Tendon Strain Apparatus:

A Non-Contacting Solution for Strain Analysis of Soft Tissue

Engineering Capstone Design
Fall 2016 - Spring 2017

The Dev-elopers:
Kelie Gonzalez – gonz6931@vandals.uidaho.edu
Gretchen Gingerich – ging4376@vandals.uidaho.edu
Craig Tedmon – tedm1302@vandals.uidaho.edu
Benjamin Perley – perl7386@vandals.uidaho.ed

Client: Dr. Nathan Schiele – nrschiele@uidaho.edu
Lead Instructor: Dr. Dev Shrestha – devs@uidaho.edu

TABLE OF CONTENTS

EXECUTIVE SUMMARY

This report provides a synopsis and evaluation of Team Dev-eloper's non-contacting strain analysis system for soft tissues. The project was commissioned in conjunction with the Senior Capstone class for Engineering at the University of Idaho and Dr. Nathan Schiele from the UI Biological Engineering Department.

The project aimed to create a camera structure linked to a LabVIEW input matrix, which would allow autonomous data and video recording. This was intended as an upgrade from the previous information collection system, which utilized a handheld personal recording device to capture relevant data.

BACKGROUND

TENDON STRAIN ANALYSIS

Tendons are the soft collagenous tissues in our bodies that are responsible for transferring forces between muscle and bone. Tendon injuries are quite common among athletes, with causes ranging from chronic overuse to acute tears. While most bone and muscle injuries quickly heal over the course of months to full capacity, tendons are unique in that the regenerative process can take over a year and often complete regeneration does not occur. The tendons in most animals' bodies tend to not only take a long time to heal but also have very poor healing capabilities. To help with the regenerative and therapy techniques used in tendon injuries, understanding the stress and strain a tendon can withstand is needed before any clinical alternatives for replacement or healing can take place.

PROBLEM DEFINITION

This project aimed to improve a system that measures strain in tendons which are subjected to a force. The system originally utilized MATLAB and LabVIEW scripts to analyze a pre-recorded video to find the displacement of a marking on a tendon. This required a lab technician to manually record and upload the video of the tendon strain cycle for the analysis. Often, the only recording equipment available was a personal cellphone camera.

Our job was to streamline and automate the process, minimizing the work required by the individual overseeing the experiment in question. This would require several steps, broken into two main projects, altering the code to allow for live data acquisition and replacing the physical components with ones that could record a tendon's displacement more consistently.

Previously, the technician would need to use separate programs for taking recordings and analyzing data. Our goal in altering the code was to combine these steps, shortening the process for the lab technician. To do so, we implemented a MATLAB plugin inside LabVIEW, which gave a user full control of video recording and data analysis inside of one window.

To help with the accuracy of data measurements, we also added an articulating arm and a camera to allow one to take recordings from a consistently fixed position. Buying a camera was also a requirement for live data acquisition, as technicians were originally unable to stream recordings directly to a computer. Ideally, adjusting the code and adding a mounted camera will greatly reduce the amount of time and work required to gather strain measurements.

PROJECT PLAN

While the majority of the project was completed by the group in tandem, there were several roles delegated to individual team members. During our first team meeting, we listed several responsibilities that would only require one group member's attention, and then agreed on who would cover which particular responsibility. Among these were monitoring the project budget, managing our portfolio, and designing the wiki page.

Given that different members had different strengths and educational backgrounds, we chose to split the main responsibilities between the two sets of engineering disciplines, allocating the MATLAB coding and physical design to the Mechanical Engineers, Ben and Craig, and the LabVIEW coding to the Biological Engineers, Kelie and Gretchen. As we transitioned toward using LabVIEW only, Gretchen and Kelie took over the bulk of the programming responsibilities.

Kelie and Gretchen purchased our camera and lens in early March, and the articulating arm was purchased shortly after, following the initial plan. Due to deviations in our programming strategy, the code was completed in mid-April, a few weeks later than we had hoped, but in time for EXPO.

Several steps:

1. Considered MATLAB and LabVIEW
2. Considered camera
3. Considered mounting, lighting

CONCEPTS CONSIDERED

CAMERA AND LENS

When we originally began this process, we knew we needed a camera that was compact as well as within our budget since this was such a large purchase for our project. Our team began to gravitate towards the idea of a GoPro camera. The idea of a durable, compact camera that is extremely popular right now not only sounded like a good idea but also sounded like it could be pretty cool to work with. After some research and talking with our client we decided that though the GoPro idea had some very likeable components there were also more things we needed to consider. We wanted a camera that was compact and lightweight. We also needed live feed from the camera to be pulled straight into LabVIEW without needing another software to fix any distortion, as well as a high resolution; these are some things the GoPro camera couldn't do.

	DCCC1645C	DCC1240C	Marlin F-131 B/C DISCONTINUED	Tucsen DigiRetina	Tucsen TrueChrome
Sensor	CMOS	CMOS	CMOS	CMOS	CMOS
Focal Length	3.5mm-75mm 18-108mm with zoom	3.5mm-75mm 18-108mm with zoom	?	?	?
Read Out Mode	Progressive Scan	Progressive Scan	Progressive Scan	Progressive Scan	Progressive Scan
Reosolution	1280x1024pixels	1280x1024pixels	1280x1024pixels	1920 x 1080p	1920x1080(Dynamic) 3264x1840 (Static)
Sensitive Area	4.61 x 3.69 mm	6.78 x 5.43 mm	N/A	N/A	
Pixel size	3.6 µm	5.3 µm	6.7 µm	1.335µm	2.8µm
Frame Rate(FreeRun)	24.9fps	25.8fps	25fps	25fps	30fps
Dimensions(HxWxD in)	1.91 x 1.73 x 1.01	1.59 x 1.26 x 1.63	2.28 x 1.73 x 1.10	?	?
Weight	0.07 lbs	0.16 lbs	0.26 lbs	?	?
Price	$355	$1,160	~$1,100	$550	$1,040 (with monitor)

Figure 1. Camera comparison chart

When we were considering lenses to work with our camera, we initially tried to find a lens with the smallest focal length available. We thought this would give us the best ability to focus on the soft tissue when the lens is very close to it. We found the Tamron 13FM22IR Compact CS-Mount Lens with a 2.2mm focal length. However, once the lens arrived and we tested it out, it had a high amount of distortion and the image was fish eyed. We learned that even though a smaller focal length can allow you to focus the image up closer there tends to be a higher amount of distortion in these lenses. After this, we determined that we could use a lens with a larger focal length and still get the same focusing capabilities needed for our system.

Concept Selected: Camera and Lens

The camera we decided on is the ThorLabs DCC1645C compact camera. It measures at only 1in by 1.7in by 1.9inches and weighs 32 grams or 0.07lbs allowing it to cause not excessive mounting to the bioreactor frame. It runs through USB hookup straight to the computer. This camera covered everything we aimed for while still being less than a quarter of our budget. To go along with our camera, we needed to purchase a lens.

We were able to purchase a lens that we had previously tested out, thanks to our lead instructor Dev Shrestha. This lens features an 8mm focal length allowing us to not have any distortion errors within our live feed as well as a manual focus to make sure it is as focused as the researchers choose. With these two components together, we were able to accomplish a compact, cost efficient, and effective camera system that can be incorporated into our programming system.

Figure 2. Camera and lens selections

PROGRAMING

Both MATLAB and LabVIEW were considered as possible coding software programs for this system. Initially, we planned to use LabVIEW to control and record the motor and bioreactor, and then analyze the recordings using MATLAB. Given that one of our objectives was to streamline the process and minimize the steps required for data collection, it was necessary to configure our programs so that they could run simultaneously. Rather than having two separate programs running two separate codes, we opted to download a MATLAB plugin that allowed one to run a MATLAB script inside of the LabVIEW window.

Concept Selected: Programing

We spent time looking over both software and the capabilities allotted in each. After discussing it with our instructor and as a team we decided to focus our code in LabVIEW. We had a few reasons for wanting to do this. For our team, no one considered themselves an expert in any software by any means. We also knew from our prior experience that it would be a relatively user friendly graphical user interface so that if someone who didn’t know the software was working with it they would still be able to look at it and see where the problem is coming from and then seek to resolve any issues if some were presented. Another reason was that our client and his researchers did have some experience. Due to these considerations, we felt this was the most appropriate route and choice for our programming software.

MOUNTING

One of the primary challenges of this project was to design a mounting system that could lock the camera in a given position consistently while also being easily adjustable in three-dimensions. Several ideas for a mounting mechanism were discussed, but mounts utilizing magnetic strips, 3-D printed plastic mounts, and mounts simply ordered off-line were the main choices considered.

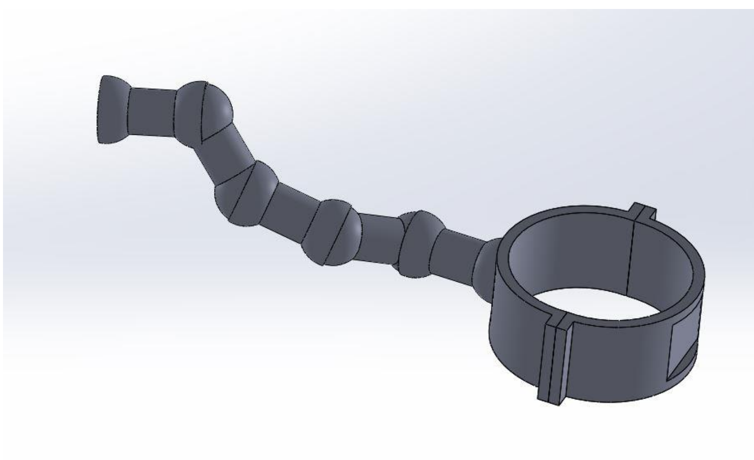

Figure 3. Initial Solid Works Model for Articulating Arm

A magnetic mount would have effectively done everything that was needed, however, it was more costly and spacious than the other choices and the frame is aluminum so the magnets couldn’t attach to it. A magnet could also potentially damage the camera, adding unnecessary risk given its day-to-day usage. A plastic, 3-D printed mount, though not sharing a magnetic mount's disadvantages, might not have been sturdy enough to keep the camera stationed in a precise location consistently.

Concept Selected: Mounting

Ultimately, we elected to purchase a reasonably priced, metal mount online from Tether Tools. The arm is made of aerospace aluminum and has one knob that releases and locks the center joint and the two articulating joints at each end to adjust the cameras position. The mount is 7” long and is capable holding devices up to 4.5 lbs. The arm has ¼”-20 male threads on both ends of it, enabling it to connect to any ¼”-20 female receptor. The arm comes with a removable clamp that can be attached to the frame of the bioreactor system at any height.

Figure 4. Selected articulating arm camera mount

SYSTEM DESIGN

Our non-contact video system displays the live video feed in the left window (seen in Figure 5) while simultaneously measuring the displacement of two points on a collagen sponge or tendon. This displacement is then calculated and graphed in the window to the right of the video feed. We placed indicators the show the numerical value of strain as the system is running, the displacement, and the number of point matches the program is picking up. We also have two error indicators which help the user to easily find what and where the errors are if there happens to be any.

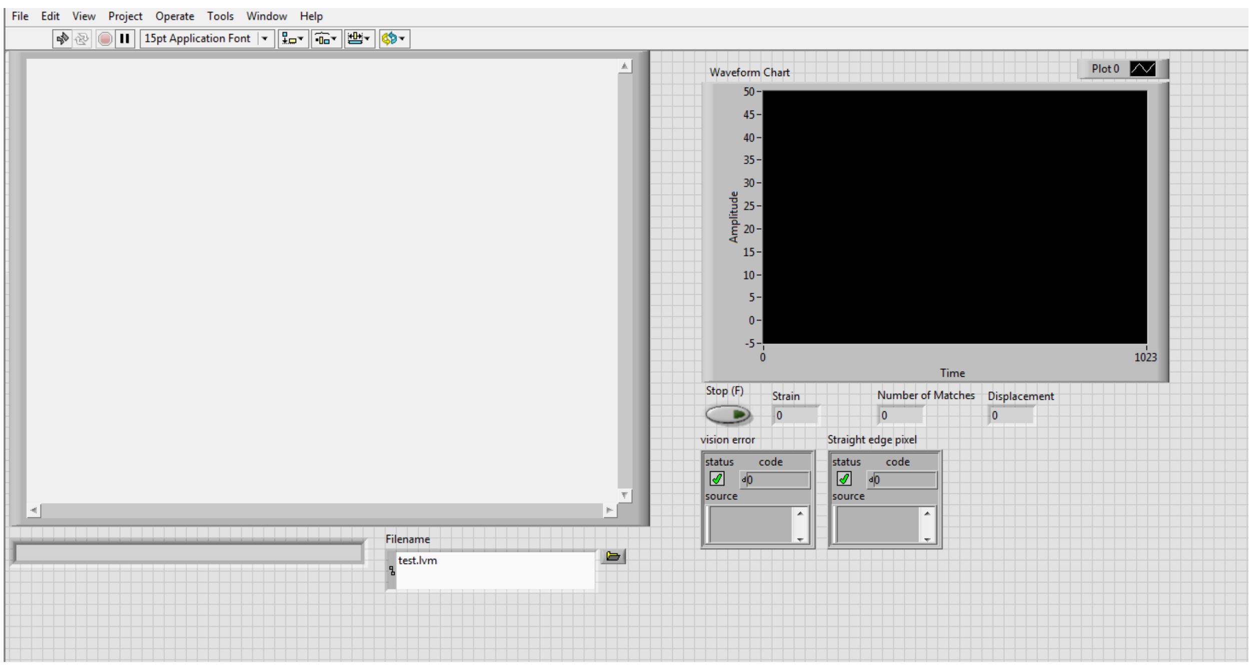

Figure 5. Front panel view of program in LabVIEW

Shown below in Figure 6 is the block diagram of our LabVIEW program. This is the behind the scenes of our programing. The live video feed comes into the 'vision acquisition' module. Within this module, you can adjust the frame rate and whether you want the image to be grayscale or in color. We decided that we wanted to use grayscale to increase the contrast of the two points on the soft tissue. From there the image is pulled into a module called ‘vision assistant’. Within this module there are many different options for image analysis. To track the change in length of the tissue we used ‘pattern matching’. A template must be manually made from the points drawn on the soft tissues but after it is saved it will work for each run of the code. Once the two points are found, our code outputs the x-values of each point. The code subtracts them and takes the absolute value to find the displacement. The smaller loop above our main loop also tracks the displacement between the two dots but it stops after it gets one data point. This gives us the initial displacement between the two points. We use the initial displacement and the changing displacement to get our strain calculation over time, $\frac{new\ length - initial\ length}{initial\ length}$. To convert this calculation from pixels to millimeters we use another ‘vision assistant’ module. A different feature within it called ‘gauge’ is used to measure the distance between did the black seal on the bioreactor in pixels. We use the known distance in millimeters to create a millimeter to pixel ratio that's unique to each time the system runs depending on camera placement. Using this ratio, we convert our strain calculation to millimeters and it is plotted into the chart displayed on the front panel. This data can also be saved as a file in the ‘write to measurement file’ module. The user can choose the file type.

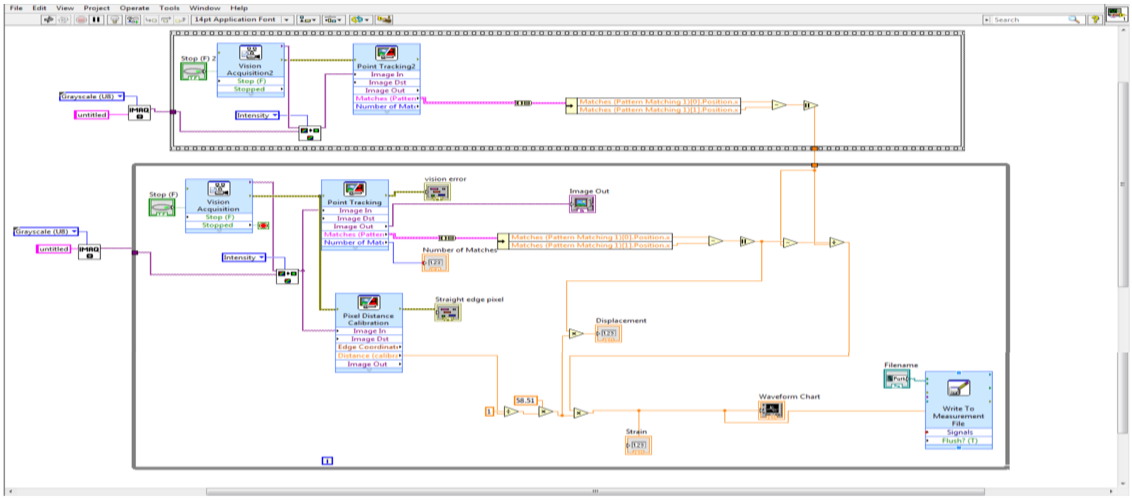

Figure 6. Block diagram view of program in LabVIEW

Shown below in Figure 7 is the assembled camera system in our client's lab, hooked up to his bioreactor and the computer. As you can see, the physical components of the system include the articulating mounting arm, the camera, and the lens.

Figure 7. Final setup of system in client's lab

Due to a large portion of our project being code/programming based we wanted to make sure anyone using the system could work it and understand what is going on without being lost. We have written and included a "how-to" guide to running our code. It is included below.

A How-To Guide to Our Code

1. Open code in LabVIEW
2. Push the Run button to run the program with the material with two identical points marked on it in the Live Feed viewer box
3. Push stop button
4. Go into Block Diagram
5. Right click on the Point Tracking module and go to 'Properties'
6. Click on 'Pattern Matching'
7. Make sure that the green box on the life feed image is adjusted so both points are positioned in it
8. Click on 'New template' (a new window should show the live feed image of the material with the two identical points mark on it, on the screen when you pushed the stop button in step 3)
9. Adjust red box around one of the points marked on the material.
10. Scribble out the background ensuring just the one point is visible
11. Click finish, this will take you back to the Pattern Matching screen
12. Make sure it is picking up both points
13. Click finish (make sure calibrated matches is selected in the controls before you click finish)
    a. Once this step is completed make sure that this box 

still shows 'Matches (Pattern Matching 1)[0].Position.x' and 'Matches (Pattern Matching 1)[1].Position.x'

14. Do these steps in Point Tracking2 in the smaller loop above the while loop

15. Once this is done, you can push run and run the program continuously.

DESIGN EVALUATION

OBSTACLES ENCOUNTERED

Throughout the course of this project, our team encountered many different obstacles to overcome. One of the biggest setbacks for our team came when our camera and lens were stolen just six weeks before the deadline for completion of our project. Without our camera system available, we were unable to run our programming system. This also set us back with the mounting system as well because we didn't have anything to physically mount to the bioreactor frame.

Though this was not something we expected to ever have to deal with, we proceeded with the purchase of a new camera system and spent our downtime developing a code to run with the webcam we had available.

Another large obstacle we have encountered over the course of this project is software compatibilities. It began with our LabVIEW software. We began working in an older version of the program but as we developed our code and began to integrate it into our client's lab, we realized he was working in a newer version. You would think that it would be a relatively easy fix but switching from different versions also meant that some of the extra vision acquisition software also needed to be reinstalled. After a few weeks of reinstalling software and converting to different versions, we were able to settle back into LabVIEW running the 2015 version.

The final obstacle we encountered was one of the most discouraging setback, not because it was the most difficult or shocking but because we were done with our code and had one final issue:

the camera we selected was not pulling live feed video into our system. After working with the IT department, our client, and the manufacturers of our camera we found that it wasn't a driver installation issue as we suspected, but it was an issue with the VI's in LabVIEW. To incorporate the live feed from our camera, an additional amount of coding needed to be added to our developed code.

As a team, we felt that though these obstacles were never anticipated and sometimes very disheartening, it showed us how to react and work around these setbacks.

BUDGET

Project Budget Breakdown	
Camera	$364.04
Lens	$84.99
Articulating Arm	$108.18
MATLAB Data Acquisition Software	$87.00
Expo Poster	$72.80
Snapshot Posters	$52.00
Total Expenses	**$769.01**
Budget Remaining	**$730.99**

Figure 8. Breakdown of budget for project expenses

Overall, we were pleased to see that our expenses were only about half of our total project allowance. The main factor that led to this was that, when possible, we tried to purchase materials that were all-inclusive. One of our earlier camera options was a GoPro Hero; however, after some research, we found that close recordings fish-eyed significantly, which would require us to purchase more software to correct distortions.

Not only was our ThorLabs camera cheaper individually, but it also wouldn't require any subsequent purchases. We also saved money choosing a basic, metal articulating arm. A typical magnetic arm costs around $100 before tax, while the Tethertools design was only $60. At the same time, we believed the convenience of having a sturdy, durable design would make the Tethertools arm a superior choice over a virtually free 3D-printed arm.

For our entire project and system, we believe that we struck a good balance between saving on expenses when possible, while also choosing products that would fit the client's design specifications and last many years.

FUTURE WORK

Overall, we believe we built an effective and cost-efficient system for our client and his researchers. Though we feel we did what we could with our experience and allotted time, this doesn't mean it cannot be improved and added onto.

There are a few components that we did not get to implement into our design. One feature we anticipated would be a good component is a lighting system. Because this system is going into an incubator, this could limit the amount of light the camera is exposed to. This could potentially be an issue with the pattern matching in the LabVIEW code.

Another component that could become an issue, with lighting, is glare. Unfortunately, because we did not get the lighting added on, we were unable to check to see if the plastic barrier on the bioreactor would cause a glare in the image.

Other than those two components, we feel that our team accomplished the major components of our project. As with anything, improvements are always possible, but our finished project taught us coding, team work, design analysis, and real-world application of our engineering skills.