— Project Report—

MEEN 433/667 – Section 500/600

Alan Gregory 30%, Logan Morris 35%, Sebastian Villa Cuellar 35%

December 1st, 2022

"On my honor as an Aggie, I have neither given nor received unauthorized aid on this academic work."

Objective and Significance

The project design was a CNC “Dice Catching” machine. The objective of this design was to create a semi-autonomous machine capable of tracking a moving object until it stops, recording the object’s final position, moving the centerpiece, and moving the object with the centerpiece to a specified location before returning to the defined origin.

The significance of the project was to push mechatronics skills beyond what was available in class. Two of the three members were already proficient in mechatronic concepts and some coding languages. Mechatronics skills were expanded on by utilizing a Raspberry Pi and an ArduCam instead of just the Arduino in the kit. While a dice catcher has no direct application to useful robotics, the principles of this machine apply to many other technologies. For example, precision controls of CNC machines, manufacturing environments where products must be moved, and machine learning algorithms are a simple extension of the optical tools used as well.

Comprehensive Design

In order to control a two axis CNC machine, at least two linear rails were needed; one for each axis. The motion would be controlled by stepper motors for their precision. For stability, two rails were used along the y-axis, preventing a large torque as the heavy stepper motor would move away from the origin. There would be gears secured to the stepper motor with a timing belt. The ends of the belt would be affixed to two screws that protruded from the surface with zip ties to provide a secure, tight fit. The opposing end would have a bearing to keep the tension. This structure was applied to both axes, although only one stepper motor was needed to control the y-axis despite there being two linear rails. The decision to abandon the second y-axis stepper motor was made to address growing concerns over time constraints and needless complexities.

In order to locate an object placed within the bounds of a machine, several options were considered, including ultrasonic sensors, touch sensors, light sensors, and finally a camera. The choice to use a camera was made after weighing these options carefully. A camera would add significant complexity to the design - since the Arduino itself cannot interface directly with an Arducam - but would also add much needed usability to the design. A Raspberry Pi was selected for this purpose, due to its accessibility (available through one of the team members) and flexibility with integration options.

The design of the camera module intended for coordinates to be exported from the Raspberry Pi to the Arduino via serial connection. The Arduino would then function as the hub of motors and drivers. This was chosen for its simplicity; once the coordinates were imported to the Arduino, they would be converted to distances and from distances to steps taken by the stepper motor through basic unit conversion calculations.

An encoder was Connected to the y-axis of the stepper motor. The encoder was in a closed loop system that sent feedback into the control system. Accurately providing the actual position of the end effector enabled the real time control of the stepper motor position and driving parameters. Part of the designed controller implemented a discrete PD controller to slow

down the response of our system is Equation 1 below. The average speed of the moving arm can be lowered by either changing the fractional step size of the stepper motors or increasing the time taken between steps. To account for steady state error, instead of utilizing a PID controller, there is a capped-minimum step size so if error exists the controller will function as a Proportional controller and will eliminate the steady state error in a few more steps. In order not to damage any components there is a maximum speed the stepper motors can move at so our true controller behaves like a PD controller transfer function with a Saturation block that limits the maximum and minimum magnitude response of the physical motors.

$$u(t) = k_p \cdot e(t) + k_d \cdot \frac{\mathrm{d}e(t)}{\mathrm{d}t} \Longrightarrow u[k] = k_p \cdot e[k] \; + \; k_d \cdot (e[k] - e[k-1])/t_{loop} \tag{1}$$

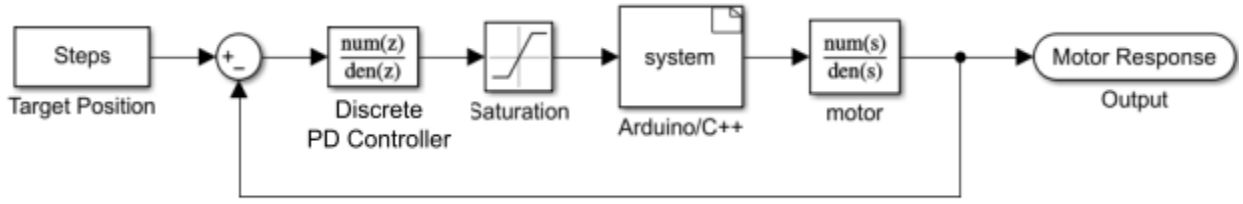

Figure 1: Closed Loop Block Diagram

Assembly

Three custom parts were designed and 3D printed. The first, drawn in Figure 2, was for connecting the side rails to the middle rail and the middle belt drive system and called the side slider mount. The part was mirrored for the opposite side. First the stepper motor and pulley mounts would be connected to the side slider mount with washers and long screws through the pill-shaped opening. Next, the middle rail would be connected to the side slider mount with screws and nuts through the two center holes on the top; the hexagonal shaped opening on the bottom would house the nuts. Finally, the side slider mount would be fastened to the side rails through the four same-sized holes equally spaced apart.

The second custom part, drawn in Figure 3, was used to connect the middle rail to the belt system and a third stepper motor which was used for “dice catching”. First, a separate mounting for the middle rail belt system would be connected to the middle slider mount through the middle two holes through the indent. Next, the stepper motor would be connected to the middle slider mount through the four holes surrounding the large bore; the stepper motor would face down with the large bore being where the motor shaft protruded towards the ground. Finally, the middle slider mount would be attached to the middle rail through the four final equidistant holes not surrounding a large bore.

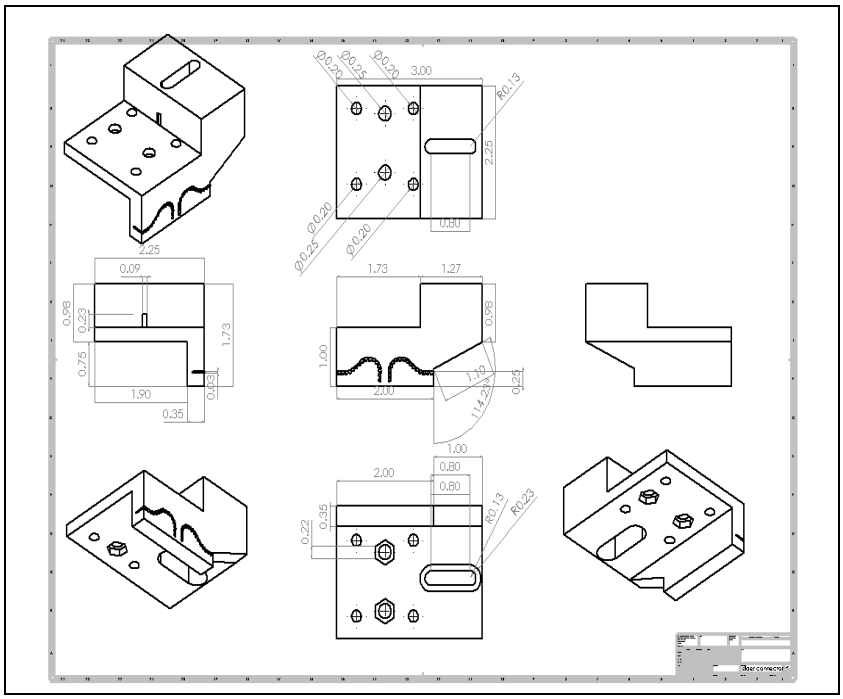

Figure 2: Solidworks drawing of the side slider mount for the side sliders. The units are in inches. The non-isometric views (from left to right, top to bottom) are top, front, right side, left side, bottom.

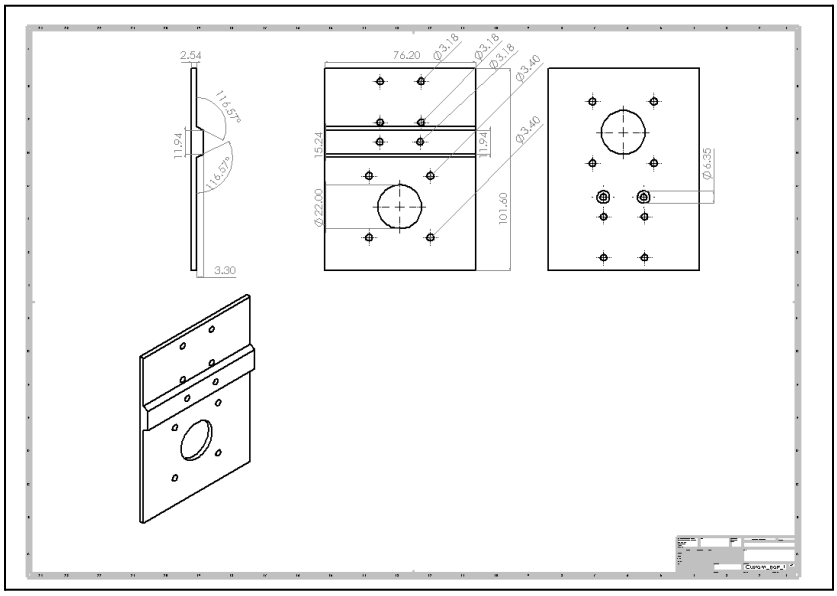

Figure 3: Solidworks drawing of the middle slider mount for the middle slider. The units are in inches. The views (from left to right) are side, top, bottom, and isometric.

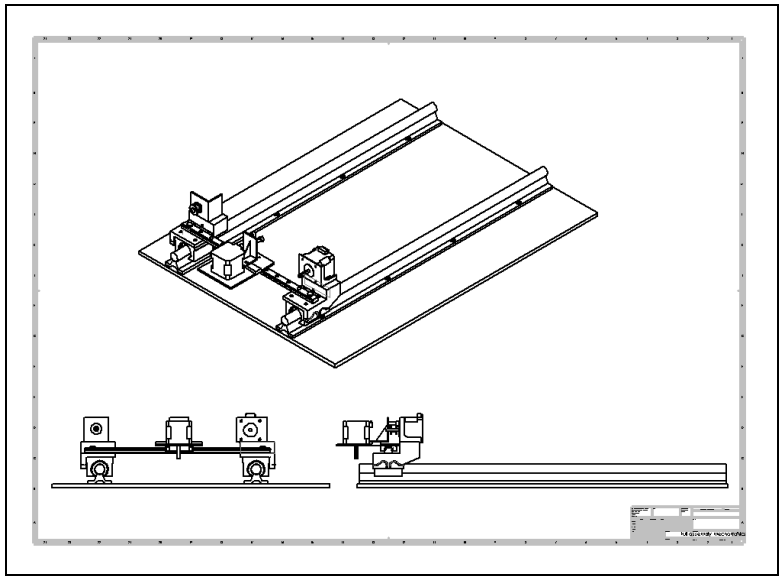

Figure 4: Solidworks Drawing of the board, rails and sliders, custom parts, and some stepper motors. The views (from left to right, top to bottom) are isometric, side, front.

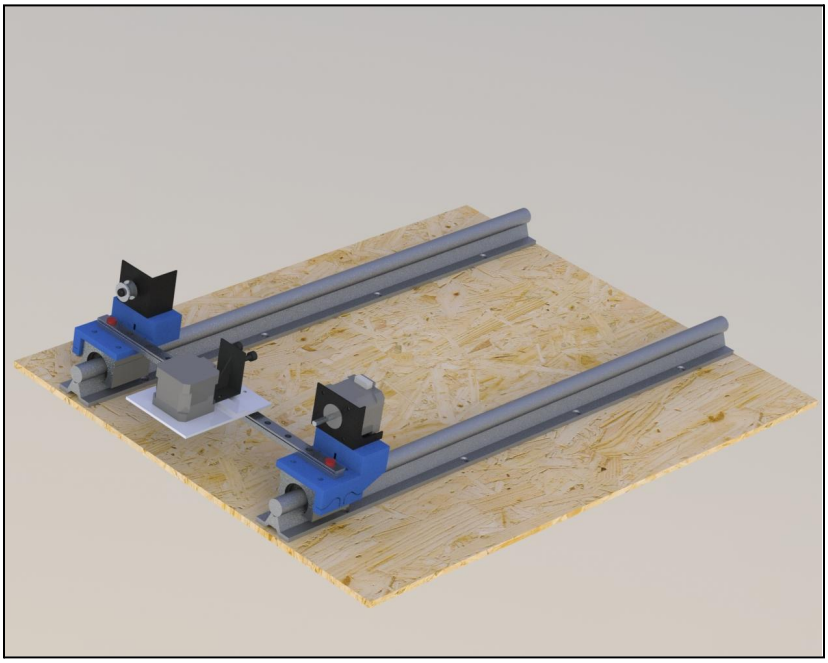

Figure 5: CAD model render of the board, side rails, middle rail, custom parts, and stepper motors.

Experimental Results

The results show that the CNC is capable of tracking a moving object until it stops, recording that final position. Separately, it demonstrated its capabilities of moving and positioning the centerpiece along a two axis grid. While ultimately, the CNC did not demonstrate its full capabilities with an integrated camera and movement system, there was much to be learned.

The motion of the machine when controlled independently was very successful. Inputting “zero” as a command would cause the rails to retract until stopped by limit switches. Inputting “move” followed by the number of steps for the stepper motors to move caused the rail to move to that specified position.

The camera detection system also worked fantastically well. The camera was able to track a rolling die within the bounds of the system and return a calculated position with reference to the origin (at the zero of both axes). The camera could detect the object with a surprising precision, as seen in Figure 7. There were a few shortcomings, such as the shading from light causing error frames, but those were minor when compared to the repeatable success from tracking the die.

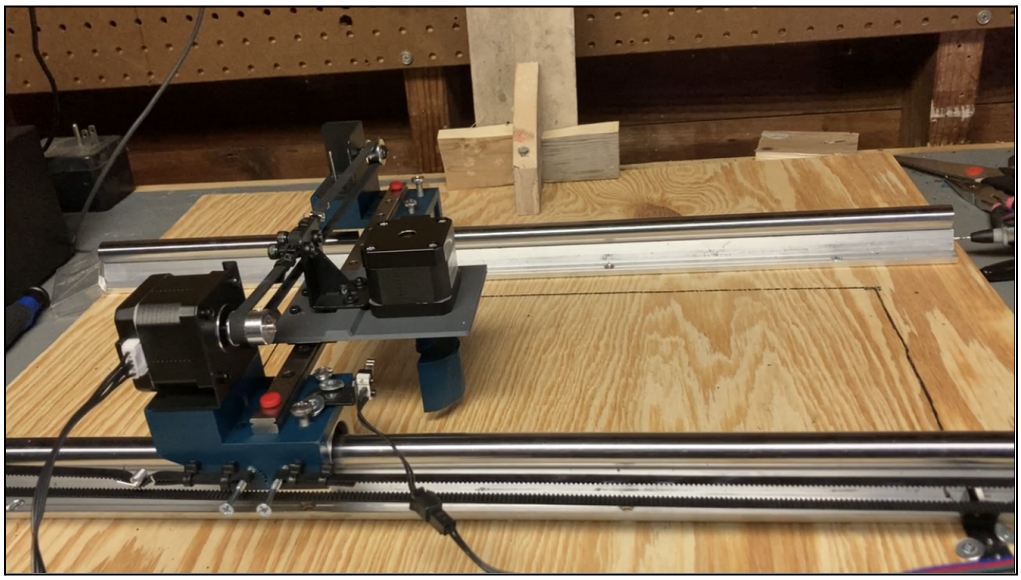

Figure 6: The rail system of the CNC.

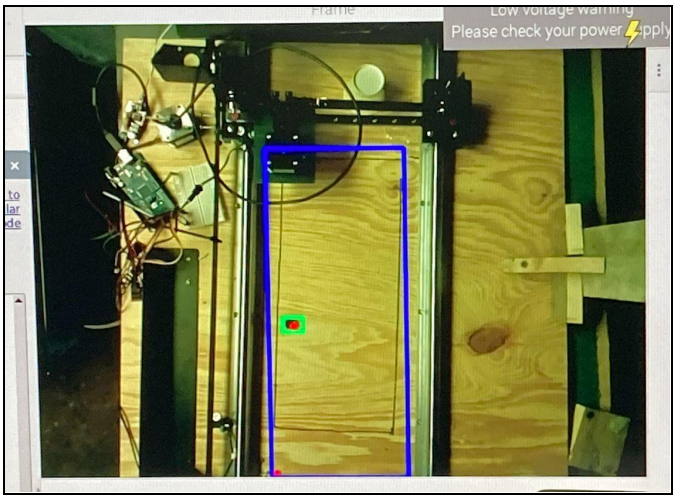

Figure 7: Perspective of the camera that is mounted above the rail system. The blue box indicates the area where movement is tracked, the green box with a red dot indicates a tracked object (a die in this case), and the red dot in the bottom of the blue box indicates the defined origin.

Shortcomings and Design Modifications

The camera was prone to recognizing objects that were not there due to changes in ambient light. The light would create shadows that resulted in the program recognizing contours of phantom objects. This was addressed by accounting for the placement of the object in the past ten frames in order to reduce erroneous results. Moreover, the system would not accept positional values until the user indicated readiness by pressing the escape button once more before proceeding to roll the die.

The Raspberry Pi would often crash when running too many processes at once. This, combined with bootlooping - being stuck in a cycle of restarting - considerably halted progress. Writing the Pi code, testing, and serial interfacing all required the Pi to be up and running so every time it crashed and got stuck, time was wasted.

Communications between the Raspberry Pi and the arduino was confirmed to have varying degrees of success. The communication between the two was occurring with regularity; both were sending and receiving signals between each other. However, the commands sent did not activate the corresponding controls in the other system. One would often be waiting for the return signal while the other was stuck in a loop.

The PD controller was to be implemented by coupling the encoder rotation to the stepper motor. This would translate the rotational motion of the stepper motor into a signal that could be analyzed in the control system. However, the attachment of the encoder faced several issues. Originally, the purchased encoder was selected because it resembled the same structure as the stepper motor: a round peg with a flat side. Unfortunately, it was slightly too big to fit into the gear. A great deal of other attachment methods were attempted, with all being unsuccessful. For

example, the large encoder was dragged alongside the chassis, attempted to be soldered to the stepper motor, and glued with putty to the gear. After all failed, the smaller encoder was used. This also was ultimately unsuccessful, with the encoder being slightly off center from the gear, just enough to make the motion of the encoder body oscillatory. When attached to the stepper motor, the adhesive was not strong enough to withstand repetitive use. The encoder was inserted between the timing belts, but not enough pressure could be applied to keep the timing consistent; too much pressure would push the encoder from the breadboard. Even fastening the encoder to a breadboard secured to a block of wood screwed into place was unsuccessful because the wood splintered. If this system would be implemented again, it would be recommended to use a motor with an encoder embedded into the system. Otherwise, the tension gear opposing the stepper motor could theoretically be replaced with an encoder, given that one strong enough of the right size could be found and used.

To simplify the electrical wiring, a CNC shield was bought that allowed a compact and clean fitting of up to 5 of the purchased stepper motor drivers (DRV8825). The shield was solely a hardware improvement based upon the RAMPS 1.4 specifications but unfortunately aforementioned drivers had issues straight out of the box. From varying resistances between the same digital pins on different drivers, to lower current limits than as rated on the data sheet. The unexpected discrepancies stalled the progression of the project and delayed the testing phase as it inhibited any movement from the stepper motors. The solution to this involved scrapping the use of the CNC shield and the DRV8825 drivers entirely and utilizing Dual H-Bridges, and a DC Buck Step Down Voltage Converter with an integrated Constant Current Power Module. The new equipment would manually limit the current from the power supply to each actuator. However, to manually control the motion of the stepper motors, a C++ library was developed from scratch to control each stepper motor individually by cycling through 4 transistor switch configurations. This caused a lot of delay in the development of the code but did fix the overall issues with the drivers and allowed for further progress.

Conclusions

The CNC “Dice Catcher” project was well accomplished despite numerous problems. The objective was to allow for expansion of mechatronics skills beyond what was available in class in CNC systems for controlling a 2 axis system. Throughout the project, the team learned how to perform image processing, communicate using a serial connection between Raspberry Pi and Arduino, and, of course, program a closed loop transfer function. The physical construction of the CNC was a learning process as well, requiring several hours of planning, purchasing, and labor. Three custom parts were designed and 3D printed for connecting the various parts together, and the team had to design the rails to work with the wiring and placement of the motors. There were plenty of shortcomings such as serial communication errors, issues with attaching the encoder, and initially utilizing low-quality drivers until they exploded. However,

the project was ultimately a successful endeavor in instructing the team in mechatronic applications.

Individual Group Member Contributions

Logan's house served as the workshop where all physical construction and assembly took place. Besides contributing tools and location, Logan also scrapped his old 3D printer for important parts. Logan did a significant portion of the physical construction of the model. Finally, Logan wrote the code for the Raspberry Pi (i.e. the camera) and developed the CAD model for the assembly as well as a portion of the 3D printed parts.

Sebastian wrote all the Arduino code, including using the control loop for the stepper motor drivers and the serial communication between the Arduino and Raspberry Pi. As the member with the most experience and expertise outside of Mechanical Engineering, he naturally took responsibility for the electrical wiring, soldering, and the Arduino C code.

Alan handled some of the material acquisition; purchasing wood board and fasteners, and designing most of the 3D printing parts. Many of the materials were old parts that Sebastian and Logan procured from previous personal projects. Alan also helped out for a good portion of the physical construction of the product. Moreover, he handled a lot of the administrative duties and coordinated team efforts.