A MATLAB Approach to Automated Bacterial Colony Counting

Jackie Marlin#1*, Jake Starr#2*

1Department of Electrical and Systems Engineering
Washington University in St. Louis
1 Brookings Drive
St. Louis, MO 63130
USA
1jackie.marlin@wustl.edu
#2starr@wustl.edu

*These two authors contributed equally to the work

Abstract The purpose of this project is to use MATLAB to create an automated method to count the number of bacterial colonies on a given agar plate, from an image of the plate. Counting the number of colonies on an agar plate is an extremely common activity in microbiology labs for medical, pharmacological, and agricultural uses. Currently, colony counting is done mostly manually because devices made for automatic counting are expensive and not user-friendly. For this project, the Circular Hough Transform is utilized to detect circular features from the images. These images were taken under differing lighting conditions for 20 combinations of agar types and species of bacteria; this provided a large dataset to develop our four different algorithms. Our MATLAB script is implemented to ultimately output a colony count with up to 98% accuracy, depending on the quality of the original image.

Index Terms—Biology computing, Edge detection, Image analysis, Image capture, Image quality, Object detection

I. INTRODUCTION

One of the most common tasks that takes place in microbiology labs is the counting of bacterial colonies that grow on agar plates. The most basic, and common, way to accomplish this task is by manually counting each colony that is present on the plate with a clicker, shown in Figure 1. However, this is extremely time consuming, requires little thought from the highly trained microbiologists, and there is significant room for human error.

A slightly more advanced manual counting technology utilizes a hatched grid and a light source (overhead or below) that holds a transparent agar plate. Moving square by square on the hatched grid, the microbiologist will tap each individual colony with a stylus. Each tap signals a noise or vibration from the apparatus that verifies the counting of each tapped colony. The device stores the total number of taps as the number of colonies present on the plate. This process is still quite time consuming, and only slightly eliminates human error. Additionally, it still takes microbiologists away from more skilled tasks. A typical manual colony counter is pictured in Figure 2.

The most advanced labs may utilize automated colony counters, but these tend to be extremely expensive, error-prone, and user-unfriendly [3]. These automated colony counters operate by capturing an image of the plate, and then using various image analysis techniques, such as altering contrast values and edge detection, process the image and count the number of colonies present. A commercial example of an automated colony counter is shown in Figure 3.

Testimonials from current microbiologists suggest that these automated counters are far from ideal, yet clearly
neither is manual counting. There is significant need for a bacterial colony counting method that is affordable, accurate, and efficient, and this project works towards the development of such a method [5].

![Figure 1](image1.png) Figure 1 (above): A clicker and a marker used to manually count bacterial colonies on an agar plate [1].

![Figure 2](image2.png) Figure 2: A more advanced manual colony counter with an attached overhead light. The circular region contains a hatched grid on which a stylus is tapped on each visible colony. The machine registers each tap and stores a count that will ultimately be the number of bacterial colonies present [2].

A. Review of Literature

In addition to research regarding currently available automated colony counters, the various filtering methods used on images of agar plates with bacteria growth present to most accurately determine the colony count were researched.

Past research shows that the main obstacles in creating an automated colony counter are illumination (shadows, reflections), bacterial colony overlap, and imperfections in the agar or plate itself (dust particles, lumps, etc). However, colony overlap and imperfections can be tackled using image binary conversion, where all pixels less than a set grayscale value are deleted from the image [7]. This eliminates potentially problematic pixels that could interfere with the counting and introduce false positives.

![Figure 3](image3.png) Figure 3 (above): Automated colony counter, with output interface shown behind. Captures image of plate, analyzes and filters image, and returns the number of colonies as well as an image of what the device determined to be colonies [4].

Our program implements the Hough Transform, which Paul Hough developed in 1960 as a “method and means for recognizing complex patterns” [12]. Hough’s intention was to streamline the analysis of bubble chamber pictures since many photographs needed analysis and when done by hand each photograph took several hours to analyze [12]. Essentially, Hough was solving the same problem for the study of atomic particles as we would like to solve for the calculation of sample bacterial concentrations. In MATLAB, the circular Hough Transform works by first analyzing the given image for high-gradient pixels -- that is, pixels that are a vastly different color than the surrounding pixels. By selecting several of these high-gradient pixels in close proximity to one another, it can be seen how they are related to one another. In our case, we are analyzing only for a circular relationship...
where the given high-gradient pixels lie on the edge of a common circle. Based on the angle and distance between the points, the program can estimate the radius and center of a found circle [13]. The circular Hough Transform is a good option to use due to its robustness in the face of image noise and its ability to detect even circles that are only partially visible (which is important for the detection of intergrown colonies).

B. Aims and Objectives

This project aims to optimize bacterial colony counters to:

- Overcome some of their main inconveniences, and
- Incorporate some requests for improvement directly from professional microbiologists who use these current models of colony counters on a daily basis.

Input from professional microbiologists, or the “voice of the customer” input, will ensure that our project will be useful to lab technicians and microbiologists today, as well as demonstrate the necessity for the development of such a tool.

This project will be completed using different processing methods available within the MATLAB Imaging Toolbox, interfaced with cameras ranging from 1-5 megapixels provided by bioMerieux, ultimately to be used to capture images of colonies of bacterial strains differing by size, color, and density. Specifically, we will be testing five different bacterial species: E. Coli (Gram Negative), Klebsiella Pneumoniae (Gram Negative), Staphylococcus Aureus (Gram Positive), Pseudomonas Fluorescens (Gram Negative), and E. Fissicatena (Gram Negative). In addition to different bacteria, different types and colors of agar will be used: SabDex Agar, Tryptic Soy Agar (TSA), Mueller Hinton Agar, Blood Agar (also known as TSA-B), and Chocolate Agar, pictured below in Figure 4. SabDex is used for a variety of Gram-positive and -negative bacteria and fungi, and is composed largely of Dextrose [5]. TSA is also largely composed of Dextrose, which is ingested by bacteria as a main energy source, but is primarily used to isolate organisms from our body. Mueller Hinton is used primarily for antibiotic susceptibility testing of nonfastidious organisms, or those that are not restricted to ingesting a particular substance. Blood Agar contains about 5% sheep blood and is used to determine the types of hemolysis present (the process by which red blood cells are ruptured). Chocolate agar is generally used to grow fastidious respiratory bacteria, such as flu viruses [6]. These choices provide a variety of pigmentation and bacterial morphologies to facilitate building a program that can cope with a high range of bacteria-agar contrasts, and will produce a good-sized sample group of images for our initial testing.

![Figure 4: The different colors and transparencies of agar to be used, from left to right: (top) SabDex, TSA, Mueller Hinton, (bottom) TSA-B, and Chocolate Agar. Each agar works with different bacteria types, which allows for a variety of contrast combinations to test with the code. Source: Own image.](image)

II. METHODS AND PROCEDURES

In the initial phase of our project, we interviewed a microbiologist at bioMerieux to learn about common experiences and preferences for an automated bacterial colony counting system. This process allowed us to obtain the “voice of the customer,” which is central to our goal of creating something that has tangible value to scientists today. Incorporating the “voice of the customer” is what separates this project from the other automated image processing-based colony counters that are available today. After the interview, we researched current open-source solutions to this problem to obtain new perspectives from other approaches to the problem, as well as to evaluate their shortcomings.

Multiple MATLAB scripts were then written utilizing the Hough Transform within the Image Processing Toolbox’s imfindcircles function. The function takes in several input parameters:

- An image file,
- A radius parameter given as a 1x2 array containing both the minimum and maximum radii of circles to search the image file for,
- An ObjectPolarity parameter specifying whether the circles are bright or dark with respect to the agar, and
- A Sensitivity parameter which sets how circular an object has to be in order to be detected (Sensitivity values are on a scale from 0 to
A. Method 1: Semi-Automatic User-Input Based Method

We developed four different methods to effectively count colonies. The first method can be described as being “semi-automatic” in nature, since it requires user-input in terms of colony radius and Sensitivity setting. The Sensitivity parameter is vital to the counter’s accuracy, as it determines the threshold at which the function will stop detecting circles. A high Sensitivity setting allows for the detection of more objects, such as overlapping colonies or imperfectly circular colonies. As a result, an excessively low Sensitivity setting will underestimate the true number of colonies, but an excessively high setting will overestimate the number of colonies.

This first method imports an image of a Petri dish with bacterial colonies present, overlays it with a pixel-scale ruler that the user can drag around to determine the minimum and maximum radii of the colonies. The function then generates two results, one for each ObjectPolarity setting. These results are then plotted over the original image and shown to the user, who is then asked to determine the best ObjectPolarity setting and report it to the program. Once the user has inputted their choice, the function will run again using the determined settings for radius and ObjectPolarity for a wide range of Sensitivity settings, from 0.85 to 0.97. This range was determined by trial and error to be best able to detect the presence of a range of colony sizes and shapes. The function then shows an image for each Sensitivity setting, and the user is asked to choose which one most accurately detected the colonies. Finally, once the optimal Sensitivity has been chosen, the function outputs the original image with each colony it found circled in red, with the total count and image name at the top of the figure. This process is demonstrated below in Figure 5.

See Appendix A for the source code for Method 1.

B. Method 2: Automatic Equal Areas Method (EAM)

The second method we developed is much more automatic with respect to the first method, though for all of the methods we have developed thus far, manual user input of minimum and maximum colony diameter is required. This is necessary since Petri dishes frequently get cross-contaminated with other species of bacteria that grow at different rates. By manually inputting the radius parameter, only the pertinent colonies will be counted, even in highly contaminated dishes. The Equal Areas Method automatically optimizes both the ObjectPolarity and Sensitivity parameters, allowing the program to run without human interference, except when it requires a colony diameter measurement at the beginning of the analysis of each image.

The Equal Areas Method uses a novel technique to determine the optimum parameters. The determination of the ObjectPolarity setting is relatively straightforward -- the program simply counts the number of colonies detected in an image at a fixed Sensitivity of 0.95 (relatively high) in both bright (with respect to the agar) and dark (with respect to the agar) modes. The optimal ObjectPolarity setting is given by whichever mode produced the higher count (typically the incorrect mode will produce very few results). If the colonies are light with respect to the agar, the program will automatically invert the colors of the image so that a bright ObjectPolarity setting can always be used.

The determination of the Sensitivity setting is much less straightforward. To achieve this, we first crop the image precisely around the boundaries of the Petri dish, setting all areas outside the dish to black. We then binarize the original image to determine the ratio of light pixels (colonies) to dark pixels (agar). The pixel-diameter
of the Petri dish remains constant as the same camera setup was used each time, therefore the relative size of the dish remains constant.

This method will run \texttt{imfindcircles} at varying sensitivities ranging from 0.85 to 0.97 in magnitude, the experimentally determined optimal range. The analysis-proper begins by drawing a black, filled circle in a new MATLAB figure, the same size as the Petri dish in the original image. For each Sensitivity setting, the detected colonies are plotted in the figure as white, filled circles using the information generated by \texttt{imfindcircles} regarding their position and radii. This figure is then saved as a high-quality PNG image (in order to preserve the circularity of the colonies). This PNG image is then re-loaded into the program, cropped precisely around the simulated edge of the dish, and binarized. From this binary image, the ratio of light pixels (colonies) to dark pixels (agar) is obtained for only the detected circles. The logic is that the ratio of light to dark in the simulated image will equal the ratio of light to dark in the original image for the optimal Sensitivity setting. The ratio in the generated image is therefore mathematically compared to the ratio of light to dark pixels in the original image and the comparison is saved with respect to the Sensitivity level. While the comparison improves, the program iterates through the remaining sensitivities. Once the program comes upon a ratio of light to dark pixels that is higher than what it was in the original image, the iteration terminates and returns the previous result as being the optimal Sensitivity.

Once the optimal Sensitivity setting has been found, the program reruns the original image using that setting and saves the image superimposed with the found colonies circled as a JPEG image. This image has the original image’s name and the number of colonies it was found to contain printed at the top of the figure. This way, a technician or microbiologist can look through all of the images at one time (this method can run through hundreds of images at once without any user input aside from the colony diameter metric using our folderOfImages auxiliary function found in Appendix E). This process is demonstrated to the right in Figure 6.

See Appendix B for the source code for Method 2.

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure6.png}
\caption{Block diagram demonstrating the Automatic Equal Areas Method.}
\end{figure}

\section*{C. Method 3: Automatic Inflection Point Method (AIP)}

A more automated method was developed in order to become more compatible with imperfect images, featuring reflections or agar impurities. From observation of highly populated plates, it was noted that when the number of colonies detected is plotted as a function of Sensitivity ranging from 0.80 to 0.99, the curve closely resembles a fourth order polynomial where the second inflection point corresponds to the optimal Sensitivity setting that will return the most accurate colony count. This led to the development of this third colony detection method: the Automatic Inflection Point Method. This method works similarly to the Method 1, but determination of the optimal Sensitivity parameter is automated, instead of being based off of user input. The script inputs an image of a bacterially populated agar plate, and asks the user to measure and enter the approximate minimum and maximum colony diameter with a pixel-scale ruler that is overlaid on the image. After this, the ObjectPolarity setting is determined by counting the number of colonies detected in the image at a fixed Sensitivity of 0.95 (relatively high) in both bright with respect to the agar and dark with respect to the agar modes. The optimal ObjectPolarity setting is given by whichever mode produces the higher count (typically the incorrect mode will produce very few results).
Then, the `imfindcircles` function is used with parameters of the measured radii, the determined `ObjectPolarity`, and `Sensitivity` values from 0.80 to 0.99. The number of colonies detected for each `Sensitivity` setting is counted, and then plotted as a function of `Sensitivity`. This method then fits a fourth order curve to the data (still assuming the data originated from a highly populated dish), from which the inflection point can be solved for. A final run through on the image will then use the initially measured radii values, the automatically determined `ObjectPolarity`, along with the inflection point-determined `Sensitivity` value to determine the bacterial colony count. Unfortunately, because this method requires a great deal of outputs and internal calculations, run time for a single image can take up to 30 minutes, which is very slow. This process is illustrated below in Figure 7.

See Appendix C for the source code for Method 3.

![Figure 7: Block diagram demonstrating the Automatic Inflection Point Method](image)

**D. Method 4: The Experimental Fused EAIP Algorithm**

Methods 2 and 3 (Automatic Equal Areas Method and Automatic Inflection Point Method, respectively) each works for a different subset of plate types and colony densities. Thus, a Fused Equal Areas-Inflection Point Algorithm would be ideal to count the number of colonies present on any given agar dish, regardless of the colony density, presence of reflections, or agar impurities. This algorithm runs by determining the density of the plate (if the colony count is above or below a certain threshold -- currently we are using $\frac{2}{3}$ as the cutoff), and then implement either Method 2 or Method 3, accordingly. This algorithm has not been completed due to time constraints, but we believe it could be a good solution to the colony counting problem. The process for Method 4 is illustrated in Figure 8 to the right.

The current source code for Model 4 can be found in Appendix D.

![Figure 8: Block diagram demonstrating the Automatic Fused Algorithm](image)

**E. Testing Procedure**

Initial testing of our program started with images (found online) of sparsely populated agar plates where the colonies could be counted easily and reliably by hand. As we worked the bugs out of our code, we graduated to denser, more heavily colonized plates to run through the code for testing. This allowed us to not only determine the counter’s accuracy, but also to find what additions could be made to make the code more robust and allow it to optimally run through different bacterial colony patterns and colors. After making these initial edits to our code to accommodate the small sample size variation in images found online, we were ready to progress to acquiring images of our own and running them through our algorithms. This way we could test the entire process from image acquisition through to colony counting while employing filtering intermediately.

An apparatus was built to facilitate image acquisition, shown below in Figure 9. It is constructed from mixed materials, due to the limited quantity of scraps available in the Wash U machine shops. There are three parts: a top camera mount made from a delrin plastic, a steel back stand, and an aluminum rectangular base. Each side of the top camera mount fits one of the two camera models used in this project (Point Grey Blackfly and Edmund Optics Monochrome); in other words, the mount is reversible to allow for interfacing with either camera model. The back stand is three feet long, to accommodate the maximum focal length of the two cameras, and mated snugly and securely into a rectangular indentation in the top camera mount, as well as a similar indentation in the base, with RC1 close sliding fits. The base is just large enough to fit a standard agar plate. It was originally planned to create a circular indentation in the base to place the plate to be imaged; however, due to the less than ideal materials and tools available, it was decided that this indentation would decrease the weight of the base and jeopardize the stability of the apparatus.
Before we were able to begin testing on our program, bacteria needed to be grown in plates. The plating and incubation was done for us by Brad Clay, our Industry Mentor at bioMerieux. Once the bacteria had incubated for the required 24 hour period, the plates were removed from the incubator, and then photographed using our image acquisition apparatus.

The first set of images was taken using experimental dilutions of the bacteria strains, which ended up producing extremely heavily populated dishes. This showed us that additional dilutions were necessary for future imaging. However, images of these plates were still taken and used to run through initial stages of our algorithms to reach a more advanced and reliable stage.

Then, a second set of images was acquired using further dilutions, this time employing varying backgrounds and light intensities. The backgrounds used were: the natural aluminum background of the apparatus, a white background, and a black background. The room available to us for imaging the agar plates had direct overhead lighting, which caused reflections off the agar and off the colonies themselves. Images were taken of the plates with this direct lighting, as well as with a shield above the plate to shade it from the direct light, in hopes of minimizing any reflections. Images were acquired featuring combinations of all backgrounds and light intensities, to create a comprehensive set of 140 images. These images were then used to run through our algorithms for each method, all while making appropriate modifications to compensate for additional needs that were not originally anticipated.

III. RESULTS

The same original image, shown below in Figure 10 was run using Method 1 and Method 2, and the results from each method are shown below in their respective subsections, to demonstrate each method.

A. Method 1: Semi-Automatic User-Input Based Method
B. Method 2: Automatic Equal Areas Method

Figure 12: Original “after” image from Method 2 showing the conversion of light to dark pixels (binarization, masking, and cropping). Source: Own work.

Figure 13: Original “after” image from Method 2 showing the overlay of white circles everywhere that a colony was detected, and black everywhere else within the detected plate. The circles shown are drawn using the MatLab rectangle function. Source: Own work.

C. Method 3: Automatic Inflection Point Method

Figure 13 below shows the curve fitting results of the Automatic Inflection Point (AIP) Method when run on a densely populated plate. The colony count vs. Sensitivity plot is well-fit with a fourth order polynomial curve, where the x-value of the second inflection point is the optimal Sensitivity value.

Figure 14: A fourth order polynomial curve is a good fit for data originating from a densely populated plate (near the area of interest at the second inflection point, located in this example at Sensitivity 0.93).

The AIP Method is less well adapted to more sparsely populated plates. Figure 14 below shows the curve fitting results of the AIP Method when run on a sparsely populated plate. It is obvious that the fourth-order polynomial does not fit the original data, because the shape of the original data resembles an exponential growth curve, which is monotonically concave up. Since there is no inflection point, the Automatic Inflection Point Method is no longer applicable.

Figure 15: A polynomial curve is not an acceptable fit for data originating from a sparsely populated plate, as the data in these cases are always monotonically concave up.

D. Overall Accuracy

Using lightly colored agar and relatively darker bacteria, our program is running about 94% accurate based on our hand-counting estimations. Our program fails to count only 5 to 7% of the colonies. On the other hand, our counter identifies a few false-positive results amounting to around 1 to 2% of the total count. In order to optimize our
counter, it would be beneficial for us to work to increase the Sensitivity so as to increase the number of false-positives and decrease the number of misses. When these two are balanced we will achieve very accurate counts.

Figure 16 (above): “Before” image: heavily populated plate to be run through our program. Image contains *Staphylococcus aureus* colonies (Gram-positive) on a Lysogeny broth (LB) agar plate [1].

Figure 17 (above): “After” image: sample run of our program on a densely populated, internet acquired image. It returned a result of approximately 1100 individual colonies. The program failed to detect about 70 colonies (~6%) and detected about 35 false-positives (~3%) for an overall accuracy of around 97%. Image Source: Own work.

Figure 18 (above): “Before” image: bacteria we grew ourselves at bioMerieux. Image contains *Klebsiella Pneumoniae* colonies (Gram-negative) in a 10,000:1 dilution on a Tryptic Soy agar (TSA) plate. Image Source: Own work.

Figure 19: “After” image: Sample run of our program, which returned a result of approximately 200 individual colonies with zero false-positives. However, it did fail to detect about 30 to 50 colonies (~13 to 20%) due to the poor quality of our initial image. We suspect our image was too blurry and pixelated for our program to detect the smaller colonies. Image Source: Own work.
IV. DISCUSSION

A. Method 1: Semi-Automatic Use-Input Based Method

The Semi-Automatic Use-Input Based Method is a good alternative to purely-manual counting techniques. It is reasonably fast, typically taking less than two minutes to analyze an image. Although this is slower than it would take an experienced counter to do by hand, the errors associated with fatigue and loss of concentration are entirely eliminated. As a result, this method can produce more accurate and speedier results in the long-run compared with manual counting. Additionally, where manually counting microbiologists are forced to make a crude estimation for dishes with counts higher than 250, this method can accurately count up to several thousand colonies.

A major drawback to this method is that it currently takes longer to run than an experienced microbiologist would take to count the dish by hand, and more specifically, that no amount of program optimization will permit it to run significantly faster than it already does. Another drawback to this method is that it requires a technician to constantly monitor the program’s progress and give it the required inputs along the way, which only partially satisfies our goal of freeing up microbiologists’ valuable time (in that this program can be operated by less-overqualified individuals).

B. Method 2: Automatic Equal Areas Method

In its current state, the Automatic Equal Areas Method is not likely a viable alternative to traditional manual counting techniques (or, indeed, the Semi-Automatic Method 1). The main factor behind this is the very slow speed at which the program runs -- it can take up to 30 minutes to process a single image. A microbiologist could manually tally 15 to 30 dishes in that time, albeit with a higher probability of error (especially on dishes with counts greater than 250). Although there is a reduction in error, it is not high enough to justify the huge deficit in speed. In this instance, much of the speed problem might be solved via careful code optimization, though even after that we suspect it would still be slower than Method 1.

Another major drawback of this method is that it is only able to work on images with no reflection in them, which requires a more complicated imaging set-up, ideally using a polarizing filter over the camera lens and a filter perpendicularly-polarized with respect to the camera’s filter affixed to the lighting. Clearly, another alternative method must be devised if an automatic solution is to be realized.

C. Method 3: Automatic Inflection Point Method

The Automatic Inflection Point Method was shown to only be useful for densely populated plates, which are the ones that take the Equal Areas Method the longest to run. The data gathered from approximately 40 images of densely populated plates showed that, when plotted, the number of colonies detected as a function of Sensitivity value could be well-fit to a fourth order polynomial curve, which gives information as to the optimal sensitivity value -- located at the second inflection point. As shown, a drawback to this method is that sparsely populated plates do not follow the same curve pattern; these plates instead resemble an exponential growth curve, which is always concave up for the region of interest. This method can be quite fast when called on appropriate images, typically taking under 3 minutes per image.

D. Method 4: Experimental Fused EAIP Method

The Fused EAIP Method is essentially an experimental combination of Methods 2 and 3 (Automatic Equal Areas Method and Automatic Inflection Point Method) that we have not completely figured out yet. This method would be ideal to count the number of colonies present on any given agar dish, regardless of the colony density, presence of reflections, or agar impurities. This algorithm would run by determining the approximate count of bacterial colonies present on the plate, to ultimately determine if the colony count is above or below a certain threshold. This would then employ either Method 2 or 3, accordingly, depending on whether the count is above or below this to-be-determined threshold. If we can work out the Equal Areas Method to get the computation time to be under 5 minutes per dish, we think this could be a viable replacement for manual counting.

E. Conclusions

Three versions of bacterial colony counters were successfully created: one being semi-automated and two fully automated after radius estimation. This was possible through the implementation of the Hough Transform within MATLAB’s Image Processing Toolbox. A fourth experimental method was created using area estimation, but it is still in the experimental phase.
F. Future Direction

Next steps to improve this project are:

1. Determine how to automate the initial radius estimation, which can take 30 seconds or longer but will make process more automatic (asking the user if the dish is contaminated, and if so, what the contaminant radius is),
2. Optimize the code used in the Equal Areas Method to make it viable,
3. Completely interface the camera with the MATLAB programs to allow for immediate analysis of an image upon capture,
4. Complete Experimental Fused EAIP Method.
5. Offer a Raspberry Pi (or similar)-based turn-key implementation.

These steps would allow for complete automation of this project, as well as optimal bacterial colony analysis.

REFERENCES


