# *Beethoven Bot*

Oliver Chang

University of Florida

Department of Electrical and Computer Engineering

EEL 4665-IMDL-Final Report

Instructors: A. Antonio Arroyo, Eric M. Schwartz

TAs: Josh Weaver, Andy Gray, Nick Cox, Daniel Frank

# *Contents*

## *Abstract*

Beethoven Bot is a mobile autonomous transcribing robot. As a casual pianist, one of the most frustrating aspects of playing piano is finding music sheets to recent songs. Most sheet music available for purchase consists of classical music or old classic songs. Finding recent or popular songs that are heard in random situations is a difficult task as the music may not be readily available or already transcribed online. In order to solve this issue, the Beethoven Bot was designed.

This autonomous robot is designed to listen to audio tones through a microphone, detect which notes have been played, and transcribe the notes onto staff paper. The audio will be received through a wireless microphone. The note is then determined through frequency analysis of the recorded music. The robot then finds the location of the note in the treble clef section. The note is then written at the correct location and the robot proceeds to the next note.

## *Executive Summary*

Beethoven Bot is an autonomous robot created to transcribe audio tones to musical notation. While there is a world of paths this robot can go onto, the objective of the Beethoven Bot is to listen to a single tone at a time before each note marking. Wireless communication is done between the microphone and the analyzing mind (Raspberry Pi) of the robot, while serial communication is done between the analyzing mind of the robot and its locomotive mind (Arduino).

Locomotion is completed through rotational movement provided by two stepper motors. One omni-directional caster is added to maintain a level base for the robot. Three ultrasonic sensors provide feedback for obstacle avoidance, while two bump sensors provide feedback for obstacle detection; all of which are placed on the front of the Beethoven Bot. The internal system of Beethoven Bot includes a two cell lipo battery, a power distribution board, a motor driver board, a solenoid with its accompanying circuitry, a servo, an LED array, the Arduino microcontroller and the Raspberry Pi.

At start up, the Beethoven Bot boots up, this takes around a minute for the Raspberry Pi. The audio program runs at boot up by an ambient noise calibration and notification through an LED. After this is completed, the Beethoven Bot continues on to listen until it hears a sound above the ambient noise threshold, after which it begins to record the audio into a wav file. This wav file is then trimmed and silence is added to the beginning and end for consistency. This signals the beginning of the audio analysis routine, in which the robot reads the wav file and performs Fast Fourier Transform on the signal to discern what frequency is read in. Each musical note has its one unique frequency, thus the note played can be determined by the calculated frequency. This is then sent to the Arduino via USB serial communication.

The Arduino waits until a note received and depending on the note, Beethoven Bot moves up or down the staff to the appropriate note location. A marking mechanism is activated and a note is marked either through a servo or a solenoid. Once the note is marked, the

Beethoven Bot returns to its center location on the staff and awaits for the next note to be received.

## *Introduction*

As a child I grew up taking classes to learn how to play piano. I greatly enjoyed playing music however all of the music that I played and had was classical pieces. I desired to played music featured in current movies, played in radios, or heard in video games. The issue was that the sheet music at musical shops was mostly classical songs and finding sheet music for these recent pieces were far and few between. In order to overcome this problem, the Beethoven Bot was created. The original Beethoven Bot was an open-loop, differential drive platform with a drawing mechanism at the center powered by a 7.4 V battery and controlled through a microcontroller and small computer.

This report will explain the design of both the mechanical and digital aspects of the Beethoven Bot. There are two main brains to this robot. The Raspberry Pi(Rpi) will listen to the audio tones and analyze the sound input to calculate the notes played through fast fourier transform. The Arduino ATMEGA 2560 receives the notes and moves the robot accordingly and activates the drawing mechanism to transcribe the note.

## *Integrated System*

The Beethoven Bot uses a Raspberry Pi for high level decisions and an Arduino ATMEGA 2560 for low level decision making. The bump switches and ultrasonic sensors provide inputs to be incorporated into the controller's computations. The melody will be taken in from the microphone analyzed with the Rpi to be sent to the Arduino for transcription.

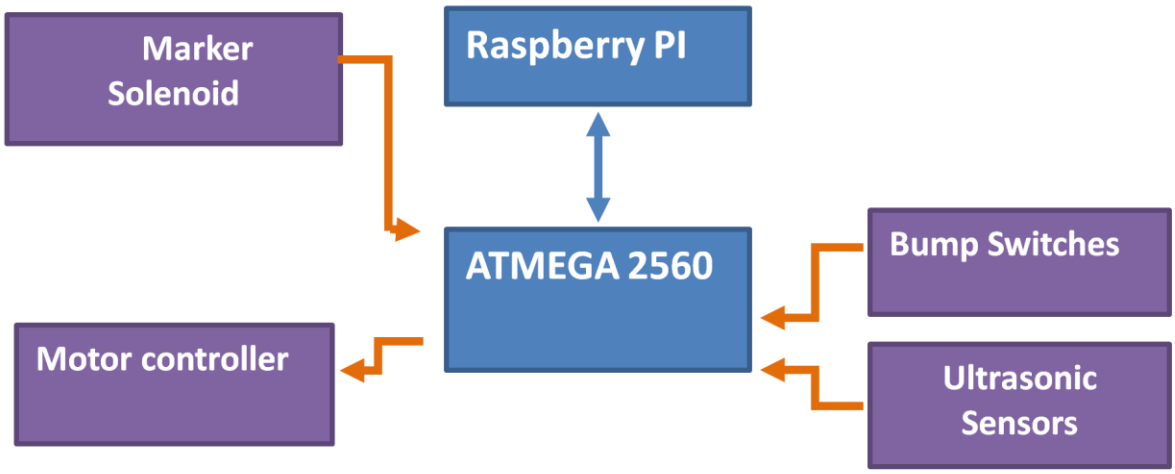



**Figure 1: Beethoven Bot Block Diagram**

## *Mobile Platform*

A two wheel differential drive system will be utilized along with a caster for level and stability. The robot is designed to be relatively small is relation to the potential area in which it is able to draw. The decision to make the robot mobile was to have more flexibility to the music sheet size ratio, this way the robot would be able to draw on all sizes of staff paper. Beethoven Bot has two possible methods of marking a note as a failsafe. A solenoid placed at the center of rotation lifts and lowers a marker to mark and indicate the location of a musical note. The second method is a servo attached to the back end of the robot, which raises and lowers a marker in an arc to mark the note. This marker may be outfitted with a stamp head, which (when inked) will stamp a musical note down. The motors are mounted to metal mounts fastened underneath the robot. In order to maintain a proper drawing formation, the caster must lift the platform to be level with the height at the wheels. The platform is made with airplane grade wood machined in lab. The components are bolted to the base platform with the exception of the Raspberry Pi, battery, and LED array, which are located on the roof of the robot. The top of the platform is held above the base through four threaded bolts, which allows easy removal for access to the central system.

## *Actuation*

Two Pololu NEMA 17 stepper motors actuate the robot. These motors are differentially driven to allow turning in-place. The actuation system is an open-loop system, due to the accurate nature of stepper motors. The decision in utilizing stepper motors for actuation was made due to the necessity of accurate position of the note in relation to the music staff. Two Pololu low voltage motor controllers provide the H-bridges to drive the motors forward or backwards by some given steps. The Arduino transmits to the motor controllers using pulse width modulation to communicate how many steps to take, while a second line tells the drivers what direction to move. A large problem that occurred and used a week of work time was the quick draining of the lipo battery on pre-demo day. This was suspected to be caused by a potential short in the power distribution board, thus a new one was populated. This problem, however, extended to cause inconsistent motion of the stepper motors as in one motor for a single direction; the motor would turn twice as fast and far as the other. This was thoroughly tested through the purchase of replacement motors and motor drivers, the systematic testing of each part individually, and the eventual realization that a common ground was causing the issue.

For a writing mechanism, a marker is pressed down through a solenoid, which provides vertical linear actuation. The solenoid forces the marker into the sheet, which marks the location of the note. The robot can then be rotated 360 degrees to draw a circle; however this circle is fixed in size due to the nature of the mechanism. The secondary mechanism provides a similar stamping motion via a servo connected to a marker arm. This servo lowers and presses the marker onto the paper, which may also be outfitted with a stamp head to ink a musical note image onto the paper instead. Weights are added on top of the mark to provide additional downward force as the stamp head cannot simply be laid on top the paper and provide a clear image. There was much struggle involved in the marking aspect, which was not originally expected to be difficult. Through much of the project, it was assumed that a simple solenoid would provide enough force to push a marker onto the music sheet, however the solenoid purchased was not nearly strong enough in addition to being far too short in arm length ( a few millimeters). Multiple iterations of extensions, different heights, and different markers were created and tested before the second marking mechanism was created along with the purchase of beefier solenoids.

## *Sensors*

Collision detection is implemented through the use of two bump sensors. These two bump sensor are placed at the front of the robot extending out left and right over the wheels since the robot's main movements will be forward and rotating right or left to reach the next line in the drawing grid. 5 volts power the switches and connect to an interrupt in the Arduino through the switches with a pull down resistor. The switch arms are extended into paddles to increase their range of detection. The three ultrasonic sensors actively measure the distance of the closest object for up to 3 feet, while the robot is in motion. When an object breaks the threshold distance of the robot's path, the robot will stop and wait until the object is removed.

## *Behaviors*

The Beethoven Bot has two modes of operation. The first being a listening mode, in which the robot will listen for a specific note and then begin reading the melody that it hears. The music heard is then calculated on the Raspberry Pi through FFT and the notes are sent serially to the Arduino to send the robot to the second mode. The second mode is the drawing mode in which the robot receives the note to be played, moves to the according location, marks the note and returns to the center location for the first mode to repeat.

The first mode of operation involved a large amount of the frustration in this project. The main code was completed without much trouble. This code consists of listening for an initial period of time to set the noise threshold. Once completed the Raspberry Pi listens until this threshold is broken and then waits until silence is heard afterwards. This makes sure the whole tone is recorded. The music file is then trimmed to cut everything before the first instance of tone and after the last instance of tone. Silence is added at the beginning and end of the file to pad it with a control. The completed file is then saved as a wav file, which is helpful for debugging purposes. The second half of this code reads the wav file and performs Fast Fourier Transform on the data. Through some simple calculations, the frequency is determined; the wav file is also played back during this time for debugging purposes. The noise values below and above the desired note range are discarded and the remaining frequencies are averaged to determine the frequency of the overall tone. This frequency tells us what note is played and that information is communicated serially to the Arduino. Serial communication is also sent during the main sections of this program for the Arduino to activating the proper LEDs to notify the user where the robot is in its code.

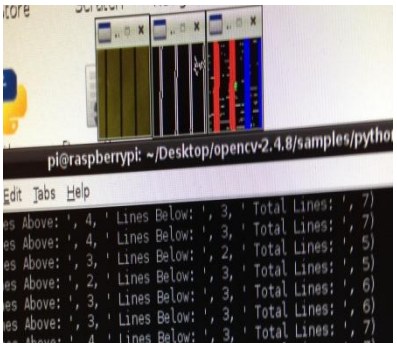
The main problem that occurred was the interfacing of an audio microphone for which the tone would be recorded through. The issue provided with the Raspberry Pi is that there is no audio card or input that may be accessed. This left the USB ports as the only method of access a microphone. Initially a Bluetooth microphone was used with success for the first half of the project. For unknown reasons, the Bluetooth microphone began failing in the latter half of the semester. The Raspberry Pi would not be able to maintain a constant connection with the Bluetooth microphone and when it did, there was no way to set up the microphone as an audio input device. A USB microphone was then installed and tested, but introduced a different set of problems. The audio quality of the microphone was much worse than the Bluetooth, causing the audio data to give too noisy samples to have a clean frequency be determined. Finally after months of work, a wireless microphone was purchased (which was known to be able to be interfaced with the Raspberry Pi with good samples). This microphone is the final iteration and works as well as expected.

The second mode of operation consists of the Arduino awaiting for the note data to be serially received. The Arduino then moves up or down the music staff from its center position of note A. After which the Arduino marks the note via a solenoid or servo, returns to its center position and moves forward to await the next transmission. The troubles regarding the marking mechanism is addressed in the actuation section above. The Arduino controls the functioning aspect of the robot. All data from the ultrasonic sensors and bump sensors are received and processed by the Arduino. Similarly all the functions of the solenoid, the servo, and the stepper motors are all transmitted from the Arduino. When Beethoven Bot is in the process of marking

its note, aka moving, the ultrasonic sensors are sequentially firing to detect objects. When the Beethoven Bot is in its first mode, which requires no motion, the sensors are put to sleep along with the stepper motors. This allows for power to be saved and to prevent over exhausting the stepper motors due to stalling.

## *Experimental Layout and Results*

Points of experimentation include the ability of the robot to drive straight, calibration of rotational motion of the robot, how accurate the note placements are, and the process of finding a proper microphone that could be used. Initially the Raspberry Pi camera was experimented with to actively compute the location of the robot in relation to the staff lines and where to go, however time and code limitations prevented this addition and the robot remains an open loop system. This experiment took nearly one month of work before abandonment. Solely the process of installing the visual processing software, OpenCV took between 1 to 1.5 weeks of research and debugging due to the newness and lack of support for the Raspberry Pi. Once installed the Hough lines function was used to determine what edges in an image was considered a line, however the sensitivity of the function causes multiple false positives to appear or not enough positives to appear to consist of seeing the entire 5 lines of the staff. An additional problem to be solved was the angle at which the lines were in relation to the robot, if this was not perfectly parallel, the lines would not be detected. It was due to this and the time constraint that the camera was removed until the project could be restarted




The initial thought of using a microphone was to utilize an electrets microphone. These microphones are cheap and simple to use. Unfortunately this meant that the microphones are terrible in quality. The microphone had to be amplified and would be placed on the robot. When experimenting with an FFT library on the Arduino, it was discovered that the microphones did not have a range larger than a few inches. Some students with similar audio robots opted to using a satellite dish method of funneling the sound to the microphone, however detaching the microphone from the robot as a whole would prove more beneficial in this circumstance. By switching to a wireless microphone, the microphone could be placed anywhere the tone source would be and the robot would still receive great audio quality. Unfortunately this added problems in itself as portrayed in the behavior section.

## *Conclusion*

The realistic goal of this project is to create a robot that will be able to transcribe a single simple tone onto a provided music sheet. The robot in its current finality is able to process and listen for a single tone per iteration and mark one of four notes on the staff paper. This can be resolved given more time and coding. The Raspberry Pi is perfectly able to discern any note on

the treble clef, while modifying the step range of the stepper motors will allow accurate movement for half note steps rather than whole note steps. The limitations that follow this work are the complexity of the melody to be transcribed, how well the music is heard and transcribed through the Raspberry Pi, and how well the robot performs in locating the note position. The robot is able to perform object avoidance through the use of the ultrasonic sensors; however there is little need for the robot to avoid objects except at the point in which the robot moves to the next note location. This is why the robot simply stops if an object is detected. The motor controller and power circuits were designed as PCBs machined on campus to create a cleaner, more professional design, however problems rose from those PCB in design and population, thus perforated boards were created in their place.

This course allowed me to experience the complexities of mechanical design and creation. It was my first experience in machining parts from a model or sketch, creating mechanism for linear motion, installing and using OpenCV and my first use of the Raspberry Pi and Arduino.

The sound analysis as the special system waits until a sound above the noise threshold is heard before beginning to record the music. After which the music is saved as a wav file and analyzed in python to determine the frequencies present through FFT. The noise values are disregarded and the frequency is averaged and the note is determined from that averaged frequency. While the robot is able to perform the most basics of its tasks, there is much room for improvement both in smoother mechanical design and more versatile audio software.

## *Documentation*

- ATMEGA 2560 Datasheet: http://arduino.cc/en/Main/arduinoBoardMega2560
- Raspberry Pi: http://www.raspberrypi.org/
- Pololu Motorcontroller: http://www.pololu.com/product/2133
- All Parts and relevant links can be found here: http://ochang.weebly.com/

## *Appendices*

- Code can be found in project website: http://ochang.weebly.com/