Robotic Musical Chimes

by Peter Buterbaugh, Dylan Flegel, Truman Larson, and Michael Rossetti

Abstract

This short paper discusses the design of a robotic musical chime machine made out of 3D printed parts, electronics, and other materials that can be easily obtained from a hardware store. This machine is an extension of a previous automatic chimes instrument created by two of the team members. The design of this instrument was chosen to act as a centerpiece and created unique challenges related to the display of physical movement while playing and the ability to play certain musical pieces. The team encountered multiple issues through the machining and build process, which led us to some new discoveries about how to design our machine and led us to make sacrifices that determined the end design and capabilities that did not fall in line with our original goals. Through the design process the team found ways to increase the performance of the machine, while also maintaining its shape and charm.

Concept, Motivation, and Purpose

Our general idea was to create a set of automatic chimes (or tubular bells) that are striked with a solenoid controlled mechanism. We planned to use a circular design with the chimes around a central turret that can rotate and strike all of the chimes. Within this turret, we planned to have multiple different striking tools/mechanisms to create unique sounds with the chimes. We had also considered adding a toggleable dampener to be able to control the duration of the notes on each strike.

In comparison to the previous automatic chimes design, we wanted to focus on the aesthetics of the instrument as well. We looked into adding percussion that ties the design together and adds different sounds. The circular shape also gives us the ability to

change the positions of the bells and provides more freedom both musically and physically.

Prior Art

Automatic Tubular Bells: 2E0WDH [1]

The circular design of this makes the machine very versatile as one striker can hit multiple chimes. This is the easiest way to achieve this, the other being achieved through linear motion which is more difficult, and eliminates many of the solenoids that were needed in the past autochimes project. However, the rotating turret is more difficult to execute as it would require a stepper motor (as used in the video) or servo to control the rotation before striking. This reduces the speed of the machine and eliminates the possibility of chords. The central turret does open the possibility of using a variety of striking mechanisms, without the need to duplicate them for each individual chime. If one were to replace or even add an additional striker to the center, the potential for new sounds increases by the number of chimes, without needing to modify a striker for each chime.

The tubes on this design are also fixed to the base of the machine as opposed to suspended by string as they were in the past autochimes project. This offers greater stability to the chimes as the tubes in the past autochimes project would often hit the frame of the machine.

Enhanced Tubular Bells: Tolaemon [2] - [3]

“Enhanced Tubular Bells (Automatic Tubular Bells)” uses a linear layout with two different, mirrored “tracks”, differing from the singular track layout that our group is using as a foundation or point of reference. As this design is better optimized for space, there is potential to expand the dynamic range of the instrument by increasing the quantity of tubes, should future revisions be made. It should be noted that the method of actuation, although powered by electromagnetic solenoids similar to those on the past autochimes project, uses a simple lever arm linkage as opposed to using the

head of the solenoid itself as a striker. This may serve to increase the mechanical advantage and, accordingly, force of the striker, consequently affecting the intensity of the emitted sound.

The modularity and customizability of the Enhanced Tubular Bells should be noted, especially in properly commenting on capability. The striker assembly is mounted in such a way that, once loosened, enables axial (up-down) motion along extruded metal tubing. This feature likely has a direct effect on pitch or timbre, and can thus be exploited when determining if the instrument is reusable/flexible. Though, it should be noted that displacing the assembly from its initial point will also displace all of the other actuators, mounted on the same frame.

Tremolo Chimes: Steven Kemper [4]

The auto wind chimes introduce a new method for exciting the tubes. This method emulates a wind effect that would be found in wind chimes which creates a unique sound from the other methods. With the concept of vibration to excite the chimes, we could potentially implement an alternative method of striking that could create a unique extended note and increase the variety that the machine can produce. This vibration could be achieved via repeated quick striking or vibration with a motor.

Jingle Bells (Previous Project): Evelyn Tran, Declan Murphy, Peter Buterbaugh, and Dylan Flegel [5]

This project was completed in the Making Music With Machines class and was the initial concept for the current project. There are eight bells, which create a major scale in the key of C. It uses a linear set of chimes suspended with string, with an individual solenoid corresponding to each note. It is also capable of being played with a MIDI controller.

Swayway MIDI Chimes: Doria Fan [6]

The auto wind chimes introduce a new method for exciting the tubes. This method emulates a wind effect that would be found in wind chimes which creates a unique sound from the other methods. With the concept of vibration to excite the

chimes, we could potentially implement an alternative method of striking that could create a unique extended note and increase the variety that the machine can produce. This vibration could be achieved via repeated quick striking or vibration with a motor.

Chaosmose: Chung Song [7]

The auto wind chimes introduce a new method for exciting the tubes. This method emulates a wind effect that would be found in wind chimes which creates a unique sound from the other methods. With the concept of vibration to excite the chimes, we could potentially implement an alternative method of striking that could create a unique extended note and increase the variety that the machine can produce. This vibration could be achieved via repeated quick striking or vibration with a motor.

Preliminary Design

The overall form factor of our instrument build would resemble a cylinder; a circular base, cut out of a material such as plastic or wood, would have extruded metal pipes to serve as our bells/chimes fixed to it. Multiple freely-rotating strikers (taking the form of electromagnetic solenoids tipped with plastic, rubber or cloth) would enable the instrument to emit sound using a MIDI input.

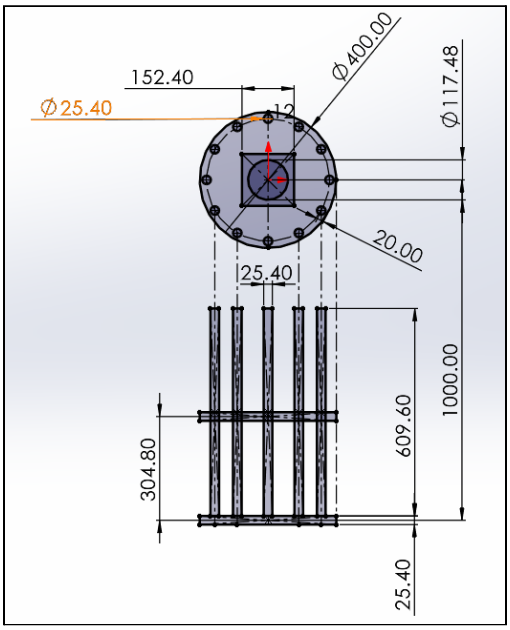


Image 1: A diagram of the initial chime holding system

In making this design choice, our objective is to be able to play multiple notes in rapid succession without having to rely on one singular striker to move into position, expanding the potential for our autochimes to play more complex pieces.

Timbre:

We will primarily try to use chimes or chime like materials to emulate tubular bells in timbre. The exact timbre would depend on the material we use for the pipes, but we are learning towards copper pipes for their availability and cost.

Time:

Since our primary focus for this instrument is the use of chime, it's important to be able to replay a melody consistently. This means that the time between beats should remain the same throughout a piece given a consistent tempo. The challenge of using a rotating striker is that the physical distance between the chimes that need to be hit will be different throughout a melody. This creates a timing challenge as we would have to account for these different distances in order to maintain a standard tempo throughout. The max timing constraint for a rotating design would be time that it takes for the striker to reach the opposite side of the rotation. A linear design would not have any issue with chords or timings as every striker would be independent.

Pitch:

Since we will be using pipes, the pitch accuracy would be determined by the length of the pipes. These lengths can be estimated mathematically and confirmed via a tuner. Intervallic leaps using a rotating design would be limited by the speed of the rotation while there is no limit for a linear design. A rotating design would most likely be monotonic, but, with multiple strikers, would be able to hit multiple notes.

Dynamics and Articulation:

The constraint of using solenoids would mean that we could only use one striking speed and thus be limited in dynamics if just using one material for striking. However, we could use a softer striking material to produce a quieter sound. If we use multiple

strikers, there is a possibility for different dynamics as we could use a different striking material for a different volume.

Visibility:

For visibility in the rotating design, we feel that the rotation would give an extra factor to the performance that would justify a lot of the drawbacks mentioned in the other sections. The linear design would not have this same visual factor because it will have less motion. We think that the visual activity provided by a rotating design would be best.

Interaction and Autonomy:

The goal is to make this completely autonomous within a song. That is once a song starts, there is no human interaction. But there is a potential between song plays to change out the pipes to achieve a different scale or range in order to play a larger variety of music.

Sonic Objects:

The sound will be produced by the chimes when they are stuck by a solenoid, either with or without a potential material change.

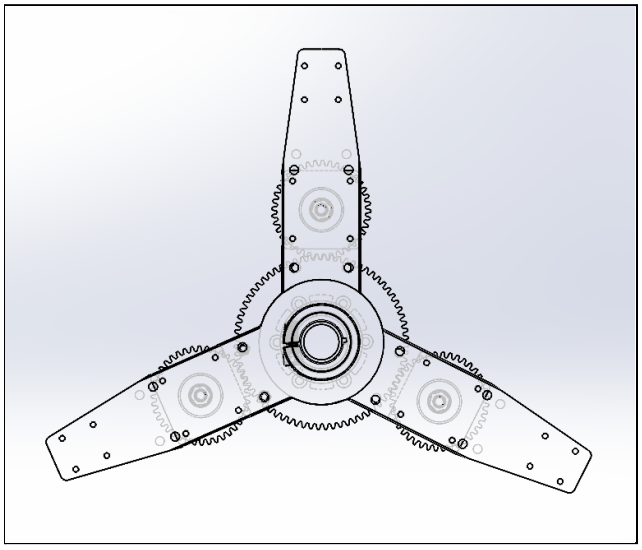


Image 2: Original design for the rotating center platform

Excitation:

The excitation of the chimes comes from the rotating solenoids. They rotate into position to strike the chime and will hit it.

Motion:

The main motion of the machine comes from the rotating platform (lazy susan or turntable) in the center of the design. This platform will hold the solenoids that strike the chimes. We think this design allows us to uphold our goals for the visual aspect of the instrument while also enabling us to strike multiple chimes at a time and with different materials throughout a performance.

Final Design

When creating the aesthetic side of our machine, we wanted it to be a "centerpiece", which is why we used a circular design. This design has the pipes arranged in a circle and includes a greater element of motion as the striker will have to move in order to hit the different notes. This focus on movement and the circular structure will sacrifice speed of play by the machine, but the team was ok with this sacrifice due to our goal of making an interesting looking machine.

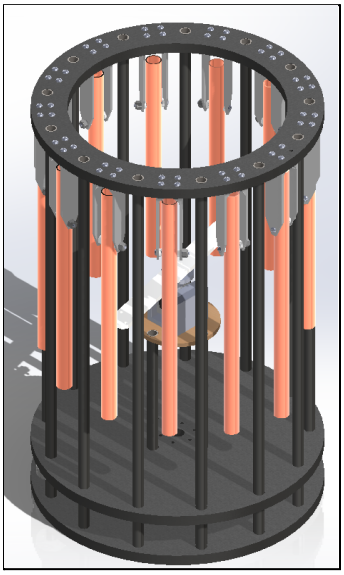


Image 3: The final CAD design of the machine

The primary material we will be using are copper pipes. These were chosen for their availability and reliability. We plan to include a full chromatic scale with both ends of the octave included. We plan for this to be C4 to C5, but it will require more testing to know the capability and sound of the copper pipes. The copper pipes will be struck with a solenoid covered in cloth to create the noise, which was the best sound created from our testing of multiple methods.

Our initial plan is to have 1 solenoid that can rotate to hit all of the notes. However, we still need to evaluate the potential of adding additional independently rotating solenoids. Ideally, we would have three that could all rotate independently. This would increase our potential speed and allow for chords. Three solenoids is our ultimate goal, but initially we are going to try to get one functional and expand from there. With only one solenoid, we have determined that the machine will be able to play in 4/4 time in 667 milliseconds.

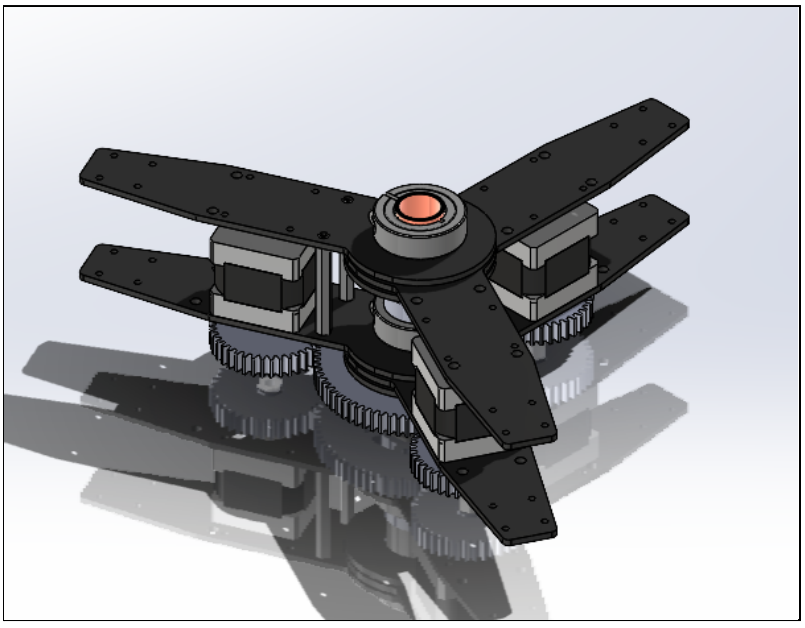


Image 4: Design for 3 independently rotating solenoids

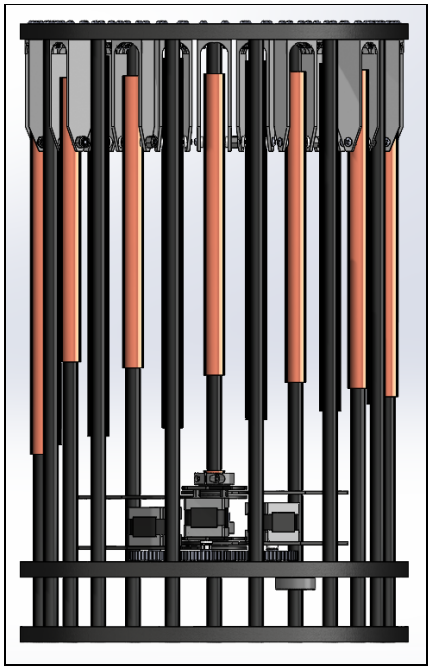
The solenoids will directly hit the chimes but they will be covered in cloth to give a less metallic sound. This material could feasibly be changed between use if needed.

The primary motion in this machine will be the centrally rotating solenoid(s) that will rotate to hit the chimes. The solenoid(s) will activate when it reaches the desired pipe and strike it to create the noise.

We will suspend the chimes from the top, fixing them such that they have a free range of motion in the direction of the strike. This will be supported by outer supports connected to a base support structure. The inner carriage mechanism that houses and controls the rotating solenoid will be in the center surrounded by the chimes.

When creating and hanging the copper pipes, each pipe will need to have been accurately cut to the proper length, and will need to have its hang point at the proper position. These values can be obtained from a spreadsheet from a DIY chime tutorial [8] that specified the lengths and hang points for each note and material we could use. Using these we can accurately estimate the lengths and hang points to achieve as close to an ideal sound as possible.

Additionally, we plan to allow the copper pipes to move freely along the axis they are struck. This will prolong the duration of the note but still cut it off as it will fall back into a soft resting point. This will be part of a passive dampening system. When the pipe returns back toward the starting position after being struck, it will hit a felt pad area that both stops the movement of the chime and dampens the sound made.



Images 5: CAD showing the suspension system for the copper pipes

The electronics for this machine consists of the two solenoids, the servo motor, an arduino, a breadboard, and the power and wiring necessary to connect everything together. It is fairly simple and allows us to use the code from the arduino to control the angle of the servo and fire the solenoids when we are facing a chime.

Our plan for the code is to create a relatively simple program that can take in a note and rotate the servo to play the correct note. This will involve knowing the angles of each of the notes beforehand and we will also need to know things like how many servos we end up having so that we can split the work of hitting the notes between them.

Bill of Materials

Link: Bill of Materials

Evaluation and Modifications

Pipe Modification:

To create the pipes to produce the sound of the instrument, we originally were going to use ½" copper pipes and measure them using a DIY spreadsheet we found

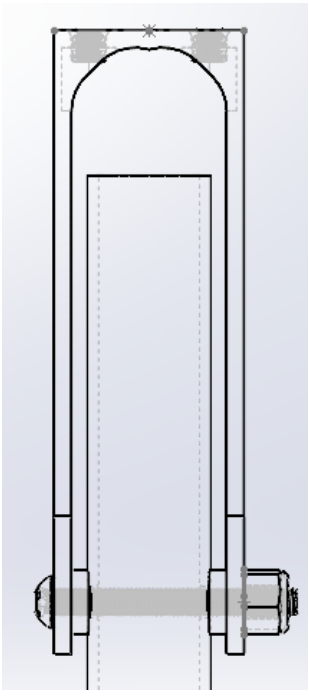


Image 6: The chime hanging mechanism that suspends each pipe at the proper node

online [8]. When we got these pipes and cut the first one, we found that it was sharper than the intended note.
This is problematic because we cut with an extra margin, so if anything, it should've been flatter.

We also noticed that the sound it produced was much quieter than we were anticipating. Knowing this, we implemented several measures to both fix the tuning problem and increase the volume of the notes. We knew that the pipes from a previous project were ¾" and produced a louder sound. We also noticed that correctly identifying the points in the pipe where the wave was at a node (aka "hang points") produced the best sound and clearest note possible, due to the vibrations created inside of the pipe.

During the build process when we incorrectly identified the node, we would primarily hear different tones being created that contradicted what we expected. Combining the larger pipe and the correct hang points vitally allowed us to increase the volume of the machine and create more robust consistent sounds from the pipes.

Mechanics:

Mechanically, we focused primarily on improving the design of the central rotating arm. While prototyping, we settled on a much simpler design, with the servo directly attached to the rotating arm. This design has a few advantages in that we do not need as many parts, and that it is easier to assemble and prototype. On the rotating arm, we improved upon the singular solenoid design to include two solenoids placed on opposite sides of the arm. This design has the advantage of limiting the rotation required to hit notes since it will split the responsibility between two solenoids.

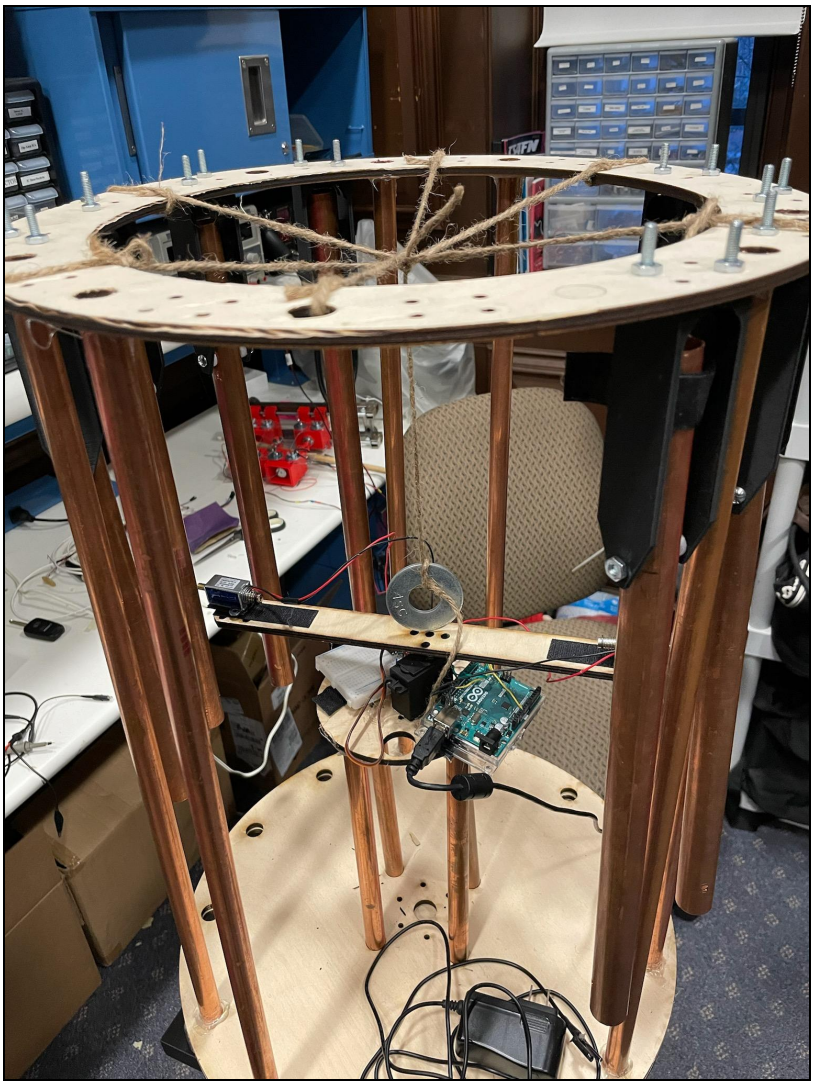


Image 7: Centering the servo and the 2 solenoid rotating platform

Code:

This week we began coding what will be the final product to start solving some of the problems that we know will encounter. This section will lay out these problems and our proposed solutions for solving them.

The main thing we are trying to do is convert a note into an angle we need to rotate to, and the correct servo we need to use once we are at that angle. To do this, we

need to configure the state of the machine into the program so that these angles can be obtained. The following aspects must be known:

- Angles of every pipe-position relative to the center
- The notes that are at each pipe-position
- The domains of each solenoid (i.e what pipe-positions each of the two servos are responsible for hitting.

The term “pipe-position” is a general term used to describe the place that a given pipe is mounted. In total there will be 13 pipe positions and they will be indexed starting at 0. The angles from the center are needed to know how far to rotate to be able to strike each pipe. The final values of these were found through measuring and verified through trial and error. We also need to know which notes are at which pipe position so that we know to which position we need to go to hit a particular note. This configuration can be changed in the code to account for different pipe arrangements. The last piece is the responsibilities that each solenoid has. Since we have two solenoids rotating on the singular servo, which can only rotate 180 degrees), both solenoids must be assigned their responsibilities so that the program will know how to rotate correctly.

With these configurations, we are able to convert a given note into the angle the servo must rotate, and the solenoid that must activate to give the correct sound. The steps are as follows:

1. Convert the raw note to a pipe-position
    a. This is easy enough. The note-pipe-position relation is defined in the configuration so we can directly use that.
2. Obtain the solenoid that has this pipe-position in its domain
    a. Again, this information is defined in our configuration so it's only a matter of searching for a particular value in the domain array of each solenoid and finding a match.
3. Convert the pipe-position to the angle (accounting for the solenoid we are using)
    a. Calculating this relies on the fact that the solenoids are 180 degrees apart. From this fact, and knowing which solenoid is centered at 0 degrees, we

can subtract 180 degrees from one of the solenoids to give the absolute angle that the servo must rotate to correctly be able to hit the given pipe.

4. Now that the angle and solenoid are known, we will rotate the servo and activate the given solenoid, playing the note.

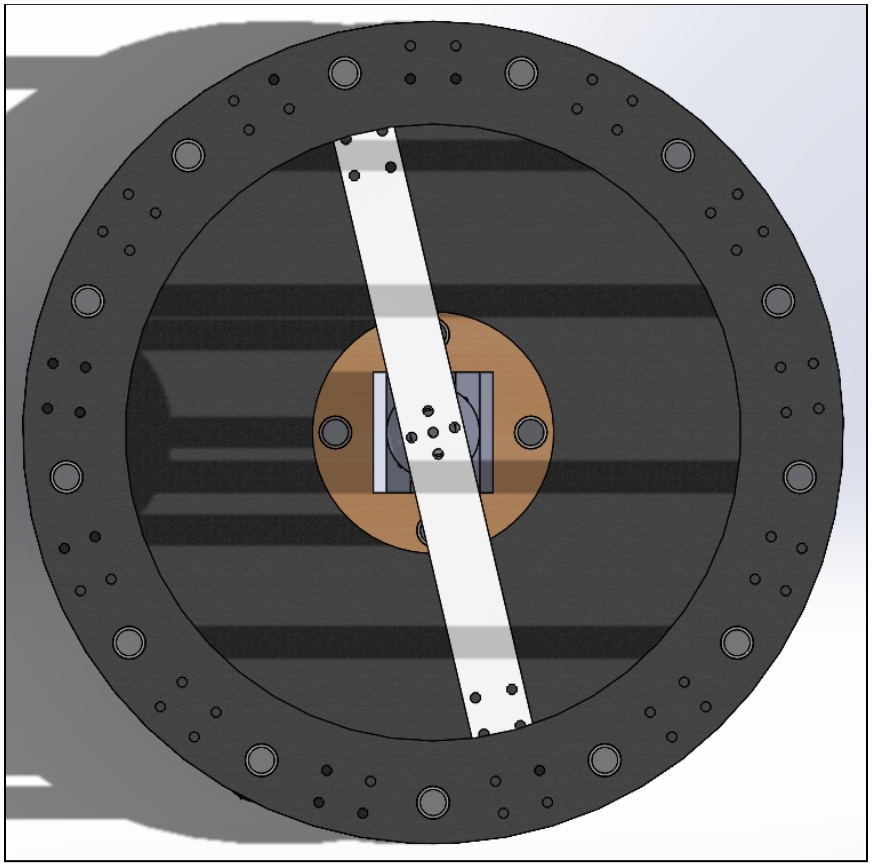


Image 8: Above view of the rotating platform showing chime placement. The solenoid platform would rotate to face a chime and then fire

Some potential issues may arise if we are given a note that we cannot play. As we receive notes, we must check if they are in our range before playing them and if they are not we must discard them.

Another potential issue is the stream of notes coming in, and achieving the correct timings. For this, we will need to know a few things:

1. The max delay of the longest movement
2. The delays between the movement of each note to every other note
3. The delay that the solenoid takes to play a note

From this, we will use the max delay as our negative delay from Ableton, then we will subtract the delay of the movement we are trying to make and start the movement that many seconds from the current point. This is the optimal configuration that allows us to hit the most notes in a given period without throwing off the delay.

This configuration, however, would require threading or knowing the time that each note is sent in order to be able to properly. This functionality is possible, but we were not able to figure it out in time and used instead the max move time for the ableton negative delay. We then move as fast as we can then fire the solenoid after that delay.

Discussion and Conclusion

One of the main things we learned from this project is the give and take that comes from building an automated instrument. Features of our machine required sacrifices and it was up to us to determine what we wanted in the final design that would help to preserve the overall goal of the project. In the beginning, we had very lofty goals, and with more time than our institution's seven week courses, we very much could have accomplished all of them. The base of our idea maintained the same throughout the process and it was a fun and interesting experience to watch as our ideas on paper became CAD models and even a finished design. We hoped to expand on the previous chimes design, Jingle Bells [5], and with our circular centerpiece design it feels like we succeeded. In an effort to focus on the spinning motion internally, the team ultimately had to make sacrifices to the playing speed and the ability to play multiple notes at once. Our original plan had us using multiple independently spinning platforms that could allow us to play chords. Although we modelled designs that may work, we ran into issues related to the wiring of these spinning platforms and not all of the team had the expertise to help in the engineering process. This ultimately led us to just using one spinning platform, but two solenoids, that each covered about 180 degrees of the machine. This design eliminated the wiring flaws and allowed us to play notes at far distances faster, but not at the same rate as our original hope. This "flaw" does make the machine imperfect when accepting Ableton input as it will be unable to play everything, but at the time it is only a limitation of the machine and it is still capable of playing quite

well. Additionally, the team focused on getting the base of the machine done along with the other mechanics which, given our time restraints, had us miss one of our goals for the number of chimes on the machine. We currently have eight chimes as opposed to the thirteen we had aimed for. This has the machine use a major scale as opposed to a chromatic scale. It would be very easy to in the future add these chimes and if made bigger, even more notes could be added. In the future, it would also be very interesting to see this design furthered with better mechanics related to the actuation of the chimes, like previously discussed. A model that allowed spinning of solenoids, but did not sacrifice speed for this aesthetic would be the ultimate goal. Additionally, such a design could also incorporate the ability to actuate the chimes in different ways allowing for a wider range of play. Apart from those changes, the team did a great job at achieving our original motivations for the project. Coming from different engineering backgrounds also gave us insight into the wide range of challenges and goals associated with both engineering and musical design.

Citations

[1]Automatic Tubular Bells. 2E0WDH, 2011

[2]Enhanced Tubular Bells (Automatic Tubular Bells) - Sample melody. Tolaemon, 2013

[3]tolaemon.com, 'Automatic Tubular Bells', 2008. [Online]. Available: http://www.tolaemon.com/atb/

[4]nime.pubpub.org, 'Tremolo-Chimes: Vibration-Motor Actuated Robotic "Wind" Chimes', 2021. [Online]. Available: https://nime.pubpub.org/pub/qx2wiy2p/release/1

[5]wp.wpi.edu, 'Jingle Bells', 2020. [Online]. Available: https://wp.wpi.edu/musicalmachines/2020/10/15/jingle-bells/

[6]nime.org, 'Swayway - MIDI Chimes', 2005. [Online]. Available: https://www.nime.org/proceedings/2005/nime2005_262.pdf

[7]nime.pubpub.org, 'Chaosmose', 2021. [Online]. Available: https://nime.pubpub.org/pub/ni5hbb4q/release/1

[8]L. Hite, "Easy DIY Chime Design and Build," DIY chime design and build. [Online]. Available: http://leehite.org/Chimes.htm. [Accessed: 15-Dec-2021].

Appendices

Ethics Statement

The team does not believe that there are any ethical concerns with our automatic musical chimes. In regards to funding, the project was self-funded by the members of the team. We also tried to use many easily accessible materials for the design so that it could be easily copied and improved in the future. The major components are copper pipes that can be easily obtained from any hardware store or, using a similar method to our own, change the material and cut pipes to the desired length. In terms of sustainability, during our process we realized that we wanted to change the diameter of the copper pipes used to help generate louder sounds from the machine. The pipes we had previously purchased were then reused as part of the structure of the machine so they would not go to waste.

Project Code

```
#include <Servo.h>

Servo servo;  // create servo object to control a servo
int solPin1 = 10;
int solPin2 = 11;

const int NOTE_NUM = 13;
const int MAX_MOVE_DELAY = 1000; // ms
const int AVERAGE_TIME_PER_DEGREE = 10; // ms; average time for the servo to
move 1 degree

enum Note {
  C, Csh, D, Dsh, E, F, Fsh, G, Gsh, A, Ash, B, Chi
};
// This is the order that the notes physically appear in starting from 0 degrees going
clockwise
int notes[NOTE_NUM] = {C, D, E, F, G, A, B, Chi};

// These are the angles that the pipes are positioned in
int angles[NOTE_NUM]  = {67, 260, 95, 287, 123, 320, 155, 348}; // TODO CHANGE
ME

//Defines the position domains for each servo
```

```
const int domain1Size = 4;
int solDomain1[domain1Size] = {0, 2, 4, 6}; // TODO CHANGE ME
const int domain2Size = 4;
int solDomain2[domain2Size] = {1, 3, 5, 7};
int noteToPos(Note n){
  for(int i=0; i<NOTE_NUM; i++){
    if (n == notes[i]){
      return i;
    }
  }
  Serial.println("Invalid Note Selected");
  return -1;
}

int posToAng(int pos, int sol){
  int angle = angles[pos];
  if (sol==solPin2){
    angle -= 180;
  }
  return angle;
}

int posToSol(int pos){
  int in1 = 0;
  int in2 = 0;

  for(int i=0; i<domain1Size; i++){
    if (pos == solDomain1[i]) in1=1;
  }
  for(int i=0; i<domain2Size; i++){
    if (pos == solDomain2[i]) in2=1;
  }
  if (in1==in2){
    if (in1==1) Serial.println("Domains overlap. Please redefine.");
    else Serial.println("Position not found in domain");
    return -1;
  }

  if (in1 == 1) return solPin1;
  if (in2 == 1) return solPin2;

  return -1;
}

/*
```

```
 * This function is used to calculate the delay we need to have before moving to be on time
 * This assumes that the built in ableton delay is -MAX_MOVE_DELAY
 * We need to calculate when we have to start such that we know when we should hit the correct note
 * diff is the difference in angle we are travelling
 * AVERAGE_TIME_PER_DEGREE is what we will use to calculate travel time
 */
void moveDelay(int diff){
  int time_needed = diff*AVERAGE_TIME_PER_DEGREE; // we dont need this for now because that would require threading to be useful,
                                                  // maybe i can figure it out tonight
  delay(MAX_MOVE_DELAY);
}

int playNote(Note n){
  int currAngle = servo.read();

  // get note position
  int pos = noteToPos(n);
  if (pos < 0){
    Serial.println("Failed to play note.");
  }
  // get note angle
  int sol = posToSol(pos);
  int angle = posToAng(pos, sol);

  if (sol < 0){
    return -1;
  }

  if (angle < 0){
    return -1;
  }

  // move to given angle
  servo.write(angle);
  int angleDiff = currAngle - angle;
  if (angleDiff < 0){
    angleDiff *= -1;
  }

  moveDelay(angleDiff);
  // delay(1000);
  // fire solenoid
  fireSol(sol);
```

```
  Serial.print("Solenoid: ");
  Serial.println(sol);
  Serial.print("Played a note at angle: ");
  Serial.println(angle);
  return 1;
}

void setup() {
  Serial.begin(115200);
  servo.attach(9);  // attaches the servo on pin 9 to the servo object
  pinMode(solPin1, OUTPUT);
  pinMode(solPin2, OUTPUT);
}

void fireSol(int solPin){
    digitalWrite(solPin, HIGH);
    delay(200);
    digitalWrite(solPin, LOW);

}

void loop() {
    if (Serial.available() > 1){
      byte note = Serial.read();
      byte velocity = Serial.read();
      if (velocity>0){
        Serial.println("velocity > 0");
        Note n = (Note) ((int) note - 72);
        int out = playNote(n);

        while(Serial.available()) {Serial.read();}
      }
    }
  }
```