

Design and Construction of a Quick Return Device for Use as a Demonstration Tool

A Major Qualifying Project Report:
submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science
By

Zachary Belohoubek

Ellyn Webber

Date: May 01, 2014

Approved:

Professor E. Cobb, Advisor

Abstract

In order to further promote a deeper understanding of the design of mechanisms, we created a Quick Return Mechanism model that demonstrates how changing design parameters can alter the motion and time ratio of the device. Data from accelerometers on the mechanism were gathered and compared to the theoretical results from a mathematical model of the linkage. With this apparatus, a professor can easily demonstrate how a quick return mechanism functions on a theoretical and practical level to further students' comprehension of the kinematics of the mechanism.

Table of Contents

Table of Contents

Abstract	1
Table of Contents	2
Executive Summary	3
Background	6
Preliminary Research	6
Basic Requirements	8
Model Specifications	9
Past Projects	9
Professor Requests	9
Research	10
Problem Statement	10
Design Specifications.....	11
Preliminary Designs.....	12
Final Design and Analysis	14
Final Design Modeling	14
Analysis	19
Position Analysis	20
Velocity Analysis.....	21

Acceleration Analysis	22
Construction of Prototype	23
Mechanism	23
Program.....	25
Evaluation and Testing	27
Summary	28
Recommendations.....	28
Aesthetic Recommendations.....	28
Functional Recommendations.....	29
Coding Recommendations	31
Other Recommendations.....	31
References.....	33
Appendices.....	33

Table of Figures

Figure 1: The Movement of a Slider-Crank Mechanism	7
Figure 2: An In-Line Slider-Crank Mechanism.....	7
Figure 3: An Offset Slider-Crank Mechanism	8
Figure 4: Crank Design with Variable Radii.....	12
Figure 5: Design for a Friction Mechanism for the Slider	13
Figure 6: Isometric CAD Model of the Frame and Base	14
Figure 7: Front CAD Model of the Frame and Base	15
Figure 8: Front CAD Model of the Mechanism	16

Figure 9: Isometric CAD Model of the Mechanism.....	16
Figure 10: Front CAD Model of the Elevator	17
Figure 11: Isometric CAD Model of the Elevator	18
Figure 12: Isometric CAD Model of the Whole Assembly	19
Figure 13: Position of the Pin Connecting the Crank and Slider Link	20
Figure 14: Horizontal Position of the Slider Block.....	21
Figure 15: Velocity of the Slider Link.....	21
Figure 16: Velocity of the Slider Block	22
Figure 17: Acceleration of the Slider Link	22
Figure 18: Angular Acceleration of the Slider Link.....	23
Figure 19: The Completed Assembly	25
Figure 20: Theoretical Position of the Slider Block.....	27
Figure 21: Actual Position of the Slider Block.....	27

Executive Summary

In today's classrooms, professors often run into trouble portraying the problems and equations in class with examples in the real world. As theory and practice are Worcester Polytechnic Institute's educational objective, we decided to build a mechanism that would highlight the two and compare them side by side. We chose to do so utilizing a quick return mechanism because they are typically more difficult for students to comprehend. By doing so, we help solve two problems with a single mechanism: theory versus practice and a specific concept within the engineering field.

To do so, we take the input values and calculate the theoretical function of the mechanism and portray it on a graph. Additionally, we acquire the function of the mechanism actually occurring at the same time by gathering data from accelerometers and speedometers on the mechanism. Then, the two functions are portrayed on the same graph to show the exact difference between the theoretical analysis and the actual analysis. Doing so not only helps students draw a comparison and an understanding between theory and practice, but also comprehend how quick return mechanisms work and real world applications.

It is critical that professors educate students to the best of their abilities using the resources provided to them. The purpose of this Major Qualifying Project (MQP) is to add to these available resources and promote a culture of learning within the Mechanical Engineering Department that emphasizes the Institution's educational objectives.

Background

In order to create a product that would fit the needs that were determined, it is important to have a deeper understanding of kinematics and a clearer competence of what professors would want the mechanism to do.

Preliminary Research

Before delving into a physical design, one must first understand the fundamentals of quick-return motion. There are several key features of the slider-crank mechanism; first being the rotation of the crank, which moves the slider arm through a designated angle. As the slider arm is rotated back and forth, an attached link moves through a rotation that is contingent upon the motion of the slider. This link forces a block through linear motion. This setup, presented in the schematic below, is the desired path of the linkage system.

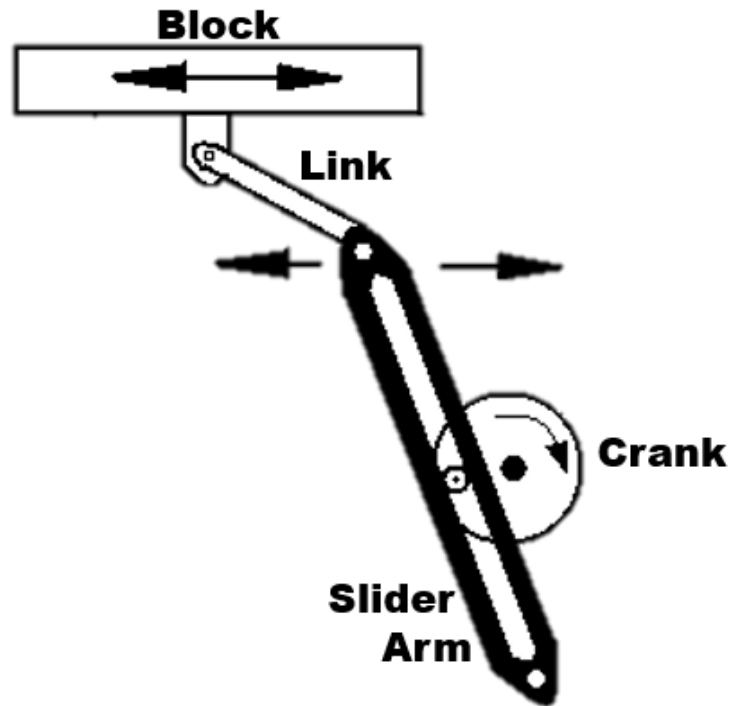


Figure 1: The Movement of a Slider-Crank Mechanism

There are two main types of slider crank mechanisms: in-line and offset. An in-line mechanism has its slider positioned such that the linearly sliding block remains on the same plane as the ground joint of the crank. This forward and backward motion is symmetrical as the crank rotates. This type of slider crank, shown below, is not a quick return model because the time ratio of the push and pull motion is always 1:1.

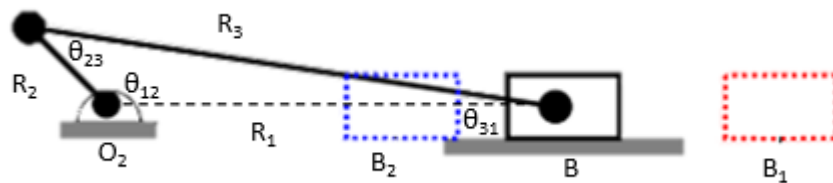


Figure 2: An In-Line Slider-Crank Mechanism

An offset slider crank mechanism has its slider positioned such that the block is not aligned with the ground of the crank, hence the name offset. This causes the back and forth motion of the block to not be symmetrical and instead is pulled back more quickly than it is pushed out. This type of mechanism, shown below, is a quick return mechanism, and is the basis for this project.

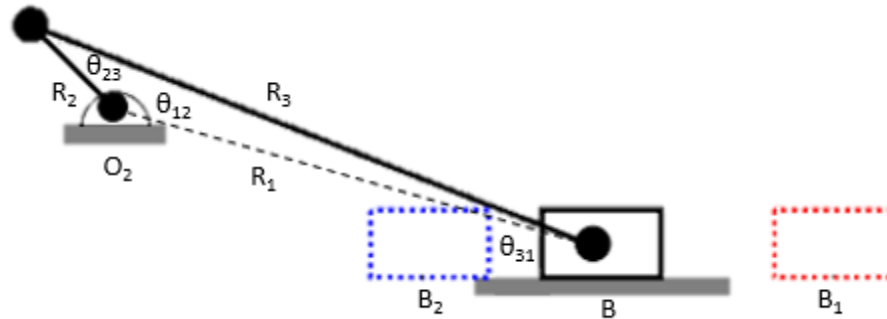


Figure 3: An Offset Slider-Crank Mechanism

The radius of the crank and length of the slider are important factors to consider when designing a quick-return mechanism. The most common quick return ratio of the time of the cutting stroke to the time of the return stroke is about 3:2 (Ramamurti, 27). For the model to be considered a quick-return the desired time ratio should determine the radius of the crank and the length of the slider.

Quick-return slider crank mechanisms have been used since the nineteenth century in a variety of machines in the engineering industry. The first application of this type of mechanism was designed by Joseph Whitworth to be used as a cutting machine. Currently, this type of mechanism is used in shapers, screw presses, and mechanical actuators.

Basic Requirements

This project is meant to be a teaching aid for a variety of classes, so there are many factors that must be considered and included in the design. In order to get a better idea of what would be advantageous in the design, professors of different classes were interviewed. From these interviews, more details of the design were considered and finalized.

As per professor request, the model is a cabinet model, rather than a tabletop. The professors said that a tabletop model would be useful as well, but it would be harder to mechanize. The model is also going to incorporate adjustable friction settings, because students need to learn how to apply friction in their analyses.

Model Specifications

When coming up with design specifications for the model, it was helpful to look at past projects, speak to the professors who would be using it, and do research on common uses for this machine.

Past Projects

Although this type of project has been completed in the past, this specific type of mechanism has not been used. When deciding on design specifications, the past projects were considered only in the aspects that are similar to this one. The size and shape of our model, for example, is very similar to that of a completed project. The types of motors, accelerometers, and hardware used in past projects influenced the types that are in this model.

Professor Requests

After interviewing professors, it was clear that certain parts of the model needed to be made adjustable. The radius of the crank and length of the slider determine the length of the horizontal stroke and the time ratio. It was suggested that both of these be made adjustable so that the professors could easily demonstrate their effect on the output. In this model, the radius of the crank is adjustable by hand, by moving the pin that connects it to the slider into a different hole in the crank face. Adjusting the length of the slider is achievable automatically by entering the desired time ratio values.

Research

Some of the design specifications were determined by research done before beginning to design the model. As previously mentioned, the typical time ratio of a cutting stroke to a return stroke is 3:2. This information was used to decide the maximum and minimum desired time ratios. The common angular velocity of the crank is 20-30 rotations per minute (Ramamurti, 25), so the model needs to have a motor that moves at around that speed.

Problem Statement

Professors and students need a mechanism to demonstrate the functionality of a Whitworth quick-return mechanism because the concept has been difficult for professors to explain to students in a way that helps them understand the movement of the mechanism.

Design Specifications

1. Product should be a Whitworth mechanism
2. Should be a quick-return model
3. Full joints are preferred to half joints
4. Full rotation of crank should be allowed, and toggle positions that prevent crank from rotating should be eliminated
5. All components must have a safety factor of 5
6. All transmission angles should be greater than 35 degrees.
7. Model should be at most 2'x4'
8. The machine should have variable speed control, able to run cycle times ranging from 0.5 to 10 seconds.
9. Model should be portable, and fit through a doorway easily
10. Model should weigh less than 50 lbs
11. Model should run on 110vac power
12. With crank held stationary, no link should be able to move more than 5°
13. Lengths of some linkages should be variable to attain different time ratios
14. Model should output acceleration on the slider block.
15. Product should be cost effective
16. Should be able to vary the quick return ratio
17. The angle that the slider arm goes through should be at most 60° .
18. All aspects of the machine should be readily accessible for regular maintenance
19. Adjusting the quick return ratio via ground link spacing should be achievable automatically, by entering preset values for height

Preliminary Designs

Going into this project, there were many aspects of the design that could be made variable. After speaking to professors, it became apparent which of these parts were the most important to include in the final design. Although designs for all of the variations were made, few of them were constructed for the physical model due to the insufficient amount of time.

The first variation of the model included a crank with variable radii, which would demonstrate how changing link lengths varies the time ratio of the mechanism. This design required putting multiple holes into the crank, so that the pin connecting the slider link to the crank could be moved to different holes depending on what time ratio was desired. This would have been achievable by using a pin that had a locking mechanism so that it could be inserted and removed easily, but would not come out unless intentionally removed. but it also would have made the crank weaker and machining the crank to have this capability would have been difficult. In addition, the adjustments would have to be made manually, which adds a safety concern. Shown below is the original CAD model that would have incorporated this feature.

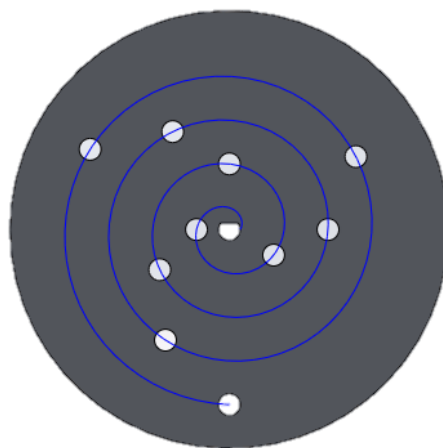


Figure 4: Crank Design with Variable Radii

Another variation of the design included a friction mechanism added to the slider block to demonstrate how the mechanism is usually used. A few designs were made that would implement this concept. The first design involved using a bearing that would lock in one direction and roll in the other. Another design utilized resistance bands that would pull on the band to add resistance in one direction and assistance in the other. A CAD model was created demonstrating how this concept would have been included in the physical model. However, incorporating this feature would have put stress on the motors that could have become a safety hazard.

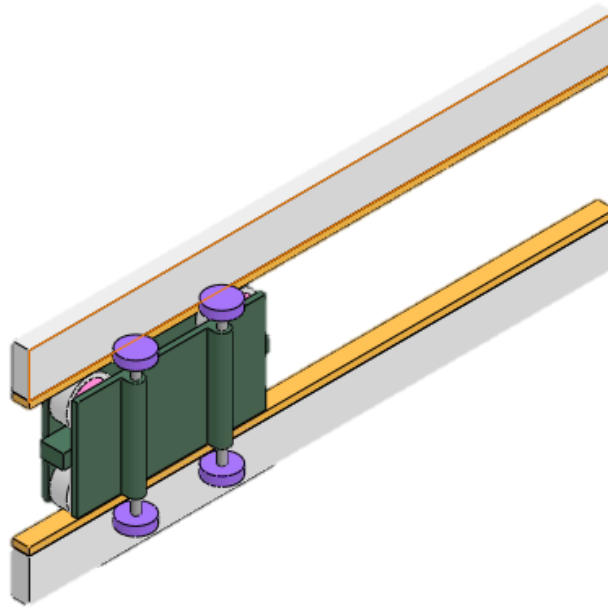


Figure 5: Design for a Friction Mechanism for the Slider

A third variation included an elevator-like mechanism that could be used to adjust the distance between the ground of the slider link and the ground of the crank. This would make it so that the horizontal distance that the slider travels is variable, as well as the time ratio. This concept was implemented in our final design by using a platform that the crank motor was mounted on, which was suspended between four pieces of the frame on slider blocks. The platform was connected to wire, which went over a pulley at the top and down to the base where a motor would

wind and unwind the wire to raise and lower the platform. Images of this concept are shown in the final design.

Final Design and Analysis

Final Design Modeling

The final design of this model was created in SolidWorks, and production methods were kept in mind while the model was being made. The frame of the mechanism was made of 80/20 aluminum framing, connected with commercially available brackets. These pieces were attached to a platform to make it easier to transport.

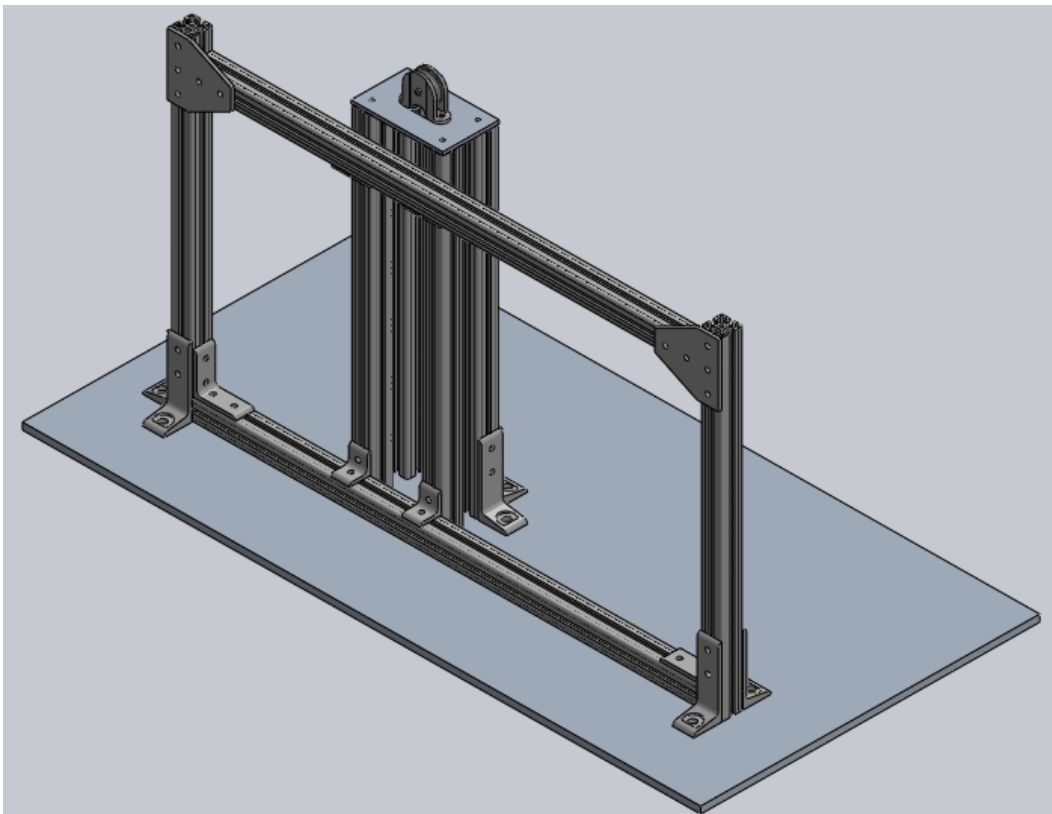


Figure 6: Isometric CAD Model of the Frame and Base

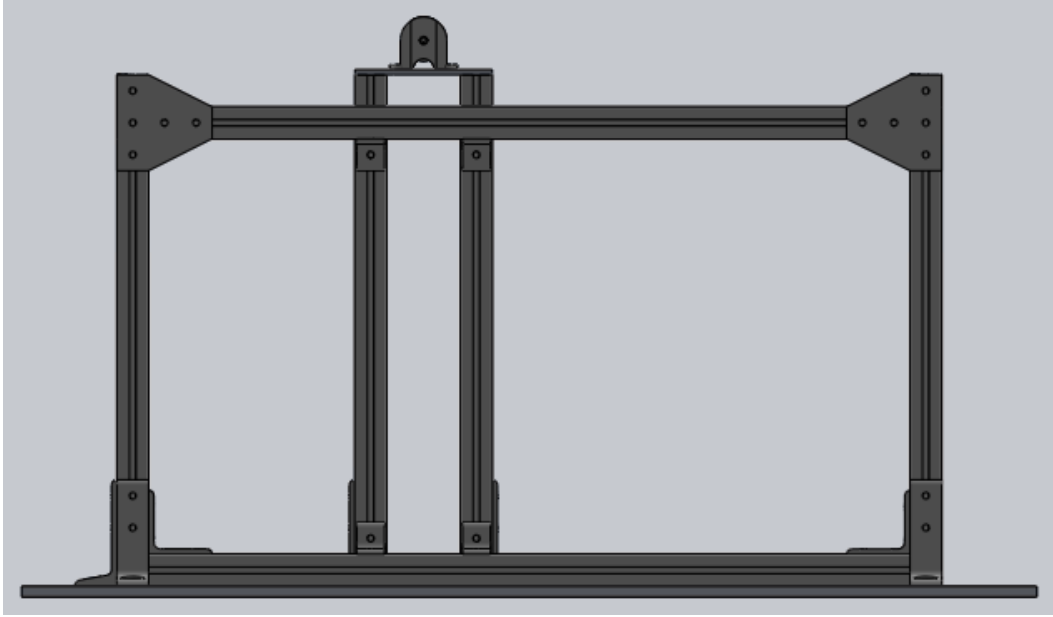


Figure 7: Front CAD Model of the Frame and Base

The mechanism itself was based off of models that were found while doing background research. The crank was made as a round disk instead of a straight link because it is easier to attach a motor to this, and it balanced the weight better than a link would have. The slider block was designed to sit in the top of the frame, so it had a small extrusion that fit into the slot in the framing. An accelerometer was mounted to the slider block to collect data.

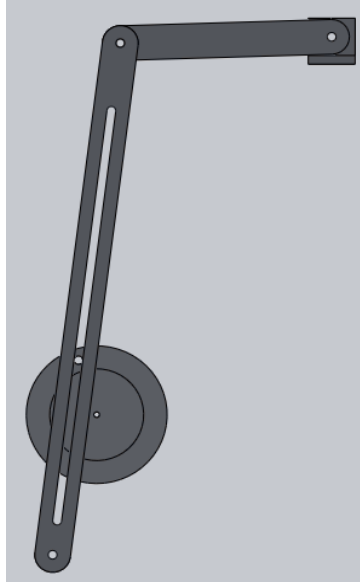


Figure 8: Front CAD Model of the Mechanism

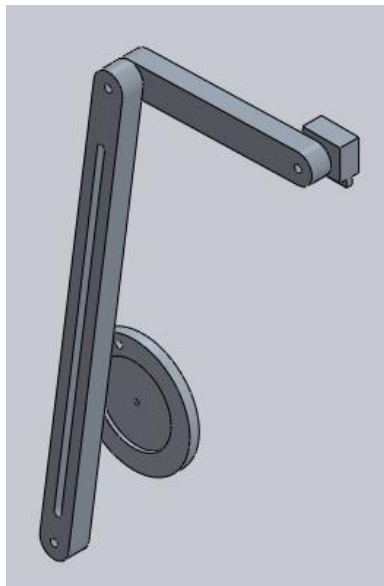


Figure 9: Isometric CAD Model of the Mechanism

The elevator that lifted the crank was designed to use sliders that were made for the 80/20 framing that would slider freely on the vertical rails. The platform that held the motor was attached to these, and the motor was mounted with a bracket to the platform. Four eye hooks were put into the platform to hold the wire that was used to raise and lower the elevator. The wire went up to a

platform at the top of the vertical rails, which held a pulley. The wire looped around the pulley and went back down to the baseboard, where it wrapped around a motor.

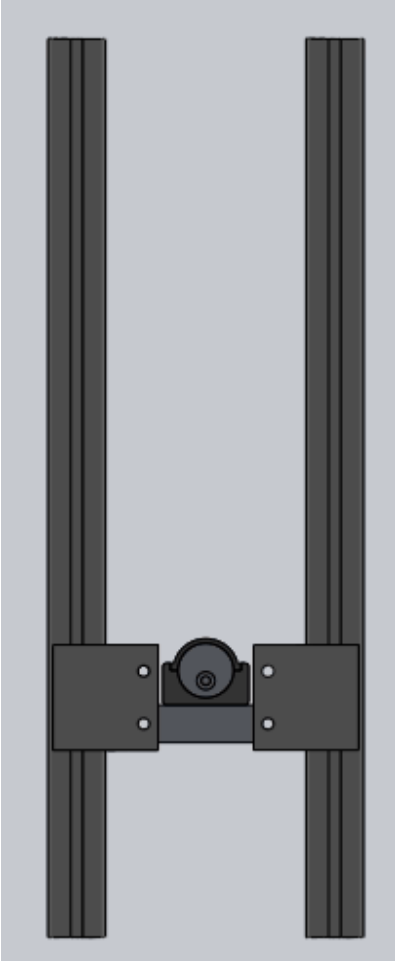


Figure 10: Front CAD Model of the Elevator

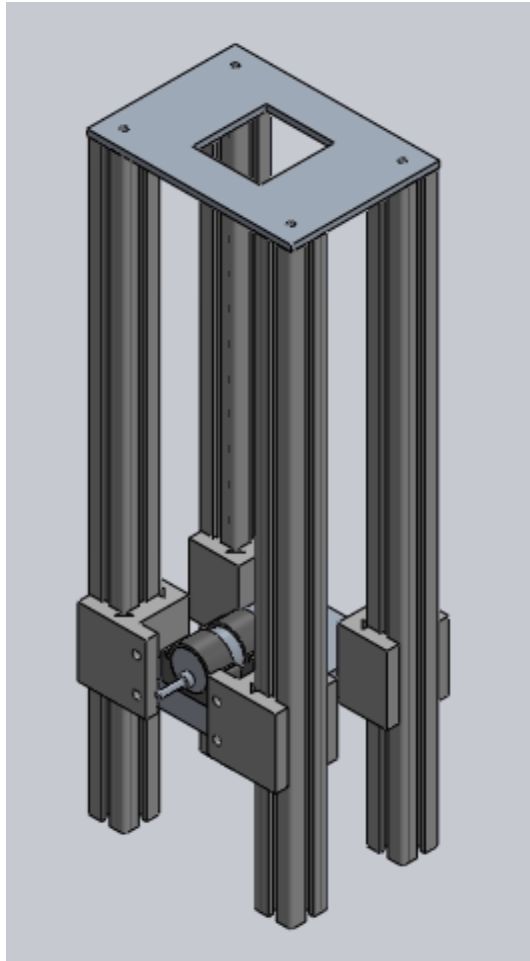


Figure 11: Isometric CAD Model of the Elevator

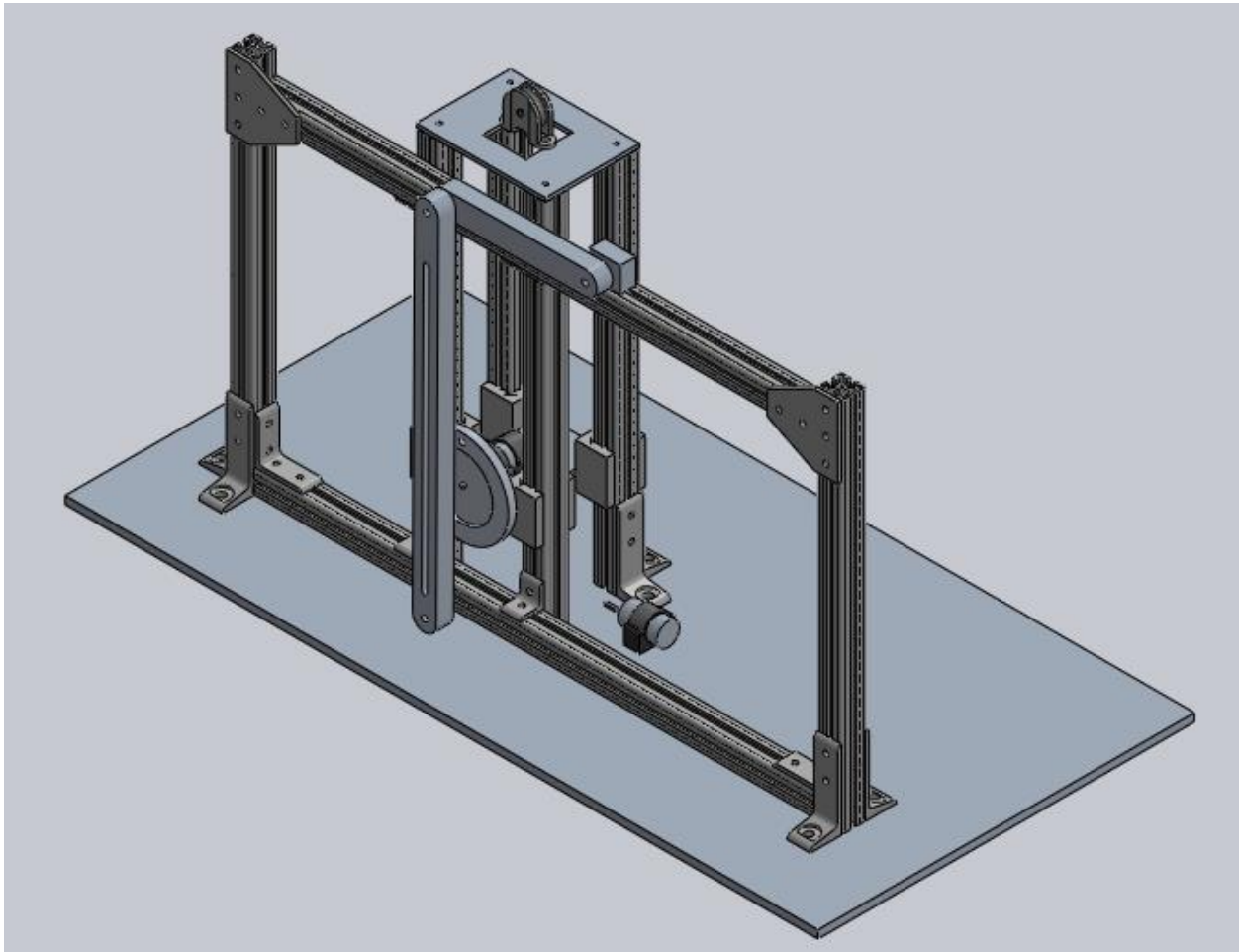


Figure 12: Isometric CAD Model of the Whole Assembly

Analysis

In order to demonstrate the theoretical values that the mechanism would output, an analysis was done to determine the position, velocity, and acceleration of parts of the mechanism based on the link lengths and motor speeds. This analysis includes graphs that represent these values with relation to the angle of the crank. These functions were used to analyze the mechanism at the

highest and lowest positions of the crank. All figures shown represent the values when the crank is in the lowest position, and the full analysis can be found in the appendix.

Position Analysis

To do the position analysis, all equations were determined as a function of the crank angle, here indicated by the variable x . First, the vertical height of the pin connecting the crank to the slider link was determined with relation to the slider link's grounding pin. This was graphed to ensure that the function and variables were set up properly.

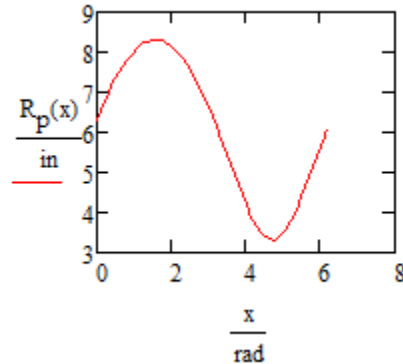


Figure 13: Position of the Pin Connecting the Crank and Slider Link

From this, the angle of the slider link to the horizontal was determined, as well as the angle between the link and the crank. Using these equations, the x and y components of the slider link were determined, and used to find both the angle between the link and the horizontal as well as the horizontal position of the slider block. Shown below is the position of the slider block with respect to the crank angle.

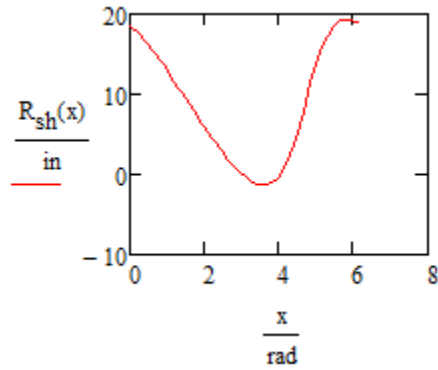


Figure 14: Horizontal Position of the Slider Block

Velocity Analysis

Using the position analysis and the motor specifications, the velocity of the transmission pin between the slider link and the crank was determined. This was used to find the velocity of the slider link and slider block.

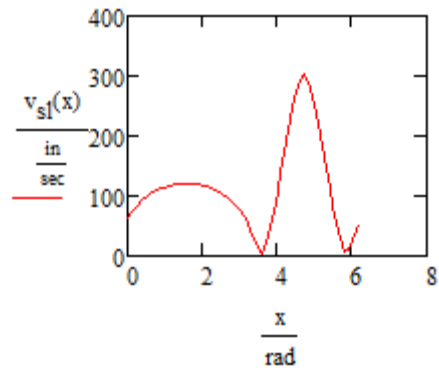


Figure 15: Velocity of the Slider Link

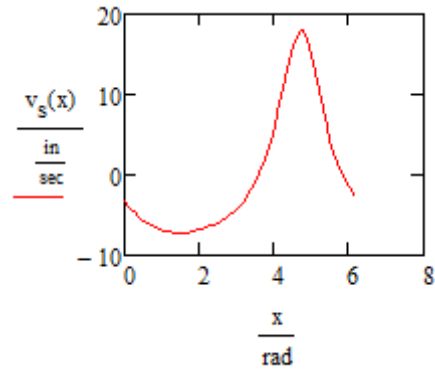


Figure 16: Velocity of the Slider Block

Acceleration Analysis

Using the velocity and position analyses, the linear and angular acceleration of the slider link was determined. Due to the nature of the mechanism, the graph of the acceleration of the slider link is the same shape as the graph of the acceleration of the slider block.

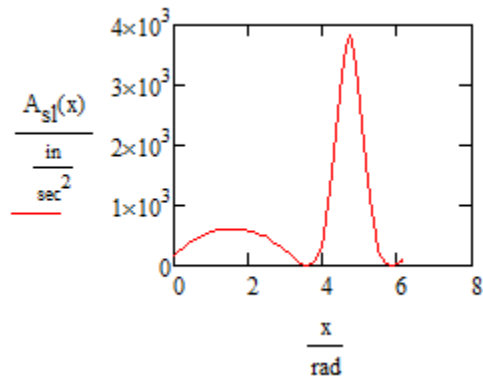


Figure 17: Acceleration of the Slider Link

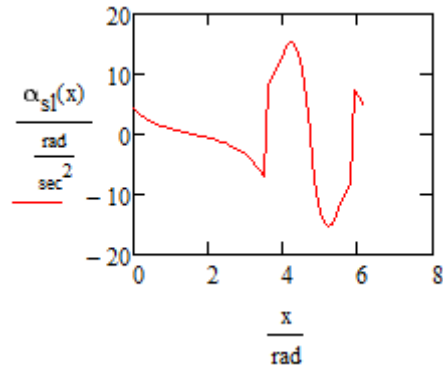


Figure 18: Angular Acceleration of the Slider Link

Construction of Prototype

Construction of the model required work in two aspects: building the physical model of the mechanism, and programming the motors to run as intended.

Mechanism

The construction of the mechanism was done very similarly to how it was planned while in the design stage of this project. The frame and base were made of 80/20 aluminum framing, and was held together with commercially available brackets, feet, and plates. When this was put together, it was bolted to the baseboard, which is a 2'x4' piece of wood.

The mechanism was machined out of aluminum. Originally, a few pieces of stock were bought and a computer-aided manufacturing program was made in Esprit to make machining on the MiniMill easier. The stock was bought to be slightly larger than the size needed for each piece, so that when material was lost in machining the pieces would still fit together properly.

The first piece that was machined was the link. This started out as a solid block of aluminum that was 12"x1.5"x1". First the block was cut down to be the correct length, which was 10.5". In one operation on the MiniMill, the ends were rounded and holes were drilled into it for the pins to fit through. Finally, the front of the link was surfaced to make it 0.5" thick rather than 1", to cut down the weight of the part.

Next, the crank was manufactured. The crank started as a 0.5" thick round disk of aluminum with a diameter of 6". A 6mm hole was drilled in the center, which was the right size to fit the motor that was used. A second hole with a diameter of 0.375" was drilled 2.5" from the center. Finally, a 0.2" deep pocket with a diameter of 4" was cut out of both sides of the disk, to reduce the weight.

The slider block started as a 2" cube that was cut in half to make it the size that was specified in the design. Two cuts were made in the bottom of the block that left an extrusion so that it could sit in the aluminum framing. The cuts went 0.375" in from the sides of the block, and were about 0.5" deep. This left an extrusion that was 0.5" tall and .25" wide. Then a hole was drilled in the center of the main block for the pin that would connect it to the link.

The slider link was the most complicated part to make, in part because it was too long to fit on the MiniMill, so a VM2 had to be used instead. It started out as a 24"x1"x1.5" piece of aluminum stock. First, the ends were rounded and holes were drilled for the connecting pins. Then the slot was cut into it using an end mill, following the path determined by the CAD model that had been made.

The elevator was constructed using wood that was screwed into the carriages that slide on the framing. The motor that ran the crank was fastened to the platform using brackets. Four eye hooks were screwed into the corners and used to attach wire that would raise and lower the

platform. The four pieces of wire were connected to a carabiner, which was attached to a piece of wire that went to the top of the elevator and wrapped around a pulley. The wire then went back down to a motor that was fastened to the base so that when the motor ran it would wind or unwind the wire, causing the platform to raise or lower. At the base of the elevator was a stop button that was mounted on a block so that when the elevator hit it, it would recognize that it was at the bottom of its range of motion.

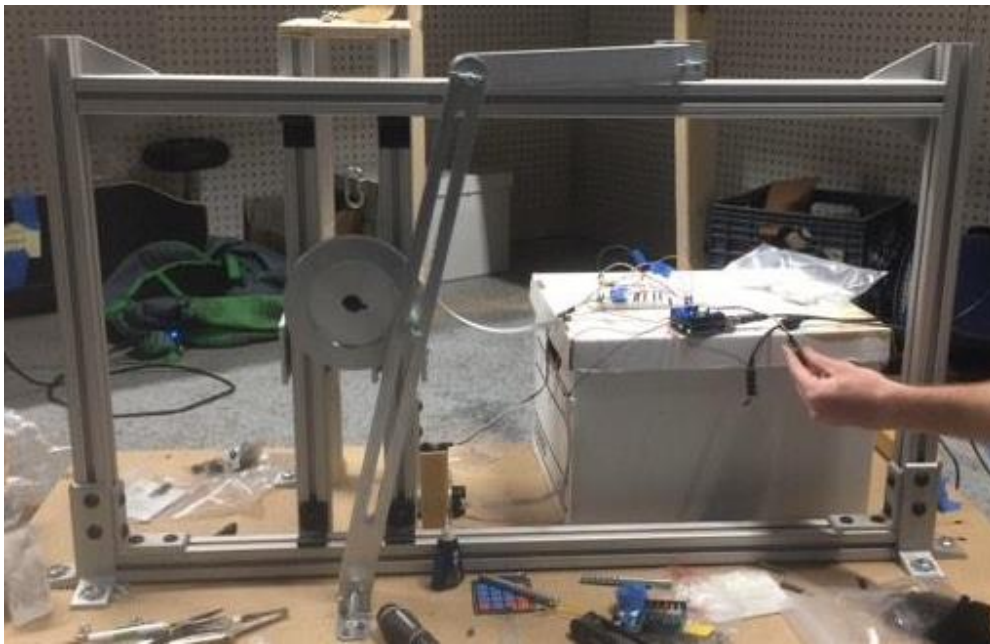


Figure 19: The Completed Assembly

Program

To use the mechanism, an external power supply is needed from your typical wall outlet and a USB connection to a computer. Upon hooking up the Arduino to power and a computer, the program must be run. The first thing the program does before it can be told to do anything is figure out where the elevator is situated, as it can be left in a position from 0-99. Thus, the program resets the device by lowering the elevator until it hits a push button. This tells the Arduino that the

elevator is at position '0'. If that button is already being pressed when the program starts running, it will not lower the elevator, as it already knows that it is at height '0'. This scale from 0 to 99 is in place to hit 100 different time ratios that are determined by where the crank is.

The most important part of programming a device like this is to ensure that there is an emergency stop in place in case of failure. Therefore, there is another push button easily accessible to the user, out of the way of any moving parts. This second button stops all motion, no matter what and also acts as the stop button for the crank motor.

The program first prompts the user for a height. Upon receiving a value from 0-99, the elevator motor rotates in the proper direction to unwind or wind up the fishing line around it as to move the elevator. The program then saves the new height of the elevator so that the next time it is to be adjusted, it can find the difference in heights and move to the next location efficiently. The push button at the bottom always acts as zero, so if the elevator somehow gets uncalibrated, the device can self-rectify the issue.

After moving to its new height, the program prompts the user for the direction of the crank motor to spin. The user enters '0' or '1' to have the crank motor rotate in one direction or the other. While the crank motor is spinning, the accelerometer gathers data on the acceleration of the slider in its one dimensional movement. The program takes the values of acceleration and then calculates the velocity. 250 data points are read out for the user to then put in an Excel sheet to graph the data.

The motor will continue to spin until the stop button is pressed, or a new height value is entered. Both motors cannot spin at the same time, as that would add significant stress to both the motors as well as the mechanism.

Evaluation and Testing

Although this product was not constructed the way that it had been envisioned, it was functional and worked decently well. When testing the device, the graphs produced by the accelerometer on the slider block were very similar to that of the theoretical graph. For the first couple of runs that were made, the average difference in the values was about 2%. There was a spike in the graph, where the error was 14%, but that can most likely be attributed to the jolting of the mechanism due to gravity.

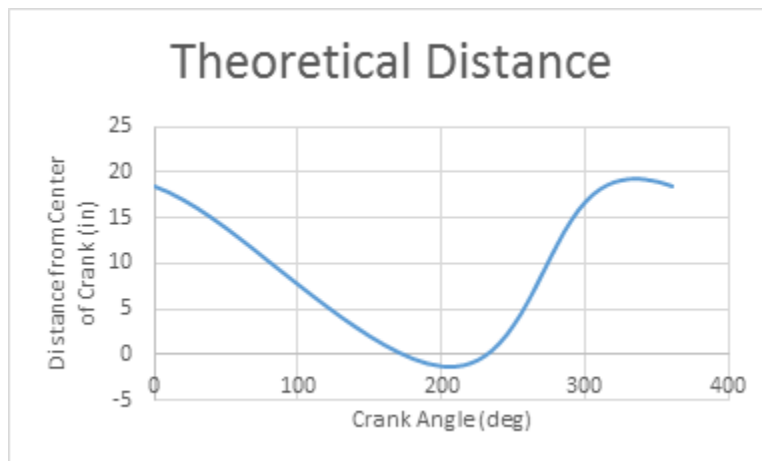


Figure 20: Theoretical Position of the Slider Block

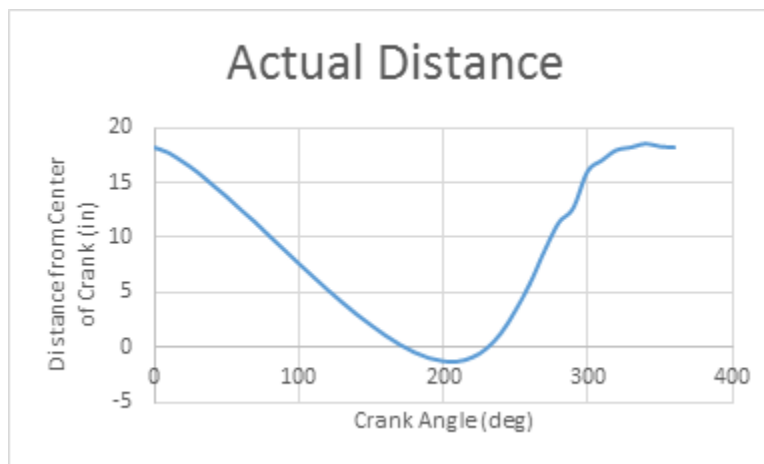


Figure 21: Actual Position of the Slider Block

Summary

The purpose of this project was to design and construct a Whitworth quick return mechanism in order to promote a deeper understand of the design of mechanisms. Beginning with background research from textbooks and professors, the group created design parameters to most accurately portray the way different variables change the time ratio of a quick return mechanism. The creation of a fully functioning mechanism incorporated many different areas of engineering. The final product is effective, but is not without its flaws. There are many areas where the project can be improved; hopefully a future team will take our project into consideration and enhance the performance of the device.

Recommendations

Throughout the duration of our MQP, we came across many ideas and concepts that we would like to have been able to implement but simply did not have the time to incorporate into our design. Some recommendations incorporate aesthetics to better show the way a quick return mechanism works while others are more functionality oriented to help teach other concepts.

Aesthetic Recommendations

Due to the fact that many engineers are visual learners and the quick return mechanism would be moving at high speeds at times, we would recommend having an LED strip behind the top of the slider along its path. This could then be programmed to light up as the point travels back and forth. Due to the slider being locked into place at the bottom of the mechanism, the point would always travel along the same path, but the distance would change due to the height variation

of the crank. This makes it simple because with only one path, a single LED strip could give the user a lot of information regarding the path and even general speeds at which the point was moving. At the faster points, the LEDs could be brighter, and during the slower parts of the rotation, they could dim.

Our mechanism needs a computer to run, as it draws power from it and utilizes the computer for analysis and the display of data. If this project were to be redone, we would like to see the project be independent of other technology. A display to the side of the apparatus itself would show all the data acquired and calculated to show the theory versus analysis we used the computer for. The cost of the project as a whole would increase due to the fact that you would need more than just the Arduino Uno we utilized in the project. A far more intense computer would be needed to not only run the mechanism and all its safety measures but also the display and analysis in real time. Additionally, there would have to be far more intricate electrical work because the device would need to be plugged in directly to a wall outlet. This would add many calculations regarding the power necessary for each part of the device as well as ensuring that something like a power surge wouldn't fry the entire device.

Functional Recommendations

One of the biggest recommendations we have for the project is the ability to change the radius of the crank. Unfortunately, this adds many safety concerns because it would have to be done manually between test runs. However, it is a relatively easy design to implement, as the crank would have multiple pinholes at different radii such that the pin could be moved at will. A simple locking mechanism on the pin could be used to make this work. With the implementation of this design, students would be able to better comprehend how the radius impacts the time ratio of a

quick return mechanism. The design of this added concept can be seen in the Preliminary Design section above.

Another key recommendation is to add a friction mechanism to the design. This would provide students with a more realistic visual of how a quick return mechanism is used. Typically, they are used as cutting tools. Adding friction to the slider when it's in the cutting stroke draws the mechanism away from the more theoretical side typically calculated in a more basic mechanical engineering course and towards the realistic application of the device seen in more complex mechanical engineering courses. There are multiple designs we came up with to add friction in only one direction of the slider. One utilizes ball bearings that lock if pushed in one direction, but roll when pushed the other way. This simulates a change in surface in one direction. Another way would be to incorporate resistance bands to show the wear on the device as it cuts through the material. The most complex way of adding friction would be to create a mechanical design that integrates with the top of the slider joint. The general concept is simple; when the end of the slider changes direction, another mechanism would shift into place to add friction. Then, when the slider changes direction again, the mechanism would disengage. This mechanism would have a rough surface and would apply pressure to the 'cutting' edge. It is also easy to change the amount of friction simulated in this design because a dial could tighten a spring to change how close the friction mechanism is to the edge. This would add more pressure to the edge when closer, simulating more friction. The concept of a friction mechanism being added to the project can be seen in Preliminary Design section above.

Coding Recommendations

A large goal of this MQP was to show the difference between theory and practice in real time so that students could better see the relationship between the two. We unfortunately weren't able to show the comparison in real time. The data acquired through the accelerometer was relayed back to the computer. At that point, the user would have to copy and paste the data into an Excel spreadsheet to quickly graph it for the class. Obviously, this is not ideal for the user or audience but there are programs that can be used in tandem with Arduino (such as GoBetwino) to start Excel, send the accelerometer data from Arduino to an Excel sheet, and graph it. This would drastically increase the ways in which the device would help students comprehend the analysis and application of quick return mechanisms.

Another recommendation for this area of the project, would be to make it an option to input the desired time ratio into the Arduino program and have it set the mechanism up to run at the desired time ratio. This would make it significantly easier for professors to show students exactly what a time ratio is and how it is changed based on different variables.

Other Recommendations

This project requires more than just Mechanical Engineering majors to unlock its full potential. A Robotics or Electrical and Computing Engineer is necessary to properly integrate the device with technology in a sophisticated manner. As Mechanical Engineers without any real background in coding, we had to spend a lot of valuable time focusing on developing a functional program for the device instead of how to make it a better learning tool.

One of our greatest struggles during the creation of our project was receiving faulty parts from manufacturers. For example, during the final stages of testing, the Arduino motor shield we

were using caught fire and caused our motors to spin out of control, damaging the physical mechanism itself. This set us back weeks in development and greatly impeded upon the completion of the project. In the future, we made sure to test each part extensively before incorporating them into the design. It is crucial for a team to do this testing despite how tedious it seems, so that they can avoid being set back weeks in production.

An important part of learning a concept is being able to see specific aspects of it in motion. While this is a great teaching tool and model of a quick return mechanism, we realized that the final product was moving too quickly to see the general concepts with the naked eye. Therefore, it would be beneficial to incorporate variability in the motor speed such that it can go slow enough for the concepts to be seen easily, while keeping the fast-paced function still possible to portray how the mechanism is used in factories across the world.

Upon completion of our project, we found flaws we had not anticipated in our design. For one, the links were heavier than we had hoped, which put a lot of unnecessary stress on the crank motor. Shaving down the links and making them thinner would have allowed for the motor to be under less stress, and have the mechanism flow smoother. Friction between the joints was also a pertinent issue which could be easily solved by placing felt between the links. Overall, the usage of clevis pins with cotter pins worked well. However, the locking cotter pin would occasionally get caught in the slot of the slider arm and jam the mechanism. Fortunately, a washer fixed the problem, but a better option for that joint should be found for future projects. We used a pulley system to move the elevator that adjusted the height of the crank. A threaded rod mechanism would have acted more consistently, as the pulley system went down faster than it went up due to the weight of the elevator compared to the strength of the motor.

References

People Interviewed:

Sarah Jane Wodin-Schwartz

Pradeep Radhakrishnan

Mechanics of Machines, V. Ramamurti

Bellisario, S. (2014, May 1). *Design and Construction of a 6 Bar Kinematic Quick Return Device for use as a Demonstration Tool*. Retrieved from:

https://web.wpi.edu/Pubs/E-project/Available/E-project-050114-141830/unrestricted/MQP_Paper.pdf

Norton, R. L. (2000). *Machine design: An integrated approach*. Upper Saddle River, N.J: Prentice Hall.

Norton, R. L. (2012). *Design of Machinery: an introduction to the synthesis and analysis of mechanisms and machines*. New York, N.Y.:McGraw-Hill.

Appendices

Appendix A: MQP Arduino Code

```
/** ELEVATOR MOTOR CONTROL:
```

```
    Motor on Channel A -> fixed speed to act as elevator, input 0 to 99 to set height
```

```
    Motor on Channel B -> fixed speed, input 1 for clockwise, 0 for counterclockwise
```

```
#include "DualMC33926MotorShield.h"
```

```
DualMC33926MotorShield md;
```

```

//*****BEGIN DEFINITIONS*****//

#define SPEED 200

/** Accelerometer controls */

#define X_PIN A5

/** Push button pins */

#define BUTTON1 A3

#define BUTTON2 2

#define BUTTON3 3

/** used to convert displacement to amount of time in ms to spin a motor */

#define SCALE_FACTOR 28

/** amount of milliseconds between BUTTON1 reads */

#define SAFETY 100

/** number of accelerometer measurements */

#define NUM_READINGS 250

//*****END DEFINITIONS*****//

//*****BEGIN GLOBALS*****//

// height of elevator, assumes initial position = 0

int currentHeight = 0;

signed int x_velocity[NUM_READINGS];

signed int x_acceleration[NUM_READINGS];

//*****END GLOBALS*****//

//*****BEGIN PROTOTYPES*****//

void setup(void);

void loop(void);

void stopAllMotors(void);

```

```
int getElevatorInput(void);

void moveElevatorUp(int displacement);

void moveElevatorDown(int displacement);

int getMotor2Direction(void);

void spinMotor2Clockwise(void);

void spinMotor2Counterclockwise(void);

void getAccelerometerData(int index);

void calculateVelocity(void);

void printData(void);

//*****END PROTOTYPES*****//

/** setup function

    only runs on startup

    initializes pin and direction for each I/O

*/

void setup() {

    Serial.begin(9600);

    md.init();

    pinMode(X_PIN, INPUT);

    pinMode(BUTTON1, INPUT);

    pinMode(BUTTON2, INPUT);

    pinMode(BUTTON3, INPUT);

    stopAllMotors();
```

```
Serial.println("DC MOTOR CONTROLLER");

moveElevatorDown(100);

}

/** runs continuously
    main function of code
*/

void loop() {

    Serial.flush();

    int desiredHeight = getElevatorInput();

    Serial.print("Move to position: ");
    Serial.println(desiredHeight);

    if (desiredHeight > currentHeight)
    {
        moveElevatorUp(desiredHeight - currentHeight);
    }
    else
    {
        moveElevatorDown(currentHeight - desiredHeight);
    }

    int motor2direction = getMotor2Direction();

    if (motor2direction)
```

```
{
    spinMotor2Clockwise();
}
else
{
    spinMotor2Counterclockwise();
}

stopAllMotors();
calculateVelocity();
printData();

delay(1000);
}

/** this function puts on the brakes for both motors
    called for safety and to protect motors
    most function calls are redundant
*/
void stopAllMotors() {
    md.setM1Speed(0);
    md.setM2Speed(0);
}
```

```

/** prompts user to input height of elevator
    polls for user input
    only accepts integers 0-99
    returns the user's desired position of the elevator
*/

int getElevatorInput() {
    Serial.flush();

    Serial.println("Input number between 0 and 99 to change elevator height: ");

    // wait for serial input
    delay(10);

    while (!Serial.available());

    if( Serial.available()){
        int serial_input = Serial.parseInt() % 100;

        return serial_input;
    }
}

/** moves elevator up to desired position
    input the displacement between where the elevator is and where the user wants it to go
    when elevator is moving,
*/

void moveElevatorUp(int displacement) {

    int timeToMove = displacement * SCALE_FACTOR;

    Serial.print("Time to move: ");

    Serial.println(timeToMove);
}

```

```

    md.setM1Speed(SPEED);

Serial.println(digitalRead(BUTTON1));

Serial.println(digitalRead(BUTTON2));

Serial.println(digitalRead(BUTTON3));

int timeMoved = 0;

while(!digitalRead(BUTTON1) && !digitalRead(BUTTON2) && timeMoved < timeToMove)
{
    delay(SAFETY);

    timeMoved += SAFETY;
}

md.setM1Speed(0);

currentHeight += (timeMoved / SCALE_FACTOR);

Serial.print("Elevator now at position: ");

Serial.println(currentHeight);
}

/** moves elevator down to desired position
    input the displacement between where the elevator is and where the user wants it to go
*/

void moveElevatorDown(int displacement) {

    Serial.println(displacement);

    int timeToMove = displacement * SCALE_FACTOR;

```



```
Serial.print("Time to move: ");
Serial.println(timeToMove);

md.setM1Speed(-1*SPEED);

int timeMoved = 0;
while (!digitalRead(BUTTON1) && !digitalRead(BUTTON2) && !digitalRead(BUTTON3) &&
(timeMoved < timeToMove))
{
  delay(SAFETY);
  timeMoved += SAFETY;
  Serial.println(timeMoved);
  Serial.println(digitalRead(BUTTON1));
  Serial.println(digitalRead(BUTTON2));

  Serial.println(digitalRead(BUTTON3));
}
md.setM1Speed(0);
if( digitalRead(BUTTON3) ){
  currentHeight = 0;
}
else{
  currentHeight -= (timeMoved / SCALE_FACTOR);
}
Serial.print("Elevator now at position: ");
Serial.println(currentHeight);
```

```
}
```

```
/** receives user input for direction of motor
```

```
    won't accept values out of range
```

```
*/
```

```
int getMotor2Direction() {
```

```
    int serial_input = 420;
```

```
    while (serial_input != 0 && serial_input != 1) {
```

```
        Serial.println("Input direction (1 for clockwise, 0 for counterclockwise): ");
```

```
        while (!Serial.available());
```

```
        serial_input = Serial.parseInt();
```

```
    }
```

```
    return serial_input;
```

```
}
```

```
/** spins motor 2 clockwise and gets accelerometer data
```

```
    takes in speed and time as an input
```

```
    function call to get accelerometer data while spinning motor 2
```

```
*/
```

```
void spinMotor2Clockwise() {
```

```
    Serial.println(digitalRead(BUTTON1));
```

```
    Serial.println(digitalRead(BUTTON2));
```

```
    Serial.println(digitalRead(BUTTON3));
```

```
    md.setM2Speed(100);
```

```
int i = 0;
for (i = 0; i < NUM_READINGS; i++)
{
  if(digitalRead(BUTTON2) || digitalRead(BUTTON3))
  {
    break;
  }

  getAccelerometerData(i);
  delay(10);
}
while (!digitalRead(BUTTON2) && !digitalRead(BUTTON3));

Serial.println("3");

md.setM2Speed(0);

// int i = 0;
// while(!digitalRead(BUTTON2) && !digitalRead(BUTTON3))
// {
//   if(i < NUM_READINGS)
//   {
//     getAccelerometerData(i);
//     delay(10);
```

```

// i++;
// }
// }

// delay(1000);
// Serial.println(digitalRead(BUTTON1));
//Serial.println(digitalRead(BUTTON2));
//
//Serial.println(digitalRead(BUTTON3));
// md.setM2Speed(0);
}

/** spins motor 2 counterclockwise and gets accelerometer data
    takes in speed and time as an input
    function call to get accelerometer data while spinning motor 2
*/
void spinMotor2Counterclockwise() {
    md.setM2Speed(-100);

    int i = 0;
    for (i = 0; i < NUM_READINGS; i++)
    {
        if(digitalRead(BUTTON2) || digitalRead(BUTTON3))
        {
            break;
        }
    }
}

```

```

    getAccelerometerData(i);
    delay(10);
}
while (!digitalRead(BUTTON2) && !digitalRead(BUTTON3));

Serial.println("3");

md.setM2Speed(0);
}

/** gets accelerometer data
    stores data in global 'array'
    data is not in terms of m/s^2
*/
void getAccelerometerData(int index)
{
    /* Sensitivity of accelerometer is 300 mV/g
    * 4V/1023 ~ 4.88V Range
    *  $0.300 / (5/1023) = 61.38 \sim 1g$ 
    */
    delay(250);
    x_acceleration[index] = analogRead(X_PIN) - 331;
}

/** calculates velocity using accelerometer data

```

```
*/  
  
void calculateVelocity() {  
    int i = 0;  
    for (i = 0; i < NUM_READINGS; i++) {  
        if (i > 0) {  
            x_velocity[i] = (x_acceleration[i]) + x_velocity[i - 1];  
        }  
        else {  
            x_velocity[i] = x_acceleration[i];  
        }  
    }  
}  
  
/** prints acceleration and velocity  
*/  
  
void printData() {  
    Serial.println("Velocity data: ");  
    int i = 0;  
    for (i = 0; i < NUM_READINGS; i++) {  
        Serial.println(x_velocity[i]);  
    }  
    Serial.println("Acceleration data: ");  
    int j = 0;  
    for (j = 0; j < NUM_READINGS; j++) {  
        Serial.println(x_acceleration[j]);  
    }  
}
```

}

Appendix B: Position, Velocity, and Acceleration Analyses

$R_c := 2.5\text{in}$ Radius of the crank

$R_{sl} := 23.875\text{in}$ Length of the slider link

$R_l := 9\text{in}$ Length of the link

$R_{gh} := 16.5\text{in}$ Maximum distance from ground of slider link to ground of crank

$R_{gl} := 5.8\text{in}$ Minimum distance from ground of slider link to ground of crank

$R_{sv} := 22.5615\text{in}$ Vertical distance from ground to slider block

$$\alpha_h := 2 \cdot \arccos\left(\frac{R_c}{R_{gh}}\right) \quad \alpha_l := 2 \cdot \arccos\left(\frac{R_c}{R_{gl}}\right)$$

$x := 0, 0.1.. 2\pi$

$\theta_1(x) := x \cdot \text{rad}$ The location of the crank (angle)

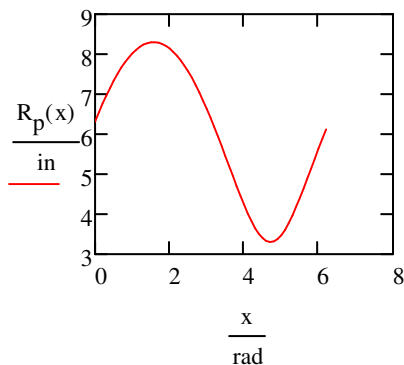
$$\frac{\alpha_h}{2\pi - \alpha_h} = 0.823 \quad \text{Time ratio when crank is in highest position}$$

$$\frac{\alpha_l}{2\pi - \alpha_l} = 0.558 \quad \text{Time ratio when crank is in lowest position}$$

When the crank is at its lowest position:

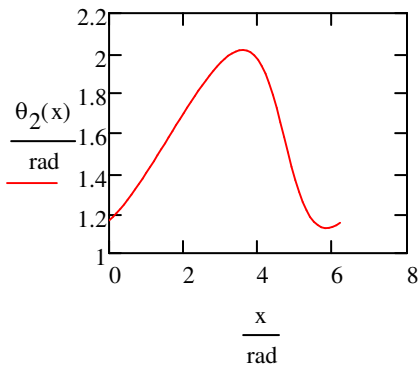
$$R_p(x) := \sqrt{R_{gl}^2 + R_c^2 - 2 \cdot R_{gl} \cdot R_c \cdot \cos\left(\theta_1(x) + \frac{\pi}{2}\right)}$$

Vertical height of the pin connecting the crank to the slider link



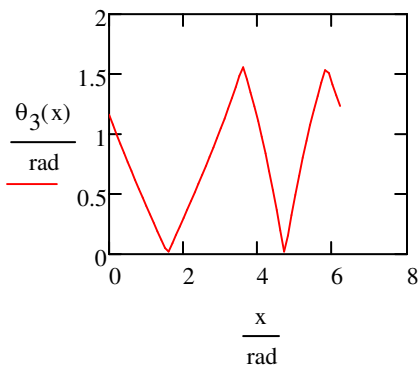
$$\theta_2(x) := \frac{\pi}{2} - \arcsin\left(\frac{R_c}{R_p(x)} \cdot \sin\left(\theta_1(x) + \frac{\pi}{2}\right)\right)$$

Angle between the slider link and the horizontal



$$\theta_3(x) := \begin{cases} \left[\arccos \left[\frac{\left(R_c^2 + R_p(x)^2 - R_{gl}^2 \right)}{2 \cdot R_c \cdot R_p(x)} \right] \right] & \text{if } x \leq \frac{3\pi}{2} - \frac{\alpha_1}{2} \vee x \geq \frac{3\pi}{2} + \frac{\alpha_1}{2} \\ \left[\pi - \arccos \left[\frac{\left(R_c^2 + R_p(x)^2 - R_{gl}^2 \right)}{2 \cdot R_c \cdot R_p(x)} \right] \right] & \text{otherwise} \end{cases}$$

Angle between transmission and crank.



$$X_{sl}(x) := R_{sl} \cdot \cos(\theta_2(x))$$

$$Y_{sl}(x) := R_{sl} \cdot \sin(\theta_2(x))$$

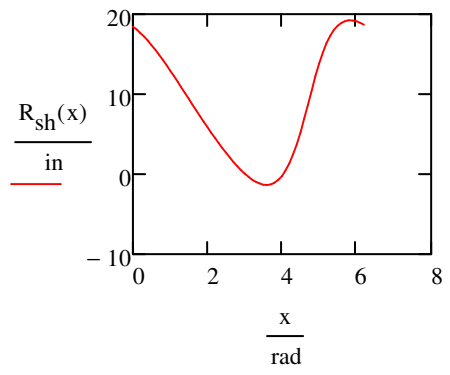
The x and y components of the slider link

$$\theta_4(x) := \arcsin \left(\frac{R_{sv} - Y_{sl}(x)}{R_1} \right)$$

Angle of the link to the horizontal

$$R_{sh}(x) := X_{sl}(x) + R_1 \cdot \cos(\theta_4(x))$$

Horizontal position of the slider block

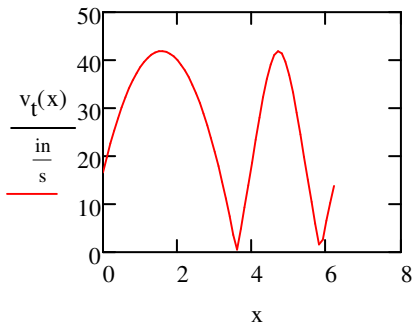


Velocity Analysis

$$\omega_c := 160\text{rpm} \quad \text{Velocity of the crank}$$

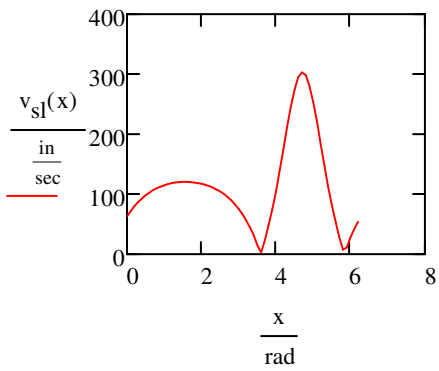
$$v_p := \omega_c \cdot R_c \quad \text{Velocity at the pin connecting the crank to the slider link}$$

$$v_t(x) := v_p \cdot \cos(\theta_3(x)) \quad \text{Velocity of transmission pin}$$

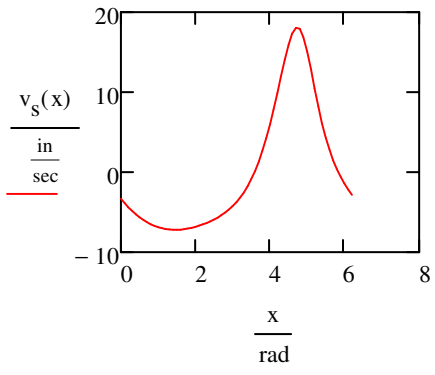


$$\omega_{sl}(x) := \frac{v_t(x)}{R_p(x)}$$

$$v_{sl}(x) := \omega_{sl}(x) \cdot R_{sl}$$

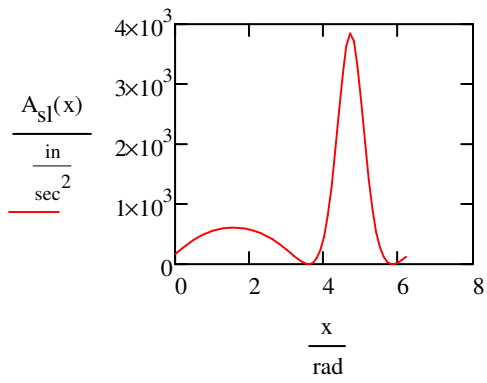


$$v_s(x) := \frac{d}{dx} R_{sh}(x)$$

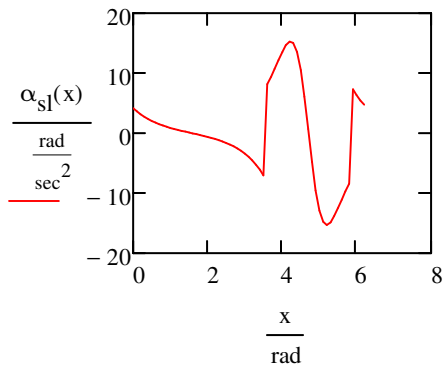


Acceleration Analysis

$$A_{sI}(x) := R_{sI} \cdot \omega_{sI}(x)^2$$



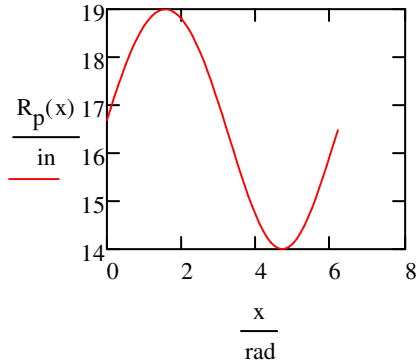
$$\alpha_{sI}(x) := \frac{d}{dx} \omega_{sI}(x)$$



When the crank is at its highest position:

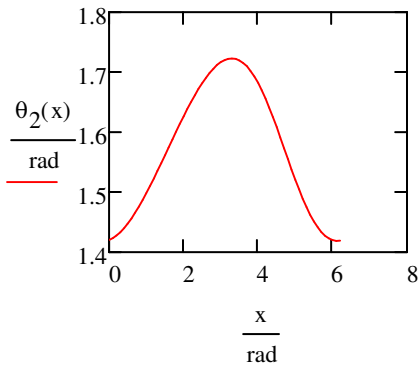
$$R_p(x) := \sqrt{R_{gh}^2 + R_c^2 - 2 \cdot R_{gh} \cdot R_c \cdot \cos\left(\theta_1(x) + \frac{\pi}{2}\right)}$$

Vertical height of the pin connecting the crank to the slider link



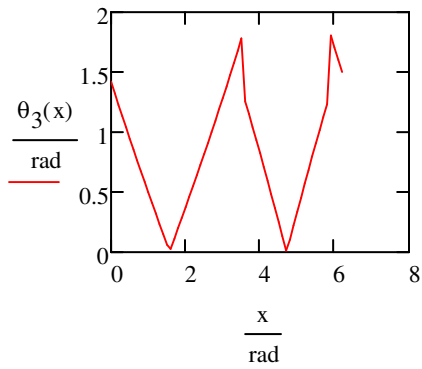
$$\theta_2(x) := \frac{\pi}{2} - \arcsin\left(\frac{R_c}{R_p(x)} \cdot \sin\left(\theta_1(x) + \frac{\pi}{2}\right)\right)$$

Angle between the slider link and the horizontal



$$\theta_2(x) := \begin{cases} \arccos\left[\frac{\left(R_c^2 + R_p(x)^2 - R_{gh}^2\right)}{2 \cdot R_c \cdot R_p(x)}\right] & \text{if } x \leq \frac{3\pi}{2} - \frac{\alpha_1}{2} \vee x \geq \frac{3\pi}{2} + \frac{\alpha_1}{2} \\ \left[\pi - \arccos\left[\frac{\left(R_c^2 + R_p(x)^2 - R_{gh}^2\right)}{2 \cdot R_c \cdot R_p(x)}\right]\right] & \text{otherwise} \end{cases}$$

Angle between transmission and crank.



$$X_{sl}(x) := R_{sl} \cdot \cos(\theta_2(x))$$

$$Y_{sl}(x) := R_{sl} \cdot \sin(\theta_2(x))$$

The x and y components
of the slider link

$$\theta_4(x) := \arcsin\left(\frac{Y_{sl}(x) - R_{sv}}{R_l}\right)$$

Angle of the link to the horizontal

$$R_{sl}(x) := X_{sl}(x) + R_l \cdot \cos(\theta_4(x))$$

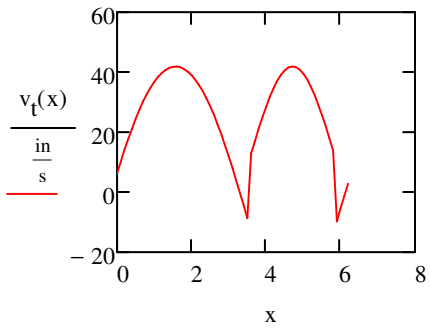
Horizontal position of the slider block

Velocity Analysis

$$\omega_c := 160 \text{rpm} \quad \text{Velocity of the crank}$$

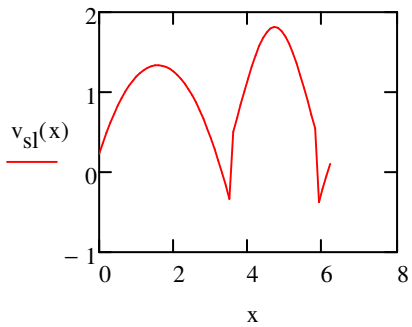
$$v_p := \omega_c \cdot R_c \quad \text{Velocity at the pin connecting the crank to the slider link}$$

$$v_t(x) := v_p \cdot \cos(\theta_3(x)) \quad \text{Velocity of transmission}$$



$$\omega_{sl}(x) := \frac{v_t(x)}{R_p(x)}$$

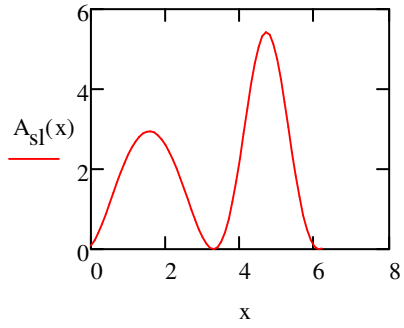
$$v_{sl}(x) := \omega_{sl}(x) \cdot R_{sl}$$



$$v_s(x) := \frac{d}{dx} R_{sh}(x)$$

Acceleration Analysis

$$A_{sI}(x) := R_{sI} \cdot \omega_{sI}(x)^2$$



$$\alpha_{sI}(x) := \frac{d}{dx} \omega_{sI}(x)$$

