Geochemistry, Geophysics, Geosystems®

RESEARCH ARTICLE
10.1029/2022GC010689

Key Points:

- The Forabot is a low-cost and open-source system for automated isolation and imaging of foraminifera
- We explore Forabot performance as a physical manipulation tool
- Forabot images are classified using deep learning with promising results for future integration

Supporting Information:

Supporting Information may be found in the online version of this article.

Correspondence to:

T. Richmond,
trichmo@ncsu.edu

Citation:

Richmond, T., Cole, J., Dangler, G., Daniele, M., Marchitto, T., & Lobaton, E. (2022). Forabot: Automated planktic foraminifera isolation and imaging. *Geochemistry, Geophysics, Geosystems*, *23*, e2022GC010689. https://doi.org/10.1029/2022GC010689

Received 14 SEP 2022
Accepted 23 NOV 2022

Forabot: Automated Planktic Foraminifera Isolation and Imaging

Turner Richmond[1], Jeremy Cole[1], Gabriella Dangler[1], Michael Daniele[1], Thomas Marchitto[2,3], and Edgar Lobaton[1]

[1]Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, [2]Institute of Arctic and Alpine Research, University of Colorado, Boulder, CO, USA, [3]Department of Geological Sciences, University of Colorado, Boulder, CO, USA

Abstract Physical inspection and sorting of foraminifera is a necessity in many research labs, as foraminifera serve as paleoenvironmental and chronostratigraphic indicators. In order to gain counts of species from samples, analyze chemical compositions, or extract morphological properties of foraminifera, research labs require human time and effort handling and sorting these microscopic fossils. The presented work describes Forabot, an open-source system which can physically manipulate individual foraminifera for imaging and isolation with minimal human interaction. The major components to build a Forabot are outlined in this work, with supplementary information available which allows for other researchers to build a Forabot with low-cost, off-the-shelf components. From a washed and sieved sample of hundreds of foraminifera, the Forabot is shown to be capable of isolating and imaging individual forams. The timing of the Forabot's current pipeline allows for the processing of up to 27 foram specimens per hour, a rate that can be improved for future classification purposes by reducing image quality and/or quantity. Along with the physical descriptions, the image processing and classification pipelines are also reviewed. A proof-of-concept classifier utilizes a finetuned VGG-16 network to achieve a classification accuracy of 79% on a validation set of foraminifera images collected with Forabot. In conclusion, the system is able to be built by researchers for a low cost, effectively manipulate foraminifera with few mistakes, provide quality images for future research, and classify the species of imaged forams.

Plain Language Summary Foraminifera or "forams" are abundant microscopic organisms found in the ocean, and their shells are a common component of seafloor mud. Mud cores can be used to understand ancient ocean conditions, and the types and chemistry of forams in a sample are useful environmental indicators. However, separating different types of forams is slow and tedious work which requires considerable expertise. We have designed and built a robot called Forabot, which picks up individual shells, takes high-quality photographs of them, and moves them to a bin for sorting. We describe the system so that other researchers can build their own Forabot at low cost. The current version of Forabot is optimized for high-quality imaging and is therefore relatively slow, but if it is instead used for classifying and sorting shells into different types, it can be optimized for speed. We discuss the preliminary performance of a classifier based on artificial intelligence, with overall accuracy of 79%. In conclusion, our robot can be built by researchers for a low cost, effectively manipulate forams with few mistakes, provide quality images for future research, and accurately classify the type of foram.

1. Introduction

Foraminifera, or forams for short, are ubiquitous in the world ocean (Sengupta, 1999). Along with their abundance, their biodiversity and extent of geologic record make their fossils of particular interest to paleontologists and paleoclimatologists (Schmiedl, 2019). The sand-sized fraction of deep sea sediments is often dominated by planktic foraminifera, of which there are about 50 extant species (Schiebel & Hemleben, 2017). Although modern benthic foraminiferal species number in the thousands, they are typically only abundant in shallow (shelf) environments and in poorly preserved abyssal sediments. For this work, we focus on deep-sea sediment and as such do not include analysis or commentary on benthic foraminifera. Due to the small size and great abundance of planktic foraminifera, hundreds or possibly thousands can often be picked from a single cubic centimeter of ocean floor mud. Foraminifera samples are normally sorted by species before they are used for either academic or

can achieve micron-level precision. Where some require manual intervention for imaging (Campbell et al., 2014; Sharkey et al., 2016), open-source projects leave room for customization for automated imaging or come with it prepackaged (Steiner & Rooney, 2021). These are all great solutions for general microscopy, but one large drawback when dealing with foraminifera is the need to manually manipulate and orient all the samples, which is time consuming by itself. For this reason, our work aims at addressing some of the specific challenges associated with isolating, manipulating, and imaging such small specimens as foraminifera.

The specific challenge of imaging foraminifera has been noted by other groups (de Garidel-Thoron et al., 2020; Elder et al., 2018) who have come up with their own solutions to address imaging and classifying foraminifera. A large data set collected by Elder et al. (2018) contained approximately 61,000 complete or damaged foraminifera samples. The images are high quality and many of the foraminifera have been labeled (Hsiang et al., 2019) with a species based on the images. Though the data is valuable for researchers, the means to repeat that level of data collection in other labs may not be as accessible. Researchers looking to expand the Elder et al. (2018) data collection by increasing the number of certain species represented, adding new species to the data set, or increasing the representations of foram orientations would require a high-end microscope, stage, and control software which may be prohibitively expensive. In addition, the foraminifera needed to be manually prepared and oriented on a slide before imaging. Again, the human-hours required to perform data collection is something that could be improved.

For automated foraminifera imaging and sorting, MiSo (de Garidel-Thoron et al., 2020) promises to be a system which could be valuable. Though the information around the system is sparse, likely due to a pending patent, they indicate that approximately 8,000 particles can be imaged per day, which is an impressive feat. While we cannot speculate on the price, the patented nature of this system does not lend itself to community improvements nor modifications. Our system aims at filling the space where low-cost open-source microscopes for generic specimen imaging intersect with high end specialized systems for automated imaging and sorting of foraminifera.

2.2. Foraminifera Classification Systems

Previous works have explored the efficacy of deep learning models tasked with species classification of planktonic foraminifera. In the classification systems mentioned below, the data collections required human intervention which may have biased the orientation selected such that the viewpoint the humans found most informative was used. Despite the human interaction component in some of the following works, they have made a great step in moving toward a fully automated classification system.

Earlier works focused on small counts of foraminifera over a limited number of species due to the human effort of individually imaging each foram (MacLeod et al., 2007; Zhong et al., 2017). Early work on species classification (MacLeod et al., 2007) showed that interest existed though the computational resources and tools did not exist to support true classifiers for unseen data. Later work (Zhong et al., 2017) began using transfer learning with larger pretrained neural networks showing that the artificial intelligence tools had finally reached the required maturity. The three pretrained networks VGG-16 (Simonyan & Zisserman, 2015), Inception-V3 (Szegedy et al., 2016), and ResNet-50 (He et al., 2016) showed a performance that was comparable to human experts at classifying foraminifera. An extension of this work (Mitra et al., 2019) further compared the variability of human classification with the neural network classifier. The neural network outperformed both experts and novices when presented with images of a foraminifera for classification.

A large data collection effort and associated classification system came out of the Endless Forams project (Elder et al., 2018; Hsiang et al., 2019). More than 34,000 planktonic foram images were collected and used for transfer learning on VGG-16 (Hsiang et al., 2019). As the first large-scale collection effort and accompanied classifier, the works (Elder et al., 2018; Hsiang et al., 2019) provide a strong case for automated species classification of planktonic foraminifera. Subsequent works (Aagaard-Sørensen et al., 2020; Marchant et al., 2020) showed similar success in correctly classifying oriented foraminifera species, often around the low 90th percentile accuracy.

Though the existing works are quite successful, other works exist which aim to extract morphological characteristics such as a chamber segmentation (Ge et al., 2017, 2021). While these features haven't been used in classification yet, they do provide an interesting extension in getting automated measurements of foram species. Each of these works benefited from directional lighting when extracting morphology. The imaging process described in Section 3.2.2 is designed with the idea of supporting future research aimed at extracting morphology in mind.

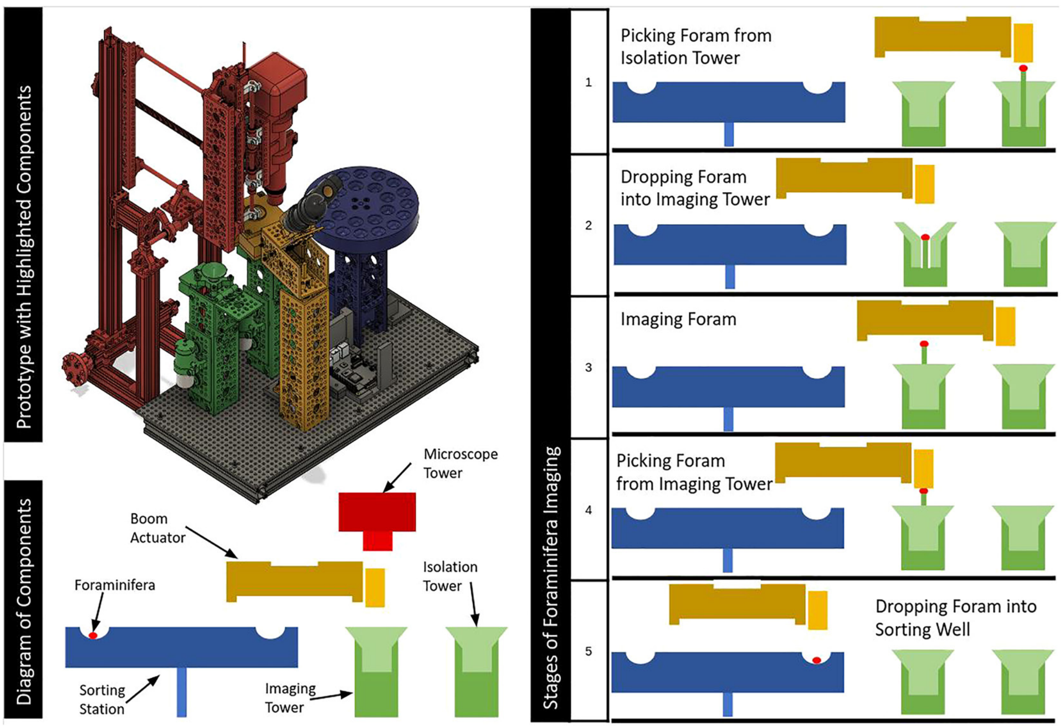
Prototype with Highlighted Components
Diagram of Components
Microscope Tower
Boom Actuator
Foraminifera
Isolation Tower
Sorting Station
Imaging Tower
Stages of Foraminifera Imaging
1
Picking Foram from Isolation Tower
2
Dropping Foram into Imaging Tower
3
Imaging Foram
4
Picking Foram from Imaging Tower
5
Dropping Foram into Sorting Well

Figure 1. To the left, a CAD (Computer-Aided-Design) model of the Forabot system hardware is shown along with an illustrative simplification of the system. To the right, the stages of imaging and sorting a single foraminifer (red circle) are shown.

3. Methods

In this section we describe the components of our system which allow for automated manipulation, imaging, and isolation of foraminifera. We split this section into two parts, hardware and software. In the hardware section, we discuss the required structural materials, motors, imaging equipment, and electronics for the system. In the software section, we discuss the software which controls Forabot system, controls the imaging conditions, and detects errors that may occur during the isolation and imaging of foraminifera. The documentation, parts lists, build steps, electronics wiring, and archived code base is made available by Richmond et al. (2022). For building a system, our experience has shown that for a first time builder two or more months may be required depending on familiarity with robotics and 3D printing. An experienced builder can fully build a Forabot in under 2 weeks. Access to the most recent updates and community questions should be accessed at the github repository (Richmond, 2022).

Let us begin by describing the major steps of foram isolation and imaging, using the system shown in Figure 1. To start, a user selects a washed and size-sieved sample of forams for analysis and deposits them into the funnel at the top of the *Isolation Tower*. The purpose of the funnel at the top of the *Isolation Tower* is to isolate a single foram for imaging and classification. A single foram is picked in the *Isolation Tower* where a *Boom Actuator* picks the foram from above using a vacuum pump and moves it to a second funnel. The second funnel is where the foram is imaged, and its associated tower is thus called the *Imaging Tower*. Once the foram is imaged and classified, the *Boom Actuator* picks the foram from the *Imaging Tower* and deposits it into a well in the *Sorting Station*. On a foraminifer's way to being sorted, it will therefore pass through the *Isolation Tower*, *Imaging Tower*, *Boom Actuator*, and *Sorting Station*. Now that the individual steps of the sorting process are defined, we explore the components in more depth and describe some secondary systems which are vital in running the system.

3.1. Hardware

Now we describe each of the four major components which can be seen with labeled parts in Figure 2. The *Imaging Tower* and *Isolation Tower* contain the same components and operate in the same manner. The purpose of

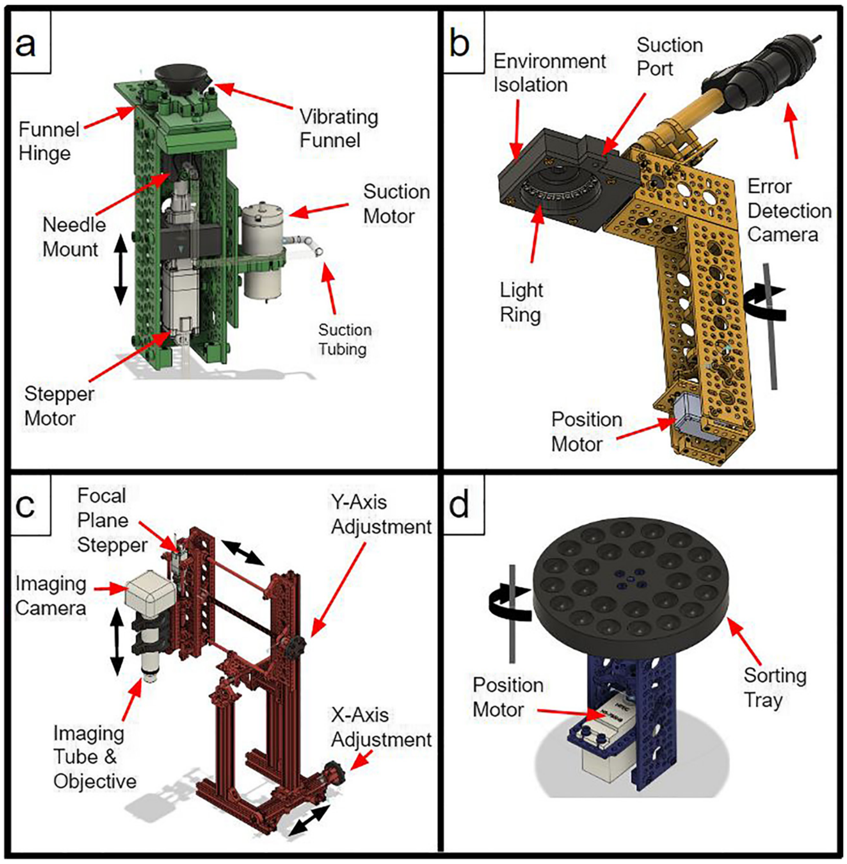

Figure 2. Showing the major components of the Forabot: *Imaging and Isolation Towers* (green), *Microscope Tower* (red), *Boom Actuator* (yellow), and *Sorting Station* (blue). Each component is more clearly labeled (a–d) where black parts indicate custom 3D printed parts, pale parts indicate purchased specialty items and electronics, and black arrows indicate direction of motion for the part.

these towers is to provide a means for reliably moving a single foram for processing. In the case of the *Isolation Tower* that processing is isolating and lifting a single foram so that it can be passed to the *Actuator Boom*. For the *Imaging Tower*, that processing includes both moving the foram to a precise distance from the microscope objective for imaging, as well as passing the foram to the *Actuator Boom* after imaging.

A tower holds a single funnel mounted to a hinged piece which can be opened to provide easy access to the needle and stepper motor within the tower. The funnel houses a coin vibration motor on the outside to encourage a foram to fall into the base of the funnel. At the base of the funnel, a hole is present just large enough for a luer lock dispensing needle to pass through. The dispensing needle is used to hold a single foram on its tip and can be readily changed depending on the size of forams being sorted. In Section 4, we provide an analysis of how well two different needle gauges are able to pick and isolate forams depending on the foraminifera size fraction. The needle is connected to both a stepper motor and a vacuum pump via a single 3D printed part. To pick a single foram, the vacuum pump is used to pull in air through the end of the needle. The stepper motor is connected in such a way that it can raise or lower the needle. The needle tip can be moved to the base of the funnel for picking a foram, to a raised position for passing a foram to the *Actuator Boom*, or at a user-specified height for imaging.

The *Actuator Boom* is the most complex part due to its multiple use cases related to physical manipulation, imaging, lighting, and error detection. It is the central component of the system which rotates on a limited angle servo. The boom arm has three important features: a suction port, a low-resolution error detection microscope, and an isolation cover with integrated LED ring.

The suction port consists of a vacuum pump connected to a downward facing circular hole with a nylon mesh filter used to prevent forams from being sucked into the motor. When transferring a foram between funnels or to a

Table 1
Array of Motor Types and Motor Controllers Used

Motor	Count[a]	Controller	Count[b]	Voltage
Servo	2	Micro Maestro 6-Channel USB Servo Controller	1	5 V
Stepper	3	Tic T825 USB Multi-Interface Stepper Motor Controller	3	4 V
DC Vibration	7	TB6612FNG Dual Motor Driver Carrier	2	5 V
Vacuum pump	3	Adafruit DC/Stepper Motor Bonnet for Raspberry Pi	1	12 V

[a]Indicates the total number of motors on the system. [b]Indicates the total number of boards on the system.

well, the suction port is positioned over the foram and the foram is pulled against the nylon mesh by the airflow. Simply shutting off the vacuum pump is often enough to release a foram, but a vibration motor is affixed to the port as a secondary precaution.

While the suction port and Luer lock needles are effective solutions for picking and maneuvering the forams, we integrated an error checking module which reduces the likelihood of failing to transfer a foram from one location to the next. The low-resolution error detection microscope is affixed to the *Actuator Boom* such that it is able to image expected foram locations. In Section 3.2.3 we describe all the error detection states which utilize the images coming from the error detection microscope.

The last component of the *Actuator Boom* is the isolation cover with integrated LED ring. In order to image a foram in a controlled environment, the isolation cover is moved into place above the foram. The isolation cover has a hole at the top which aligns with a microscope objective to prevent as much ambient light as possible from reaching the foram during imaging. By isolating the foram from environmental light, the LED ring is able to fully control the lighting conditions.

Next we describe the *Sorting Station* which has the least complexity. The purpose of the *Sorting Station* is to provide isolated wells for forams to be deposited into once they have been imaged and processed. The *Sorting Station* contains a servo which spins a 3D printed sorting tray. The sorting tray has individual wells; each of which can be associated with a particular species once a classifier is fully integrated. The *Sorting Station* rotates to a well so that the *Actuator Boom* is aligned and can deposit the foram sample. The well selected during data collection is the next well over in the counter-clockwise direction, but future works to integrate a classifier could return the well index of a species for sorting.

The final component of the system is the *Microscope Tower* for imaging the forams. The microscope is a single tube system with a 4X objective and a c-mount 18MP camera. A frame, which is attached to the base structure of the system, holds the microscope tube. Fine adjustment knobs have been integrated to allow for a simple mechanism when manually aligning the microscope over the imaging needle. Due to the limited depth of the objective, the foram is imaged in a z-stack with the microscope moving in 60 μm increments between focal planes. As such, the microscope is attached to a stepper motor which controls the focal plane of each individual image. We empirically determined the targeted distance between focal planes, but due to mechanical reliability the distance was kept conservative to ensure there is at least some small overlap between the 90 μm depth of field images at each focal plane.

The Forabot uses four motor types: stepper motors, servo motors, DC vibration motors, and DC vacuum pumps. The electrical components for driving these motors are off-the-shelf driver board solutions with industry supported operation. Table 1 shows all the boards which are used for controlling the individual motors. The boards are all controlled by a single low-cost, credit card sized computer common in the education and makers communities. The control computers supported are the Raspberry Pi 3b+ (2017) and Raspberry Pi 4. All electronic components are powered through a single "laptop" power supply which can take an alternating current from 90 to 264 V as input and produce a single 12 V DC output. A small electronics breadboard must be soldered to properly distribute the 12 V power to each of the electrical components of the Forabot.

Table 2
Runtimes for the Five Stages of the Forabot

Stage number	1	2	3	4	5
Stage description	Isolation pick	Transfer to imaging	Imaging	Imaging pick	Sort
Runtime (s)	5	13	97	6.5	10

3.2. Software

In this section, we discuss the software used for controlling the system. The Forabot is run as a client server architecture where a Raspberry Pi acts as the server and a personal computer (PC) is the client. Tested versions of the Rasbperry Pi include 3b+ (2017) and 4 running Raspberry Pi OS Buster while the PC has been tested to run on both Windows 10 and Ubuntu 18.04. We begin by describing the controls system running on the Raspberry Pi, which controls all hardware and electronics other than the camera sensors. Afterward, we detail the image acquisition and processing which run on the PC. Lastly, the PC-run error detection steps are detailed including the motivation, procedure, and potential resulting actions for the Raspberry Pi to resolve errors.

3.2.1. Controls System

The Raspberry Pi is setup as a server such that when it is powered on, it homes the hardware system so that all components have accurate and known position, are in their starting locations, and are ready to receive forams for processing. Once the Raspberry Pi homes the hardware, it sits in an idle state waiting for a connection from a client. In order for the PC to connect to the system, the PC must be connected via USB ports to the error detection microscope, the high resolution microscope camera, and a serial-to-USB cable to the Raspberry Pi. When the client code is run on the PC, the Raspberry Pi and PC establish a connection using serial communication. Serial communication was selected as opposed to a network protocol as it is expected many of the computers used will be university owned and potentially have network protocols blocked.

Once the connection between client and server is established, the server (Raspberry Pi) controls all manipulation of the foram steps outlined in Section 3.1. The server and client have a synchronized relationship so that only one of the Raspberry Pi or PC is in control and the other is waiting for commands. When images need to be taken, the server requests the client to take an image and waits for the client to acknowledge that the image is taken. When a foram is imaged, the server controls the focal plane and lighting direction of the Forabot while the client controls the image capture. The server must not move the focal plane or change the lighting direction until the image is fully captured or imaging artifacts such as blurring could occur.

In addition to communication about imaging synchronization, the Forabot server also requests classification information from the client for both error detection and species identification. In the case of error detection, the client can issue commands which change the state of the server. After taking an image either through the microscope or error detection camera, the client either responds by letting the server know it has completed imaging and everything looks okay, or it will respond with an error state outlined in Section 3.2.3. Lastly, since the client contains all images, it performs classification of the imaged foram when requested by the server.

The processing states of the Forabot controls system are depicted at the right in Figure 1. Table 2 depicts the run times of each of these stages. The first stage consists of isolating a single foram and performing an error check. Next, transfer to imaging consists of moving the foram from the isolation needle to the imaging funnel and performing another error check. The imaging step consists of picking the foram, moving the microscope into place, and the full imaging of the foram from a single orientation. After imaging, the foram is picked from the imaging funnel and a species classification is requested. In the last step, the foram is moved from the imaging funnel to the appropriate sort well and a final error check is performed. The total time required is about 2.2 min.

As of now, these timings are conservative. The stepper motors driving the needles move slowly so they are less likely to skip steps and are more precise. Due to the small foram size, having a reliable needle location is vital to take good images. Similarly, the servo motor driving the *Actuator Boom* is speed limited to allow for observing the system more closely during data collection. The third stage of the Forabot pipeline consists of taking 30 images which will be described in Section 3.2.2. A short delay between each of the 30 images is taken as a precaution to prevent exposure from previous focal planes or lighting directions. While the runtime is possible to improve by tweaking the system's motor speeds, the largest gain would be to reduce the image resolution and/or number of images, as the image acquisition and write time have the largest impact. The current version of Forabot is optimized for the collection of high quality images, whereas classification could ultimately be done with smaller/fewer images and hence greater speed.

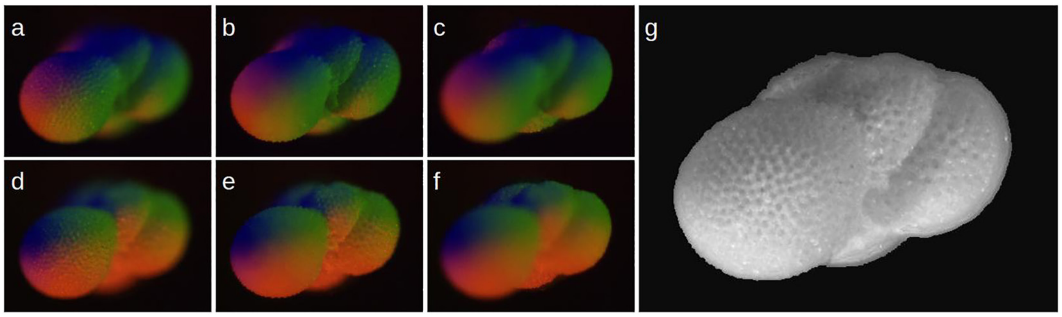

Figure 3. Example RGB images under the first (a–c) and second (d–f) lighting conditions. Representative focal planes are shown for the top of the foram (a, d), the middle of foram (b, e), and the bottom of foram (c, f). The corresponding extended depth of field image (g) is generated as a grayscale image.

3.2.2. Imaging

The server-client relationship is required for one major reason—offloading image processing to a more powerful computer (the client). The two imaging sources are the error detection microscope and the high-resolution microscope. The error detection microscope will be described in Section 3.2.3. For now, we focus on the high-resolution microscope and resulting images.

The specific high-resolution camera used in the presented system utilizes a proprietary driver from ToupTek which is only supported on x86 architectures (PC) and not ARM (Raspberry Pi). Due to the lack of driver support, large image sizes, and support for cloud data backup the client PC was integrated. Since the Forabot is a data-collection system in the current stage, the highest quality images are of importance for research. If the Forabot develops into use as a classification system, running all components on a single Raspbery Pi is quite doable with a change in imaging sensor which would likely have lower resolution and thus faster imaging throughput.

Though the imaging setup is highly configurable, the parameters are set for work presented here. The camera is set to have an exposure time of 200 ms, an analog gain of 200%, and a resolution of 4,912 × 3,684 pixels. In total each foram is imaged from 15 focal planes where the first focal plane has the end of the imaging needle in focus. As previously described, the focal planes are offset by 60 μm resulting in a total z-axis coverage of 900 μm. At each focal plane, the foram is imaged from two different lighting conditions. A representative selection of focal planes under the lighting conditions described here can be seen in Figure 3. Each lighting condition contains a single LED pixel of red, blue, and green for a total of three lighting directions. The three colors on the LED ring are evenly spaced with each color getting captured by a single channel on the RGB imaging sensor. The second lighting condition is similar to the first, except the lights are offset by 60°. By controlling the lighting in this manner, we are able to extract 6 grayscale images of the foram where each grayscale image has a light source from a unique direction. The further processing of these images to generate a single EDF image is detailed in Section 5.

The *Imaging Funnel* exists separately from the *Isolation Funnel* to allow for the Forabot to take images of a single foraminifer from multiple orientations. By isolating a foram in the *Imaging Funnel, the foram can be picked on the needle for imaging and then a vibration motor can encourage the foram to fall. Afterward the needle lowers to pick the foram again at a random orientation.* The ability to image a single foram from multiple orientations is helpful from a data collection viewpoint as it allows for a human to use all orientations for labeling the foram's species. Once a classifier is integrated, the Forabot is not able to choose the orientation for imaging. If a classifier does not have high confidence when classifying an image, the foram can be imaged again from a different viewpoint which may have more identifying characteristics.

3.2.3. Error Detection

Manipulation of individual foram tests is an important step in both imaging and species sorting. Manipulating objects which are measured on the micrometer scale is a challenge which is naturally prone to failure. To boost the performance of the proposed imaging and sorting system, we integrate error detection steps utilizing the OpenCV library (Bradski, 2000) to ensure the state of the system is as expected. Below we outline three error check procedures, namely Error Checks I, II, and III, along with their motivation and resolution procedures in the case of a detected error.

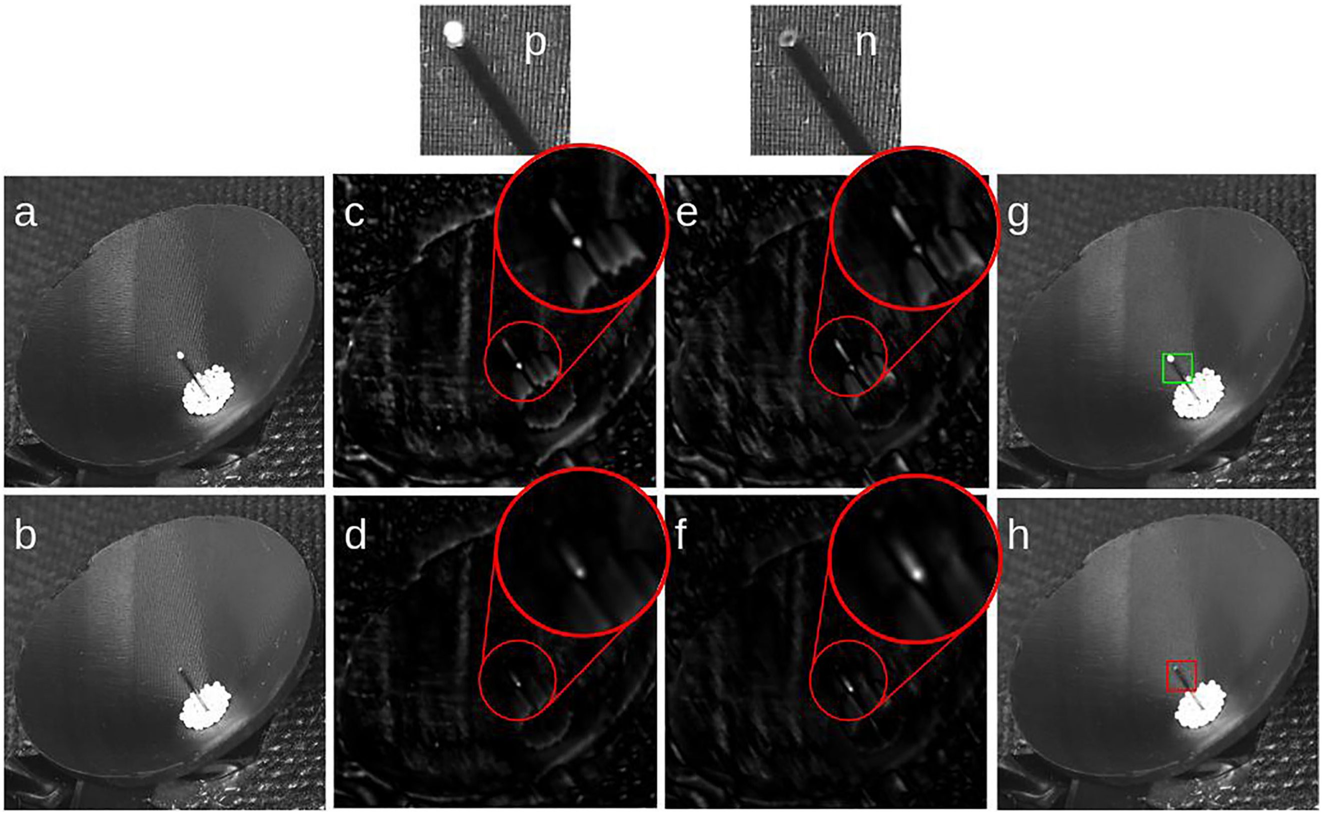

Figure 4. The template matching algorithm for picked foram detection must identify a needle as either having a foram at the end or not. The template matching is run using the positive (p) and negative (n) templates at the top. The input images (a, b) are run against the templates to produce response maps associated with the positive template (c, d) and the negative template (e, f). The detected highest response locations are shown (g, h) with a green box if the highest response is associated with the positive template and a red box if the highest response came from the negative template response map.

Error Check I occurs when isolating a single foram in the *Isolation Tower* from the deposited group of forams as shown in Figure 4. As a first step in sorting forams, the Forabot isolates a single foram for imaging. Before the foram is passed to the *Actuator Boom*, the error detection microscope takes an image of the raised needle. The image is analyzed to detect the end of the needle and check if there is a foram on the end. If a foram is picked, it is passed to the *Actuator Boom*. Otherwise, the needle lowers and repeats the process of attempting to isolate a single foram. By checking a successful foram isolation at this step, processing time is sped up since the Forabot will not go through the processing steps of imaging nothing.

For Error Check I to determine whether a foram is on the tip of the needle in the *Isolation Tower*, a pattern matching test is used. Representative images of the needle end with and without a foram are used as templates to search for the end of the needle in the image. The template images are convolved with an isolation attempt image. Whichever template image convolution has a higher response is how the image is classified. Figure 4 shows both the template images as well as the maximal response locations from the convolution. A shortcoming of the approach used is that the template images do not consider or detect the isolation of multiple forams. Thus the Forabot does not have any conditions in place to determine whether a single foram or multiple forams were isolated for transfer or imaging. See Section 4.3 for the effectiveness of the error detection as well as the prevalence of multiple foram isolations.

After Error Check I, the Forabot proceeds to correct the error in the case one is detected or continues with processing the isolated foram. If no foram is detected, the Forabot simply repeats the isolation process. If a foram is detected, the foram is passed to the *Actuator Boom*. When the *Actuator Boom* handles a foram, the foram is not visible from either of the cameras on the system. As such, there is some uncertainty when handling undesired states that occur when moving a foram from one location to the next. To determine if any unexpected states have

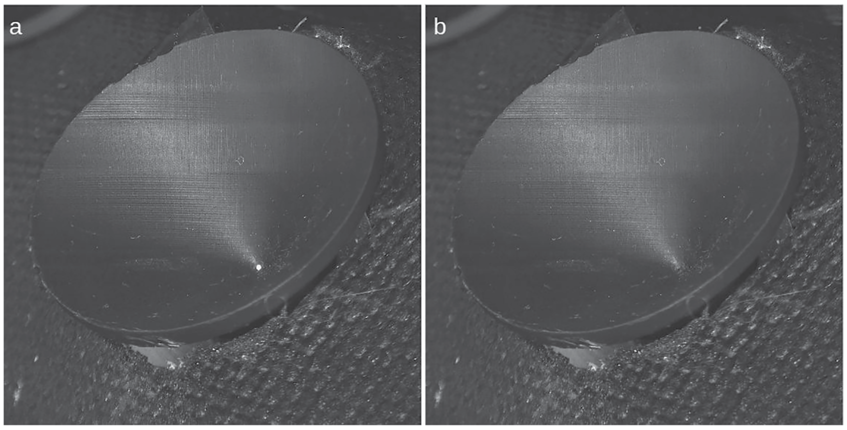

Figure 5. Foram detection in a funnel based on blob detection of a large grouping bright pixels on a black funnel. The funnels with a foram (a) and without a foram (b) are shown where the presence of a bright grouping of pixels at the center of the funnel is a foram.

occurred when moving the foram, Error Check II determines whether there is a foram present in the *Imaging Tower*.

Error Check II is a simple blob detector which finds groups of bright pixels with a large enough area. Since the funnels are printed using a black resin, detecting a white object on a black background is quite simple and extremely accurate. Figure 5 shows the results of the detection when a foram is present and when one is absent. Though both of the failure detection algorithms are quite basic, they work well for the system in its current state. With a larger user base, more environments, and more data to consider, the error detection could become more sophisticated.

Error Check II is run at two different steps during the Forabot processing. The first time it is run is right after the attempt to transfer an isolated foram to the *Imaging Tower*. If a foram is present in the funnel, then the system continues with imaging. If no foram is present, the foram was not passed to the *Actuator Boom* or the foram is stuck in the *Actuator Boom*. The *Actuator Boom* attempts to drop the foram again using a vibration motor to dislodge any stuck forams. The Forabot will repeat the drop attempt as long as Error Check II fails, up to a predefined number of attempts. If the predefined number of failures occur, it is assumed there is no foram in the *Actuator Boom* and the system will go back to isolating a foram.

The second instance in which Error Check II runs is after the Forabot attempts to transfer a foram to the *Sorting Station*. In this case, Error Check II passes when no foram is in the *Imaging Tower*. A foram may be left in the *Imaging Tower* if multiple forams were picked or if the handoff to the *Actuator Boom* failed. In either case, the *Imaging Tower* must be cleared so the sorting process will repeat until the foram is transferred successfully.

Lastly, Error Check III is performed using the images from the camera on the *Microscope Tower*. At the start of a session before any foram isolating, imaging, and sorting takes place, the Forabot images the needle in the *Imaging Tower* with the imaging camera at each focal plane and lighting direction. The initial needle imaging provides information useful for aligning data collections as well as providing a control image for what it looks like when no foram is present. The histogram for each control image is saved for detecting if a foram is present during imaging. Each image taken when attempting to image a foram has its histogram compared to the control for the corresponding focal plane and lighting direction using a chi-square comparison of the histograms (Figure 6). If the correlation is below a predefined threshold, the needle lowers to pick the foram again for imaging. The Forabot overwrites any images which were taken before an Error Check III failure occurs, so all focal plane and lighting direction images must have a foram present for the set of images to be kept.

The error checks are not only in place to prevent single foram manipulation or imaging failures, they are also used as a mechanism for indicating to a user that the system may need to be cleaned if too many errors occur. There are a few states which the system attempts to resolve on its own, but some cases may require user intervention. The two most common states that may need human intervention are a foram getting caught in the nylon mesh of the suction port and a foram sticking to the side of the imaging funnel due to static charge. In the case of a foram

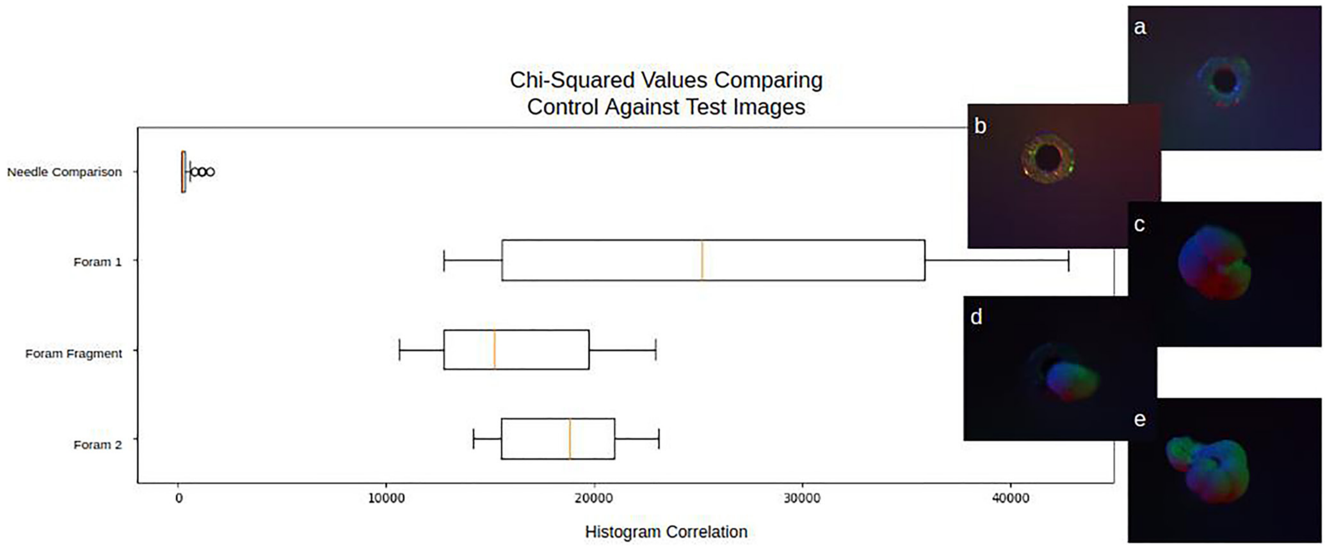

Figure 6. Box plots of the Chi-Squared values when comparing a control image against different samples that may be seen during imaging. Example images from the first focal plane are shown for the control set (a), a set of images without a foram present (b), a set of images with a foram (c), a set of images with a foram fragment (d), and a set of images with a different foram (e). The box plots are generated using the 30 Chi-Squared value generated; one for each focal plane and lighting direction of the set of images.

sticking to the nylon mesh, it not only fails to drop but also prevents other forams from being picked. Simply running a paintbrush around the mesh will dislodge the foram. In some cases, a staticky foram sticking to the imaging funnel is not able to be knocked loose by the vibration motor. Simply removing the foram with a paintbrush resolves the error. Each of these instances should be checked if the Forabot stops running before expected due to the inability to resolve failures in the system.

4. Mechanical Evaluation

The main objective of this section is to quantify the Forabot's ability to manipulate forams and describe how failures impact throughput. The foraminifera sample used in the manipulation analysis is presented first. We begin by analyzing the physical movement of a single foram through the stages of imaging and sorting. To do this, we present a simple test which is run to show how often the Forabot fails to manipulate a foram specimen as intended. Next, the failure error detection algorithms are described along with their impact on improving foraminifera throughput as compared to a system without checks. In Section 3.2.3, we explained how the Forabot aims to resolve unintended situations. We use the resolution procedures along with the prevalence of failure to find an expected runtime for a single foram. Lastly, we conclude by presenting the efficacy of the Error Checks presented in Section 3.2.3.

4.1. Description of Sample

The tests described here are performed on washed (>63 μm) and size-sieved foraminifera from the 1–2 cm interval of multicore MGL1208-38MC, which was retrieved from 2,859 m water depth at 6°49.6’N and 161°2.5’W in the tropical central Pacific Ocean. These sediments are rich in calcium carbonate and preservation of planktonic foraminifera is good.

4.2. Physical Manipulation

In order to properly analyze the physical manipulation efficacy of the Forabot, we describe the foram sample and preparation done for the data presented in this section. Foraminifera are sieved into >600, 500–600, 425–500, and 355–425 μm size fractions. Exactly 100 forams are then randomly chosen from each size fraction. Due to the size of the foraminifera, we have found that they can be noticeably influenced by electrostatic forces. As a result of the electrostatic forces, the forams may stick together which causes failures in isolating a single foram, they may stick

Table 3
Failure Rates of Common Foram Size Fractions Tested Using Two Needle Gauges, With Stages as Given in Table 2

Needle gauge	Size fraction	Stage 1 error	Stage 1->2 error	Stage 4->5 error	Expected additional runtime (seconds)	Multiple forams picked
27	**355–425**	0.08	0.07	0.05	2.5	0.07
	425–500	0.16	0.07	0.02	2.7	0.08
	500–600	0.12	0.04	0.03	2.1	0.01
	600+	0.35	0.02	0	3.4	0.02
25	500–600	0.04	0.05	0.05	1.8	0.05
	600+	0.05	0.04	0.03	1.5	0.04

Note. The Forabot is able to recover from Stage Errors, but unable to detect and recover from treating an isolation of multiple foram specimen as a single foram specimen (far right column). Average expected additional run times are based on the failure rates and the time to correct a given error. Size fractions are bolded under the recommended needle gauge.

to the boom actuator which results in failure when moving a foram, and they may stick to the edge of a funnel which results in failures when trying to pick a foram in the imaging funnel. To reduce the impact of these complications, each group of foraminifera is wetted using a paintbrush dipped in water and allowed to fully dry. Lastly, the foraminifera from a given size fraction are all placed in the isolation funnel for processing.

We present a size fraction analysis of the four different groups of foraminifera. Two different needle gauges are tested for picking the forams, 25 and 27 gauge. In order to have a comparable test across changing needle gauges, the same 100 forams are used for each size fraction. The lower bound on the size fractions tested for each needle was selected based on the ability to isolate a single foram from a group. When foram size is too close to the inner diameter of a needle, finding a suction strength which is able to hold a single foram without also picking up multiple other forams proves difficult. As such, the performance presented can be used as a guide when selecting a needle gauge for imaging a sample of foraminifera.

The analysis includes running through an entire imaging pipeline with 100 forams initially deposited in the isolation funnel. As forams are imaged, the isolation funnel depletes until it is empty when all 100 forams have been imaged. If multiple forams are ever moved from the isolation to imaging funnel, then all but one foram are manually returned to the isolation funnel after the system images and sorts the forams. As the sorting tray would become more cluttered with forams, the tray is cleared for every 10 forams deposited to ensure all forams are tracked and accounted for at each stage of the process. Clearing the sorting tray is only done to ensure proper tracking of the performance and is not necessary in a normal operation. No other manual intervention is done to the system during the size fraction analysis to represent a normal use over 100 forams.

The stages of processing a foram and the probabilities of failure in moving on from each stage are shown in Table 3. The most important takeaway we want to emphasize is that the needle gauge has a substantial impact on the performance of Forabot. The 27 gauge needle performs significantly better on forams under 500 μm, while the 25 gauge needle performs better on forams over 600 μm. As noted above, the 25 gauge needle performance on the smaller foram size fractions aren't presented as there was no suction strength which was able to find a reasonable balance between isolating a foram and not always picking multiple forams. By selecting the proper gauge needle when sorting forams, a user can expect to see forams successfully moved at any single step more than 90% of the time with some variation into the mid-eighties depending on size. The probability of successfully moving a single foram through the full pipeline with no issue ranges from 77% to 88%.

While success rates provide insight, we show the impact of failing to move a foram as expected on the overall run time for sorting a single foram in Section 4.3. One previously mentioned condition which we cannot handle through error detection is the isolation of more than a single foram. In these cases, the Forabot images a group of forams that is isolated as a single specimen and sorts them all into the same well. The last column of Table 3 shows the rate at which more than a single foram was isolated over the 100 imaged forams. As the foram size becomes larger relative to the size of the needle, they tend to block the opening of the needle and prevent the suction from picking multiple forams during isolation. Wider ranges of foram sizes (e.g., everything greater than 355 μm) would likely result in poorer performance because of degraded optimization of the needle size.

Finally, there is a possibility of breaking specimens during manipulation. During setup the needle height is adjusted to prevent the foram from forcibly contacting the nylon mesh on the suction port. Once that is set, breakage is infrequent, but we have not quantified it. In Section 5.2 we present data collected with the Forabot which imaged over 300 specimen from three orientations where no forams were lost or completely destroyed. Approximately 5 were shown to have some breakage, though the samples were not checked prior to imaging and thus we cannot comment on the impact of the Forabot. Very poorly preserved specimens, of the sort that break when touched with a picker's paintbrush, would likely be damaged by Forabot.

4.3. Failure Detection

The failure detection checkpoints presented in Section 3.2.3 are able to make running the system with the presented failure rates manageable. Resolving an isolation failure (Stage 1) takes 9 s. A transfer to imaging failure (Stage 2) adds 14 s, because three repeated three-second drop attempts over the imaging funnel ensure the foram isn't stuck to the top suction, and repeating an isolation pick takes another 5 s. To correct a foram falling off during imaging or not being successfully picked for imaging (Stage 3), the imaging needle needs to attempt to pick the foram again and the microscope may need to slightly adjust focal planes which takes approximately 14 s total. It should be noted that although rare, the foram could fall off toward the end of imaging which means that all the time that it took to image the failed orientation is lost. We don't take into account the foram falling off during imaging since it hasn't been observed in an undisturbed Forabot. Though for a Forabot in an environment where the system or its foundation may be bumped into or otherwise disturbed, the Error Check III is vital for preventing the Forabot from imaging a needle with no foram. In an earlier iteration of the Forabot, Error Check III true positives were more prevalent as the needle moved to change focal planes with respect to a fixed microscope; it was not uncommon for forams to fall when moving to a new focal plane. Lastly, the foram could fail to be transferred to the sorting well (Stages 4 and 5). The Forabot will simply repeat the sorting procedure which takes approximately 15 s. Using these error correction times and the corresponding failure rates, we get the expected time increase per foram considering the foram size fraction and needle gauge which is presented in Table 3.

The performance of error detection itself does have an impact on runtime, though it is not included in the expected additional runtime per foram presented in Table 3. Each error check algorithm was run using a data set with 100 positive and 100 negative samples. Both Error Check II and Error Check III were able to correctly classify all 200 images as containing a foram or not. Though there is a possibility of failure in these error checks, the chance of a misclassification is low enough that they do not have a negative impact on runtime. Due to the slightly more complex classification problem for Error Check I, the classifier was able to get all of the failed (negative) isolation images correct and 92 of the successes (positives) correct. It is preferred to have Error Check I with a low false positive rate since the system will have to go through more steps to process a foram when nothing is picked. The associated increase in false negatives is not as impactful since it only consists of performing the isolation step again. In the worst case of the presented isolation successes in Table 3, a false negative rate of 8% increases the expected additional runtime of a foram by 0.7 s.

5. Image Processing and Classification Evaluation

Given that the system is able to move a foram through each stage in its imaging and sorting pipeline with a high degree of reliability, we are next interested in getting a classification of the foram's species so it can be accurately separated from other species. As discussed in Section 2, prior works have shown that the classification of foraminifera species at an accuracy comparable to that of human experts is possible (Hsiang et al., 2019; Marchant et al., 2020; Mitra et al., 2019). For this reason, we do not focus on improving the classification in this paper, but only aim to show that the presented system is capable of capturing images which can be successfully used in existing classification pipelines. The rest of this section will focus on the data collection efforts, the image processing steps, the challenges associated with classification in our imaging pipeline, and lastly the performance evaluation of transfer learning using our data to refine a classifier trained using the data set from Hsiang et al. (2019).

5.1. Image Processing

For each orientation of a specimen, the 30 images from 15 focal planes and 2 lighting conditions are fused into a single grayscale EDF image. First, the two lighting condition images at each focal plane are fused into a single grayscale image by converting each lighting condition into a single grayscale image, and then averaging the pixel intensity between the two grayscale images. The result is a 15 focal plane grayscale image set which is then stack aligned to correct for any shift in the foram between focal plane images. A rigid-body translation alignment provided by a python library for stack registration (Thévenaz et al., 1998) is used. The Sobel operator (Tinku & Ajoy, 2005) is applied to each image as a metric to determine which focal planes are most in focus. The metric has the high responses at specularities, which are bright spots where light is more intensely reflected to the camera. Specularity metric values decrease but remain large relative to the rest of the image as they are moved out of

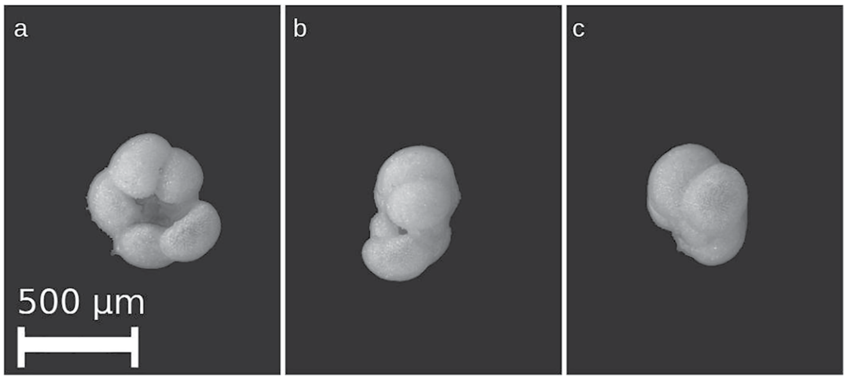

Figure 7. View point variability shown from three random samples of a single *Neogloboquadrina dutertrei* specimen during data collection. The umbilical view (a) and two side views (b, c) are shown.

focus, but that same blurring increases their area in an image. To prevent the blurred area of a specularity from being confused with in-focus features of the foram, an in-focus focal plane is determined for each specularity of a foram. The in-focus focal plane for each specularity is considered the root of a specularity growing algorithm. The algorithm takes a mask of the in-focus focal plane specularity and compares that mask with a specularity detection of the next focal plane. All pixels in the focal plane of comparison mask which are reachable from the in-focus specularity mask (i.e., that form a connected component) are considered a blurred specularity. This comparison is done for adjacent focal planes starting with the in-focus plane and moving to subsequently more blurred focal planes. At the end, each focal plane has a mask which is used to ignore blurred specularities as these locations falsely appear to be in focus due to their high metric response. The specularity masks are combined with the Sobel responses to generate the final EDF image by taking the focal plane with maximal response at each pixel where blurred specularities are ignored via the specularity mask.

5.2. Species Classification

As just mentioned, the initial data set used to train a base classifier for foraminifera classification is obtained from Hsiang et al. (2019). A similar network structure is used, but since our images use directional lighting we cannot source accurate colors from the test of the foram. Due to the difference in data collection methods, we modify the network presented by Hsiang et al. (2019) to take in grayscale images which don't have directional lighting. After training the initial grayscale classification network, we perform fine tuning of the network using our data set so that we can evaluate the performance on images coming from our imaging domain.

The data set collected from our system is comprised of 6 species which were all present in the (Hsiang et al., 2019) data set with at least 500 specimen present in each species. The species used (with associated counts) were *Globigerinella siphonifera* (51), *Globigerinoides sacculifer* (53), *Globorotalia menardii* (56), *Neogloboquadrina dutertrei* (59), *Globigerinoides conglobatus* (43), and *Pulleniatina obliquiloculata* (54). These species were selected as they were available in high counts at size fractions greater than 355 µm. Each specimen in the collected data set is imaged from three random orientations which provides more orientations for training at the cost of increasing the total time of imaging (Stage 3) from 97 to 291 s Figure 7 provides an example of a specimen that was imaged during data collection which provides one umbilical view and two different lateral views.

The extended depth of field images are used as a proof of concept input when training a simple foram species classification network. The classification network largely follows (Hsiang et al., 2019) with some modifications due to image modality differences. Both the data collected for fine-tuning in Hsiang et al. (2019) as well as data collected with the presented system are used in training. Because the automorph (Hsiang et al., 2019) EDF data is in RGB format and the EDF data as part of this research is in grayscale, the automorph data is transformed to grayscale and back to RGB by repeating the intensity into all channels. The data collected as part of this research is first cropped so that the foram occupies approximately 80% of the image, then the intensity channel is repeated in to the RGB channels (Figure 8). A VGG-16 (Simonyan & Zisserman, 2015) network was fine-tuned using Tensorflow (Abadi et al., 2015) with the model weights ported from the original model trained using Caffe (Jia et al., 2014). The preprocessing used when training the VGG-16 model was used in fine-tuning on the foram data

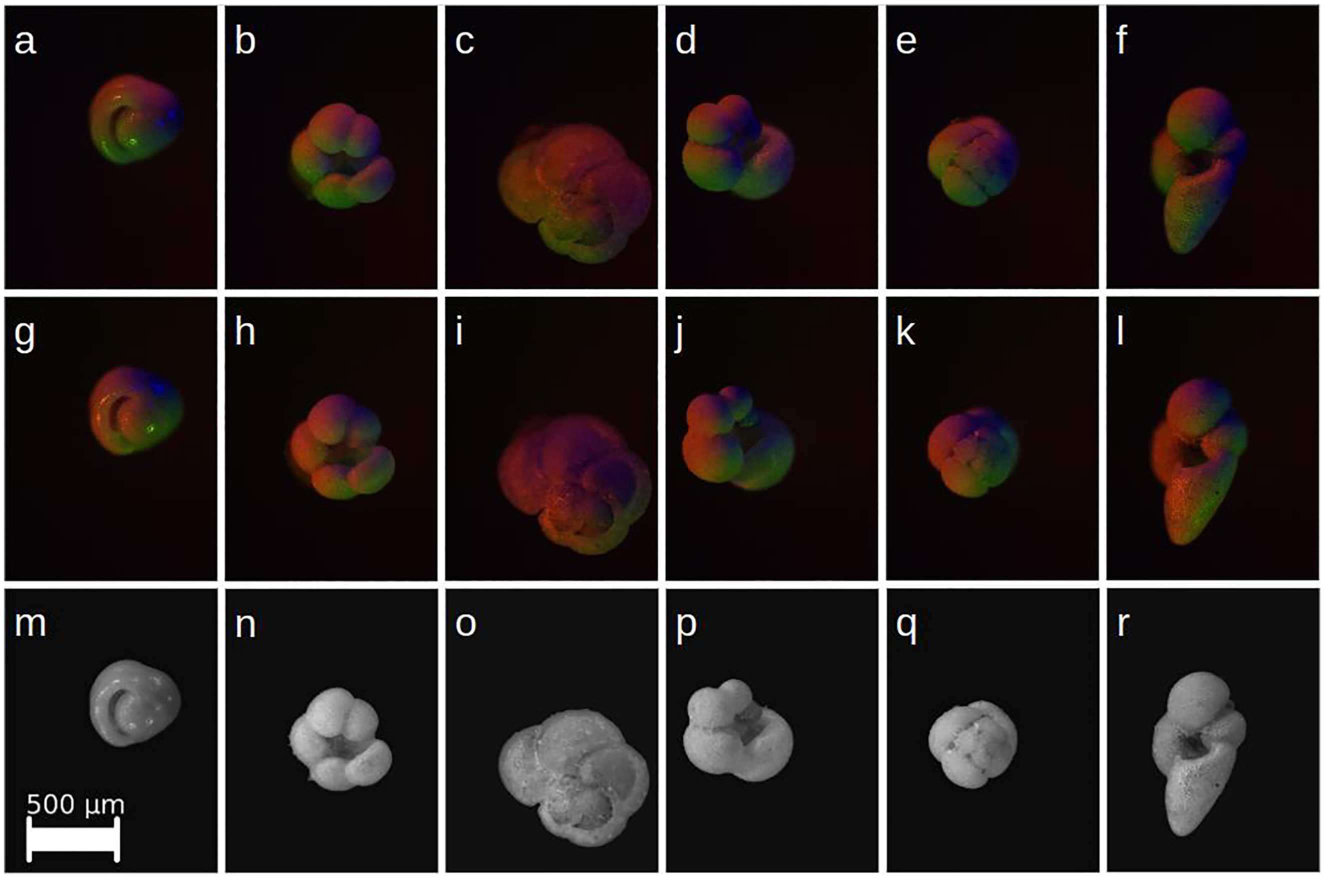

Figure 8. For species (left to right) *Pulleniatina obliquiloculata*, *Neogloboquadrina dutertrei*, *Globorotalia menardii*, *Globigerinella siphonifera*, *Globigerinoides conglobatus*, and *Globigerinoides sacculifer*, the first (a–f) and second (g–l) lighting directions are shown for a selected focal plane. The reconstructed extended depth of field images (m–r) are also shown for each species.

set. Preprocessing is comprised of rearranging the channels of the RGB image to BGR (which has no affect in our data) and subtracting the mean BGR value of the original training set from each pixel in an image.

The final fully connected layers of the original VGG model are replaced for fine tuning with two fully connected layers consisting of 512 and 36 nodes to generate logits for the 36 classes of foraminifera species in the (Hsiang et al., 2019) data set. The layer with 512 nodes uses a dropout of 50% with a ReLU activation function. The final logit connection uses a softmax activation function. Only weights representing connections with the two added layers are set to trainable, while the rest of the VGG-16 network weights are fixed during training.

The network was trained with a learning rate of 0.0001 following (Hsiang et al., 2019) over 200 epochs. Training data consisted of both data sets shuffled for each epoch, and the validation set consisted of only Forabot images. Figure 9 shows a confusion matrix of the validation data after completing training. Note only 6 species are present in the validation set. The classifier was able to discern that each of the validation images were from the Forabot imaging domain as it only classified the validation data as one of the six Forabot species. The full accuracy for the validation set was 78.75% which is good considering the Forabot images only contributed to less than 3% of the training set. By continuing to build the data set of images from the Forabot system over a larger group of species, we would expect to see a classification accuracy in the high 80th percentile as reported by Hsiang et al. (2019).

6. Conclusion

The work in this paper describes an imaging and manipulation system which could benefit the paleoceanography community by providing a standard system for sorting cleaned planktonic foraminifera based on species. Through low-cost and easily accessible parts, the system is able to move, image, and isolate individual forams

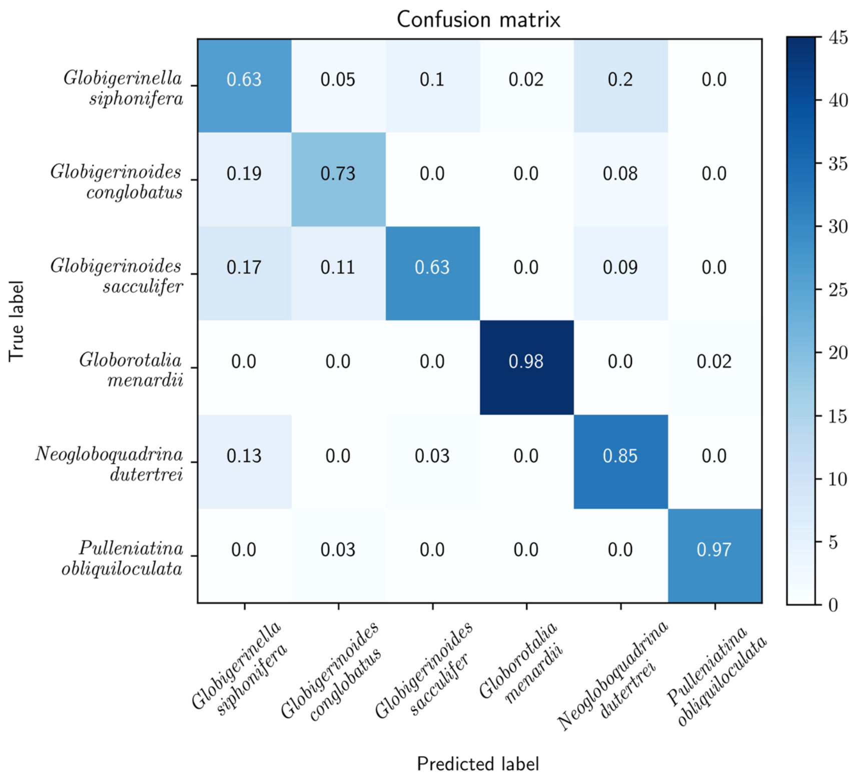

Figure 9. Confusion matrix of the validation data after 200 epochs of training.

to aid researchers in their efforts. Though the system's throughput as presented isn't particularly high with an approximate throughput of 648 samples sorted per day, slight modifications can be made to change the system from a high fidelity data acquisition system into a classification system optimized for speed. The three factors which affect throughput the most are image resolution, number of focal planes, and number of orientations. When considering the system as a classification system for automated sorting, future work will aim to optimize these factors. Decreasing the resolution and number of focal planes will directly improve the presented throughput. A classifier with the option to resample orientations would need to be optimized to consider the cost of time in its loss function to minimize resampling orientation.

As the Forabot currently stands, it is a system which can isolate and image forams for data collection. Though the system presented only is run as a data collection system, it was built with the aim of being used as a classification and sorting mechanism once a sufficient data collection has occurred for training a classifier with a larger number of species. Over time, the database of species labeled forams imaged using the Forabot will grow enough to train a classifier on enough species to deem it useful as a classification system. The future work of building the data set to enable a classifier which works well with the Forabot is the vital last step to enable it to become a classification and sorting mechanism. Once the classifier is completed, the code base and hardware were designed to allow for a seamless transition from a data collection system to a species sorting system.

Despite the limited presented throughput, the system end-user may be most interested in human effort per sample. Limiting the user interaction was a main objective when designing the system presented here. At the time of writing there exist no other publicly-available automated means for researchers to image, classify, and sort cleaned foraminifera samples. With continued use of the Forabot, future improvements based on feedback could greatly improve the usability, efficiency, and classification performance. As the system matures, it is entirely possible other groups such as the benthic foraminifera research community may test the system and find small adaptations to make an effective system for their efforts.

Data Availability Statement

An example of the raw data collected using the Forabot is available for download along with software for validating the research presented. The software for generating figures, merging raw image stacks into EDF images, error detection examples, as well as the species classification is available under the same archive. Lastly, the build documentation, parts lists, electronics wiring guide, and full CAD model is available as a reference for interested researchers to build their own Forabot (Richmond et al., 2022). All data and software is available through the joint cooperation of Dryad and Zenodo via a single DOI at https://doi.org/10.5061/dryad.6hdr7sr44 under a CC0 1.0 license for the data and GNU-GPL3 license for the software. Additionally the most up to date software for the Forabot is available at https://github.com/ARoS-NCSU/ForaBot.

Acknowledgments
This material is based upon work supported by the US National Science Foundation Grants OCE-1829930 and OCE-1829970. The authors thank those who have helped with previous versions and incremental improvements of a system over the years. Namely, we thank two teams: Eric Davis, Robert Macpherson, and Joseph Berman as well as Harrison Klas, Akshay Paruchuri, and Shannon Pinnel for their work on two earlier iterations of a foram picking system.

References

Aagaard-Sørensen, S., Haugland Johansen, T., & Junttila, J. (2020). First-order machine learning based detection and classification of foraminifera in marine sediments from arctic environments. In *EGU general assembly conference abstracts* (p. 7606).
Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Retrieved from https://www.tensorflow.org/
Bradski, G. (2000). *The OpenCV library*. Dr. Dobb's Journal of Software Tools.
Campbell, R. A. A., Eifert, R. W., & Turner, G. C. (2014). Openstage: A low-cost motorized microscope stage with sub-micron positioning accuracy. *PLoS One*, *9*(2), 1–18. https://doi.org/10.1371/journal.pone.0088977
de Garidel-Thoron, T., Marchant, R., Tetard, M., Adebayo, M., & Gally, Y. (2020). Automated recognition and picking of foraminifera using the miso (microfossil sorter) prototype. In *EGU general assembly conference abstracts* (p. 18067).
Drew, L. W. (2011). Are We Losing the Science of Taxonomy?: As need grows, numbers and training are failing to keep up. *BioScience*, *61*(12), 942–946. https://doi.org/10.1525/bio.2011.61.12.4
Elder, L. E., Hsiang, A. Y., Nelson, K., Strotz, L. C., Kahanamoku, S. S., & Hull, P. M. (2018). Sixty-one thousand recent planktonic foraminifera from the Atlantic Ocean. *Scientific Data*, *5*(1), 180109. https://doi.org/10.1038/sdata.2018.109
Ge, Q., Richmond, T., Zhong, B., Marchitto, T. M., & Lobaton, E. J. (2021). Enhancing the morphological segmentation of microscopic fossils through localized topology-aware edge detection. *Autonomous Robots*, *45*(5), 709–723. https://doi.org/10.1007/s10514-020-09950-9
Ge, Q., Zhong, B., Kanakiya, B., Mitra, R., Marchitto, T., & Lobaton, E. (2017). Coarse-to-fine foraminifera image segmentation through 3D and deep features. In *2017 IEEE symposium series on computational intelligence (SSCI)* (pp. 1–8). https://doi.org/10.1109/SSCI.2017.8280982
He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
Hsiang, A. Y., Brombacher, A., Rillo, M. C., Mleneck-Vautravers, M. J., Conn, S., Lordsmith, S., et al. (2019). Endless forams: >34,000 modern planktonic foraminiferal images for taxonomic training and automated species recognition using convolutional neural networks. *Paleoceanography and Paleoclimatology*, *34*(7), 1157–1177. https://doi.org/10.1029/2019PA003612
Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., et al. (2014). Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093.
MacLeod, N., O'Neill, M., & Walsh, S. A. (2007). A comparison between morphometric and artificial neural network approaches to the automated species recognition problem in systematics. In *Biodiversity databases* (pp. 37–62). CRC Press.
Marchant, R., Tetard, M., Pratiwi, A., Adebayo, M., & de Garidel-Thoron, T. (2020). Automated analysis of foraminifera fossil records by image classification using a convolutional neural network. *Journal of Micropalaeontology*, *39*(2), 183–202. https://doi.org/10.5194/jm-39-183-2020
Mitra, R., Marchitto, T., Ge, Q., Zhong, B., Kanakiya, B., Cook, M., et al. (2019). Automated species-level identification of planktic foraminifera using convolutional neural networks, with comparison to human performance. *Marine Micropaleontology*, *147*, 16–24. https://doi.org/10.1016/j.marmicro.2019.01.005
Richmond, T. (2022). forams-bot. Retrieved from https://github.com/trichmo/forams-bot
Richmond, T., Cole, J., Dangler, G., Daniele, M., Marichatto, T., & Lobaton, E. (2022). Forabot supporting information [Dataset, Software]. Dryad. https://doi.org/10.5061/dryad.0vm37
Schiebel, R., & Hemleben, C. (2017). Planktic foraminifers in the modern ocean. *Planktic Foraminifers in the Modern Ocean*.
Schmiedl, G. (2019). *Use of foraminifera in climate science* (Vol. 4). Oxford University Press. https://doi.org/10.1093/acrefore/9780190228620.013.735
Sengupta, B. (1999). *Modern foraminifera*. Kluwer Academic Publishers.
Sharkey, J. P., Foo, D. C., Kabla, A., Baumberg, J. J., & Bowman, R. W. (2016). A one-piece 3D printed flexure translation stage for open-source microscopy. *Review of Scientific Instruments*, *87*(2), 025104. https://doi.org/10.1063/1.4941068
Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International conference on learning representations*.
Steiner, R. A., & Rooney, T. O. (2021). Piautostage: An open-source 3D printed tool for the automatic collection of high-resolution microscope imagery. *Geochemistry, Geophysics, Geosystems*, *22*(5), e2021GC009693. https://doi.org/10.1029/2021gc009693
Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *2016 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2818–2826). IEEE Computer Society. https://doi.org/10.1109/CVPR.2016.308
Thévenaz, P., Ruttimann, U., & Unser, M. (1998). A pyramid approach to subpixel registration based on intensity. *IEEE Transactions on Image Processing*, *7*(1), 27–41. https://doi.org/10.1109/83.650848
Tinku, A., & Ajoy, K. R. (2005). *Image processing: Principles and applications*. Wiley-Interscience. Retrieved from https://proxying.lib.ncsu.edu/index.php?url=https://search-ebscohost-com.prox.lib.ncsu.edu/login.aspx?direct=true%26db=nlebk%26AN=140433%26site=ehost-live
Zhong, B., Ge, Q., Kanakiya, B., Marchitto, R. M. T., & Lobaton, E. (2017). A comparative study of image classification algorithms for foraminifera identification. In *2017 IEEE symposium series on computational intelligence (SSCI)* (pp. 1–8). https://doi.org/10.1109/SSCI.2017.8285164