Asterism: Motorized Astrophotography Mount

Team B1

Yuyi Shen
Electrical and Computer Engineering
Carnegie Mellon University
yuyis@andrew.cmu.edu

Kenny Ramos
Electrical and Computer Engineering
Carnegie Mellon University
kdramos@andrew.cmu.edu

Joy Gu
Electrical and Computer Engineering
Carnegie Mellon University
jagu@andrew.cmu.edu

***Abstract*—Avenues of automating and optimizing a motorized camera mount for astrophotography were explored using both computer simulation and a physical prototype, including the usage of a computer vision routine for the automatic tracking of astronomical objects and the effectiveness of software-based tangent error compensation as a means of simplifying mount construction. On average, it took about 250 frames (10 seconds, given a camera frame rate of 30 FPS) for error from the center of a simulated camera's field of view to converge across several simulations of the CV-based tracking routine, while a tracking error of 50.175 arcseconds/minute was attained across 30 minutes worth of stacked exposures of a known constellation with the mount compensating for the rotation of the Earth and tangent error.**

***Index Terms*—Astrophotography, Circuits, Computer vision, Motor control, Object detection, Object mapping, Object tracking, OpenCV, Optical flow, Raspberry Pi, Real-time, Software, Stepper motor, Robot**

I. Introduction

Astrophotography is a common pastime amongst photographers and amateur astronomers, and oftentimes involves the taking of long exposures of the night sky or of a specific astronomical object. This usually requires the usage of a motorized camera mount to compensate for the natural rotation of the night sky or the natural movement of a relatively close-by object. The type of mount most accessible to hobbyists (cost and complexity-wise) is known as a barn-door tracker, and typically consists of a set of hinged plates that are driven apart by a motorized threaded rod upon which a camera is mounted [1]. This basic setup only compensates for the natural rotation of the night sky (seen as star trails in uncompensated long exposures) and can be mounted atop a motorized pan and tilt setup to enable compensation for the movement of astronomical objects that move relative to the night sky (moon, planet, comet, etc.). Such devices often achieve tracking errors on the order of 19.75 arcseconds/minute [2] for low plate angles and up to 4 hours of operational time if a more mechanically complex design is used to compensate for the plates' inherent tangent error [2]. This operation time corresponds to a total power consumption of 17.5 W if the mount is powered by a 70 W-hr battery (as is typical for the capacity of motorized mount batteries [3]).

Due to the constraints induced by CMU's move to remote instruction this semester, our project consists of two components: a computer simulation of a computer vision routine's ability to use feedback from a simulated camera to pan and tilt said camera to track a specific object, and a physical prototype of a barn-door tracker mounted atop a motorized pan and tilt mechanism (as opposed to the fully integrated equatorial mount design that was described in the previous report). The computer vision routine was designed to hypothetically link a camera atop a motorized mount with the mount's motor controllers to track specific astronomical objects, while the barn-door tracker prototype was used to test the efficacy of software routines for the removal of tangent error (an inherent tracking error that appears in the simplest barn-door tracker designs). We aimed to attain a maximum tracking error of 19.75 arcseconds/minute with the physical prototype, and a maximal 5% positional error in the computer simulation (as calculated in terms of pixel distance/simulated object size in pixels). Because the physical prototype makes use of 3 times as many motors as a typical barn-door tracker due to the pan and tilt mechanism, we targeted a power consumption of 1.5 times the figure cited earlier (26.25 W). This specification assigns a typical operational endurance of 8 hours per motor [3] (assuming a 70 W-hr battery), and multiplies the resulting power consumption per motor figure by the total number of motors. Lastly, because of the large amount of power consumed in the physical prototype, we aimed to ensure that the quiescent operating temperature of all electronic components remained below 60 C as a safety measure. Physical contact with objects above this temperature can cause serious burns in 5 seconds or less [4], and keeping all components below it ensures that a careless operator's reflexes have plenty of time to kick in.

II. Design Requirements

The principle requirement for the computer-vision portion of the project relates to its ability to track an object in a simulated sky using a camera's pan and tilt controls, while the main requirement for the physical portion relates to its ability to compensate for the rotation of the night sky during a long exposure.

A. *Physical Prototype Sky-Tracking Accuracy*

Tracking the movement of the sky involves turning the camera along the polar axis at a rate near 4.178×10^{-3} degrees per second, one full rotation over the course of a

sidereal day, which is approximately 23 hours, 56 minutes, 4.0905 seconds. Note that this figure is equivalent to the rate of the Earth's rotation, which results in star trails that are 15 arcminutes long per minute of exposure time in a long exposure photograph of the night sky if it isn't compensated for.

Because proper statistics on the tracking errors incurred by barn-door trackers are effectively nonexistent due to the highly informal nature of astrophotography, a specification for the accuracy of the physical portion of the project had to be generated from a poster on an astronomy forum that had noted a tracking error of 1185 arc-seconds over an hour of operation [2], which corresponds to a velocity error of 19.75 arcseconds/minute. This level of error would result in the formation of 19.75 arcsecond long star trails for an exposure time of 1 minute. This numerical specification can then be compared against the angles of observed star trails taken with the camera mount, after normalization to the exposure time.

B. *CV Routine Object-Tracking Accuracy*

The computer vision-based section of the project involves implementing object tracking, for objects which do not move at the same rate as the rest of the sky. Because this section of the project was validated in simulation, the accuracy requirement is derived with a case study of a real object tracking camera mount:

The setup procedure for such a mount involves the taking of multiple long exposures within a reasonable amount of time in order to identify object movement based on the drift of point lights (stars) in a set of photographs. Furthermore, the option to track the movement of a single object through the same set of captures is given. Using EQ.1 it is found that most astral objects have an apparent size in the range of 1-30 arcseconds in the night sky.

$$\theta_{apparent} = 2\arctan\left(\frac{ObjectDiameter}{2\times ObjectDistance}\right) \quad (1)$$

This informs the requirement for the accuracy of object tracking. Due to the diagonal angle of view of 63.00° a typical lens[11] in a far field capture and an 18MP resolution dimensions of (5184x3456) this translates to 6230 diagonal pixels at a angular resolution of (63/6230) ~ 0.0101 of a degree per pixel, or ~36 arcseconds per pixel. For a more accessible lens[13] the listed angle of view is 89.9° and assuming the same 18MP resolution we find an arc resolution of 52 arcseconds/pixel. These are, of course, bigger than most of the point lights "should" be given their angular size. However, due to things like atmospheric scattering, apparent magnitude, and the discrete way in which photons are detected in DSLR camera sensors this increases their apparent size in a photograph. Given that, their apparent size still proves to be a good metric for understanding the overall path of the source of the light being captured as an error in the tracking of the source's center translates quite nicely to an error in the captured result.

Given all of this our target is to eliminate noticeable errors in object captures. For this we qualitatively found tolerable levels of error for typical captures using these lenses of both a nearby object (The Moon) and a far away object referenced before (Jupiter). We settled at a 5% total position error for The Moon and a 40% error for Jupiter. This would translate to a 1'42.5'' position error for The Moon over a given capture and an 11.6'' position error for Jupiter given their respective angular sizes.

One note is that these numbers mean different interpretations for different capture qualities/resolutions. That is to say, for large objects we expect our system to be accurate within 1'42.5'' and for smaller objects the metric of 11.6' seems adequate given experimentation.

Another final note is that it does not make sense as implemented to measure the accuracy of the object-tracking routine as a constant percentage or angle given that the error can vary over time. An aspect of object-tracking not considered at the time of the design report is the time to convergence of the object-tracking (e.g. of a PID controller or similar), which is explored further in the following section.

III. System Architecture

The physical portion of the project consists of a barn-door tracker mounted on a pan and tilt setup composed of a motorized lever arm and turn-table. The 2 NEMA 17 motors actuating the lever arm and turn-table are connected to a controlling Arduino through individual breadboard-based H-bridge drivers, while the unipolar stepper motor driving the barn-door tracker is connected to said Arduino through a ULN2803 Darlington transistor array being driven by an Arduino stepper motor driver library [16].

The computer vision section of the project takes in images from the simulation and generates pan and tilt information to allow a simulated camera to track a user-selected object. Had there been time, this module would have been integrated with a sky-tracking routine and a software driver for the H-bridge drivers controlling the mount's pan and tilt motors in a GUI through which the user would have been able to select the mode of operation, and input information so that the mount can either select an object to track, or turn sky-tracking on and off.

Originally, the three stepper motors would have been connected to a centralized motor controller board, which would then have interfaced with a Raspberry Pi running the CV routine through the GPIO pins. There would have also been a gyroscope connected to the Raspberry Pi to sense the position of an un-motorized axis. The latter approach was discarded after considering the excessive cost of such a large PCB, and the move to a fully motorized barn-door tracker-based design rendered the gyroscope unnecessary.

Later in the design process, an accident with the mount's power supply destroyed the Raspberry Pi with too little notice to order a replacement, making it impossible to use a Pi at all in our final design and forcing a switch to an Arduino. This

precluded the possibility of integrating the CV routine and GUI into the mount as well, requiring the usage of simulation to validate said software routine. Furthermore, the same incident destroyed the PCB that had been designed to drive the unipolar stepper motor responsible for actuating the mount's barn-door tracker, requiring a switchover to a generic ULN2803 driver chip and Arduino library from said PCB and a custom motor driver library we'd written for the Pi.

Fortunately, the onboard 5V SMPS that had been on the dead PCB survived the incident, and was ultimately recycled as a 5V power supply for the ULN2803 chip.

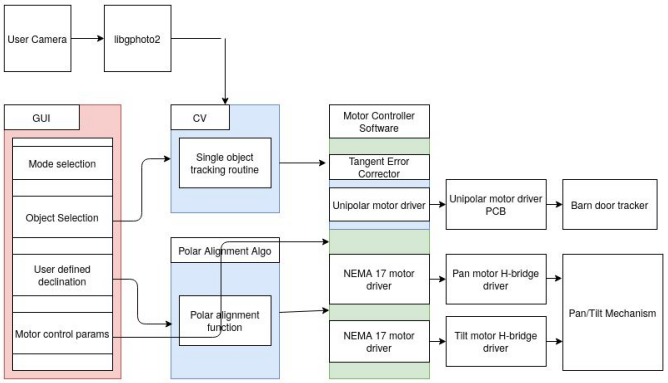
After the switch to a barn-door tracker-based design, the software running on the Raspberry Pi (see Fig. 1.a.) used step-counting to maintain information about the positions of the camera mount's axes. The GUI would have allowed the user to calibrate the software to the specific camera used. The image-processing aspect of the software would have polled the camera via libgphoto2 for images at a user selected interval and then processed it to calculate inputs to the motor controllers.

A semi-automatic polar axis alignment mode had also been planned as a part of the GUI in which the system would have instructed the user through a set of calibrating steps that involve pointing the mount's finder scope to astronomical objects that are both user known and stored inside a database by our system. The user would have chosen to align with two of these predefined objects and, after calibration, the system will have approximate knowledge of the user's zenith position relative to the polar axis. This would then have allowed the system to align the axis of the "compensator" to the polar axis. Unfortunately, complications with the integration of the object-tracking mode into the GUI precluded the inclusion of this feature.

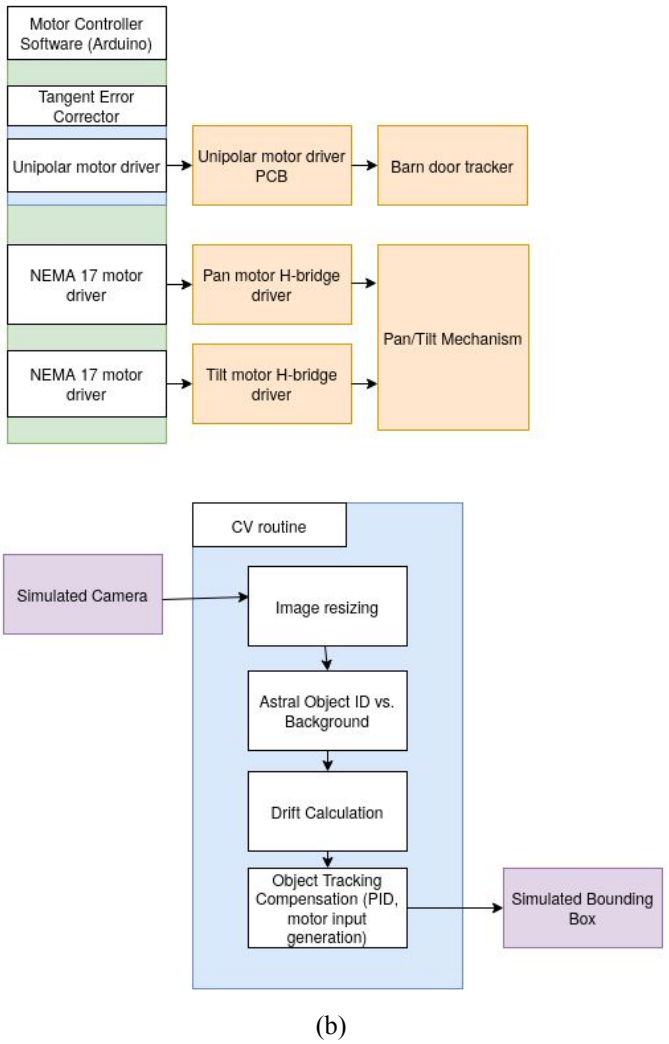
After the destruction of the Raspberry Pi, the computer vision algorithm of the object tracking mode that had been planned for the GUI was modeled in simulation to evaluate its functionality instead of being integrated into the full design. It processes an input image for the location of the tracked object and performs calculations based on the previous locations of the object to determine inputs to the motor controller in order to compensate for the motion of the object.

Similarly, the sky-tracking mode of the Raspberry Pi code had to be replaced by an equivalent Arduino script due to the Pi's non-functionality. Fortunately, it was not necessary to test this section of the project in computer simulation.

Section V goes into system architecture in further detail.



(a)



(b)

Fig. 1. System picture. (a) Full system prior to Pi destruction. (b) Final system diagram.

IV. Design Trade Studies

When designing the system, we considered a number of different configurations for the overall system and its subsystems in order to meet the design requirements specified in Section II.

A. *Motor Selection*

When considering motorization of the mount, it was necessary to decide on the type of motor to use and whether different motors should be used for different axes.

Initially, a NEMA 17 bipolar stepper motor with a step angle of 0.9 degrees, with a rated voltage of 3V, a rated current of 1.7A/phase, and a holding torque of 48 N-cm, was chosen. It was selected for its high resolution and reasonable cost per motor. In addition, with a stepper motor, a motor encoder sensor for feedback is unnecessary, because motor steps can be counted to approximate position. This motor also satisfied the torque requirements for each motor axis by a decent safety margin.

The torque was calculated for the tilt motor (which needs to lift the barn-door tracker (modeled as a shaft), compensator motor, and camera):

$I_{comp} \approx 0.3\ kg \times (0.02\ m)^2 = 0.00012\ kg \cdot m^2$

$I_{shaft} = \frac{1}{3} \times 0.234\ kg\ \times (0.305\ m)^2 = 0.0072\ kg \cdot m^2$

$I_{cam} = 2 \times 1.9\ kg\ \times (0.305\ m)^2 = 0.353\ kg \cdot m^2$

$I_{tot} = 0.36\ kg \cdot m^2$

$\omega = \frac{180\ \text{deg}}{10\ sec}\ [arbitrary]$

$\rightarrow \alpha = 0.0628\ rad/sec^2$

$\tau = 2.3$ N-cm

Lastly, the torque requirement for the mount's pan motor (placed as a turntable under the tilt motor) was found:

$I_{tilt\ motor} = 0.5 \times 0.3\ kg \times (0.02\ m)^2 = 6 \times 10^{-5}\ kg \cdot m^2$

$I_{tot} = 0.36\ kg \cdot m^2$

$\omega = \frac{180\ \text{deg}}{10\ sec} \rightarrow \tau = 2.3$ N-cm

To simplify driver circuitry, the same model of motor is to be used for both axes of the mount's pan and tilt assembly. Note that these calculations were initially made for an equatorial mount of equal size, and carried over into our final barn-door tracker-based design. Prior to making these torque calculations, a large set of motor types was considered qualitatively before settling on bipolar hybrid stepper motors.

TABLE I. Motor Types and Considerations

Motor Type	Pros	Cons
Brushed DC motor	Low driver circuit complexity, decent torque. Permanent magnet DC motors are very cheap.	Difficult to model transfer function between motor power and speed (might require a motor encoder sensor for feedback), datasheets usually don't specify torque vs. current curve. High motor speeds mean that a higher-ratio gearbox is required (expensive).
Brushless DC motor	Built-in speed control and tachometer sensor, decent torque.	Decently sized ones are on the order of 80 dollars (Digikey). There is one model that costs ~20 dollars, but it is low in stock.
Synchronous AC motor	Rotates at speed of powerline frequency, accuracy is only dependent on frequency control	Anything larger than a vibration motor is on the order of ~$50.
Permanent magnet stepper motor	Produces detent torque (remains stationary when undriven). Gives higher torque than variable reluctance motors at limited speeds. Reasonable prices.	Moves in steps (usually 100 per rev.) Requires a polyphase driver circuit.
Variable reluctance stepper motor	Less torque drop-off at higher speeds.	No detent torque, notorious for noise.
Hybrid stepper motor	Highest resolution of stepper motor types, cheap on Pololu, can be half-stepped or microstepped for further resolution benefits.	Usually more expensive than the other stepper motor types. Bipolar motor drivers are effectively H-bridges, and are harder to implement, while unipolar motor drivers require more wiring.

Fig. 2. The types of motors considered and characteristics of each [14,15].

For the polar aligned axis used for compensation of the sky's movement in the barn-door tracker section of the mount we chose a 28BYJ-48 stepper motor due to the lower torque requirement of the barn door tracker design (discussed later)

and the lighter and higher step count the unipolar stepper motor utilizes.

The power consumption of these motors totals to 21.325 W, with the 28BYJ-48 stepper motor taking in 0.925 W and the two NEMA 17 motors taking in 10.2 W each. Since an Arduino uses roughly 0.29 W [17], this translates to a total system power consumption of 21.6 W, which is well below the power specification of 26.25 W.

B. *Gyroscope Selection*

Although we considered using a gyroscope in the design phase of this project we decided that with the changes faced after the start of remote instruction made the gyroscope unnecessary. More specifically, the design changed over to one in which all mechanical degrees of freedom were fully motorized by stepper motors, making it possible to simply count motor steps to keep track of motor shaft angles.

C. *Custom Motor Controllers vs. Store-bought*

The large NEMA 17 stepper motor we selected is rated for 3V and a continuous current of 1.7A/phase. To get an idea of the specifications of commercially available motor drivers, we inspected Pololu's online catalog of stepper motor drivers. Only the TB67S249FTG, AMIS-30543, TB67S128FTG, and DRV8711 were listed as capable of supplying 1.7A of continuous current per phase without additional cooling, and the one requiring the lowest supply voltage still needs at least 6V to operate. Therefore, we found it necessary to design our own stepper motor driver for a 3V supply voltage in order to support the specific motor we selected.

We also examined the possibility of procuring a commercially available motor driver for the 28BYJ-48 motor we selected. Because the typical driver for this particular motor (ULN2003) was not available on Sparkfun, we considered using the ULN2803 Darlington array as a unipolar motor driver. However, due to the relatively poor performance of bipolar devices as switches, we elected to simply assemble an array of discrete MOSFETs on a PCB as a driver. As a contingency for the possibility of this design failing, we also ordered a handful of ULN2803s. This turned out to be useful when an accident with a 12V battery destroyed the majority of the PCB that we had assembled.

Later on, these decisions proved to be fortuitous. The hottest electrical component in the mount (a 3V 9W buck converter for powering the NEMA 17 motors) ended up reaching a temperature of 51.3 C, which is well below the 60 C safety specification cited earlier in this paper. Meanwhile, the MOSFETs used in the custom driver circuits that were designed for the NEMA 17 bipolar motors reached a mild 36 C during operation, while the PCB driver transistors reached 31.6 C. When the PCB driver had to be switched out for the ULN2803 after the destruction of the Raspberry Pi, the ULN2803 devices reached a slightly higher 34.7 C. Thus, the usage of custom-designed circuits was beneficial for the thermal performance of the overall system.

D. *On-Board Computer Selection*

Microcontrollers are unlikely to be able to meet the memory and processing power requirements, and a full laptop computer would be cumbersome to attach to the unit, so single-board computers received more consideration. The main candidates for an on-board computer were the Raspberry Pi 3B+ (or a similar model) and the BeagleBone Black development board. Both can run a full Linux kernel and are compatible with any candidate software libraries. Both are powerful enough for basic computer vision applications. Both can use SPI to communicate with peripheral devices and have USB ports for peripherals.

The BeagleBone Black, at $62.38 from Digi-Key[5], is slightly more expensive than the Raspberry Pi 3B+, which is provided at no cost by the course staff. It has 65 GPIO pins, more than the Pi, which has 26 pins. Both have enough pins for the motor controller board, which requires 4 pins per motor and 4 pins for SPI for the gyroscope, which comes out to 16 pins total. Unlike the Raspberry Pi, it lacks a GPU, which would be useful for accelerating image processing. Due to these factors, the Raspberry Pi seems more suitable for the project.

In the end, due to the destruction of the Raspberry Pi towards the end of the semester in an accident with a battery lead and the lack of time to acquire a replacement, the Pi was replaced with an Arduino to get the physical component of the project ready to demo while the CV routine that was originally to be demonstrated on the Pi was validated in a computer simulation.

E. *Software Tools and Algorithms*

For the project, in order to implement object-tracking, it's necessary to be able to identify an object across several images and to be able to identify the movement of that object from frame to frame.

Some libraries for computer vision applications include OpenCV, TensorFlow, CCV (unfortunately known to have issues compiling on the Pi)[7], SOD (an embedded computer vision library with licensing fees for trained models).

Known algorithms for object detection include YOLO[6], Faster R-CNN object detection, and SSD object detection. SSD and YOLO are single-shot object detectors and are typically faster than Faster R-CNN object detection. YOLO is known to struggle with small objects within an image, so it may not be suitable for our application, tracking stars. Object tracking involves taking in an initial object detection and tracking the object as it moves across video frames. It's not necessary that the object tracking algorithm we use be robust to occlusion, because we intend for the object to remain within frame, and the objects we intend to track will be moving at slow rates.

Ultimately, object detection was not necessary for this project, but object tracking algorithms were. OpenCV

provides several tracking algorithms, including the MedianFlow tracker, Kernelized Correlation Filters, MIL, and the Boosting tracker. Given the application at hand it was important to respond to object-tracking failure reliably and important for the tracker to update quickly. MedianFlow and Kernelized Correlation Filters met this criteria.

Both OpenCV and TensorFlow have pre-trained models available for free that implement such algorithms. It may be necessary to train our own computer vision models, but this is not preferred, since it can be time-consuming and difficult. We are likely to use OpenCV.

In order to interface with a camera, we plan to use libgphoto2, a library designed to allow access to a digital camera by external programs. It is compatible with many cameras, including some smartphones. Downsides to using libgphoto2 and a user-provided camera include latency of image transfer. An alternative could be to use a Raspberry Pi camera module for visual feedback. This would reduce the latency of image transfer and make calibration of object tracking easier given a consistent position on the mount and a consistent zoom level. However, the Raspberry Pi camera module may have insufficient image quality compared to a user-provided camera, and it may be difficult to consistently ensure that its position and direction corresponds with that of the user-provided camera.

F. *Mechanical Considerations*

Due to Carnegie Mellon University's switch to remote instruction and the limitations that come along with it we decided to adjust our design to a barn door tracker while maintaining three degrees of freedom. We planned for three motorized axes: one for the sky-tracking compensator to turn along, and two to align the polar axis or assist with object tracking.

Our choice of using a barn door tracker required an alternative route for gearing down to the required rate. Pairing a #20 pitch carriage bolt and a 4096-step stepper motor our project could utilize either a constant or function defined rate in order to adjust for the movement of the stars in the night sky. Due to the nature of a barn door tracker system, it is not necessary to account for holding torque for parts driven by the polar aligned axis. Furthermore, the motor is able to provide increased torque due to the lower rate at which the motor is required to rotate (~0.5-1.0 rpm).

There is some inherent error in using constant linear actuation due to the non-constant rate of change in angle between two sides of a triangle when the third side changes in length. This referred to as the "tangent error" and it can be derived given in the following formula:

k = motor rpm
sidereal rate = 2pi/86164s
arm_length = length between polar axis and the bolt that linearly actuates the upper arm of a barn door tracker = 250mm
thread_pitch = pitch of the linear actuating bolt = 1.27mm/rotation

$$E(t) = (\text{Sidereal Rate}) * t - 2 * \arcsin(k * t/60s * (\text{thread\_pitch}) / (2 * \text{arm\_length}))$$

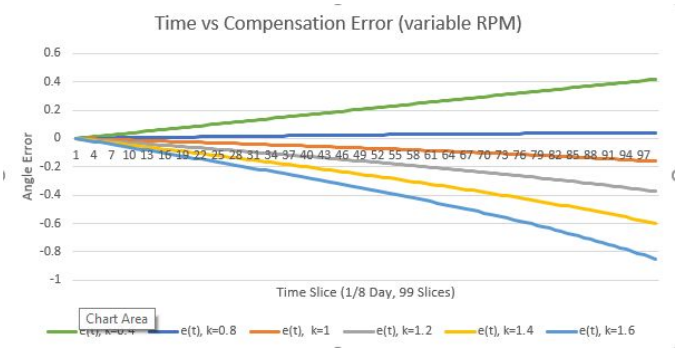


Fig. 3. Angle Error from real given different k values.

Fig. 4. Barn Door Design

V. System Description

The original design for our system had two major parts, a circuits subsystem which included a motor controller board (with the accompanying power distribution setup), and a software subsystem which included a computer vision component and an embedded component. For testing and verification, we initially planned to build a testing rig to simulate the motion of the sky.

As budget and time constraints became increasingly clear, a significant proportion of the original system design had to be abandoned. The construction of the testing rig had to be cancelled in favor of testing the system directly by taking long exposures of the night sky, while the motor controller board design had to be split into a PCB for driving our camera mount's unipolar 28BYJ-48 motor and breadboard-based H-bridges for driving the larger NEMA 17 motors. This had to be further revised after the accidental destruction of the Raspberry Pi, with the partially damaged PCB being swapped out for a discrete ULN2803 chip and the software component being changed over to an object tracking CV routine that could be demonstrated in simulation and a manual sky tracking routine written for the Arduino that ended up replacing the Pi to get the physical component of the project ready to demo.

A. *NEMA 17 and 28BYJ-48 Motor Controllers*

Originally, the mount was to make use of three individual motor controller boards with logic buffered inputs, each of which would be supported by a 8V power supply and a high power 3V power supply. These were meant to drive three NEMA 17 motors in the original system design, which was structured as an equatorial mount instead of the barn-door tracker-based design we ended up with. After the change to a

barn-door tracker design with 2 NEMA 17 motors, it was decided to construct H-bridge drivers on breadboards after reconsidering the power requirements of said motors and the potential need to quickly slot in replacement components. As a result, each SMD component in the original circuit had to be replaced by an equivalent or superior thru-hole device. In total, 4 GPIO pins were required to control each motor.

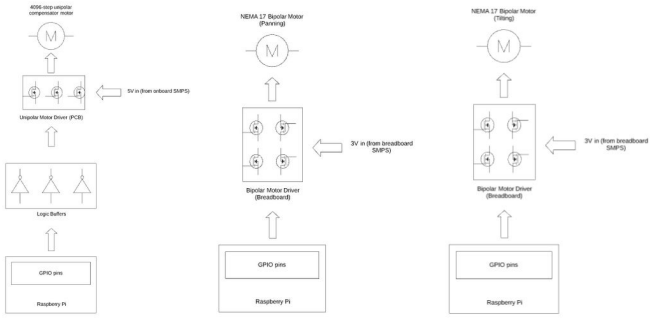


Fig. 5. Block diagram for all motor controllers. Note that the Raspberry Pi ended up being replaced by an Arduino after its destruction.

The full schematic of a single H-bridge (2 per motor controller board) is shown below:

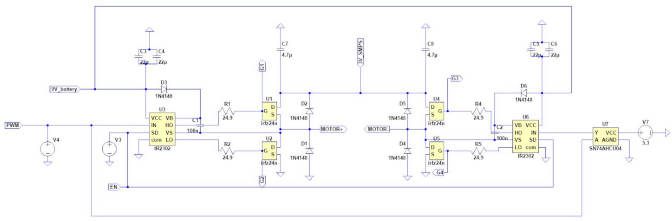


Fig. 6. H-bridge circuit schematic

SPICE simulation with the above schematic verifies that the circuit successfully switches current in both directions through a motor coil:

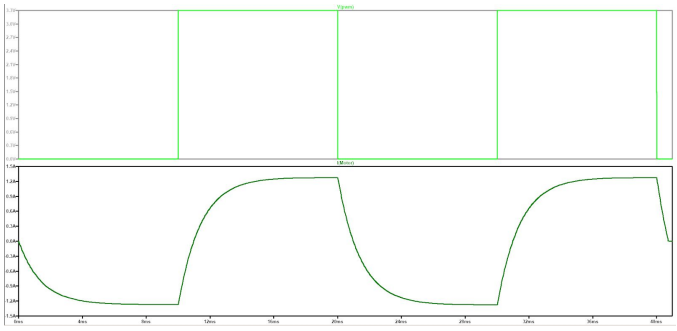


Fig. 7. Current switching back and forth in a motor coil.

Furthermore, SPICE simulation for the voltage at the motor coil terminals indicates that it will remain within 1 diode drop of the power rails despite the development of sharp spikes whenever the H-bridge switches, meaning that the flyback diodes included for the protection of the H-bridge MOSFETs are functioning as planned.

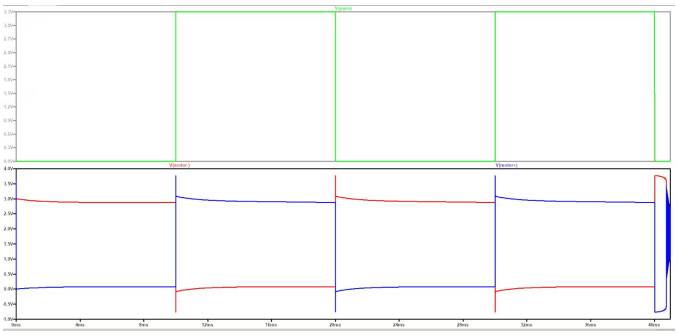


Fig. 8. Voltage spikes at motor terminals.

Due to the relatively low power consumption of the 28BYJ-48 motor, it was initially decided to design a PCB-based driver consisting of an array of large MOSFETs:

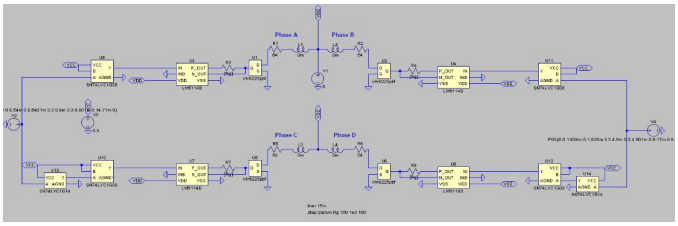


Fig. 9. 28BYJ-48 driver circuit schematic

The ability of the circuit to drive current through consecutive motor phases was verified in SPICE:

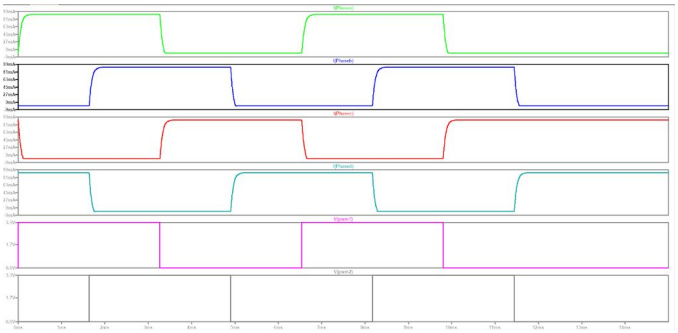


Fig. 10. Simulation of Current through Individual Phases of 28BYJ-48 Motor (top 4 waveforms).

However, after said PCB was partially damaged in the same incident that destroyed the Raspberry Pi, it was decided that it would be too much trouble to repair a board with so many SMD components and the PCB was replaced with an electrically equivalent (if less efficient) ULN2803 chip.

B. *Power Distribution*

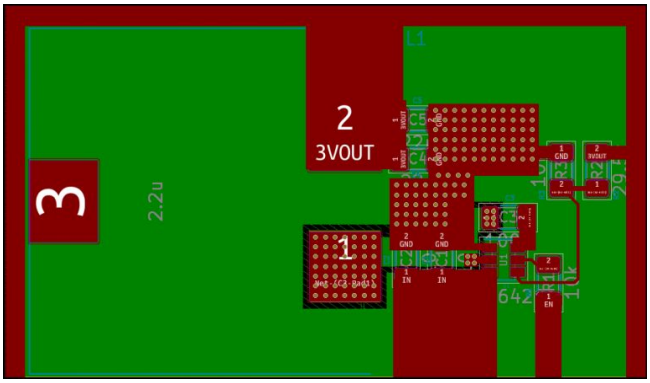


Fig. 11. PCB layout for 3V 4A power supplies.

As a part of the original design that called for PCB-based NEMA 17 motor drivers, a 3V 4A buck converter was designed and laid out using a TPS564201DDCR buck converter controller IC early in the semester. Although an equivalent to this design using through-hole components was eventually substituted for it in order to power said NEMA 17 motor drivers, the same schematic and layout were also repurposed through the modification of feedback resistor values to supply 5V for the 28BYJ-48 motor used in the system's barn-door tracker. A separate 8V gate driver power supply was found to be unnecessary, as said drivers could simply be switched out for 9V drivers that could be directly powered from a 9V battery. Considerations were made for powering logic buffers from a Raspberry Pi and the originally proposed test rig's laser diodes, but were found to be unnecessary. The Pi was ultimately swapped out for an Arduino, which could directly power said logic buffers with 5V, while the test rig became infeasible to construct in the context of the time constraints later in the semester.

C. *Software Subsystems*

As described earlier in this document, our original plan was to run a computer vision algorithm on a Raspberry Pi which would process images from a user-provided camera and calculate the corresponding inputs to the motor controller board given a user-selected function.

For testing the software subsystem, our main concern dealt with object tracking accuracy and our ability to accommodate multiple capture types, resolutions, and rates. The most pressing issue was creating a subsystem test for our implementation of OpenCV object tracking. For this we assumed that our sample rate for object detection given by the user is high enough that star and object movement is roughly linear. Using this assumption we are able to generate tests by simply applying linear offsets to captures of desired and scoped for objects and constellations (e.g. Jupiter, Ursa Major) and assessing the performance of our object drift calculations by comparing "inverse" calculations with our generating parameters. Furthermore, due to the large error from sample to sample that will likely be seen for very high sample rates (e.g. 1Hz) we want to explore samples that are more temporally separate, such as two captures that are ~60s apart. For nearby objects (i.e. "large", e.g. The Moon) we expect about half of our aforementioned 5% "large object error" to be attributed to this portion of our system due to the larger margin of error that can be seen when trying to find centers of large objects in a photograph. For faraway objects (e.g. Jupiter) we expect this error to decrease the more time we spend gathering sequential samples. As a result we expect about ⅓ of our 40% small object error (or less) to come from this subsystem. Instead of generating images with linear offsets, we used an OpenGL simulation which would rotate a simulated sky and independent objects. We planned to add noise to the simulation to further approximate realistic operating conditions.

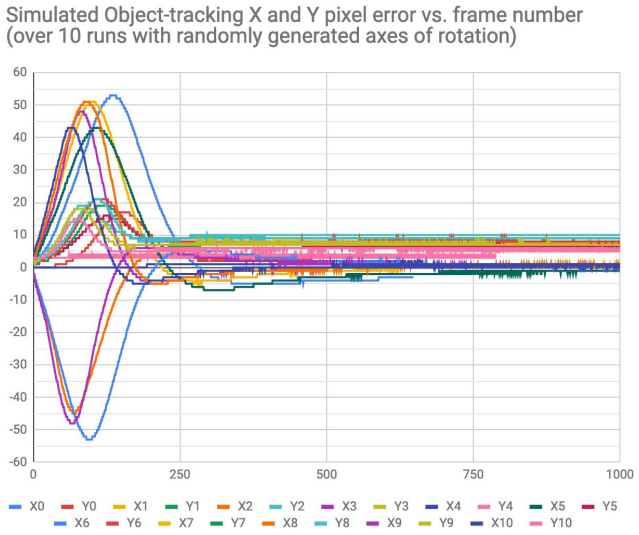


Fig. 12. Simulated Object-tracking X and Y pixel errors over 1000 frames. Frame number is used as a proxy for time. These numbers were collected from 10 simulated runs with randomly generated axes of rotation for the tracked object.

Object-tracking was implemented to track objects in an OpenGL simulation. The object-tracking routine used OpenCV to track an object across frames from the simulated camera and used a naive PID control loop with X and Y pixel error as inputs to calculate the required motor speeds to compensate for the object's movement. On average, error converged at around 250 frames. Given different PID parameters, convergence could potentially be achieved in fewer frames. The correspondence between frame number and time is not clear but would depend on the expected speed of the object tracked and the sample rate of the camera. There is some jitter past the convergence point, and convergence took longer for some runs than others.

For accommodating different resolutions/capture sizes we

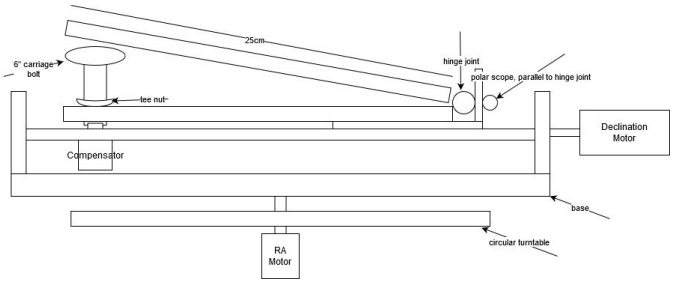
planned to use an object-detection mode calibration step where the user first calibrates their camera to a desired capture spec. Then the system would have taken a few samples with known offsets (offsets that will be known due to a previous star-tracking calibration step) and stored a 1:1 correlation between resolution space and real space. Under the assumption that camera specs do not change or are recalibrated upon change, this correlation is used to map desired resolution space offset detected by CV to actual space offset in the actuators. This will then be tested for rigor given different generated samples and output correlations.

Due to time and equipment constraints late in the semester, a significant portion of the original plan for the software component of this project did not come to fruition. The accidental destruction of the Pi by a misplaced battery lead effectively killed our ability to integrate the software into the mount, rendering the above described calibration step moot. In the end, the above mentioned computer vision algorithm was integrated into a computer simulation as a substitute for a full demonstration.

D. *Mechanical Subsystems*

The mount was constructed with a mix of standard 5/8th's plank wood (for the compensating arm) and mdf for the lower portion (alignment). Due to this we were able to utilize simple turntable bearings from McMaster-Carr as previously discussed in design proposals. The upper polar-aligned axis gearing ratio was accomplished with the aforementioned high-step ratio stepper motor and a bolt-tee-nut combo that simulates a worm gear. The mount was constructed using several hand tools and an electric drill due to remote instruction limitations. Supplies that were not previously planned for, (such as a hinge joint connecting the two compensator arms) common household items and parts were used due to latency, practical, and budgetary constraints [e.g. a door hinge was used].

(a)



(b)

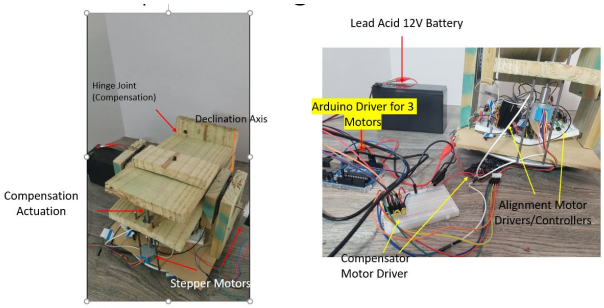


Fig. 13. CAD drawings. (a) Mount Diagram b) Mount picture

.

E. *Testing*

Due to the fickle and often inconvenient nature of stargazing a testing rig was planned. It would have consisted of an array of 12 laser diodes fixed in a foam ball, which we can turn on a motorized turntable or by hand. The motorized rig will then project these lights onto a surface (either walls or large sphere) and rotate at roughly a sidereal or faster rate. To this projection an additional laser diode can be used to simulate an object that moves at a different rate and direction to help test whole system object tracking. However, timing constraints lead to this portion being discarded.

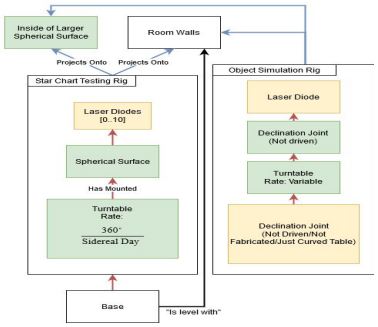


Fig. 14. Block diagram for testing rig. (LEGEND: Green: Fabricated, Yellow: Purchased, Blue Arrow: Projects onto, Red Arrow: Parent-to-child mechanical relation)

Instead, we decided to do field testing. One of the challenges was finding available spaces to conduct said tests given the widespread closures and tightened security due to the recent epidemic.

The test was conducted by taking a series of 31 evenly spaced captures of the night sky using the camera and our mount in each mode, for a total of 93 captures. These captures would then be stacked and main star clusters (in this case those of the constellation Ursa Major) would be measured in both offset from origin point and overall spread from geometric center.

Furthermore, for captures that were compensated, the minor axis of the smallest ellipse containing the star trail would be

measured.

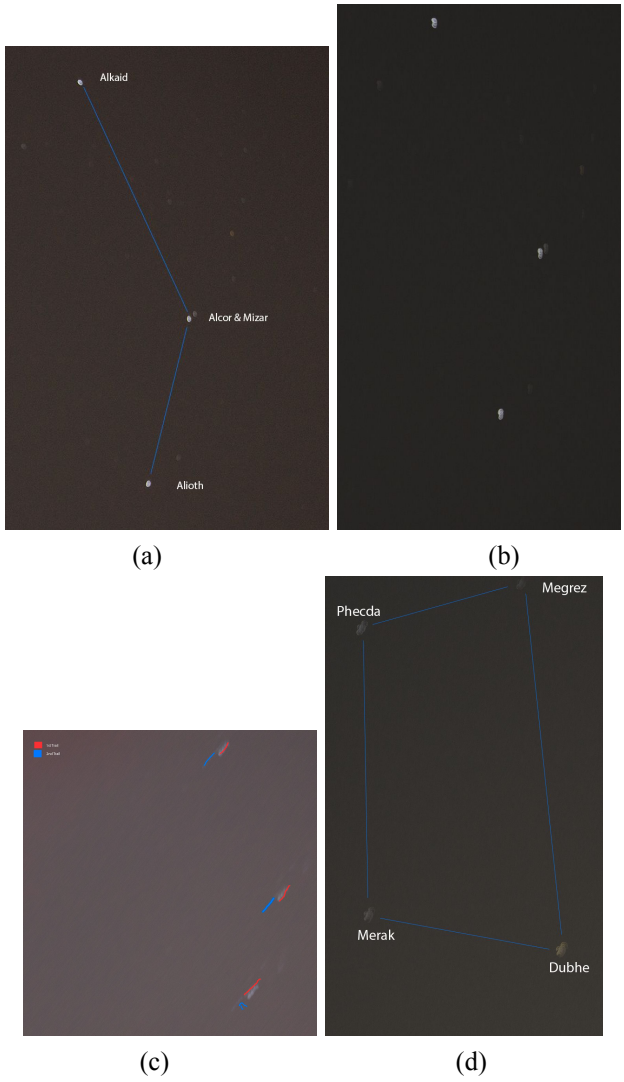


Fig. 15. (a) Annotated tail of Ursa Major (Compensated, No-Tangent Compensation) (b) Compensated star trail (No-Tangent Comp) (c) Annotated Uncompensated Star Trail (d) Tangent Compensated Star Trail "Bear Body" Ursa major ("dipper" of big dipper)

Our captures were taken with camera settings: 15 second capture, f4 aperture size, iso800 for near horizon captures and iso1600 for near zenith captures. These iso and capture time choices were mostly made out of necessity due to the light pollution present in available spots during the quarantine. A longer exposure would have revealed errors more accurately and better capture conditions would have reduced the noise floor.

Our tests resulted in an error rate of 29.4 arcseconds/minute with sky movement compensation without tangent error correction and 50.2 arcseconds/minute with tangent error correction. For our uncompensated capture we measured an error rate of 887.3 arcseconds/minute which is within 3% of the expected error for non-compensated captures (905.28 arcseconds/minute). We believe the higher error seen in the tangent compensation is mostly due to the noise floor inherent in the measurements given the aforementioned limitations being larger than the benefits of tangent compensation.

VI. Project Management

A. *Schedule*

The main milestones for the project are completion of circuit layout, construction and assembly of the mount and gearing, prototyping and testing of the computer vision software for object tracking, and integration. It is expected that construction and assembly of the mount, prototyping of the computer vision software, and fabrication of mount circuitry will be completed by the end of March, leaving April for more testing and integration.

Refer to the end of this document for a full Gantt chart.

B. *Team Member Responsibilities*

Yuyi is responsible for designing the motor controller circuits and power supplies. This consisted of laying out a PCB for the 28BYJ-48 motor driver, designing the circuits for the breadboard-based NEMA 17 motor drivers, and designing the circuits for the SMPSs meant to power them. Yuyi also advised Kenny on their assembly.

Kenny is responsible for assembling the various mount components into a complete system, testing the resultant construct, and mechanical design of the mount. He also worked on the polar alignment function of the mount.

Joy is responsible for design and construction of the mount, prototyping of the object mapping software, and integration of the object mapping software with the mount movement.

It's anticipated that some tasks will be shared between members and that some task divisions will change over time.

C. *Budget*

Refer to the end of this document for a full parts list.

D. *Risk Management*

Some notable risks for our project deal with failure to implement certain subsystems effectively or in a timely manner.

One risk we faced is a failure to implement a working motor controller. This could have happened due to several factors including stress and voltage spike test failures, and controller-driver interfacing errors. A solution for this subsystem failure would have been to forgo implementing our own motor controller. Purchasing a driver controller module as a replacement would have posed it's own challenges as power distribution design would have to be altered to accommodate for the new module. We also investigated means of debugging the motor controller upon arrival, in case there wasn't time to order a pre-packaged motor controller in the face of coronavirus-induced shipping delays. This took the form of including means of monitoring the motor controller

within the PCB (test points, current sensing resistors, etc.) and investigating the usage of an Arduino as a makeshift oscilloscope for interfacing to these means (now that an actual oscilloscope is no longer available). Such a failure actually occurred in the last few days of the semester, when a misplaced battery lead blew up a portion of a motor controller. Luckily, a replacement for said motor controller had been ordered as a contingency and was slotted in.

In our original risk management plan, we mentioned a contingency for the failure of our project's mechanical compensator in the form of a switch to a mechanically simple barn-door tracker-based compensator. Owing to the unavailability of CMU fabrication facilities and thus the inability to build our original design, this contingency was in fact implemented.

A third subsystem that could have suffered failures is the computer vision software. If object-tracking is inaccurate, slow, or buggy, or if the latency of image transfer between the user-provided camera and the Raspberry Pi is too high to allow for real-time correction, real-time object-tracking would not have been feasible. We could have instead implemented blind tracking based on known information about celestial objects or given input from the user. We could also have repurposed the computer vision for drift alignment for polar alignment. Fortunately, no such issues manifested themselves over the course of the project, and the latency of image transfer between the Pi and the camera became a nonissue after the accidental destruction of the Pi by a misplaced battery lead.

One major risk that manifested in the move to remote work is the increased difficulty of debugging the software components of the project. The group member primarily responsible for our project's computer vision routine is located in Pittsburgh, while the group member performing the integration of the mount's hardware components is located in Miami, Florida. To alleviate the distance, we investigated the possibility of setting up a ssh server on the camera mount's Raspberry Pi so that remote debugging is possible for the person located in Pittsburgh. The principal challenge of this is making sure that the ssh server doesn't compromise the security of the Miami-based group member's network.

Fortunately, there were few risks to the successful integration of the mount's hardware. The team member in charge of physical assembly was familiar with both the mechanical and electrical sides of the project, and assured the group that he was in possession of the equipment necessary to accomplish his tasks. This indeed turned out to be the case.

One major risk that we failed to account for was the possible destruction of the Raspberry Pi, the centerpiece of our project. The ease with which it could be damaged had not occurred to us, and the accidental destruction of the Pi late in the semester caught us flatfooted. Luckily, some frantic parts swapping with an Arduino meant that we could still manage a partial demonstration of our project, at the cost of changing the computer vision portion's demonstration to a computer simulation.

VII. Related Work

Some similar products have been designed when it comes to polar alignment. For example these two patents[8][9] detail an implementation for a product that uses positioning protocols to derive the user's location and/or altitude-azimuth in respect to the polar axis, in effect allowing it to point the system to stars and constellations inside of a manufacturer defined controller. However, there is no mention of using visual feedback to account for drift or the possibility of tracking objects outside of the manufacturer defined astronomical object database which are both defining parts of our system.

Another patent exists that describes the use of an intelligent motor controller system for a telescope mount[10]. This patent seems to be often used in conjunction with the aforementioned two and it details the use of optical encoders and servos in order to more accurately position a user's telescope. Furthermore, the patent discusses the option to use a brushless mount for communication between the top and bottom of the system in order to eliminate problems that might arise from wires wrapping around the mount as it rotates. These two are both distinct from our system in that we employ the use of stepper motors and a gyroscope in order to derive and affect position, rather than encoders and servos. Furthermore, our system does not scope for the possibility of a full 360° rotation, instead limiting it to the polar-aligned compensator to a range of ~120° and a limited single rotation turntable for the bottom.

VIII. Summary

Although our design met the power consumption and operating temperature specifications by a reasonable margin, we were unable to achieve the 19.75 arcsecond/minute tracking error specification. Indeed, the minimum tracking error we were able to attain was a slightly higher 29.4 arcseconds/minute, without software compensation for tangent error. Applying said compensation resulted in a surprisingly higher tracking error of 50.2 arcseconds/minute. This implies that the inability of our design to meet this specification does not stem from the intrinsic tangent error of our project's mechanical portion, but from jitter in the mount's motors or external vibrations. Therefore, our design's performance is likely limited by its ability to dampen extraneous motions that are transmitted to the camera. Given more time, we could have attempted to place the mount on some form of vibration dampening material (foam) to mitigate this.

A. *Future Work*

We do not plan to continue working on this project.

B. *Lessons Learned*

The move to remote instruction in the middle of the

semester effectively threw a spanner in the works. Assembly tasks had to be lumped upon a single team member to avoid having to mail partially completed project subsystems halfway across the country, and shipping delays intensified. Furthermore, the distance between team members became excessive, making it difficult to communicate regularly. Unresponsiveness led to a sense of uncertainty, which caused stress. Some team members would drop out of contact for days at a time in the last weeks of the semester. Phone calls were occasionally effective, but were inconvenient on both ends of the conversation. Last of all, the high shipping delays meant that replacement parts could take an excessive amount of time to arrive. The lack of a backup Raspberry Pi meant that we were caught off guard when the original was destroyed in an accident with a battery lead, and some frantic last-minute changes had to be made to swap out the Pi for the only other available microcontroller, an Arduino.

IX. References

[1] Lodriguss, Jerry. "Constructing a Barn-Door Tracker." *Beginner's Guide to DSLR Astrophotography*, Jerry Lodriguss, Sept. 2009, www.astropix.com/bgda/sample2/sample2.html.

[2] "Barndoor Tracker Error Estimates." *Cloudy Nights*, Astronomics, 29 Oct. 2011, www.cloudynights.com/topic/348662-barndoor-tracker-error-estimates/.

[3] "The Ultimate Guide to Powering Your Telescope: Behind the Scenes with Celestron PowerTank and PowerTank Lithium Family of Products." *Celestron - Telescopes, Telescope Accessories, Outdoor and Scientific Products*, Celestron, 4 Oct. 2019, www.celestron.com/blogs/news/the-ultimate-guide-to-powering-your-telescope-behind-the-scenes-with-celestron-powertank-and-powertank-lithium-family-of-products.

[4] "American Burn Association Scald Injury Prevention: Educator's Guide." *American Burn Association*, United States Fire Administration Federal Emergency Management Agency, Apr. 2017, ameriburn.org/wp-content/uploads/2017/04/scaldinjuryeducatorsguide.pdf.

[5] "BBB01-SC-505-ND." *Digi-Key Electronics.*

[6] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement." arXiv, 2018.

[7] "Does not compile correctly... - Issue #75 - liuliu/ccv." Github Issues. https://github.com/liuliu/ccv/issues/75

[8] Baun, Kenneth W., et al. *Fully Automated Telescope System with Distributed Intelligence*. US6304376B1, 16 Oct. 2001, https://patents.google.com/patent/US6304376B1/en.

[9] Baun, Kenneth W., and John E. Hoot. *Fully Automated Telescope System with Distributed Intelligence*. US6392799B1, 21 May 2002, https://patents.google.com/patent/US6392799B1/en.

[10] Baun, Kenneth W., et al. *Telescope System Having an Intelligent Motor Controller*. US6563636B1, 13 May 2003, https://patents.google.com/patent/US6563636B1/en.

[11] Canon. *Canon 51JV-IcjWBL Lens Specifications*. https://images-na.ssl-images-amazon.com/images/I/51JV-icjWBL.pdf.

[12] Beish, Jeff. "Design a German Equatorial Mount." *The Association of Lunar and Planetary Observers*, ALPO, www.alpo-astronomy.org/jbeish/Equ_Mount.pdf.

[13] *Canon B1FDpTkTjKS User Manual.Pdf*. https://images-na.ssl-images-amazon.com/images/I/B1FDpTkTjKS.pdf. Accessed 2 Mar. 2020.

[14] Collins, Danielle. "Stepper Motors: Permanent Magnet, Variable Reluctance, and Hybrid." *Linear Motion Tips*, 26 Apr. 2018, www.linearmotiontips.com/stepper-motors-differences-between-permanent-magnet-variable-reluctance-and-hybrid-types/.

[15] Williams, John. "Types of Electric Motors." EE 410/510 - Electromechanical Systems, The University of Alabama in Huntsville, www.ece.uah.edu/courses/material/EE410-Wms2/Electric%20motors.pdf.

[16] Tyler Henry, CheapStepper v0.2, (2016), GitHub repository, https://github.com/tyhenry/CheapStepper

[17] Mitchell, Alan. "Operating an Arduino for a Year from Batteries ." *Analysis North Blog*, 2 Oct. 2011, analysisnorth.com/articles/arduino-for-a-year.html.

TASK TITLE	TASK OWNER	START DATE	DUE DATE	Week 4 2/3	5 2/10	6 2/17	7 2/24	8 3/2	SB 3/9	9 3/16	10 3/23	11 3/30	12 4/6	13 4/13	14 4/20	15 4/27
Fabrication and Mechanical																
~~CAD Mount (not including gearing)~~	JG															
~~CAD Mount with gearing~~	JG															
Design compensator ~~gearing~~ barn door mount	KR															
Order parts and supplies	JG + KR + YS															
~~Constructing Equatorial Mount~~	JG + KR + YS															
Assemble mount	KR															
~~Obtain Camera Adapter~~	JG + KR + YS															
Circuitry																
Motor Driver and Power Supply Redesign + Fab	YS															
~~User Interface Specification~~	YS															
CV System and Interface																
Polar alignment algorithm	JG + KR															
Interface PA with mount circuitry	YS + KR															
Object tracking prototype (with video)	JG															
Interfacing Camera with object tracking	JG															
Integrate with Mount Movement	JG + KR															
Calibrate for mount	JG + KR															
GUI																
Software implementation	JG															
Obtain and install peripherals	JG + KR + YS															
Testing of software implementation	JG + KR															
Verification																
Skychart laser array circuit design	YS + KR															
~~Skychart laser array board layout+fab~~	YS															
Assembly of Skychart laser array	KR															
Sky tracking and object tracking tests	JG + KR + YS															
Integration and Additional Testing	JG + KR + YS															
~~Mock input generation~~	KR															
Oscilloscope/testing instrument implementation	KR															
Course Logistics																
1st Status Report	JG + KR + YS	2/10	2/15													
Design Presentation	JG + KR + YS	2/17	2/24													
Design Document	JG + KR + YS	2/17	3/2													
Final Presentation	JG + KR + YS		4/27													
Final Report	JG + KR + YS		4/27													

Part Name		QTY	Cost ($)	TotalCost
400-step stepper motor		3	$17.95	$63.10
Gyroscope Sensor		1	$8.95	$8.95
Camera Tripod		0	$43.99	$0.00
Tripod Camera phone adapter		0	$6.89	$0.00
Telescope Finder Scope		1	$13.79	$13.79
Pi-compatible display		0	$47.99	$0.00
Square turntable bearing (6031K160)		2	$2.40	$4.80
Round turntable bearing (6031K21)		1	$7.76	$7.76
4ft aluminum U-channel		1	$6.79	$6.79
(PKG of 10) Stainless Steel Phillips Flat Head Screws, 1/4"-20 Thread Size, 4" Long		1	$11.03	$11.03
(PKG of 25) Stainless Steel Phillips Flat Head Screws, 6-32 Thread Size, 1" Long		1	$3.36	$3.36
(PKG of 10) Stainless Steel Phillips Flat Head Screws, 10-32 Thread Size, 1-1/2" Long		1	$4.21	$4.21

(PKG of 100) Socket Head Screw, 4-40 Thread Size, 1/2" Long		1	$4.20	$4.20
2ftx2ft 1/2" MDF		0	$4.70	$0.00
8mm shaft ballbearing (608ZZ) (PKG 10)		1	$6.00	$6.00
D5mm Planetary Gearbox Nema17 Stepper Motor Speed Reducer High Torque 30:1		1	$38.12	$38.12
Motor driver components		0	$18.76	$0.00
Motor power supply		0	$8.18	$0.00
Clear Cast Acrylic (2ft X 2ft X 6mm)		0	$15.70	$0.00
ABS filament		0	$10	$0.00
Circuit Board Fabrication (Small Motor Driver, 3 boards per order, fast order)		1	$57	$57.20
8v Power supply (Voltage Regulator + Caps)		0	$1.12	$0.00
Laser Diodes		1	$5.49	$5.49
Laser Power supply		0	$0.79	$0.00
Laser Driver		12	$1.43	$17.16
5v Raspi Power supply		0	$18.50	$0.00
Small Motor Driver Components		3	12.92	$38.76
Small Motor Driver Power Supply Components		3	4.96	$14.88
Small 4096 Step Motor		3	8.32	$24.96
12V Battery		1	18.99	$18.99
3V Through-hole SMPS		2	15.73	$31.46
Through-hole NEMA-17 Motor Controller		2	26.87	$53.74
9V battery clip		1	0.48	$0.48
2 Universal mounting hubs (5mm)		2	$7.49	$14.98
1/4" D x 1.5" L shaft		0	$1.50	$0.00
ULN2803 Darlington Array		3	1.95	$5.85
5V 4A wall power supply		1	12.95	$12.95
1/4" - 5mm shaft adapter		2	$4.99	$9.98
1/4" 36 in. threaded rod		1	2.28	$2.28
1/4" 6 in. carriage bolt		2	0.57	$1.14
1/4" wing nut		1	1.18	$1.18
1/4" tee nut		3	1.18	$3.54
TOTAL COST				$487.13

Tools	Price	
OpenCV	$0	https://opencv.org/
libgphoto2	$0	http://www.gphoto.org/proj/libgphoto2/
wxWidgets	$0	https://www.wxwidgets.org/

Microcap Circuit Simulator	$0	http://www.spectrum-soft.com/index.shtm
KiCad	$0	https://kicad-pcb.org/
LtSpice	$0	https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html
SolidWorks	$0	https://www.solidworks.com/
Qt	$0	https://www.qt.io/