2D except that is adds to direction check for the new dimension.

After the b/w, size restricted image has been component labeled a loop just collects the area and intensity values for the now segmented blobs. As an output option, the user can display the following plots:

- Histograms for receptor size and average intensity
- Cumulative histograms for size and average intensity
- Scatter plot of receptor size vs avg intensity

Additionally, the data may be exported to Microsoft Excel or to the MATLAB workspace for further analysis. This was particularly useful for the researchers in performing T-Tests on their accumulated data. The other display options are for viewing an image with the blob numbers shown or an RGB overlay that shows identified receptors in red overlaying the blue background image.

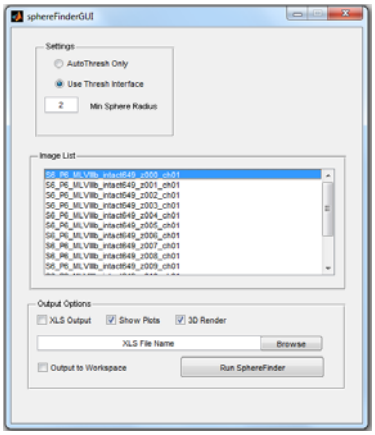


Figure 8: 3D Receptor ID Interface

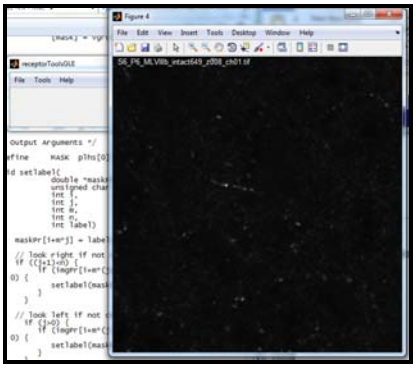


Figure 8: Image Set is displayed as loaded

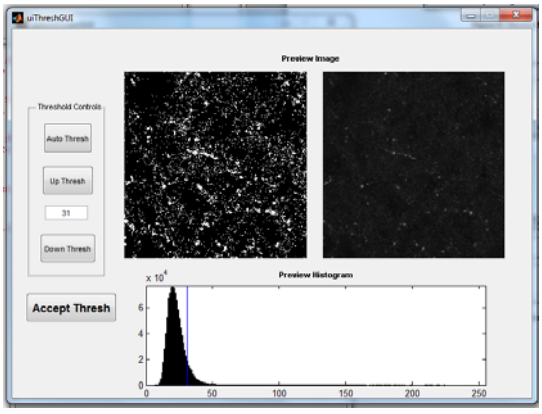


Figure 9: Threshold Interface

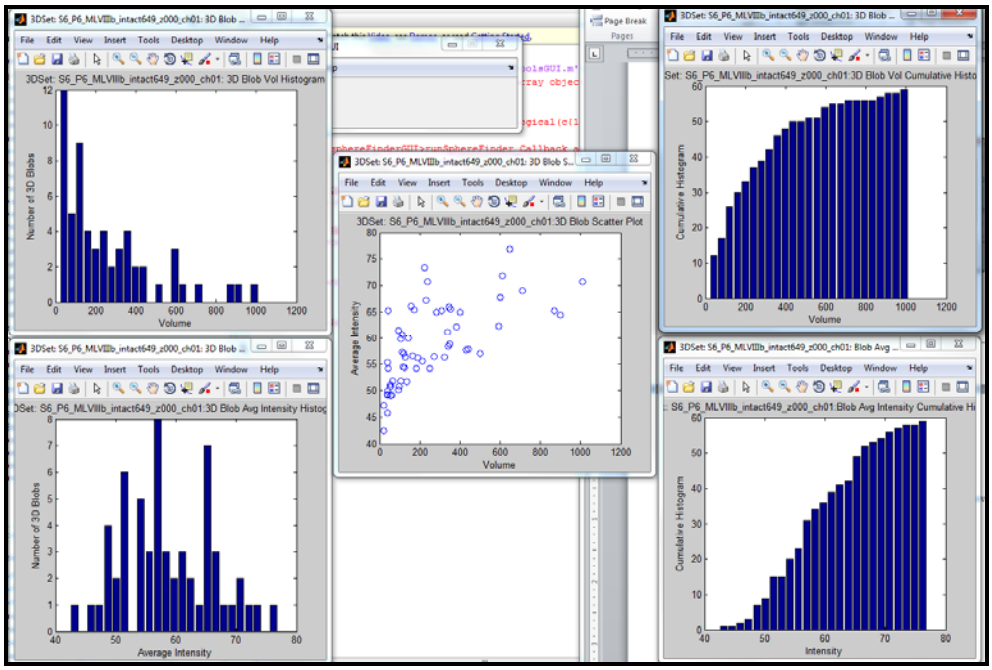


Figure 10: Output receptor data plots

3.3 Three-Dimensional Data Set Rendering

The 3D rendering of the identified receptors is generated using MATLAB's isosurface function. Before rendering, the identified receptors are given a size category number. Small receptors (volume < 800 voxels) are labeled first. Once they're labeled the label data is passed to the isosurface function along with the label value. This function looks for points in the data with the specified label value and tries to connect then with patches. These patches are calculated as vertices and edges and are then passed to the patch renderer function with instructions to make them green. The same process happens for medium receptors (800 > volume > 3000) that are colored yellow and large receptors (volume > 3000 voxels) which are colored red. These rendered patch surfaces are displayed on a graph with dimensional aspect ratio adjusted to account for the physical distance between confocal images and the pixel to pixel distance. For most cases in data I've seen the following values determine the aspect ratio:

```
pixSize_xy = 91.55e-9;
pixSize_z = 1.18e-6;
zFactor = pixSize_z/(pixSize_xy*2);
```

The factor of two in the denominator of the zFactor is due to the fact that the 3D view is rendered at half resolution (512x512 instead of 1024x1024). Some lighting and background color is also added to the plot to help the 3D effect. The max projected image is also shown on the floor of the plot to help the researcher relate the 3D rendering with the 2D images.

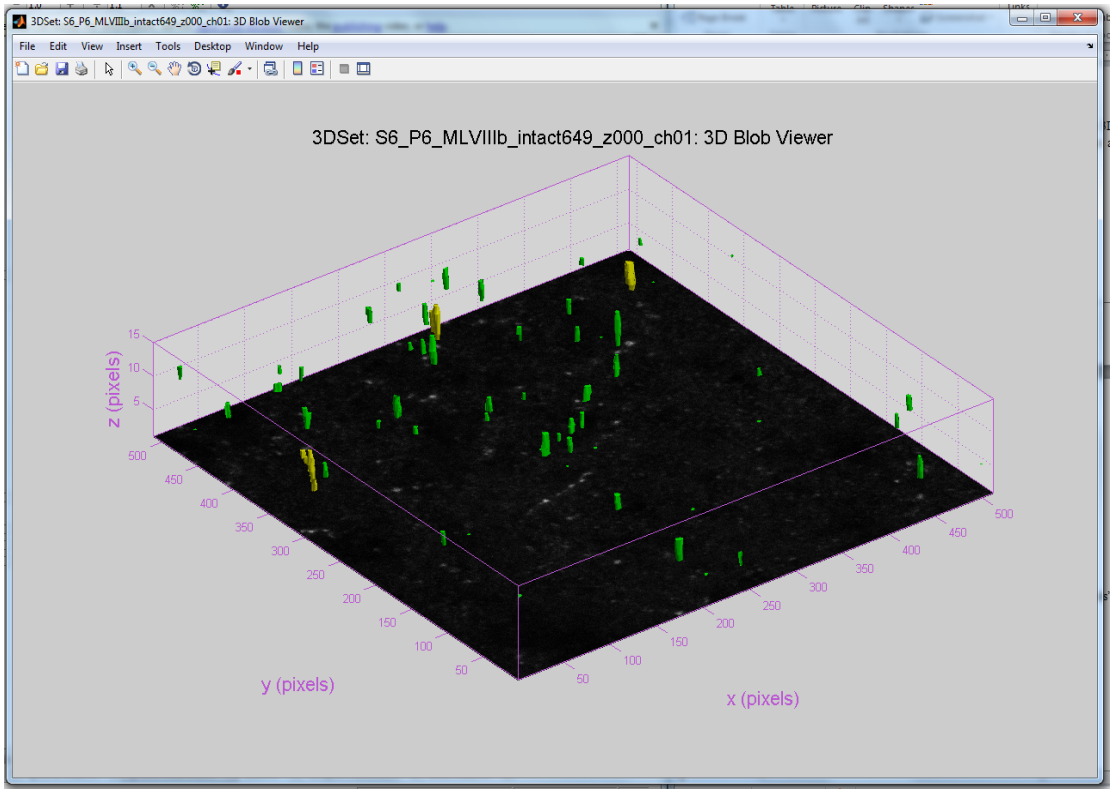


Figure 11: 3D Rendering for a sample data set

4. TESTING AND SYSTEM QUALIFICATION

In order to qualify the methods used above, it was important to create a known data set for the 2D and 3D setups. I wrote a script to populate a 3D volume with a specified number of spheres of a specified radius and intensity gradient in order to qualify the techniques and 3D rendering with a known data set and dimensional aspect ratio. Slices of this populated data set were used to test the 2D tools.

The approach to use spheres with linear intensity gradient by radius was chosen to best mimic the actual data sets in way that was easy to test accuracy. To choose a test radius and gradient factor I just tried a few options until the test set visually looked like the real data set. The most important thing to set in the artificial data set was the mean intensity. This value was quite low in the actual data sets and so the values needed to match to properly test the display and identification capabilities. The figure below shows a slice of one the test data sets.

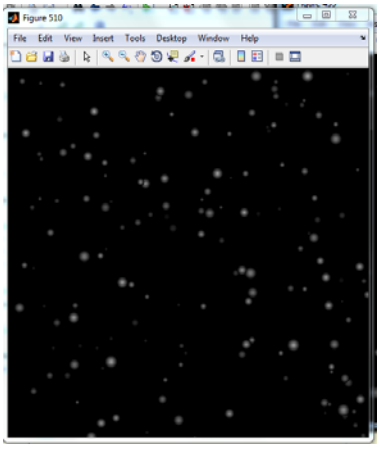


Figure 12: Artificial Data Set Slice

5. RESULTS AND DISCUSSION

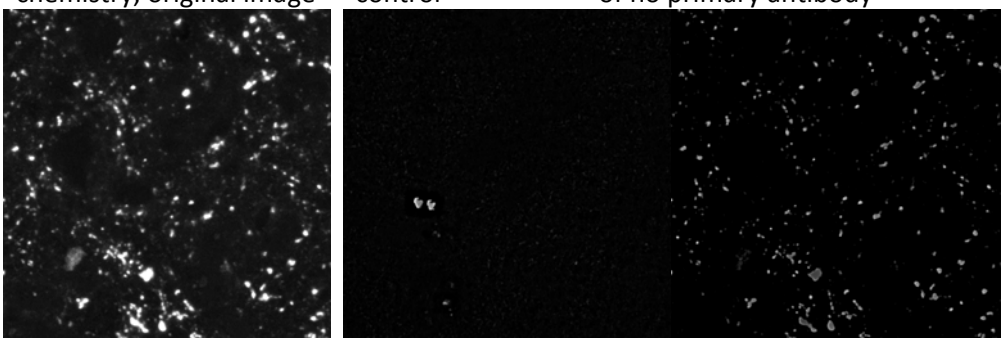
The end product of this project is fully functional to the requirements asked by the neurobiology lab and has surpassed their expectations. More importantly, the software has already aided in the research conclusions of an undergraduate researcher, Gabrielle Van Patten, as part of her Research Honors program thesis. The following extract from her paper was generated using the receptorToolGUI:

--- Reproduced with Permssion from [4] ----------

A. $5HT_{2C}$ immunohisto-chemistry, original image

B. No primary antibody control

C. Image B after subtraction of no primary antibody



D. Binary output after setting intensity threshold

E. Mask for image C

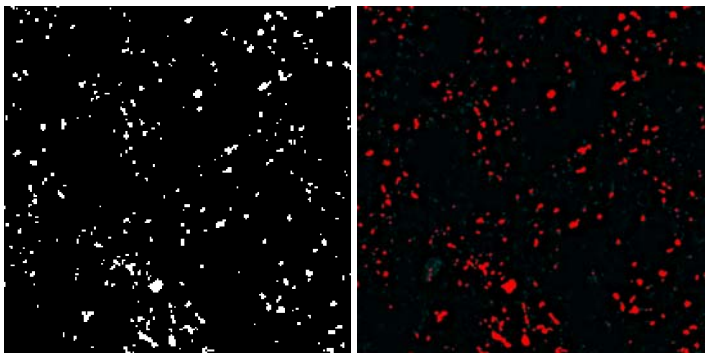


Figure 7. Matlab analysis to determine the size and intensity of $5HT_{2C}$ receptor clusters in intact, SCI/saline, and SCI/quipazine mice. First, the average intensity of the no primary antibody control (***B***) is subtracted from the original image (***A*)**, to adjust for tissue auto-fluorescence and non-specific binding of the secondary antibody. The resultant image (***C***) then has an intensity threshold applied in Matlab (***D***), and a minimum cluster radius is set for identifying regions of interest. The program uses a morphological close operation to remove regions under the cluster radius restriction, and then segments them to give a labeling mask structure for the deconvolved, no primary subtracted image (***E***).

--- Reproduced with Permission from [4] ----------

The result of the paper is also reproduced with permission below:

"We found that after SCI, the number and average size of 5HT2C receptor clusters increases, as well as the area and mean intensity of CaV 1.3 channels. Hayashi et al. (2010) found that SERT is down-regulated after severe spinal cord contusions, and Murray et al. (2011) found that there are more constitutively active 5HT2C receptors after. Likely, many more receptor subtypes and channels are affected in different ways by SCI, and only by using multiple approaches such as immunohistochemistry and electrophysiology can we hope to elucidate all the effects of spinal cord injury, and how we can mediate those effects to increase quality of life." [4]

It is evident from these two sections that the software analysis tools were indeed useful in confirming the paper's conclusion that more (in area and intensity) receptors were seen in SCI mice. I expect that the tools provided by this project will be useful in further confirming the lab's receptor sensitivity hypothesis in future publications.

6. Future Considerations

In an effort to make the software created for this project most usable for the neurobiology in future work I made the infrastructure modular so implementing new functionality is simple and comes with instructions in the User's Manual. If, for example, they wanted to add functionality for characterizing the web-like Calcium channels they could simply write the script for it, connect it to the menu, and implement the function I/O as per my instructions in the User's Manual.

7. Acknowledgements

I would like to thank Bruce Land, Ron Harris-Warrick, and Gabrielle Van Patten for their help with this project. They have helped to gain a better understanding of basic neurobiology, image processing techniques, and especially the process of creating user interfacing tools.

8. REFERENCES

1. Murray, Katherine C., Aya Nakae, Marilee J. Stephens, Michelle Rank, Jessica D. Amico, Philip J. Harvey, Xiaole Li, R Luke W. Harris, Edward W. Ballou, Roberta Anelli, Charles J. Heckman, Takashi Mashimo, Romana Vavrek, Leo Sanelli, Monica A. Gorassini, David J. Bennett, and Karim Fouad. "Recovery of Motoneuron and Locomotor Function after Spinal Cord Injury Depends on Constitutive Activity in 5-HT_2C Receptors." *Nature America*16.6 (2010): 694-701. Print

2. Hutcheon, B., L. A. Brown, and M. O. Poulter. "Digital Analysis of Light Microscope Immunofluorescence: High-resolution Co-localization of Synaptic Proteins in Cultured Neurons." *Journal of Neuroscience Methods* (2000): 1-9. Print.

3. Prasad, V., D. Semwogerere, and Eric R. Weeks. "Confocal Microscopy of Colloids." *Journal of Physics: Condensed Matter* 19.11 (2007): 113102. Print.

4. Van Patten, Gabrielle N., and Ronald M. Harris-Warrick. *Immunohistochemical Quantification of 5HT2C Receptors and CaV 1.3 Channels after Spinal Cord Injury in the Upper Lumbar Mouse Spinal Cord*. Tech. Print.

APPENDIX

USER'S MANUAL

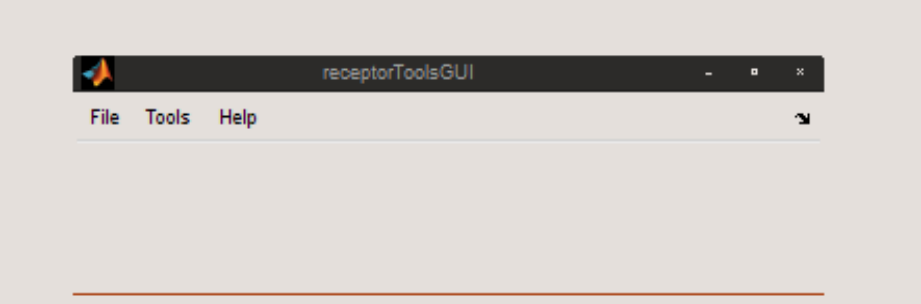


Table of Contents

1 About

The software interface outlined in this document was created to the specific needs of Cornell's Harris-Warrick lab in the department of neurobiology.

Last Updated - May 2011

by Adam McCann ajm232@cornell.edu

2 System Specifications

All MATLAB scipts were created and tested in the MATLAB 2010b version with image processing toolbox included.

3 Installation

All software scripts and files are included in the zipped file

receptorToolsGUI.zip

Including the following scripts:

blobFinder.m
makeHistGUI.fig
makeHistGUI.m
putvar.m
receptorToolsGUI.fig
receptorToolsGUI.m
removeRefMeanGUI.fig
removeRefMeanGUI.m
saveImageGUI.fig
saveImageGUI.m
sphereFinder.m
sphereFinderGUI.fig
sphereFinderGUI.m
uiBlobFinderGUI.fig
uiBlobFinderGUI.m
uigetBitDepth.fig
uigetBitDepth.m
uiThreshGUI.fig
uiThreshGUI.m

the following documents:

McCann – MEng Report.pdf
McCann – Meng Report – User's Manual.pdf

as well as a folder containing a pair 2D sample images, one 3D sample set, and one PSF image.

To install and use the software, place all the scripts above in the same directory, change the current directory in MATLAB to this directory and run receptorToolsGUI.m.

4 Input/Output

As mentioned above, to start the program use the following command:

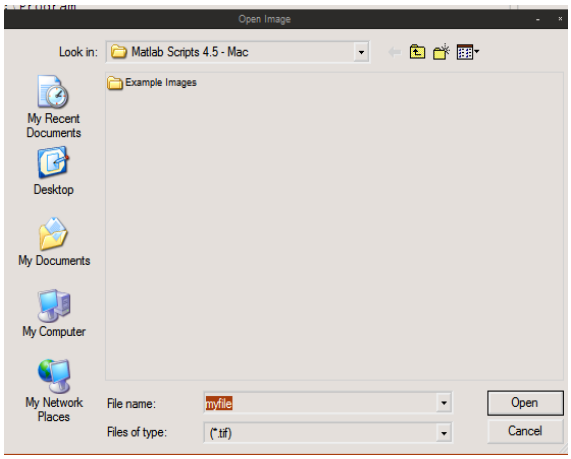
```
>> run receptorToolsGUI
```

For processing 2D images, open the images first then select the desired process from the menu bar on the figure titled receptorToolsGUI. For 3D data sets (image sequences), select the desired process from menu bar on the figure titled receptorToolsGUI and it will prompt you to open a sequence of images.

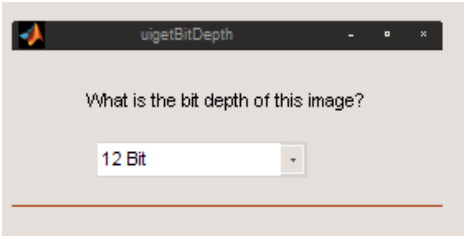
4.1 Opening Images

In the receptorToolsGUI figure, under the file menu select Open Image or use the shortcut CTRL+O when the receptorToolsGUI is active.

You will be prompted to select a file



You may select .tif images of bit depth 8, 12, or 16. After selecting open, you will be prompted to enter the bit depth of the image that's just been selected.



The software will then open a figure with the properties:

Tag	'image#'
Name	['bitDepth' filename]
User Data	raw image data (in raw bit depth)

The image that is displayed inside the figure is the selected raw image converted to 8-bit and resized to be half of its original size. We convert to 8-bit for all displayed images because the human visual system can really only distinguish on the order of 256 different grey levels. The resizing just help the I/0 to run quickly and allows the user to keep many images open at once with minimal memory load.

WARNING:

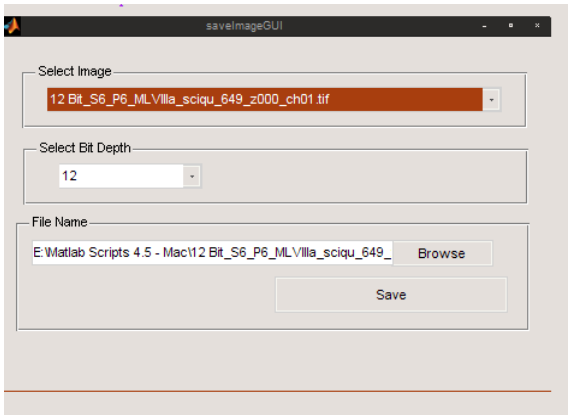
Do not use the File > Save function in the image figures to save the images. This will save a "screen shot" of whatever is in the figure window, so you will be saving an 8-bit half sized version of your original image.

TROUBLESHOOTING TIPS:

If you load in images and they look darker than they are supposed to be, it's probably because you got the wrong bit depth. Typically, images coming directly from the microscope are in 12 bit and deconvolved images are in 16 bit. It is helpful to label saved images with their bit depth.

4.2 Saving Images

In order to save an image that is open in one of the figures select File > Save Image on the receptorToolGUI figure. Presumably you would want to do this after you've performed a Max Z Project or a Reference Mean Subtraction. The interface will bring up a list of currently open images for you to choose from along with a bit depth to be saved in and the file name.



WARNING:

Saving in 12-Bit format will really be in 16-Bit format, but will leave the last four bit blank.

4.3 Close All Images

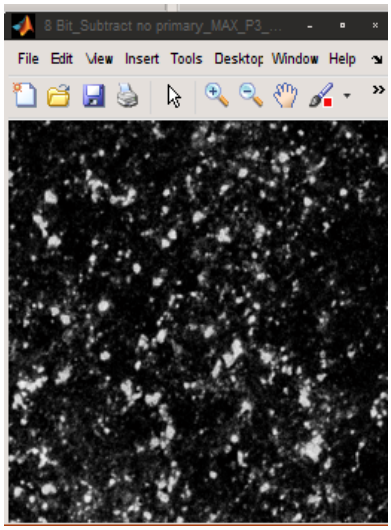
In the receptorToolsGUI figure, selecting File > Close All Images will close only the images with the 'Tag' starting with the 'image', or equivalently all the figures that aren't plots or interfaces.

5 Tools

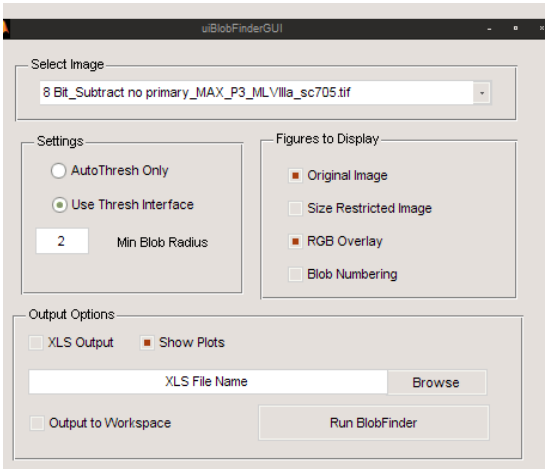
5.1 2D BlobFinder

The blobFinder tool is used to identify and characterize the blobs in a given image. The output is a list of blobs and their corresponding area, total intensity, and average intensity values.

I'll use the 8-bit example image, Subtract no primary_MAX_P3_MLVIIIa_sc705.tif, for the following walk-through.



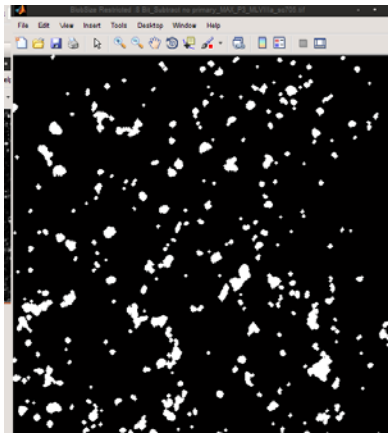
Selecting Tools > Filters > 2D BlobFinder in the receptorToolsGUI figure will bring up the following interface.



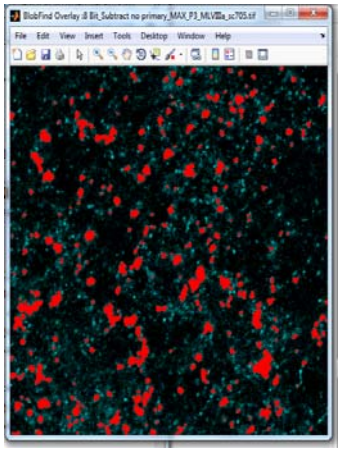
The Select Image drop down box will list all the currently open images, I've selected the example image shown above. In the Settings group, the AutoThresh Only will just disable the user interactive thresholding of the image and just use the value of the image mean intensity + one standard deviation of the intensity. The Min Blob Radius setting controls the radius of the minimum allowable blob.

The possible figures to display are:

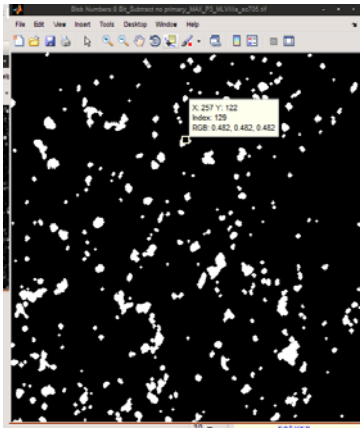
- *Original Image* – When selected, this will keep the original image from closing.
- *Size Restricted Image* – When selected, the software will show the black and white, thresholded and blob size restricted image.



- *RGB Overlay* – When selected, the software will show the identified blobs in red overlaying the original image in blue and green.



- *Blob Numbering* – When selected, the software will show the identified blobs and if the user selects the Data Cursor in the figure window they can click on a blob the label number will be shown.



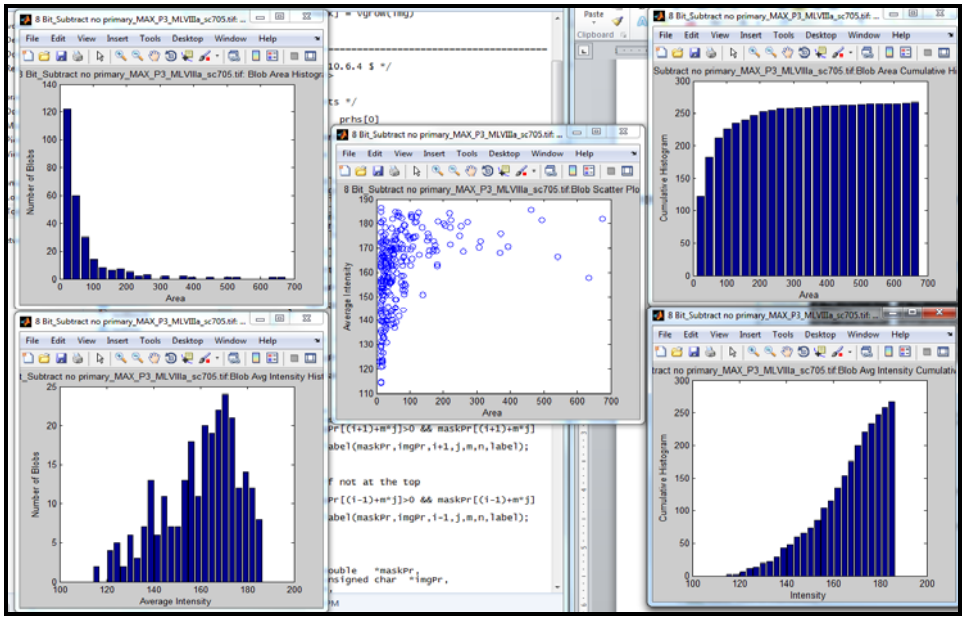
The possible output options are:

- *XLS Output* – When selected, this will save a copy of the blob characteristic data to an XLS file of the name specified by the user in the text dialog box below the XLS Output button. You cannot use this

function to append the results onto an existing XLS spreadsheet because the data values will always be written in the first three columns of the first sheet.

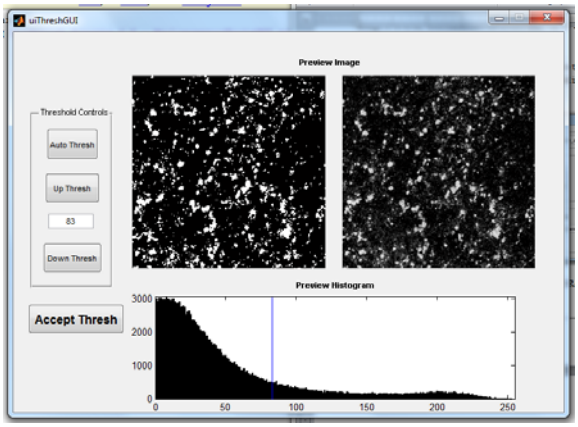
- *Show Plots* – When selected, the software will display the following 5 plots regarding the blob characteristic data:

    - Histograms for receptor area and average intensity
    - Cumulative histograms for area and average intensity
    - Scatter plot of receptor area vs avg intensity



- *Output to Workspace* – When selected, the software will place a copy of the 3xnumblobs characteristic data matrix in the workspace under the variable name blob_info. If there is a variable in the workspace already call blob_info, this will not append the new data to the old data, but will instead write over the blob_info variable.

Selecting <Run BlobFinder> will open up the user interactive threshold panel if auto thresh was not selected.



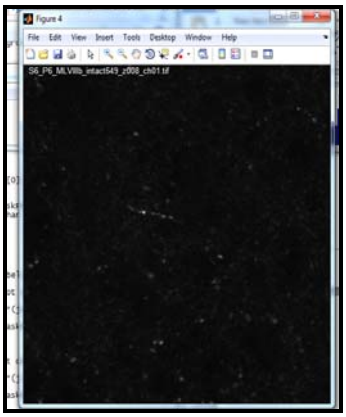
The user can adjust the threshold with the up or down buttons or by just typing it in the box and pressing enter. When the selected threshold changes, the left image in the Preview image set will show the resulting, thresholded image while the right image shows the unchanged original. The user may also want to use the image histogram provided at the bottom of the panel with a blue line to indicate the current threshold. Selecting the Accept Thresh button will cause the program to finish and output all the selected options.

5.2 3D BlobFinder

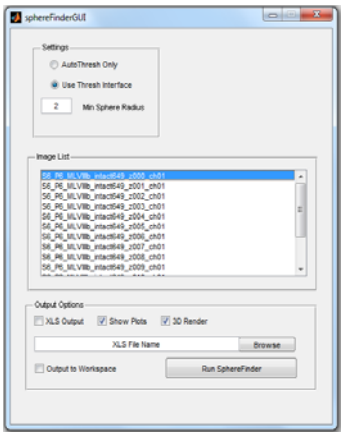
The 3D blobFinder tool is used to identify and characterize the blobs in a given image sequence. The output is a list of 3D blobs and their corresponding volume, total intensity, and average intensity values.

I'll use the 12-bit example sequence in the folder 3D Test Set.

Selecting Tools > Filters > 3D BlobFinder in the receptorToolsGUI figure will prompt the user to select a group of files. It is assumed that the files to be analyzed are all the same bit depth and are sorted in the file so that when they are loaded the stack is in the correct order. This is simple to do if put the z parameter in the file name. Upon selecting the images, they will be displayed as they are loaded in along with their filename in text.



After the files have been loaded the following interface will appear.



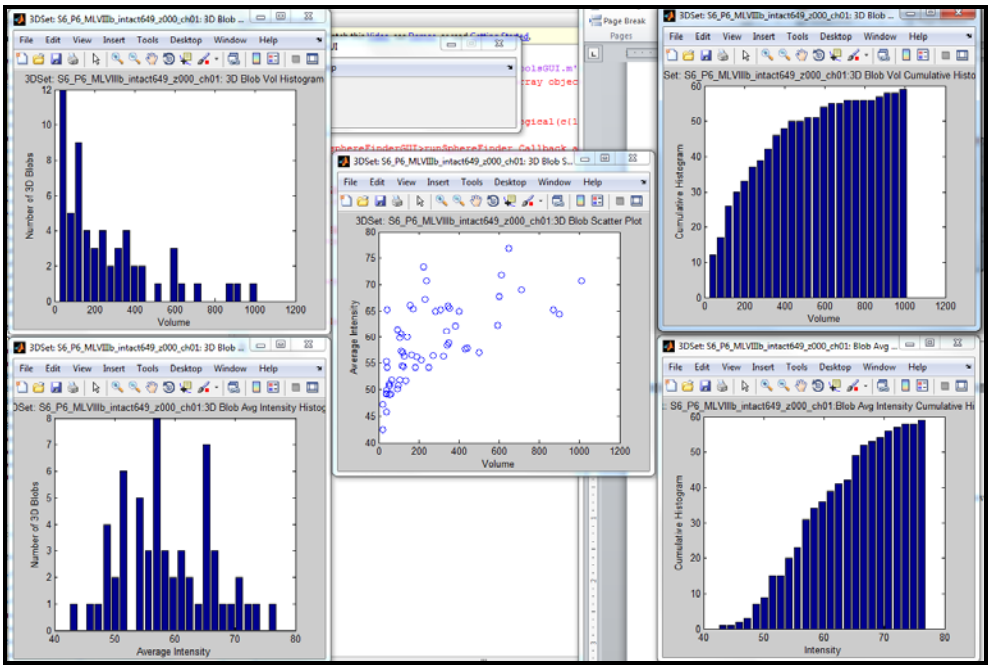
When clicked, the Image List will list all the image loaded into the sequence in the order that they will be stacked. In the Settings group, the AutoThresh Only will just disable the user interactive thresholding of the image and just use the value of the image mean intensity + one standard deviation of the intensity. The Min Blob Radius setting controls the radius of the minimum allowable 3D blob.

The possible output options are:

- *XLS Output* – When selected, this will save a copy of the 3D blob characteristic data to an XLS file of the name specified by the user in the text dialog box below the XLS Output button. You cannot use this function to append the results onto an existing XLS spreadsheet because the data values will always be written in the first three columns of the first sheet.

- *Show Plots* – When selected, the software will display the following 5 plots regarding the blob characteristic data:

    - Histograms for receptor volume and average intensity
    - Cumulative histograms for volume and average intensity
    - Scatter plot of receptor volume vs avg intensity



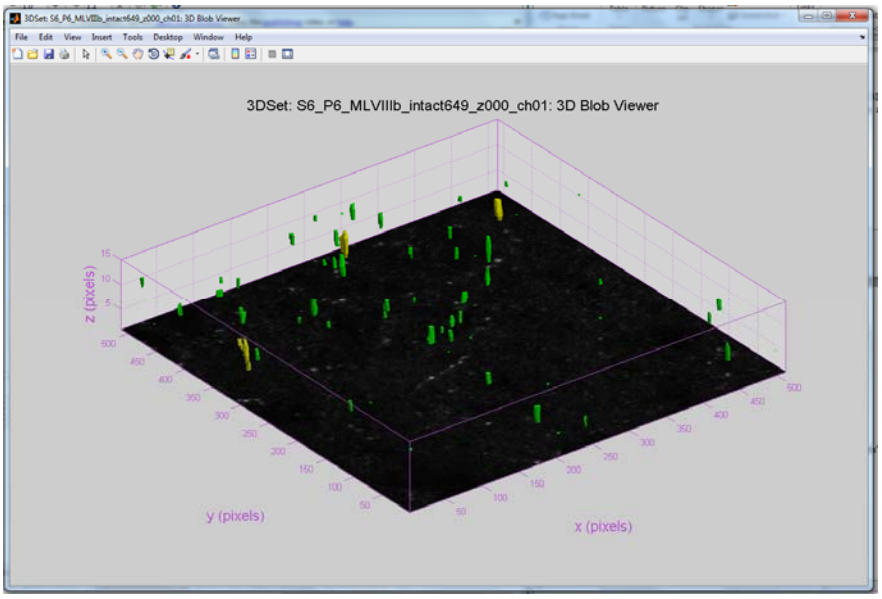
- *Output to Workspace* – When selected, the software will place a copy of the 3xnumblobs characteristic data matrix in the workspace under the variable name sphere_info. If there is a variable in the workspace already call sphere_info, this will not append the new data to the old data, but will instead write over the sphere_info variable.

- *3D Render* – When selected, a 3D rendering of the identified receptors is generated using MATLAB's isosurface function. Before rendering, the identified receptors are given a size category number. Small receptors (volume < 800 voxels) are labeled first. Once they're labeled the label data is passed to the isosurface function along with the label value. This function looks for points in the data with the specified label value and tries to connect then with patches. These patches are calculated as vertices and

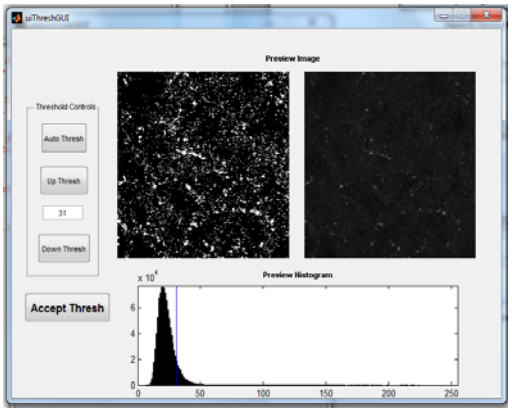
edges and are then passed to the patch renderer function with instructions to make them green. The same process happens for medium receptors (800 > volume > 3000) that are colored yellow and large receptors (volume > 3000 voxels) which are colored red. These rendered patch surfaces are displayed on a graph with dimensional aspect ratio adjusted to account for the physical distance between confocal images and the pixel to pixel distance. For most cases in data I've seen the following values determine the aspect ratio:

```
pixSize_xy = 91.55e-9;
 pixSize_z = 1.18e-6;
 zFactor = pixSize_z/(pixSize_xy*2);
```

The factor of two in the denominator of the zFactor is due to the fact that the 3D view is rendered at half resolution (512x512 instead of 1024x1024). Some lighting and background color is also added to the plot to help the 3D effect. The max projected image is also shown on the floor of the plot to help the researcher relate the 3D rendering with the 2D images.



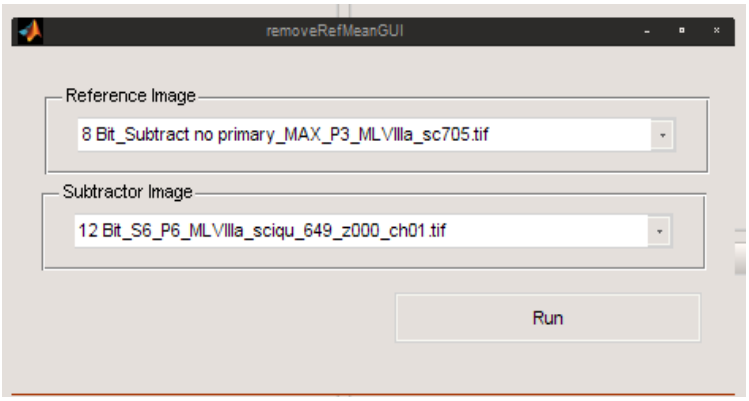
Selecting <Run SphereFinder> will open up the user interactive threshold panel if auto thresh was not selected.



The user can adjust the threshold with the up or down buttons or by just typing it in the box and pressing enter. When the selected threshold changes, the left image in the Preview image set will show the resulting, thresholded Max Z projection of the set while the right image shows the unchanged original. The user may also want to use the image histogram provided at the bottom of the panel with a blue line to indicate the current threshold. Selecting the Accept Thresh button will cause the program to finish and output all the selected options.

5.3 Remove Ref Mean

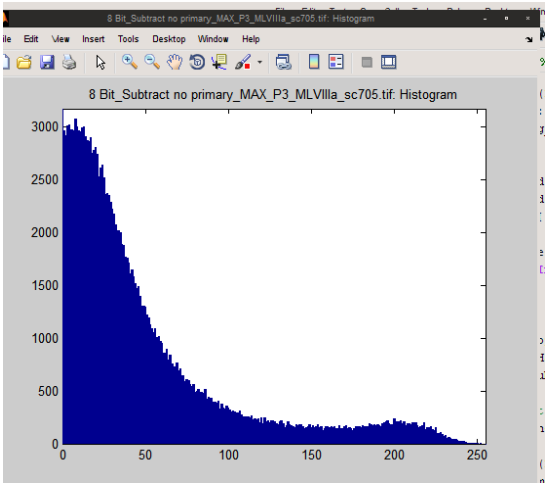
This tool is used to subtract the mean of one image from another. Selecting Tools > Math Operators > Remove Ref Mean in the receptorToolsGUI figure will open the following dialog box.



The mean of the reference image will be subtracted from the subtractor image. The resulting image is always in 8 Bit and will labeled so in the name of the resulting figure.

5.4 Histogram

This tool is used to show the histogram of a selected image. Selecting Tools > Histogram in the receptorToolsGUI figure will open an image select list. Selecting <Generate Histogram> will produce the histogram for the image. If the image is mostly black the histogram will let the zeroth bin of the histogram run off the chart and display a warning box.



5.5 Max Z Project

This tool is will compute the Maximum intensity for the each z projected pixel in a image sequence. For all notation in the software and in this manual, the image axes are x and y while the axis perpendicular to the plane of the image is the z plane.

Selecting Tools > Histogram in the receptorToolsGUI figure will open an image select list. Selecting multiple files of the same bit depth will compute the max z project and create a figure of the same bit depth with the annotation Z PROJECT in the figure name.

6 Sample Work Flow

In this example, the goal is to use a 12 bit raw image and its 12 bit corresponding reference image to get a blob_info data set.

- Open both images with receptorToolsGUI
- Use the Remove Ref Mean to subtract the raw reference mean from the raw image.
- Save image (not the reference image) in 8-bit
- Open the 8-bit image in ImageJ and apply deconvolution, then save in 8-bit from ImageJ
  - ImageJ has a plugin called Iterative Deconvolution. Run this 10 iterations.
- Open the 8-bit deconvoled image with receptorToolsGUI
- Apply 2D BlobFinder and output data to workspace

7 Adding Future Functionality

Adding new functionality to receptorToolsGUI can be very easy if you use the following steps and copy certain sections of existing code in order to get the I/O right.

If for example, I wanted to add a function that generated a random integer for every pixel in a specified image. I'd call this function randomize and save it in a file called randomize.m. It is important the raw image and it's bit depth be input parameters into the function. The script for randomize.m might be something like this, where I've included the bitdepth conversion just to show:

```
function [img_8_rando] = randomize(raw_img,bitDepth)

   imgHi = 2^bitDepth-1;
   imgLo = 0;

   img_8_rando = uint8(255.*double((img-imgLo))./double((imgHi-imgLo)));
```

```
for i = 1:size(img_8,1)
      for j = 1:size(img_8,1)
            img_8_rando = randi(255);
      end
end
```

To connect this function to receptorToolGUI, start by typing the command

```
>> guide
```

Open receptorToolsGUI.fig and Tools > Menu Editor. I'll add randomize to Tools and give it the Label: Randomize along with the Tag: menu_tools_randomize. Save receptorToolsGUI.fig and open receptorToolsGUI.m.

In the receptorToolsGUI.m file there should be a new function listing called menu_tools_randomize_Callback.

Now I want to use randomize on the images that are already open in receptorToolsGUI, so I'm going to implement a new GUI called randomizeGUI.m and randomizeGUI.fig

Again type `>> guide` and this time select Blank GUI. Add a pop up menu for image list and make tag for it 'imageList' using the property editor for the pop up menu. Next add a run button and give it the tag 'runRandomize'. Save the figure as randomizeGUI.fig and it will generate the file randomizeGUI.m.

In randomizeGUI.m, there should be a function called imageList_CreateFcn. Paste the following code under it:

```
ihandles = findobj('-regexp','Tag','^image');
ilist = cell(0);

for i=1:size(ihandles,1)
    ilist = [ilist;cellstr(get(ihandles(i),'Name'))];
end

if isempty(ilist)
    ilist = cellstr('No Images Open');
end
set(hObject,'String',ilist)
```

Next, under the function runRandomize_Callback add the image passing code, a call to the original function randomize, and the output image passing to figure code

```
%% Image Selection Data Passing
imgList = get(handles.imageList,'String');
listVal = get(handles.imageList,'Value');
imgName = imgList{listVal};

if isequal(imgName,'No Images Open')
    errordlg('No Images Selected','make Hist Error');
    return
end

imgHandle = findobj('Name',imgName);
img = get(imgHandle,'UserData');
bitDepth = str2double(imgName(1:2));
```

```
  %% Call to function
  img_8_rando = randomize(img,bitDepth);


  %% Pass resulting image to figure

 figure('Name',[num2str(bitDepth) ' Bit_Ref Remove_'...
imgName],'NumberTitle','Off',...
                'Tag','imageRefRemove');
 set(gcf,'UserData',img_8_rando);
 imshow(img_8_rando)
 delete(handles.randomizeGUI)
```

Finally, in receptorToolsGUI.m under the function menu_tools_randomize_Callback put

```
randomizeGUI();
```