ECE 44x - Group 3
Automated Microbial Analysis

Next Steps Document

Jorian Bruslind

Mack Hall

Zach Bendt

May 22, 2020

Contents

Abstract

This document was created to provide information and insight into the final status of the Automated Microbial Analysis Capstone Project (ECE 441 - 443: Team 3). The overall goal of the project was to create an automated system to analyze, sort, and log petrifilm microbial data for Analysis Laboratory. To that end, the team decided to create a delta robot mechanism with a Raspberry Pi 4/Raspberry Pi Camera V2. In the end, the project was left unfinished due to the coronavirus outbreak.

1 Introduction

The Automated Microbial Analysis project aims to develop a system which is able to analyze a series of microbial samples on a special media called PetriFilm [**?**] automatically. This is a 3M product that is used in a variety of scientific capacities, mainly in food science and the medical field.

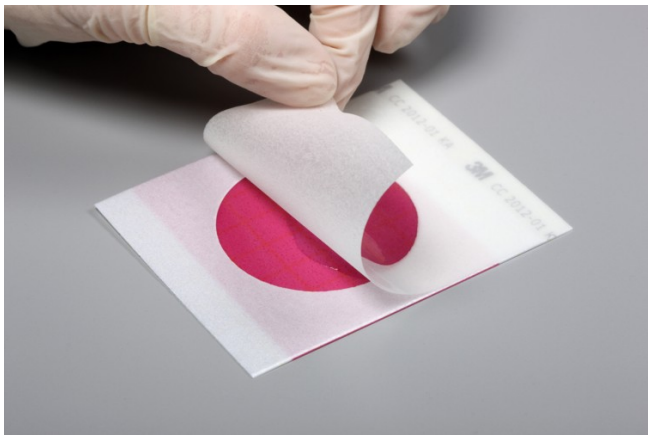

Figure 1: 3M PetriFilm Single Sample

PetriFilm is a very thin paper-like substance which has 2 layers: on one there is an agar media film that is used to grow bacterium, On the other there is a wax coating that is used to seal the sample from outside interference. A liquid sample can be applied to this agar film and left to incubate over a period of 1 - 2 days. After this time, the bacteria that were present within the sample will have had enough time to grow into colonies which are visible to the human eye. These colonies can be counted and the original number of bacterium per unit volume can be known from the original sample. The process to count the bacteria samples and categorize them into datasets is often tedious and can take a human operator around 1 - 2 minutes per sample [**?**]. With sample sizes of 50 - 60, this process can easily take over an hour. The goal of our project is to use an automated system to count these bacteria colonies in sample sets of upwards of 100+ in as little as 20 minutes. This data will be automatically stored and tabulated within the client's original database.

For this process it is necessary to create a physical manipulation system that will be able to move paper-like objects in an efficient and precise manner. These samples are somewhat delicate and require careful handling. In addition, they must be placed in a specific orientation so that the bacterial samples can be easily read by some visual component. Therefore it is necessary to create a mechanical system that can be easily made precise and is able to be used for our intended implementation. In addition, there will be designated areas for loading/unloading of samples so that the overall process requires as little human intervention as possible.

Once the loaded, the system will take over and begin the counting process for each sample. There will be error correction and state information for each portion of the counting cycle to ensure the

samples are not damaged and to reduce the chance of a miscount. After the sample has been counted and cataloged, it will be placed into one of two sections designated "Pass" or "Needs Review" where they can be sorted by a human operator.

2 Next Steps/Improvements

2.1 Mechanical

The mechanical system is almost complete and only requires a few components to be fully functional. There are plenty of aspects that could be improved however.

1. **Custom Model Material Upgrades**

   The mechanical system uses a few custom models that were designed for simplicity/easy integration. These parts are currently made using deposition PLA material made from a 3D printer. One of the goals of this section was to use a professional protoyping firm (such as Shapeways) to create these custom models with more rigid, precise materials. PLA is a great material for prototyping and providing a proof of concept model for use in the system, but there are many other materials available that can be made lighter, stronger and more precisely than what a 3D printer and PLA can accomplish.

   Recommendation

   Our recommendation would be to look into contracting with an external company and creating the models out of Nylon 11 or RPU70 plastics. Nylon 11 would be the preferred material due to its superior tensile strength to PLA (PA11) and its cheap cost compared to RU70 [1]. RU70 is the best material for this task however as it has the highest tensile strength (1900Mpa RU70 vs 1600Mpa PA11) and highest shore hardness (80 D RU70 vs 75 D PA11) [2] (RU70)

2. **Better Ball Joint Ends**

   One of the most important components of the delta robot mechanism is the rotating joints at the end of the upper and lower arms. These joints allow for full rotation in one direction and partial (60 deg) rotation in the other directions. This motion allows the end effectors motion to be controlled by the 3 arms while maintaining a parallel configuration to the work surface. The original ball joints purchased were repurposed RC car ball joint and are of low quality. Towards the end of the project we were able to replicate this 3 dimensional rotation using 3D printed parts, but ulitmately the motion is still stiff and unreliable.

   Recommendation

   Our recommendation for this issue would be to look into purchasing official ball joint ends from a certified supplier such as McMaster Carr or any other fastener vendor. This would allow for more information about the available motion of the component while also giving much better precision/accuracy in the parts provided.

3. **Small Pneumatic System**

   The pneumatic system wasn't fully implemented due to the issues encountered in spring term. However our initial implementation shows that the pneumatic tubing and suction cup/adapter assembly would be too large for the end effector to properly maintain. Our preliminary testing showed that the end effector was over-burdened and would slightly bend/bow in certain directions due to the weight of the tubing.

   Recommendation

Our recommendation would be to find a smaller tube set that still provides the necessary -1.4 bar vacuum while also being small enough to not impact the end effector motion. Another idea would be to add a spring damped system to the end effector and the top center of the robot so the end effector is naturally drawn to the center of the robot (and stays parallel to the workspace).

4. **Motor Mount Improvement**

   The current motor mounting system is sufficient for the application but could be made better by creating a component that encloses more of the motor surface area. The main problem is the motors are exposed and the wires have no mounting management. This is necessary so the wires are moved out of the way of the main mechanism and the overall system looks clean.

   Recommendation

   Our recommendation would be to modify the existing structure of the motor mounts to enclose the entirety of the motors. These models can also be modified to add wire mounting points so the control/feedback wiring can be structured and kept out of the way.

2.2 Electrical

The main difficulties of the electrical subsystem have already been completed. What remains are fairly simple, but nonetheless important, tasks that were not able to be completed due to the global health crisis. The majority of the time spent on the electrical subsystem was spent on custom PCB design and getting the motors to move according to a coordinate format.

1. **Relaying True Motor Position**

   Each Auxiliary Stepper Driver PCB contains an ATtiny85, which reads electrical impulses from an integrated encoder on each stepper motor. Each pulse represents another step taken by the motor, so the true motor position can be accounted for through counting the pulses. The ATtiny85s are currently set to send the position individually, but no information accompanies it to notate which motor is sending the information.

   Recommendation

   In order to solve this issue, the ATtiny85 must be re-flashed with updated code. The current code can be found here, on the project GitHub, which should be modified to send another unique character per motor, so the information can have a unique identifier. Then, the Auxiliary ATmega328 should read and parse the information, and pass it along to the Raspberry Pi via serial communication. If the relayed true motor position is different from the expected position, then the Raspberry Pi should trigger a homing sequence in order to realign the expected and actual motor positions.

2. **Accounting for the Size of the Incoming Sample Stack**

   The user can place an arbitrary number of samples to be processed into our system. Additionally, the incoming sample stack will have a varying height throughout the processing procedure, as the incoming stack will shrink with each passing sample being processed. As such, it is important that the height of the end effector is not too high nor too low when attempting to pick up the next sample from the incoming stack.

   Recommendation

   Adjusting the height of the end effector will have to take place on the Raspberry Pi, as that is what sends the coordinates for the motor arms to execute. The user already inputs the number of samples to be processed into the GUI at the beginning of every session, so we need to find the height which the incoming stack decreases for one sample. Once we find that, we will know how

far down the end effector has to travel in order to be able to pick up the first sample and each subsequent sample thereafter. An easy way to implement this would be to have a set coordinate that places the end effector directly over the incoming stack. After traveling to that coordinate, the end effector will be lowered straight down by a varying amount down onto the stack. Due to the delta configuration, moving the head directly up and down is very easy to implement, as all three motors will move by the same amount. The amount that the head will have to move vertically will be incremented by the height of a single processed sample, after each sample is processed. In theory, this will allow the end effector to always have the appropriate height when taking a sample off the incoming stack. Such code needs to be written and implemented into the coordinate execution routine on the Raspberry Pi.

3. **The Homing Sequence**

   The motors are equipped with incremental encoders, which means that they only relay if the motor moves forwards or backwards, not motor position; once power is lost in the system, so is the motor position. We can know if the motor is moving, but that information is not very valuable if we do not know where the motor arm currently is. The homing sequence remedies this by moving the motors all the way in a single direction until they cannot move any further. Since we know how far each motor can extend, we know where is motor arm is now located. From there, we are able to calculate the motor arm position no matter where it is moved, since we know the initial position and all subsequent movements.

   Recommendation

   The homing sequenced needs to be implemented. Luckily, the software used to relay motor movement, grbl, has a dedicated command for homing sequences. Once the limit switches are hooked up to their respective Auxiliary Stepper Driver PCB, grbl will take care of the rest. However, the software found on the Main ATmega328, which runs grbl, will have to be modified. Variable $22 holds a boolean value for the homing sequence enable. Currently it is set to 0, but should be set to 1 for grbl to do it automatically on startup. To do this, simply run the code in the Arduino IDE and type "$22=1" into the serial monitor. This will save the variables value even after a power cycle.

4. **Minor PCB Improvements**

   As this project was completed on a strict budget of both time and money, additional revisions of the PCB could not be ordered once small issues were discovered.

   Recommendation

   The PCBs, if re-manufactured, need to have a few adjustments. On the Auxiliary Stepper Driver PCBs, the output pins for the DB9 encoder connector need to be adjusted. The trace between the ATtiny85 physical pin 2 and the pin for EA- or Pin 4 on the DB9 connector should be removed. Then ATtiny85 physical pin 2 should be connected to the pin for EB+ or Pin 2 on the DB9 connector. This will allow for the encoder to pass on the motor position to the ATtiny85 correctly. On the Central Processing PCB, the 5VDC line for the 5V headers should be modified. Pin 2 of P9 or P10 respectively need to be connected to the 5V supply. Unfortunately, an oversight during designing the PCB lead to these pins being connected to the wrong net, so they are not receiving any power without a jumper currently.

2.3 Computer Science

1. **Computer Vision Improvements**

   The computer vision system needs to be improved to increase the accuracy of the counts. This is a critical part of the project and is directly related to engineering requirements. The requirement

stated that The system should correctly identify and count the microbial colonies to within a 10% error of a human but ideally, the system should to be as accurate at counting microbial samples as a human eye.

Recommendation The computer vision should be honed using the five parameters that are set in the colonyCounter.py file. The five parameters are intensity threshold, area, circularity, convexity and inertia. These parameters should be set once the testing environment is stable. This is done by mounting LED's around the camera to make the amount of light on the sample when the image is captured the same for every image. Tuning these parameters before the lighting is stable could lead to imprecise results.

2. **Automation Controls Integration**

   The automation controls are calculated by the RaspberryPi which then sends the control messages via serial interface to the ATmega328 where they are decoded and used to set the motors to the proper angles. These control messages are integrated into the user interface software along with the computer vision software. The functions for the controls have yet to be written.

   Recommendation The automation controls functions have already been stubbed out in the robocontrols.py file, which can be found on the GitHub. They currently just print to the console for debugging purposes. They just need to send the proper (x,y,z) coordinates to the ATmega328's. The encoders on the motors can be used to find the proper coordinates to get the right sequence of motor positions. There may also be some timing issues to work out depending on how long each step of the process will take. This can be easily taken care of by implementing wait() function calls as needed and using trial and error to find the optimum sequence.

3. **PefriFilm Size detection**

   This prototype was made specifically for Analysis Laboratories and There are multiple sizes of petrifilm that can be used to perform the food sample testing. To make the bot more marketable to other companies the bot should be able to adjust the height at which the image is captured. This would require the computer vision system to be able to detect the size of the petrifilm. This is because the camera is mounted on the bottom of the base and the position and focus are fixed.

   Recommendation

   In order to implement size detection, the range of motion of the end effector could be translated into a size range of the petrifilm based on the field of vision of the camera. There would also need to be edge detection to find the size, in pixels of the circular testing area on the petrifilm which could then be translated into millimeters based on how far away the end effector is from the camera when the image is captured.

4. **Gas detection**

   During one of the initial informational meetings with our client there was a discussion about adding gas detection to the computer vision system. Gas is produced by some of the microbes that grow on the petrifilm and create bubbles under the material that is used to seal the petrifilm. The challenge in adding gas detection to the computer vision analysis would be integrating it with the microbial colony detection while maintaining the accuracy of the basic detection.

   Recommendation

   To add gas detection would require edge detection to be integrated into the computer vision to detect the bubble around a microbial colony. In addition, there would need to be a separate sorting stack for the samples that are found to have gas bubbles. Any addition to the computer vision portion of the project might also require additional processing power. This could be implemented using a more specialized PCB used in place of the RaspberryPi.

References

[1] S. Inc., "3D Printing PA11 - Shapeways," May 2020, [Online; accessed 20. May 2020]. [Online]. Available: https://www.shapeways.com/materials/pa11

[2] "Shapeways + Carbon," May 2020, [Online; accessed 20. May 2020]. [Online]. Available: https://www.shapeways.com/partnership/carbon#page-block-o067l2oj9l