WORKS COMPLETED:

For this week, I’ve worked mainly on setting up the camera. The camera came with a CD that contained all the necessary driver and software for the camera. The driver was installed to my personal laptop and the camera is connected to the laptop through USB. The camera’s own software was used to assess the imaging quality of the camera. It was found that the spacer that was ordered with the camera was unnecessary because it increases the focal distance of the camera to a very high value. Thus, the spacer was not used in conjunction with the camera. Overall, the camera gave pretty good image quality although it took a really long while to figure out to focus the camera.

After the imaging quality of the camera was assessed, I tried to communicate with the camera through LabVIEW. First, I used MAX to try to configure the camera. I found out, however, that MAX was unable to detect the USB camera. One possible reason I could think of was that my laptop was either my laptop was dysfunctional, the driver of the camera was not installed correct, or my version of LabVIEW was outdated. Thus, I tried to connect and installed the camera to a desktop in the lab. The desktop has the latest version of LabVIEW. After I carefully installed the driver for the camera, I opened MAX and still, MAX could not detect the camera.

I went online to search for reasons why the camera could not communicate with LabVIEW. Some forums said that LabVIEW does not support USB devices. However, there is a driver called IMAQ for USB 1.0 that can be installed into LabVIEW. This driver allows LabVIEW to communicate and interact with USB devices, such as webcam, microscopes, scanner, etc. These devices, however, must support DirectShow. Basically, DirectShow is a multimedia framework that records media files on demand at the request of the user. The camera that our team purchased supports DirectShow, and thus I installed the IMAQ for USB into LabVIEW, hoping that it would communicate with the camera. After the installation, however, the camera still could not be detected by LabVIEW.

I am currently in the process contacting the camera’s manufacturer, Mightext, and National Instruments to try to resolve these issues. I am awaiting their responses. Aside from contacting the manufacturer, I only went online and did a lot of research on how to resolve
These issues. I also read a lot of forums. In many of these forums, there are a few people experiencing the same problem as I do. The cameras they purchased could not communicate with LabVIEW. For the majority of them, their resolution was to install the IMAQ for USB. On the contrary, some suggested a method of importing a dynamic linked library (DLL) for the camera into LabVIEW.

A DLL is a shared library, and a shared library is a section of precompiled codes that are dynamically linked to the main program at run time. A DLL for a specific device can be imported into LabVIEW through the use of an importing shared library wizard. The wizard can be found in the LabVIEW program. The function of the wizard will be to read the codes within the DLL and compiled them into VIs that can be used in LabVIEW. These VIs will be located in the user library in the LabVIEW directory, and can be called directly from LabVIEW. I’ve tried to import the DLL for the camera into LabVIEW. Unfortunately, the generation of the VIs from the DLL was unsuccessful due to some unexplainable errors within the DLL.

Another method I’ve tried to resolve the issue was installing a third party SDK driver called IVISION LabVIEW Toolkit. An SDK stands for software development kit. The purpose of this SDK driver was to act as a middleman between LabVIEW and the USB device. The driver, however, is not free and quite expensive to purchase. This will be my last resort if I cannot find any other resolution.

Another feasible method is through the use of NI VISA. NI VISA is a high level API used to communicate with instrumentation buses. VISA can be used to communicate with USB device, which in turn can be used by LabVIEW. The VISA involves creating a driver file (INF) for the device and telling the computer to use VISA as the default driver for the device. The device can then be configured in MAX. The use of VISA, however, is extremely complicated and requires that the device must not have its own driver. I am currently researching more about this method and considering this possibility.

Through my research, I also found out that in order to communicate with the USB camera through LabVIEW, using IMAQ for USB, the camera must have a LabVIEW supported DirectShow. It is questionable whether the purchased camera has a DirectShow that supports LabVIEW. I am also in the process of contacting the manufacturer to straighten this out and ask if they can send me a LabVIEW supported DirectShow.

In addition to trying to resolve the camera issue, I’ve also worked on the image acquisition and image processing code for the camera in LabVIEW. Fig. 1 below shows the working version for this code. Basically, the camera will be initialized using the camera initializing VI. Then, the image grab acquire VI will be inside a while loop. The image grab acquire VI allows the camera to capture images. The purpose of the while loop is to allow
continuous capturing of images. Within the while loop is a case structure. The purpose of the case structure is to allow the image to either or not be processed. When the case is true, the image is processed through thresholding, adjusting the brightness and contrast, etc. When the case is false, the image will not be processed and the output will only be the raw image. The case structure allows user to setup the camera to capture optimum raw image before processing the image. Once the image is processed, another VI, called the particle analysis VI, is used to track the markers on the image. This VI tracks the (x,y) positions of each marker, in pixels, on the image. These positions are important in the calculation of strain.

![Image Acquisition/Processing Code](image)

**Fig. 1 – Image Acquisition/Processing Code**

**FUTURE WORKS:**

Future works in the next couple of weeks include contacting National Instruments and Mightex to try to resolve the issue with communication with the camera using LabVIEW. Hopefully, Mightex has a DirectShow driver that is compatible with LabVIEW, because with this driver, the camera can be communicated directly in LabVIEW through the use of the IMAQ for USB. However, if the manufacturer does not have this driver, then I will have to explore other options, such as using NI VISA or importing DLLs to LabVIEW.

Other future works include working on the program. Currently, the motor module is completed and the camera module is almost completed pending the interaction with the camera and LabVIEW. Once the camera is successfully connected to LabVIEW, the camera module will be completed. Other additions to the camera, however, includes a subroutine to graph the positions of the markers so that as the tissues moves, the user will be able to see the markers’ movement along with the tissue simultaneously.

Finally, what remains the testing module and the integration of the calculation module. The testing module, however, is primarily the integration of the motor module and the camera module along with a subroutine to output relevant information into text files. According to Eric, the calculation module is almost completed and once my part of the program is complete, we will integrate it together.