Bachelor's Thesis

Information and Communications Technology

2022

Hassan Farah

MICROSCOPE AUTOMATION: MACHINE VISION TO SUPPORT IN-CELL SIGNAL DETECTION

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Bachelor's Thesis | Abstract

Turku University of Applied Sciences

Information and Communications Technology

2022 | 35 total pages

Hassan Farah

MICROSCOPE AUTOMATION: MACHINE VISION TO SUPPORT IN-CELL SIGNAL DETECTION

Automation has become increasingly important in our life. With automation, we can achieve a reduction in cost, time, mistakes, and a multitude of other aspects that hinder the workflow in production.

The objective of this thesis was to use computer vision on fluorescence automated microscopy to help cell imaging in detecting signaling events from cells based on calcium spikes.

Automated microscopy has been used in drug discovery for quite some time, but there has only recently been a rapid advance in automated detection and analysis of cells, which would enable an amplitude of possibilities with the data obtained from cells that are analyzed after they have been detected.

This thesis introduces the processes necessary to achieve functional cell detection and analysis. The cell detection algorithm, which utilizes image segmentation, and the cell analysis algorithm, which employs ROI, are presented in detail with code examples in this thesis. The steps taken to improve the cell detection capability utilized processes in image segmentation. The processes that were used include blurring, greying, binary thresholding, and contour.

In conclusion, the data received from the detected cells offers numerous l possibilities such as decision making, visualizing the data, and predicting outcomes with the use of machine learning to name a few. The collected data is not only limited to the intensity of the cell, but also other attributes including size, color, and shape can also be collected for analysis purposes.

Keywords:

Machine Vision, Neuroscience, Drug discovery, Microscopy, Python, OpenCV, AI, Edge Detection, Image Segmentation

Contents

Figures

List of abbreviations

AI	Artificial Intelligence
DAQ	Data Acquisition Box
RGB	Red Green Blue
ROI	Region of Interest
v4	version 4

1 BACKGROUND AND INTRODUCTION

The main goal of drug discovery is to improve our quality of life and health. To boost the quality and speed of this work, any applicable technology development is worth investigating further. Historically, drug discovery has been a slow process because testing and research are mostly manual and human-input dependent. Often, researchers need to analyze thousands of test samples and the process can be very tedious and lengthy.

Automation has played an important role in the development of any industry. It is expected, and it is the main hypothesis of this thesis that automation will have a strong impact on the drug discovery industry, helping to speed up processes and the work of researchers and scientists. Recent advances in artificial intelligence (AI), computer vision and machine learning may apply to this field, and that is what this work is about.

The optogenetics sector is gaining popularity in the drug research and neuroscience industries. Optogenetics is the method of controlling neurons with the usage of lights. Optogenetics allows for rapid regulation of neural spiking with millisecond accuracy. As an approach that employs both optics and genetics. Optogenetics is beginning to translate and transit into drug discovery in several key domains, including target discovery, high-throughput screening, and novel therapeutic approaches to disease states. (Song, 2015)

Automation has long been prevalent in the computer industry, and it has resulted in significant complex advancements in what tasks computers can perform. One of these advancements that needs to be highlighted is the computer vision sector. Computer vision in simple terms is the ability of the computer to derive information about its surroundings using a device such as a camera. Computer vision receives information from cameras as data so that computer hardware and software can process, analyze, and evaluate numerous features for decision making.

Fully Automated microscopy would allow rapid evaluation of responses to a range of conditions as well as compound screening for drug discovery using samples in multiwall plates.

The objective of this thesis is to use computer vision on fluorescence automated microscopy to help cell imaging in detecting signaling events from cells based on calcium spikes.

This chapter began by addressing the study's history and context. The second chapter examines the importance of automated microscopes in health science as well as the many types of automated microscopes available. The third chapter covers the microscope that is picked for our thesis and its functionalities. The theory underlying the use of cell detection will be thoroughly discussed in the fourth chapter. With code examples, the methods presented in the fourth chapter will be put into application in the fifth chapter which will demonstrate how to set up cell detection and analysis. Finally, chapter six summarizes the actions taken in this thesis, highlights the accomplishments, and provides recommendations for future development.

2 AUTOMATED MICROSCOPY IN HEALTH SCIENCES

2.1 Importance of Microscopy in Drug Research

The use of fluorescence microscopy on living cells has become an essential aspect of current cell biology. The term "fluorescence microscopy" refers to any microscope that generates images using fluorescence. Fluorescence is the emission of light by a material that has previously absorbed light. Cells become fluorescent when the light of a certain wavelength (or wavelengths) is shone on the specimen, which is absorbed by the molecules that leads them to produce light of longer wavelengths. These molecules are known as fluorophores.

Due to fluorophores, fluorescent probes can detect cellular signaling events, which are necessary for an organism's biological responses. Because of the wide variety of cells present, cell responses can be dynamic, making the evaluation of its performance difficult. An efficient method that will be discussed in the fourth chapter of this thesis is that each cell's reaction must be monitored regularly to better understand how it has been performing.

2.2 Automated Microscopy

Until now, microscopes require a user to manually fine-tune the microscope parts to obtain an adequate result that matches what the user wanted to observe, e.g., sliders, objective lens, brightness, and so forth. This fine-tuning made the measurements especially difficult in operations requiring many repeated observations over an extended period of time. This is where an automated microscope improves the previously mentioned manual components of the microscope by computerizing them. Since these components are computerized, they do not require regular user involvement to calibrate.

Microscopy automation removes the constant user moderation from result acquisition. The automation lowers the user intervention that is needed for

calibrating the microscope functionalities. This procedure is especially beneficial in situations that demand a large number of repeated observations over time.

Automated microscopy is a technique that saw an increase in usage within fields that utilize microscopic functionalities. While automated microscopy also transmits imagery data from its optical lens, it also allows the user to calibrate the microscope through imaging software and, in optimal cases, record these settings as metadata. The imaging software controls the microscope's built-in electrical components that correspond to it its manual counterpart, to obtain a satisfactory result.

Motorized microscope components and accessories, or their solid-state counterparts enable the investigator to automate live-cell image acquisition and are particularly useful for time-lapse experiments that range in timescale intervals from milliseconds to tens or hundreds of minutes. (The Automatic Microscope, n.d.)

2.3 Manufacturers of automated microscopy

There are as many as a dozen and growing number of manufacturers of automated microscopes, all ranging in different prices, functionalities, sizes, and most importantly design. This section introduces the most important manufacturers and their products. The third chapter in this thesis covers in detail the automated microscope used during the current research.

2.3.1 Finchscope by Gardner Lab

Finchscope (Figure 1) is a lightweight automated open-source microscope intended for the use of both wired and wireless microscopy. The device, as it occurs with most of the microscopes presented in this section, comes with its software. The software runs only in MacOS as of now. Its Data acquisition device, also called DAQ, is based on a prototype electronic hardware device called Arduino.

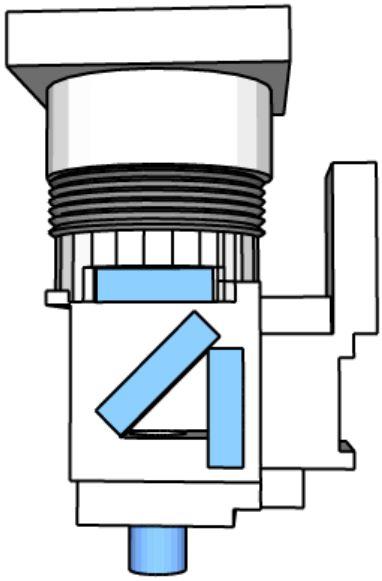

Figure 1. Finchscope Miniscope. (FinchScope, 2017)

2.3.2 Inscopix Miniscope

The commercial Inscopix miniscope (Figure 2) collects fluorescent changes in real-time. Imaging, optogenetic, and behavioral data are transmitted via a flexible cable to the data acquisition (DAQ) box, and the images are remotely streamed live on a computer or mobile device. (Inscopix, n.d.)

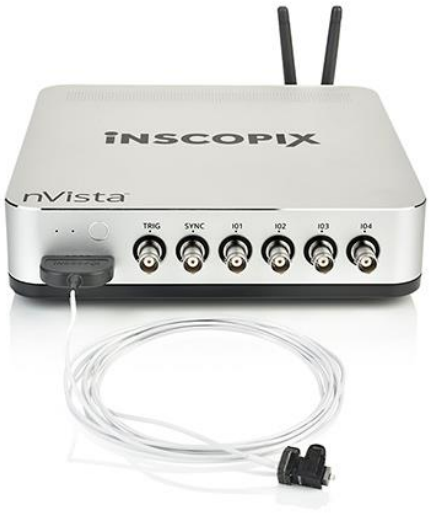

Figure 2. Inscopix Miniscope. (Inscopix, n.d.)

2.3.3 CHEndoscope

This open-source miniature microscope as illustrated in Figure 3 device is designed to provide an accessible set of calcium imaging tools to investigate the relationship between behavior and population neuronal activity for *in vivo* rodents. The CHEndoscope is open source, flexible, and consists of only 4 plastic components that can be 3D printed. (CHEndoscope - OpenBehavior, 2018)

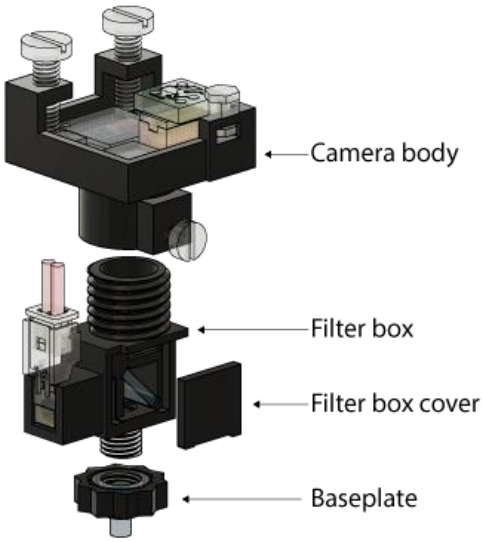

Figure 3. CHEndoscope Miniscope. (CHEndoscope - OpenBehavior, 2018)

3 MICROSCOPIC FUNCTIONALITIES

This chapter aims to cover the microscope, and its functionalities, that are used in this thesis. The automated microscope which is shown in Figure 4 is Miniscope v4 (Aharoni and Golshani, n.d.). This device was picked for this testing due to having a higher than the average frame rate and its focus comes as a linear slider.

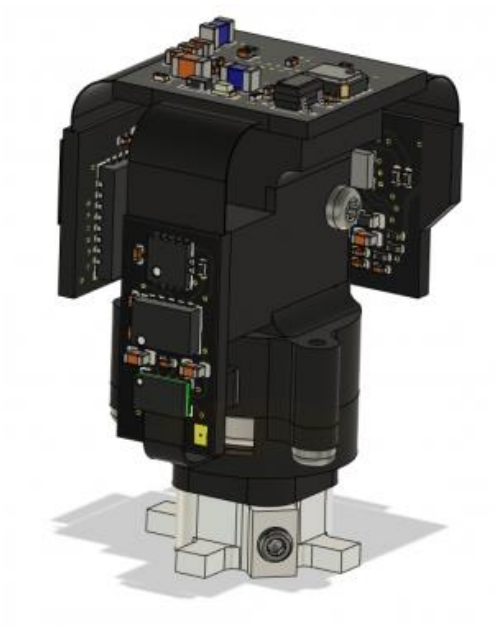

Figure 4. Miniscope v4.

3.1 Miniscope v4 - Microscope

As presented in Figure 5, a DAQ box and computer software are included in addition to the Miniscope to handle the information transmitted by the Miniscope.

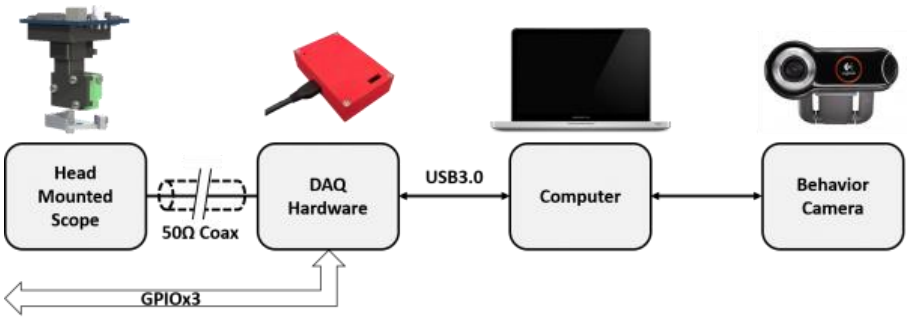

Figure 5. Miniscope v4 Assembly.

3.2 Miniscope v4 - DAQ

A data acquisition device is a hardware device that connects the computer and the sensors. DAQ samples signals that measure physical occurrences and translate them to a digital data that can be managed by a software

3.3 Miniscope v4 – Software

There is dedicated software provided for this microscope. The software allows control of the parameters of each component that make up the microscope such as focus slider, brightness intensity, frame rate, gain, and upper/lower levels of the frame. This software is designed entirely for Windows, however, it does support other operating systems in limited ways because of the use of QTcreator.

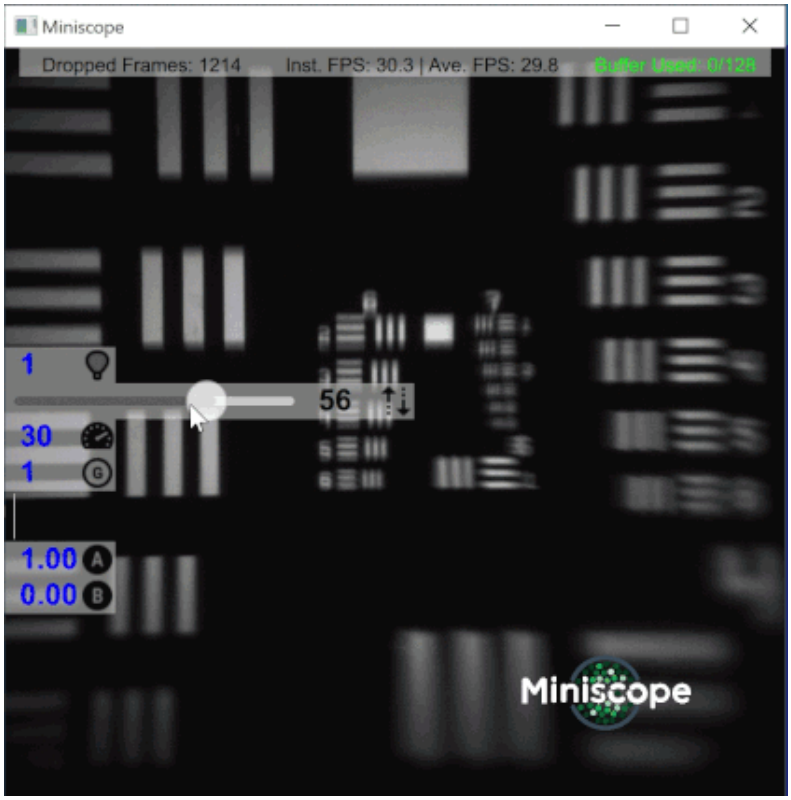

Figure 6. The automated microscope software interface.

Each of the parameters depicted in Figure 6 that relate to the components stands for:

: LED intensity.

: Focal plane

: Frame rate

: Gain

: Upper level of displayed frame. Does not affect saved data.

: Lower level of displayed frame. Does not affect saved data.

4 UNDERSTANDING THE USAGE OF CELL DETECTION

In this part, the hypothetical cell detection procedure and each operation that is a part of image segmentation as a whole and will be discussed in detail and layman's terms.

The image segmentation method may be divided into many parts as seen in Figure 7, which is discussed in further detail in the upcoming practical topic of this thesis. Greyscalling, binary thresholding, and contour detection will be covered in the next few sections.

Figure 7. Theoretical Image Segmentation Process.

4.1 Automation

The initial procedure that was taken to work on the miniscope was to make it efficient and fully automated. This meant that the miniscope could, but was not limited to, detecting and reporting on every active cell.

Detection algorithms are image processing techniques used to find spots in a digital image that have discontinuities, or sudden changes in image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image. (Settem, 2021)

Edge detection focuses on regions where the strength of light intensity varies fundamentally. Nonetheless, contours are dynamic assortments of illuminations and segments relating to the states of the objects in the image. Accordingly, we can control contours in our program by counting several contours, utilizing them

to order the states of objects, trimming objects from an (image segmentation), and substantially more.

Detection algorithms have been introduced like Canny edge detection, Sobel edge detection, Contour detection, etc. Each of these methods aids in the detection of cells. Contour detection is the one we'll be concentrating on in this topic because it helps us recognize cells more accurately in the long term. Figure 8 depicts an example of what Canny edge detection can accomplish. Regions with significant pixel color intensity variations are highlighted and presented on a black picture background.

Figure 8. Contour edge detection example.

4.2 Greyscale

The perception of colored images differs as there are many different colors in an image that our human eyes can perceive as opposed to the computer, which has a far broader spectrum of possibilities. Most modern computer monitors can display millions of colors. The discovery of materials from which LEDs (light-emitting diodes), phosphors, and liquid crystals are made makes these colors possible. The smallest unit on the computer screen or a camera is called a pixel. (JCE staff, 1998)

Images are made up of pixels and channels. Each pixel of a Standard RGB image value ranges between 0-255 (8-bit unsigned integers), linearly

representing the intensity of each of the three channels, Red, Green, and Blue. The standard RGB image is 24 bits but 48 bits size also exists. Figure 9 presents the three channels as a reference.

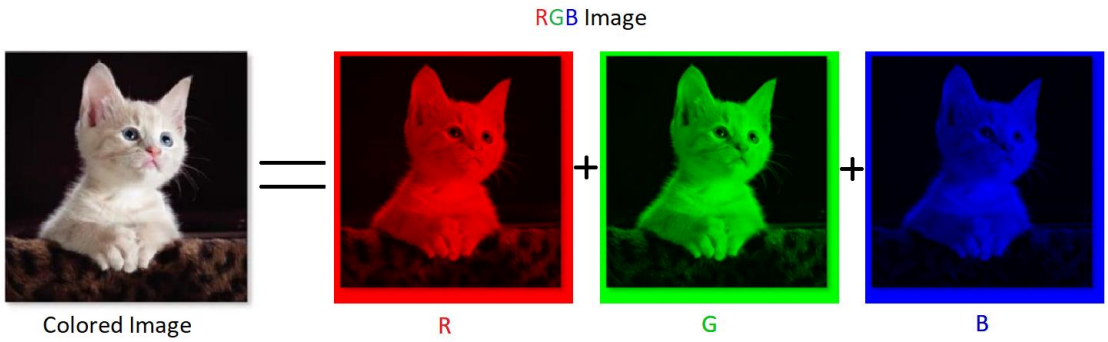

Figure 9. Image color layering example.

Due to RGB images including a lot of information that, in some cases, maybe redundant for image processing, it may impede or slow down the edge detection process. Alternatively, the three channels may carry different information that can be used and combined. In the present case, a one-channel image will be considered.

A single-channel image can be obtained by converting a multi-channel image to a single channel or converting the image to greyscale. Ultimately, this is important for preparing an image for the next step of image segmentation, which is thresholding.

4.3 Binary Thresholding

Binary thresholding is a type of *image segmentation applied to simplify the analysis of images. In our context, RGB data will be converted to only black and white images.* Figure 10 shows the effect of binary thresholding, where the original image including different grey levels (left) has been changed to black and white only (right). The outcome of running a binary threshold on the right. (Thresholding – Image Processing with Python, 2022).

Binary is a computer word that refers to any system with only two possible values, for instance, 0 and 1. The information of each pixel in the figure will be compared to an arbitrary threshold, and everything over and under that threshold will be converted to black or white respectively.

Binary thresholding handles two procedures in the image. The first being the binary and the second being the thresholding of that. The binary part deals with an 8-bit image that ranges from 0 to 255 for each pixel. A binary image is an image that has just two potential values of pixel Intensity. They are 0 for dark and 255 for white.

Figure 10. Binary thresholding example.

4.4 Contours identification

Contouring is a process for shape examination and object location. Contour is a boundary around something that has distinctly defined edges, which implies that the machine can compute contrast in angle, attempt to check whether the similar result proceeds and structure a conspicuous shape and draw a boundary around it.

In Figure 11, all the preceding steps have been completed, and the binary threshold is placed on top of the original picture. This results in the contour effect.

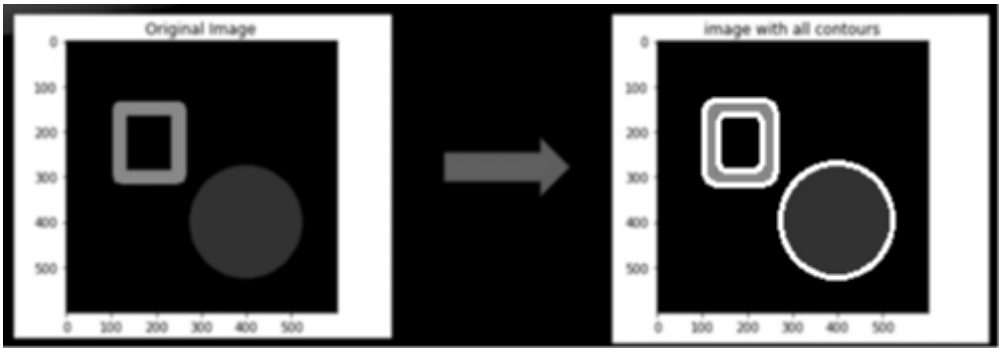

Figure 11. Contour example. Original (Left) and annotated (Right).

5 APPLICATION OF CELL DETECTION

This section presents the contribution of the author to the field of automatic microscopy for drug research (or other application in life science). In practice, the miniscope v4 will be used to capture an image of cells, which is thereafter interpreted by the programmed algorithms. The theoretical sequence presented in Section 4 is still followed for the most part, but two additional adjustments were made as seen in Figure 12. The additional adjustments are noise reduction to remove distortions in the image, and Region of Interest to aid in cell tracking and image analysis. Some sequential procedures are required because each step has a goal to achieve.

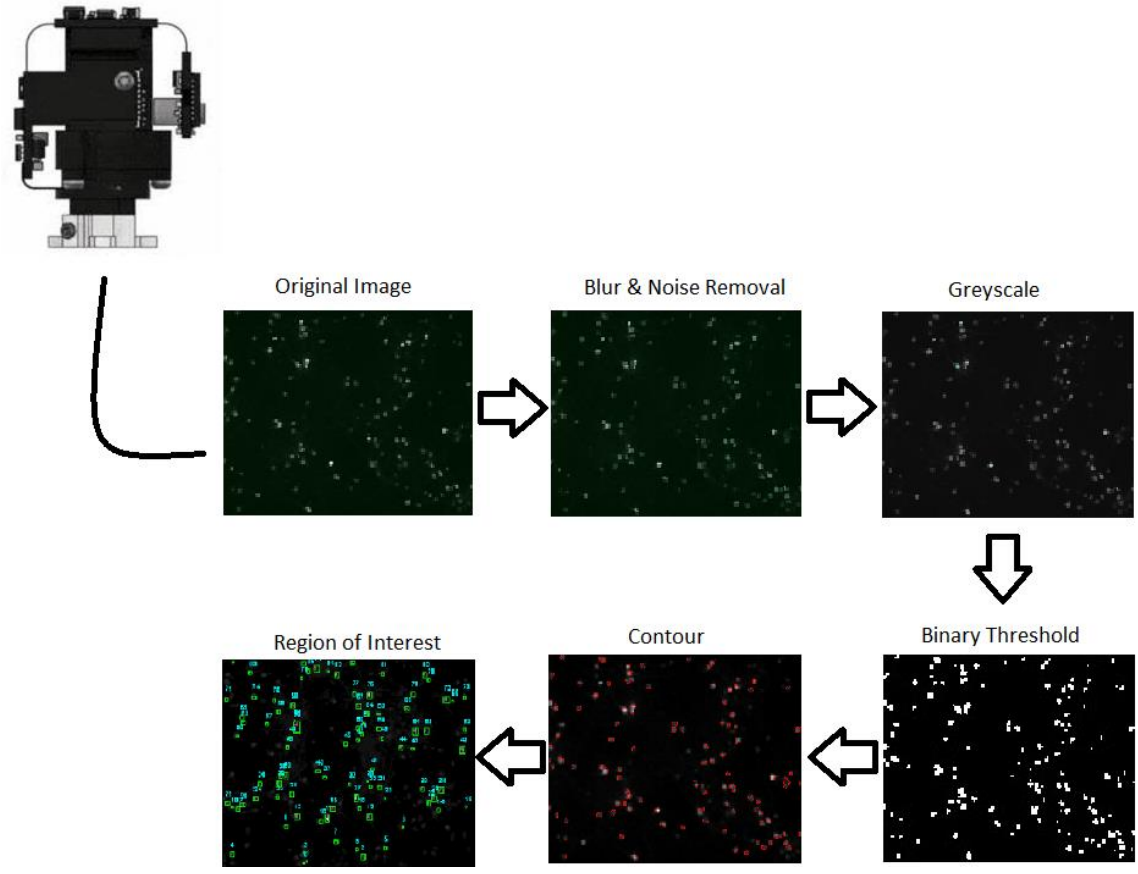

Figure 12. Practical Image Segmentation Process.

5.1 Image Segmentation

5.1.1 Get Image

There is an open-source and cross-platform programming library for this purpose. The library is called OpenCV and it is completely free. OpenCV is a leading open-source library for computer vision, image processing, and machine learning, and now it features GPU acceleration for real-time operation. (Nvidia OpenCV, 2019).

The python interface version was used in this case as it doesn't need any additional installations. The code below is essential for the first step of the procedure to produce Figure 13. The first step is to connect the camera output through code.

```
1. # import the OpenCV library
2. import cv2
3.
4. #Variable to hold capture object
5. vid = cv2.VideoCapture(0)
6.
7. while(True):
8.     # Camera Capture is enabled and shown frame by frame
9.     ret, frame = vid.read()
10.
11.      # Resulting frame is shown
12.      cv2.imshow('frame', frame)
13.
14.      # Video capture is stopped
15.      # if "Esc" is pressed
16.      # the last frame is saved as png
17.      c = cv2.waitKey(1)
18.      if c == 27:
19.          cv2.imwrite('images/image.png',frame)
20.          break
21.
22.  # After the capture is done
23.  # it is a good idea to release
24.  # the capture object
25.  vid.release()
26.
27.  # Destroy all the windows
28.  cv2.destroyAllWindows()
29.
```

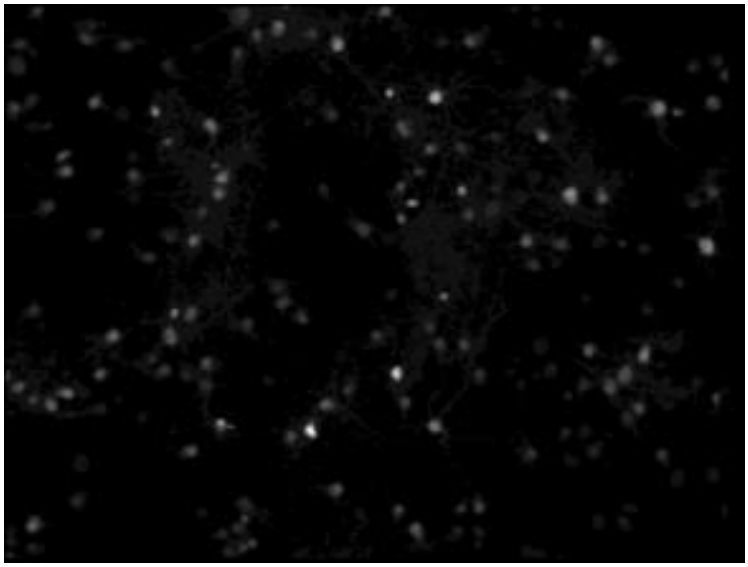

Figure 13. Cell Image, Original image.

5.1.2 Noise removal

Most images include noise or distortions that make distinguishing the shapes or objects difficult. These constraints must first be reduced by processing the image. Typically, images are preprocessed to remove noise or smooth them out. Because of these limitations, further steps mentioned at the start of this section are necessary when approaching binary thresholding through code than the hypothetical approach. Some image preprocessing techniques include, but are not limited to, Gaussian, Median, and Bilateral blur. In our scenario, it should be sufficient to use median blur, which removes outliers (speckle noise) and limits size distortions.

Because images include numerous pixels, blurring an image allows the color to shift smoothly from one side of an edge to the other. The concept of edges is crucial in identifying objects in an image: the lines that represent a transition from one group of similar pixels in the image to a different group. The pixels in an image that indicate the boundaries of an item, where the image's backdrop ends and the object begins, are an example of an edge.

```
1. # Remove noise from the image
2. blur = cv2.medianBlur(frame, 3)
```

By blurring the image, we had captured, we end up with less sharp edges and perceivable differences in background and object as presented in Figure 14.

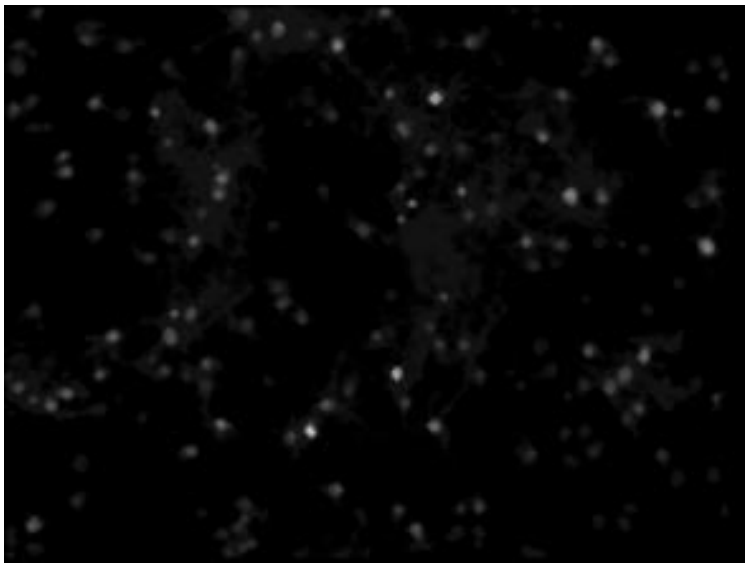

Figure 14. Cell Image, Blur.

Filtering image data is a standard process used in almost every image processing system. Filters are used for this purpose. They remove noise from images by preserving the details of the same. The choice of filter depends on the filter behaviors and type of data. (Swain, 2018)

5.1.3 Greyscaling

Greyscale helps with lessening the information held in each pixel as further explained in Section 4.2.

Blurring or removing noise from an image also reduces the size of an image. With an appropriate blurring function, we can *deconvolution* the blurred image into the original image. This can be very helpful in transferring data that is vast in size. (Pradia Naufal, 2020)

```
1. # Greyscaling the capture object output
2. grey = cv2.cvtColor(blur, cv2.COLOR_BGR2GRAY)
```

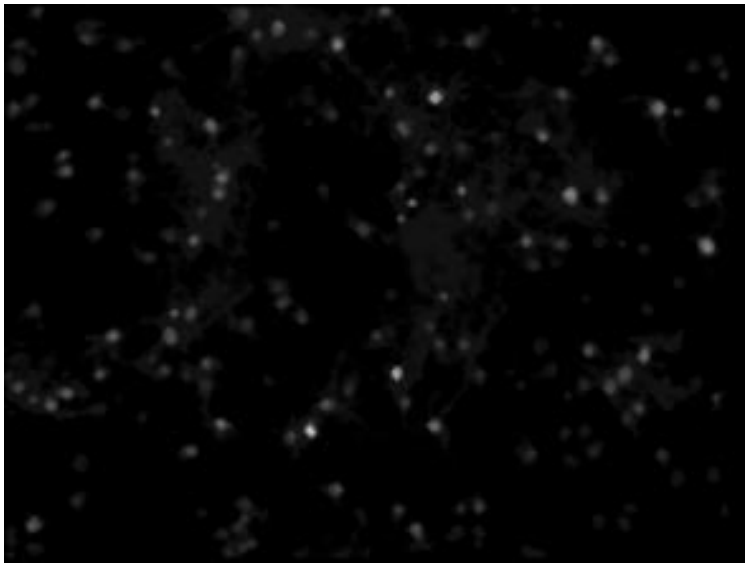

Figure 15. Cell Image, Grey.

Figure 15 presents our image after the greyscaling process. The outlook does not differ much from the image presented in Figure 14, because our data was already captured in grey tones.

5.1.4 Binary Thresholding

In Binary Thresholding, each pixel value is compared to the threshold value. If the pixel value is less than the threshold, it is set to zero; otherwise, it is set to the maximum value which is always 255.

```
1. # Convert to black and white
2. thresValue = 40
3. maxValue = 255
4. thresh = cv2.threshold(grey, thresValue, maxValue,
   cv2.THRESH_BINARY)[1]
```

In some cases, where the optimal threshold value is unknown, automated threshold algorithms such as Otsu thresholding might be useful. Otsu's thresholding method involves iterating over all possible threshold values and computing a measure of variance for the pixel levels on each side of the threshold, i.e. pixels in the foreground or background.

The aim is to find the value at which the sum of the foreground and background variance is minimal.

```
1. # Convert to black and white
2. # using Otsu thresholding
3. maxValue = 255
4. thresh = cv2.threshold(grey, 0, maxValue, cv2.THRESH_BINARY |
   cv2.THRESH_OTSU);
```

Figure 16. Cell Image, Binary.

5.1.5 Finding contours

Following the successful discovery of the boundaries of each object in our image as seen in Figure 16, moving on to the next stage would mean that the contours should be extracted and identified. Simply overlaying the detected edges on top of the original image will give us the precise space occupied by each object.

```
1. # Draw contours on original image
2. cnts = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
   cv2.CHAIN_APPROX_SIMPLE)
3. drawnCnts = cv2.drawContours(image, contours[1], -1, (0, 0, 0),
   2)
```

Because we now have access to each object location, ROI can be useful in constructing a method to identify and track each cell for subsequent study. This

helps in further interpreting the image. As presented in Figure 17, ROI may improve our image analysis by assigning a region of interest to each contour for further examination of its behavior such as movement, intensity, size, and so on. Once the contour is found, the rectangle shape is placed on top of each cell to help in understanding where each cell is positioned.

```
count = 0
# Iterate through all the contours
for contour in contours[1]:
    # Find bounding rectangles
    x,y,w,h = cv2.boundingRect(contour)
    # Draw the region of interest
    cv2.rectangle(drawnCnts,(x,y),(x+w,y+h),(0,255,0),1)
    #optional labeling for the region of interest
    cv2.putText(drawnCnts, str(count), (x, y-6),
cv2.FONT_HERSHEY_SIMPLEX, 0.2, (255,255,0), 1)
     count += 1
```

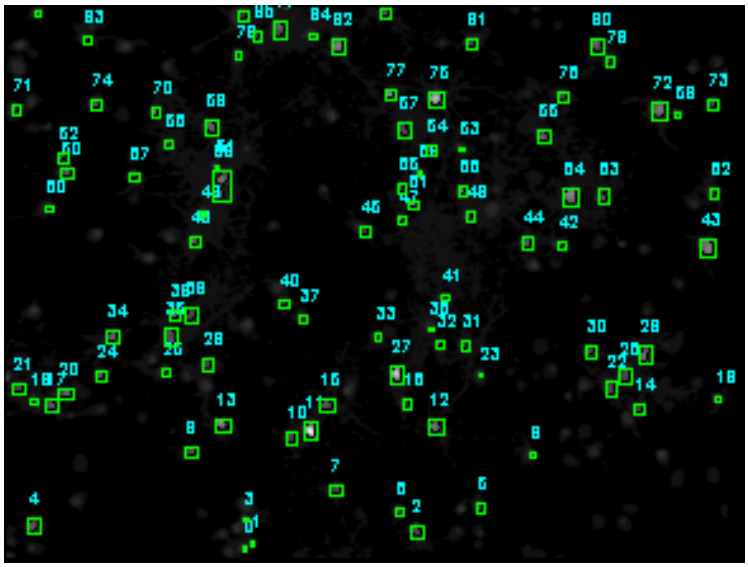

Figure 17. Cell Image, ROI.

5.2 Tracking activity of the ROI

The ROI's activity can be determined by tracking its shapes or light intensity. Therefore, in this stage, concentrating on the intensity of the light rather than

the shape is a viable option. Graphing our light intensity will offer us an indication of what is occurring in our cells.

However, a single image is insufficient for graphing. In this situation, the first image might be utilized as a starting point for defining the cell positions. To graph the cell activity during the procedure, numerous images of the same cell acquired at different time points must be used. A video is another option, but it would have to be analyzed frame by frame.

In this stage, different samples are captured from a single cell which is analyzed to assess its activity over time. In Figure 18, the horizontal axis represents time, while the vertical axis represents the intensity of light.

Currently, a line plot is being used to represent the data that is provided, other styles can be used as well such as but not limited to scatter plot, bar graph, or histogram. This is all depending on what is needed to be understood from the data collected.

```
1. # Data variable contains contour of one ROI
2. plt.plot(data[:, 1])
3. plt.show()
```

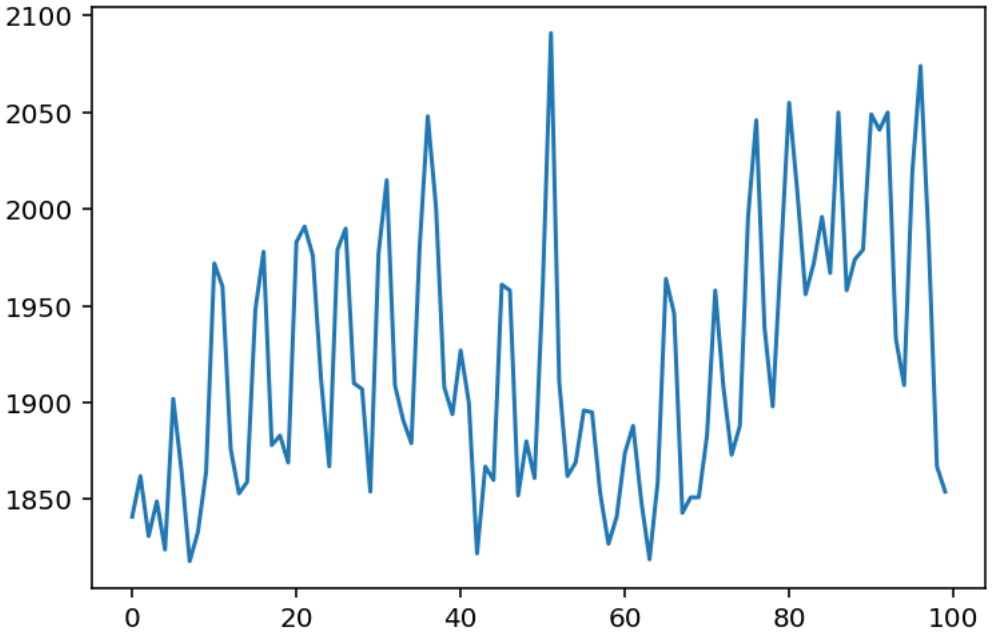

Figure 18. Cell image, Single graph.

Following the success of our graphed cell as seen in Figure 18, the data presented is essential in understanding how the cell behaved during the experiment. It is also possible to do this for each cell. Tracking numerous ROI would imply that each cell would need to be iterated through to graph them properly.

```
1. # Data variable contains all contour locations of one ROI
2.
3. current_shape = 0
4. image_total = len(contours[1])
5. plot_size = int(np.sqrt(image_total)+1)
6.
7. i_total = plot_size
8. j_total = plot_size
9. figure, axis = plt.subplots(i_total,j_total, figsize=(20,20))
10.  for i in range(i_total):
11.    if current_shape >= len(data[0]):
12.      break
13.    for j in range(j_total):
14.      axis[i][j].plot(data[:, current_shape])
15.      current_shape+=1
16.      if(current_shape >= len(data[0])):
17.        Break
18.  plt.show()
```

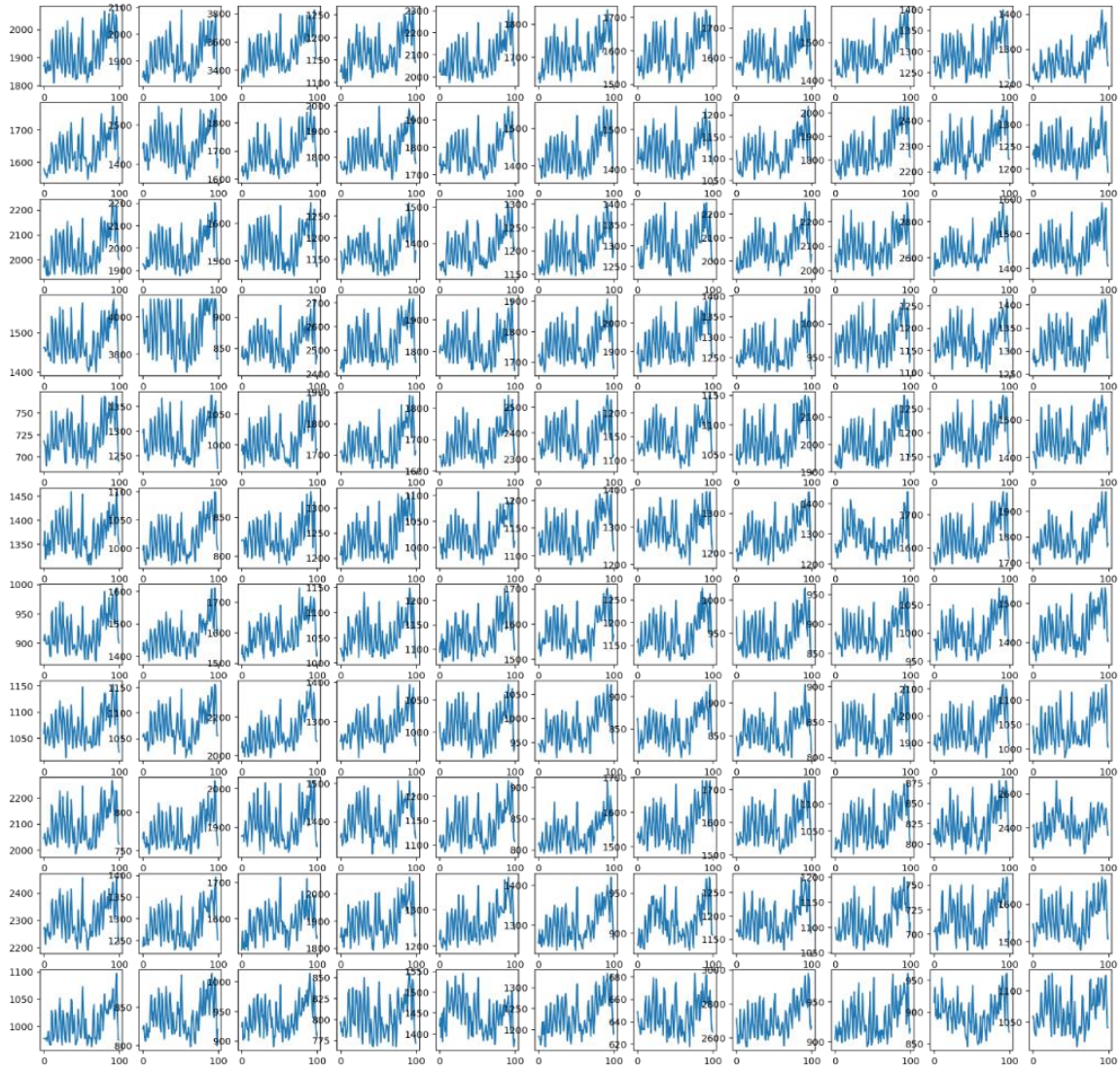

Figure 19. Cell image, Multiple graphs.

The tracking of each cell, as shown in Figure 19, provides a whole different viewpoint on how the procedure has progressed. A single targeted cell does not reveal much, but numerous targeted cells may be stitched together to have a clearer understanding of how the cell experiment went.

Some additional steps that are taken commonly to manipulate the data are:

- As each cell may have a different beginning intensity of fluorescence, each line graph is normalized to the original signal. When studying disturbances of a steady-state, normalization based on the original signal of the cell is useful.

- The background signals are subtracted to isolate the signals that belong to the cell. This is done in case the image has a background problem or uneven illumination.

6 CONCLUSION

The objective of this thesis was to use computer vision on fluorescence automated microscopy to help cell imaging in detecting and analysing signaling events from cells based on calcium spikes. This thesis presented the algorithms developed and demonstrated in the previous chapter to improve the automated microscope result. The set of algorithms constitutes a form of machine vision that is expected to improve the quality and efficiency of research projects applying microscopy.

The success of this project is determined by the compatibility of the automated microscope and detection code written. The practical outcome revealed that a working prototype with both assets integrated is achieved.

With the automated microscope, it is now possible to execute cell imaging activities even when the researcher is not physically present in the same room. This, along with the extra benefit of wireless operation and adjusting, has the potential to transform the future of microscopy.

The possibility of programming the microscope adds value on an unprecedented level. This approach would make it so that graphing of each cell is carried out and reported to the user. Various other data could also be provided to the user such as timing, size, and the intensity of each cell.

Further development could be achieved with increased manpower and elimination of time constraints, therefore rendering this project more valued and accomplished.

Some additional features for future development that were planned but not implemented in this thesis included:

- A microscope component automating feature to autofocus on cells
- A notification system to warn the user about cell activity
- A feature to have an array of automated microscopes communicating with each other
- An app to stream the microscope content.

- Editing the software that came with the microscope to include my algorithm within it

Some other challenges the author faced while working on this thesis have been:

- Coming into this research with little to no experience in cell imaging or optogenetics. The author had previously worked on a comparable project, which aided him slightly with certain niche elements.
- Programming and determining what aspects of the project would be addressed in the thesis. Additional programming for the project has been completed, however, part of it is not included in this thesis.
- Testing and learning about the automated microscope environment. Ultimately this has taken the majority of time working on the project

REFERENCES

Aharoni, D. and Golshani, P., n.d. *UCLA Miniscope - The BRAIN Initiative*. [online] The BRAIN Initiative. Available at: <https://www.braininitiative.org/toolmakers/resources/ucla-miniscope/> [Accessed 1 June 2022].

Datacarpentry.org. 2022. *Thresholding – Image Processing with Python*. [online] Available at: <https://datacarpentry.org/image-processing/07-thresholding/> [Accessed 1 June 2022].

GitHub. 2017. *FinchScope/im1.png at master · gardner-lab/FinchScope*. [online] Available at: <https://github.com/gardner-lab/FinchScope/blob/master/FinchScope/img/im1.png> [Accessed 1 June 2022].

JCE staff, 1988. *How Many Colors in Your Computer? Discovering the Rules for Making Colors*. [online] ACS Publications. Available at: <https://pubs.acs.org/doi/abs/10.1021/ed075p312A> [Accessed 1 June 2022].

Inscopix (a), *Stanford scientists capture mouse memories | Inscopix*. [online] Inscopix.com. Available at: <https://www.inscopix.com/stanford-scientists-capture-mouse-memories> [Accessed 1 June 2022].

Inscopix (b),. *Miniature Microscope Technology & Solution | Inscopix*. [online] Inscopix.com. Available at: <https://www.inscopix.com/miniature-microscope-technology#technologyminiature> [Accessed 1 June 2022].

Nikon's MicroscopyU. n.d. *The Automatic Microscope*. [online] Available at: <https://www.microscopyu.com/applications/live-cell-imaging/the-automatic-microscope> [Accessed 1 June 2022]. Naufal, A. P., 2020. *Medium.* [Online] Available at: https://medium.com/swlh/how-image-blurring-works-652051aee2d1

NVIDIA Developer. 2019. *Nvidia OpenCV*. [online] Available at: <https://developer.nvidia.com/opencv> [Accessed 1 June 2022].

OpenBehavior. 2018. *CHEndoscope - OpenBehavior*. [online] Available at: <https://edspace.american.edu/openbehavior/project/chendoscope/> [Accessed 1 June 2022].

Pradia Naufal, A., 2020. *How Image Blurring Works*. [online] Medium. Available at: <https://medium.com/swlh/how-image-blurring-works-652051aee2d1> [Accessed 1 June 2022].

Settem, S., 2021. *What is Edge Detection – An Introduction*. [online] mygreatlearning. Available at: <https://www.mygreatlearning.com/blog/introduction-to-edge-detection> [Accessed 1 June 2022].

Song, C. and Knöpfel, T., 2015. Optogenetics enlightens neuroscience drug discovery. *Nature Reviews Drug Discovery*, [online] 15(2), pp.97-109. Available at: <https://www.nature.com/articles/nrd.2015.15 > [Accessed 1 June 2022].

Swain, A., 2018. *Noise filtering in Digital Image Processing*. [online] Medium. Available at: <https://medium.com/image-vision/noise-filtering-in-digital-image-processing-d12b5266847c> [Accessed 1 June 2022].