LISA-T Solar Array Deployment System and Articulation System
User Manual

Team Contact Information:

Name: Alexander Mueller
Email: alm558@nau.edu
Phone: 623-466-4875

Name: Daniel Krollman
Email: djk258@nau.edu
Phone: 815-354-8129

Table of Contents

I. Introduction

Initially, we would like to thank you for choosing our group out of multiple applicants to develop a deployment, retraction, and articulation system for the LISA-T prototype. We feel through this project we have been able to showcase our engineering skills we have proctored over the last four years. There are multiple needs that needed to be implemented in order to make the LISA-T system more robust. The deployment and retraction of the array ensure that no research instruments are obstructed during spaceflight. Additionally, the articulation of the solar array is needed to ensure maximum power generation on the CubeSat and also allows the array to function autonomously. We have showcased a robust deployment/retraction and articulation system to suit your needs.

First off, our system uses a scissor-lift mechanism which is able to deploy and retract the solar array, although it needs more development in order to perform its functions smoothly. Additionally, we have developed an automatic solar tracking mechanism which autonomously tracks the position of greatest light intensity. This subsystem parallels an additional manual control subsystem that takes in user input and adjusts the positioning of the solar array. This subsystem is needed in order to make any minor adjustments to the array's positioning in case of algorithmic or hardware failure that might occur during normal operations. All of the controls for our prototype are contained in a control panel. The control panel has three toggle switches, two potentiometers, and one pushbutton. The three toggle switches are used to determine which mode the user will choose when push button is pushed. The first toggle switch is for the activating the scissor lift. This is not set into motion by the pushbutton but rather is checked by the program continuously and set into motion if this switch is flipped. The second and third toggle switches control whether the user wants to manually adjust the rotation or the tilt of the array respectively. In parallel, the potentiometers control the amount that the array is articulated on each axis. Lastly, the push button is used to activate the manual control of the system. All of these control are located on an easy-to-use control panel to create a user-friendly environment.

This user manual will go more into detail in the following sections in order to help ensure the longevity of this product and its usefulness. The following sections will contain instructions on installation, configuration, maintenance, and troubleshooting in order to ensure the functionality of our product. Next will be our individual accounts of which subsystems we worked on and what type of work we performed during our development of our system. The work breakdown structure will convey all tasks complete or incomplete and detail the process to completion or incompletion. We thank you for your time and hope that this user manual will help ensure our product will last in the future.

II. Installation:

Our design is based on an Arduino Uno microcontroller. This device runs the circuits that drive the entire system. There are also many 3D printed parts that make up the mechanical motion and housings for the product. The 3D Printed parts and electrical system are detailed in this section.

Electrical Installation:

The entire circuit diagram can be seen in Appendix A. The arduino itself runs off the power supply coming from the USB connection to the computer. Once the code is uploaded (using the arduino-branded software), the arduino can also be powered through the DC input on the board connected to a 9V battery. All the peripheral circuits run off the 5V line on the arduino board itself. The 4 photoresistors utilized in the automatic sun tracking system are tied to the analog input pins on the arduino to measure the voltage change as the light changes across the photoresistor array. The 3 servo motors are tied to 5V, ground, and digital IO pins on the arduino board for control. Specifically, the four photoresistors are in series with their own 100K ohm resistor. The amount of voltage is read from the junction of these two devices and each are read by analog pins A0, A1, A2, and A3. The servo motor that controls rotation is connected to the digital pin D4, tilt is connected to D5, and the lifting servo is connected to D6. A full circuit schematic can be seen in appendix A.

Wiring Connections:

There are multiple ports coming off of the arduino shield to ensure that the correct wiring is easily obtainable.

1.) First connect the ribbon cable that comes with the assembly to the port labeled “sensors”. This port is used to read data from the sensors into the microprocessor.
2.) Next, connect the servos. The servos have labeled ports named “tilt.s”, “rot.s”, and “lift.s”. These names mean that these are the necessary ports for the data line to the servo motors themselves.
3.) The next step is to connect the control board. There are ports named “IRQ”, “lift.c”, “rot.c”, tilt.c”, “rot.p”, and “tilt.p”. These naming conventions go hand-in-hand with the names on the control board. The two knobs connect to the “.p” ports, the push button labeled “set pos” connects to the IRQ port, and the three toggle switches connect to the “.c” ports.
4.) After the electrical wiring has been completed the code can be uploaded to the board via a USB connection.

Programming the Arduino:

In order to ensure the Arduino has the correct program, it is recommended that the arduino be programmed using the “manualcontrol.ino” file given to the user.

1.) Initially, after the wiring is completed the shield needs to be connected to the Arduino. The shield connects to the top via the headers on top of the Arduino. Ensure that the digital and analog I/O pins are in the right configuration and connect the shield to the Arduino.
2.) Afterwards, connect the USB cable to the Arduino and upload the “manualcontrol.ino” file to the Arduino.
3.) These steps should ensure that everything is wired correctly and your product should start articulation automatically.

Mechanical Installation:

The mechanical system can be broken down into the different components we designed on creo for this project. Table 1 below shows the different file names and the purpose they serve within the product. Figure 1 displays the full CAD model of the system. Note: the CAD model in figure 2 also only displays 3 “links” of scissor lift; this can be altered reach a desired extension height. The mechanical pieces are held together using #10-24 x ⅜” Pan head phillips machine screws.

Table 1: 3D printed CAD files and their purposes

File Name	**Quantity**	**Purpose**
2ucs.prt	1	2u Cubesat Chassis, mounts for scissorlift/scissorlift drive mechanism
DepPlate.prt	1	Photoresistor Array Housing/”Tilt” servo mount
Tiltaxis.prt	1	Axis that connects the “Tilt” servo and “rotation” servo
Topplate.prt	1	Top plate of the scissor lift/ “rotatio” servo mount
Type1.prt	As required	Type 1 of scissorlift “link” pieces, hole configuration: large/small/large
Type2.prt	As required	Type 2 of scissorlift “link” pieces, hole configuration: small/large/small

Contservomount.prt	1	The mount for the bolt that ties the ballscrew to the continuous servo to drive the scissor lift
Boltthreadpiece.prt	1	Mount for the ball screw to be threaded around the “bolt” that is connected to the continuous servo to drive the scissor lift

180 degree Servo Mount Installation:

There are two 180 degree servos and one continuous servo used in this project. The mounting procedure for these servos is documented below:

1.) Mount one 180-degree servo “horizontally” flat against the bottom of the “depplate.prt” piece use the “mounting pegs” on the bottom of this piece to anchor the servo and use hot glue to fix the servo to that plate.
2.) The axis for the first servo will fit into the upper horizontally oriented hole of the “tiltaxis” piece
3.) The second 180-degree servo sits in the rectangular hole in the center of the “topplate” piece. Use hot glue to anchor the servo in place once seated.
4.) The axis for the second servo will be seated into the lower vertical hole of the “tiltaxis” piece.

Preparing Ball Screw Pieces:

1.) Gather contservomount.prt piece and boltthreadedpiece.prt piece.
2.) Gather two nuts that fit with desired bolt size (bolt will be the actual driver of the ballscrew)
3.) Using highly adhesive glue (krazy glue/gorilla glue), fix the nuts to the hexagonal holes in each piece.

Scissor Lift Construction Process:

This section details how to construct the scissor lift for the deployment system and how to install the scissor lift into the cubesat chassis.

Stage 1: Building Individual “links”

1.) Gather one “type1” piece and one “type2” piece.
2.) Thread a screw through the middle “large” hole on the type2 piece.

3.) Seat that screw into the middle “small” hole on the type1 piece (you should have a pivotable “cross link” from at this point.
4.) Repeat steps 1-3 for desired amount of “links” for each side of the scissor lift

Stage 2: Connecting Individual Links

1.) Flip two “link” assemblies over so that the heads of the middle screws you seated in the previous stage are face-down on the table.
2.) Layer the Large hole at the end of the type 1 pieces above the small holes of the type 2 pieces.
3.) Thread a screw through the large hole of the type 1 pieces, and fasten that screw into the small hole of the type 2 pieces.
4.) Repeat steps 1-3 necessary to link as many pieces together as desired.

Stage 3: Anchoring the bottom of the scissor lift to the cubesat body

1.) Gather the two sides of the scissor lift and the 2ucs.prt cubesat chassis.
2.) From the inside of the cubesat chassis, align the large hole on the end of the type1 piece (red) with the anchor hole on the cubesat chassis.
3.) From the inside of the chassis, thread a screw through the large hole of the type1 piece, and fasten that screw into the small hole of the cubesat body.
4.) From the outside of the cubesat chassis, thread a screw through the “rail” cutout, and fasten into the small hole at the end of the type2 piece. Additionally, thread this screw through the hole on the “boltthreadpiece” to fix the type2 piece to the boththreadpiece.
5.) Repeat steps 1-4 for opposite side of scissor lift.

Stage 4: Anchoring the top of the scissor lift to the topplate

1.) Gather the cubesat chassis with assembled scissor lift sides assembled in stage 3 and the Topplate.prt piece
2.) Align the large hole of the type1 piece with the small hole on the top plate piece
3.) Thread a screw through the large hole of the type 1 piece and fasten it into the small hole of the type 2 piece.
4.) Align the small hole of the type2 piece with the “rail” on the top plate.
5.) Thread a screw through the rail on the top plate, and fasten it into the small hole of the type 2 piece.
6.) Repeat steps 1-5 for opposite side of the scissor lift.

Installing the continuous servo and scissor lift drive mechanism:

This section details how to install the continuous servo on the cubesat chassis as well as the ball screw mechanism that drives the scissor lift.

1.) Gather the continuous servo and “contservomount.prt” piece.
2.) Using the “crosshair” mounting pad on the servo, align the contservomount piece with the crosshairs for optimal fit and rotation
3.) Using an electric drill, drill through the crosshair piece into the contservomount piece to create holes to fix the servo to the 3D printed piece with screws.
4.) Use screws to fix the continuous servo to the contservomount piece
5.) From the inside of the cubesat chassis, seat the continuous servo in the rectangular hole in the cubesat chassis. Use hot glue to fasten the servo in place
6.) Once everything else is installed (servos, scissorlift, threaded crossbar piece), thread the bolt that will drive the ball screw through the circular hole in the cubesat chassis
7.) Thread the bolt through the Boltthreadpiece.prt at the base of the scissor lift.
8.) Before threading the bolt into the contservomount piece, add strong adhesive (krazyglue/gorilla glue) to both the threads of the bolt that will be going into that nut, and the threads of the contservomount piece itself (this step is very important, the bolt must be fixed to the rotational motion of the servo, so if it is not adhered correctly, the bolt can slip and no rotational motion will be applied to the bolt to drive the scissor lift)

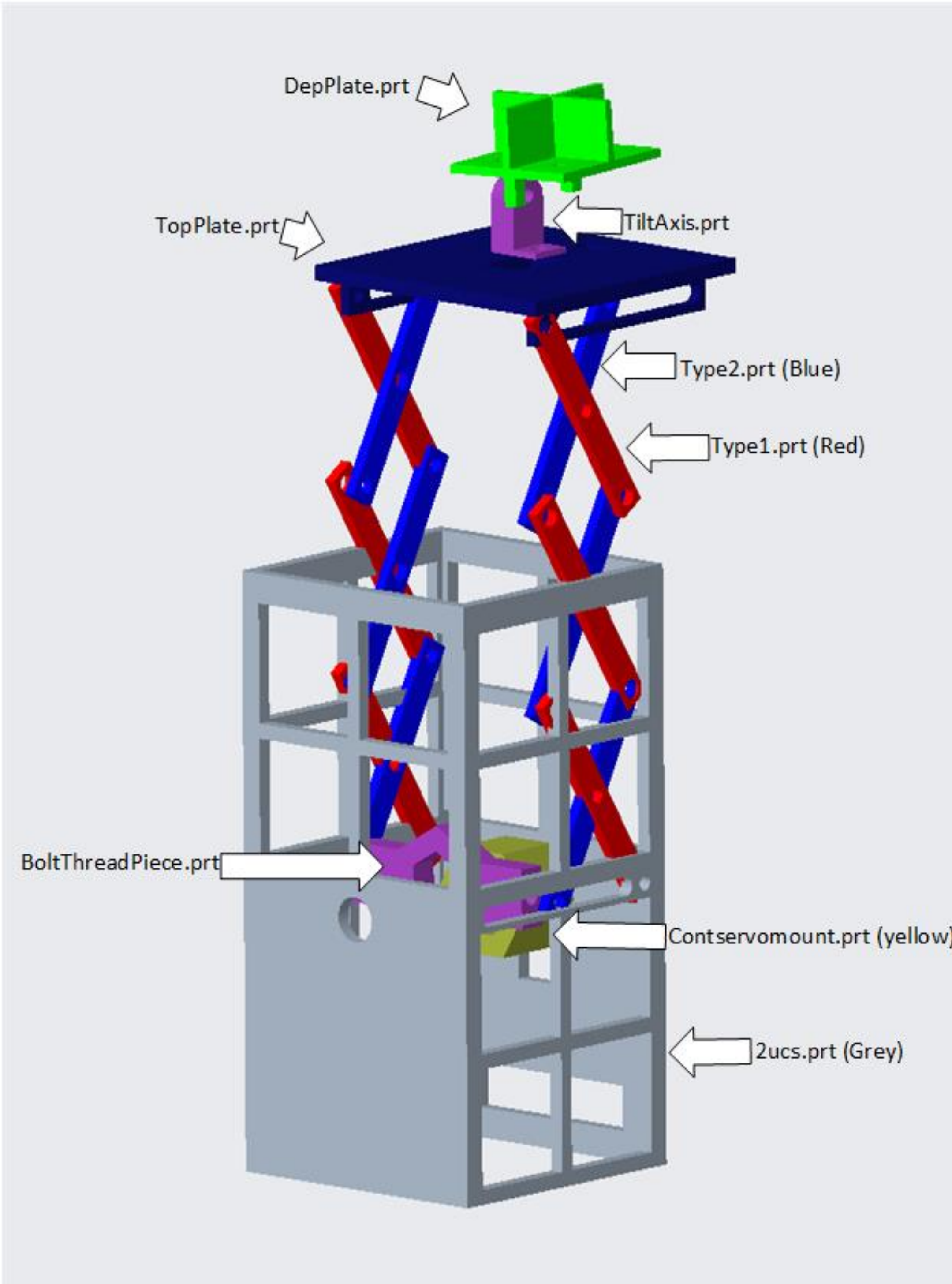

Figure 1: Full assembly CAD Model of Deployed system

III. Configuration and Use:

Articulation System:

Once the product is fully assembled and the circuit and arduino are ready, upload the .ino arduino code to the board. Once the code is successfully uploaded, and everything is set up correctly, the articulation system will first snap to its middle position flat deployment plate relative to the ground (middle of range of both rotation and tilt servos). Once it reaches the initial position, the articulation system will automatically start tracking the strongest light source. No user input is needed for the light tracking, the system will just try to point the "face" of the deployment plate at the strongest light source based off the readings from the photoresistors.

Manual Control:

Initiating manual control to correct the articulation system is fairly straightforward. The steps to initiate manual corrections are detailed below. See Figure 2 below for an image of the control board for reference on the different switches/toggles needed for this part.

Figure 2: Image of the user control panel

Initiating a correction on the tilt axis:

1.) Toggle the “tilt” switch up
2.) Set the position of the “tilt” potentiometer to the position you want the tilt axis to snap to (this may take some experimentation to find the position you need, depending on which orientation you mounted the servo)
3.) Press the “Set Pos” button, and the tilt servo will snap to the position specified by the tilt potentiometer
4.) Once the servo snaps to the desired position, the automatic articulation system will resume based on the readings from the light sensors.
5.) Toggle the “tilt” switch down to put the control board in safe mode (no manual control)

Initiating correction on the rotation axis:

1.) Toggle the “rotation” switch up
2.) Set the position of the “rotation” potentiometer to the position you want the rotation axis to snap to (this may take some experimentation to find the position you need, depending on which orientation you mounted the servo)
3.) Press the “Set Pos” button, and the rotation servo will snap to the position specified by the rotation potentiometer
4.) Once the servo snaps to the desired position, the automatic articulation system will resume based on the readings from the light sensors.
5.) Toggle the “rotation” switch down to put the control board in safe mode (no manual control)

Controlling the deployment system:

When the code is first uploaded to the arduino board, it assumes that the deployment system is fully retracted. If the deployment system is extended initially, please unplug the arduino from the USB or DC power and retract the system by hand. Failure to start the system in the undeployed position could result in damage to the ball screw mechanism that drives the scissor lift.

Once the code is uploaded to the arduino board and the system is in a undeployed state, the deployment system is ready for user interaction. In order to extend the scissor lift, the user needs to toggle the “lift cntrl” switch on the control board up. Once the servo starts spinning, set the “lift cntrl” switch back to the down position. The servo motor will continue spinning until the ballscrew has reached its deployed state. Failure to return the “lift cntrl” switch back to the down position will result in the deployment system automatically retracting the scissor lift as soon as it is done extending.

To retract the deployed scissor lift, quickly toggle the “lift cntrl” switch up and down to start the servo motor, and retract the ball screw driving the scissor lift. Again, failure to switch the “lift cntrl” back to the down position will result in the deployment system automatically extending once the ballscrew is fully retracted.

IV. Maintenance & Troubleshooting

Overall, because this project is just a prototype for a potentially space-faring component, there is not much expectation for this project to last longer than the initial test phase. Therefore we do not predict much, if any, maintenance will need to be performed during this system’s life-span.

Troubleshooting:

Articulation System doesn’t track light source correctly:

- Ensure that the photoresistors are connected to the correct analog pins on the arduino relative to their location on the deployment plate, and also the position specified in the arduino code.
- Ensure that when mounting the articulation servos, they are mounted initially in a “center” position i.e. each servo has 90 degrees of rotation from the “initial” point when the code is first uploaded to the board.
- Ensure that the servos 3-pin connections are connected to the correct pins on the board, each servo has a 5V pin, GND pin, and a digital communication pin on the arduino board. They need to be tied to the correct terminals otherwise the servos won’t move.
- Ensure that the digital communication pin from each servo is connected to the correct digital pin on the arduino (specified in the code).

Manual control doesn’t work correctly:

- First check to ensure the automatic articulation system is working correctly (see above)
- Ensure all of the toggle switches, push button, and potentiometers are connected to the correct pins on the arduino board (specified in the arduino code)
- Check for shorts/bad connections in the wiring between the arduino and the control board.
- When trying to do tilt and rotation corrections using manual control, only one can be done at a time. If both “tilt” and “rotation” toggle switches are in the up position, no

manual control will happen. You must first set a manual rotation position, then a manual tilt position.

Scissor Lift Deployment doesn't work correctly:

- Ensure that the manual control toggle switch on the control board is connected correctly (see arduino code for pin placement)
- If the is servo is spinning, but the ballscrew isn't, re-glue the base of the ball screw into the nut held by the servo motor.
- If the servo is spinning, but it is under a lot of strain, make sure that the scissor lift links aren't getting caught on the cubesat chassis. Also ensure that the screws holding together the scissor lift are not getting snagged on anything.
- If there is a lot of friction in the scissor Lift system, check each screw individually for tightness, ensuring that the scissor lift links are tight enough to create a rigid structure once deployed, but not so tight that the scissor lift links cannot pivot around the screws.

V. Work Breakdown Structure

In this section, our team will explain which portions of the project each individual was responsible for completing, if these tasks were completed or left unfinished, and the reasoning for incompletion if any part was not finished.

ID	Activity/Task	Description	Deliverable(s)	People Involved
1	**Solar Tracking & Articulation System**			
1.1	Solar Tracking	Read Photoresistors & Calculate posistion of light source		
1.1.1	Calibrate Photoresistors	Install & Test photoresistors to reliably sense changes in light	Reliably read light sensors	Dan, Alex
1.2	Articluation			
1.2.1	Design 3D Housing for Motors and axis	CAD Design compoents for articulation system	3D Printed Prototypes	Dan
1.2.2	Write Arduino Code	Using sensor values, calculate posistion of light and move motors accordingly	Track light position Reflect with motors	Dan, Alex
1.2.3	Install 3D components with electrical hardware	Integrate printed components with arduino hardware	Prototype ready for testing	Dan
1.2.4	Test/Optimize	Test and re-work code/hardware to preform optimally	Working Tracking & Articulation system	Dan, Alex
1.3	Manual Control			
1.3.1	Prelim Design & Purchasing	Research components & complete preliminary design for manual control system	Parts Purchased	Alex
1.3.2	Build Circuit	Build manual control circuit	Completed manual control circuit	Alex
1.3.3	Implement Arduino Code & Motors	Write arduino code to utilize the manual controls and test with motors	Circuit and Hardware ready for test	Alex
1.3.4	Test/Optimize	Test and re-work code/hardware to preform optimally	Working Manual Control subsystem	Dan, Alex

The solar tracking and articulation system is split up into 3 subtasks: Solar tracking, articulation, and manual control. Dan and I were able to accurately read light from the sensors and moved on to the next task.

The articulation took more time than we had originally planned. Alex ended up spearheading this portion as well as the manual control portion, so Dan was able to focus solely on the scissor lift mechanism. Dan also completed and finalized the 3D design for the articulation axis. Everything under the articulation subtask was completed and working by our presentation date.

ID	Activity/Task	Description	Deliverable(s)	People Involved
2	Deployment System			
2.1	Design			
2.1.1	3D Design	Develop 3D Model for Scissor Lift	Printable CAD file	Dan
2.1.2	Preliminary Prints	Print prototype for scissor lift	Printed ScissorLift	Dan
2.2	Motor & Software Integration	integrate motor with scissor lift and write software for control	Testable prototype ready	Dan
2.3	Test/Optimize	Test Scissor lift and optimize 3D Design for CubeSat	Prototype ready for integreation with other systems	Dan

The deployment system became the most difficult aspect of this project. We struggled to get a working (hand powered) model of the scissor lift until about mid-March. We had gone through many iterations of prints and we kept having pieces that wouldn't fit together correctly or would be scaled oddly with oval-shaped holes. We didn't get a full scissor lift mechanism built until April, so we were already behind schedule by the time we had to integrate the servo motor to drive the scissor lift. We also went through delays in designing the ball screw mechanism that drives the scissor lift. Initially, we planned to 3D print this entire mechanism, however once Dan began the 3D CAD modeling, he found it to be more difficult to create threaded pieces to fit together smoothly. That, coupled with the knowledge of the low-resolution prints the makerbots at Cline Library produce led us to believe that 3D printing this ball screw mechanism wasn't our best course of action. Instead, we decided to get a 6-inch bolt from Home depot, as well as two nuts that would thread around the bolt. One nut would be fixed (with glue) to the rotational axis of the continuous servo, and the other nut would be fixed to a crossbar at the end of the scissor lift and be able to thread smoothly around the bolt as the servo motor spins the bolt. This would give us the threading motion needed to convert the rotational motion of the continuous servo to linear motion needed by the scissor lift. Ultimately, we got all of the pieces assembled, and the ball screw mechanism worked as expected. However, the scissor lift itself had too much slack in the design and the ball screw was only able to extend the scissor lift a few inches rather than the .5 meter we expected. A tighter-fitting, better designed scissor lift may solve this issue. We believe that if we had a mechanical engineer on our team to spearhead this aspect of the project, the scissor lift would have turned out better than it did.

ID	Activity/Task	Description	Deliverable(s)	People Involved
3	**Integrate Articulation Subsystems**			
3.1	Combine software/hardware systems	Integrate software and hardware for the completed system		
3.1.1	Combine software	Combine arduino code for manual control and solar tracking	Arduino script	Alex
3.1.2	Combine hardware	Combine hardware for manual control and solar tracking to one board	Complete circuit on breadboard	Alex
3.1.3	Testing	Test for accuracy and to check for any unwanted effects	Debugged system	Alex
3.2	Implement Control Circuit Shield for Arduino	Research, order, build, and test shield		
3.2.1	Research and order proper shield for arduino	Ensure the shield will have enough space for all components	Part purchased	Alex
3.2.2	Copy final circuit to shield	Recreate control circuit to fit on arduino shield	Circuit chematic	Alex
3.2.3	Review shield schematic	Review with team the entire circuit schematic	Checked schematic	Alex, Dan
3.2.4	Final Implementation	Solder and secure all components of final circuit	Finished shield	Alex
3.2.5	Testing	Test connections using voltmeter	Checked circuit	Alex, Dan

For the integration of articulation subsystems tasks, Alex completed all tasks by the presentation date. The software was relatively simple to integrate together, but the hardware was challenging. Combining the hardware into one Arduino shield proved difficult for two reasons. The first reason was that certain tools were needed to be purchased for performing cable management techniques to clean up how the device looked. Additionally, the second reason being that there was a substantial amount of delicate soldering that needed to be done in order to successfully implement a shield apparatus for the Arduino. This was completed solely by Alex and the shield tested correct for all functionalities.

ID	Activity/Task	Description	Deliverable(s)	People Involved
4	**Final Integration of Subsystems**			
4.1	Assemble completed subsystems	Mount articulation on completed cubesat housing with integrated scissor lift		
4.1.1	Redesign 3D assembly for optimal fit	Modify currect CAD files to fit final design	Printable CAD file	Dan, Alex
4.1.2	Print final 3D assembly	Send CAD file to library for printing	Printed pieces	Dan
4.1.3	Build 3D assembly	Construct final assembly of printed pieces	Constructed 3D housing	Dan, Alex
4.1.4	Integrate 3D assembly with other subsystems	Construct final working prototype with all subsystems integrated together	Constructed final prototype	Dan, Alex
4.1.5	Test	Test system for failures and correctness	Debugged deployment/articulation system	Dan, Alex

Integration of subsystems went fairly smoothly with the exception of integrating the scissor lift. Once the automatic solar tracking was implemented successfully, it was fairly straightforward to add manual control to that subsystem. Once we had a good idea of how the manual control and articulation system worked with the user input devices, we were able to optimize the "depplate.prt" piece, and "tilt"axis piece to be ready to integrate with the scissor lift. We also designed the control panel piece at that time. However, because of our troubles with designing the scissor lift, integration of the three subsystems into a cubesat chassis was at a standstill for most of March/early April. Once we had a working scissor mechanism, and had figured out the best way to drive the scissor lift, we were able to re-design our cubesat chassis with mounts for the scissor lift, ball screw mechanism, and continuous servo. We only had a single print of the ball screw mount pieces, as well as the cubesat chassis. Once we had those pieces, we were able to integrate the scissor lift and articulation system into the cubesat chassis, and once we performed some by-hand modifications (such as cutting off some of the “crossbar supports” from the cubesat chassis to remove pieces that were obstructing the scissor lift) we had a working model of our prototype. The articulation system worked well, and the manual control was responsive and easy to use. The scissor lift ball screw mechanism worked the way we expected it to, but the looseness of our actual scissor lift links left a lot of wiggle room in the scissor lift, so the ball screw was only able to extend the scissor lift a few inches.

VI. Conclusion

Overall, we were pleased with the way our prototype turned out. The articulation system worked well and was able to reliably track a light source automatically. The manual control system allowed for adjustments to the articulation system, while immediately returning to the automatic tracking method. As for the scissor lift, while it didn’t perform exactly the way we wanted it to, served as a valid proof-of-concept for this type of deployment design to be implemented with the LISA-T Solar array.

We wanted to thank you again for challenging us this year with the given problem, and for funding this project and our trip out to Huntsville. We hope that our prototype can help you create a flight-rated deployment, retraction and articulation system.

With best wishes from your product developers,

Dan Krollman
Alex Mueller

Appendix A: Circuit Schematic