Gait Controlled Treadmill

By

Jiangtian Feng,

Charles Lewis IV, and Jacob Pruiett

Final Report for ECE 445, Senior Design, Spring 2019

TA: Anthony Caton

05 May 2019

Project No. 43

Abstract

The standard equipment for cardiovascular conditioning and physical therapy is the treadmill. The ease of use and on board velocity feedback are some of the main draws as to why people all over the world have prioritized usage of the device for years. Although, despite the positive considerations, many still prefer to run outside. In this document an alternative design for the standard treadmill is presented and may pave the way for the development of a buttonless positional/velocity controlled device. Here we explore the fundamental modules needed for implementation and integration of a system that bases speed variations of the motor off of the movement of the person or object on the treadmill by way of ultrasonic or lidar sensor feedback while minimizing velocity steady state disparity between belt and object. In addition, supplemental methodologies for past and future design considerations will be explored.

Contents

1. Introduction

There is a distinction made between the process of running indoors on a treadmill and outside without. Aside from the haptic contrast between the two processes, the most salient of considerations is the method of control between running on and off machine. The most obvious of this consideration is the method used for varying speed. When running naturally, a person need only speed up or slow down by changing their gait speed, but when using a treadmill the method of varying speed is by pressing buttons. In the following project, a method to eliminate is sought.

If the need for button control on a treadmill is eliminated, the user better able to focus on the originally intended process without the need to concern themself with the manipulation of speed control. The mitigation of the automation process is akin to a planes ability to engage an autopilot mode; allowing the pilot to refocus attention and micromanage marginal processes. The integration of this automation process' foundation lies in one of two ways of implementation: velocity control or positional control. Both of these methods require the employment of a error analysis feedback based algorithm that simultaneously determines velocity or positional information from a stream of data via sensor and consolidates the information by throwing out outliers and averaging the accurate data within a predetermined error range. Said data is then be fed into a module designed to output a pulse width modulated signal (PWM) with the appropriate duty cycle relevant to the incoming data feed and drive a speed control circuit designed to vary motor voltage and thus motor speed.

As mentioned previously, the main selling point of the product is to implement a unique series of subsystems that are capable of tracking and analyzing user velocity changes and outputting a control signal to a motor via a speed control circuit. This initial design had to be modified due to physical changes to the treadmill as a result of improper specifications for the treadmill belt. The verification procedure for velocity control requires an accurate and quantitatively sound testing method. The foundation of such a test requires a procedure where changes in velocity are measurable; thus, a belt is needed so that an object (i.e. RC car) could be placed and varied at differing readable speeds in order to make comparisons. The resulting design application was to be a positional based tracking system insomuch that the belt was not required in order to obtain quantitatively sound feedback for verification. The original block diagram for the velocity based system is shown in figure 1.

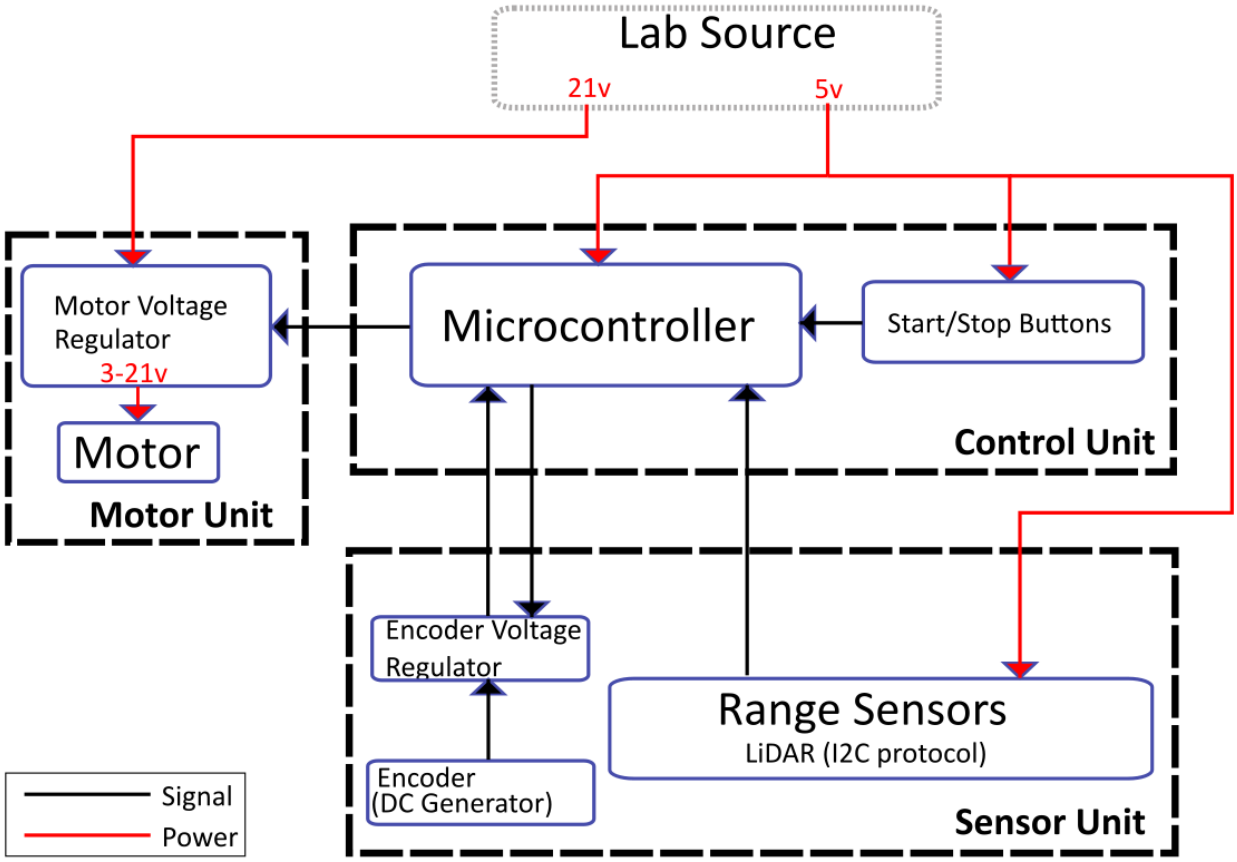

Figure 1: Block diagram depicting macro level perspective of all subsystems.

From the diagram high level perspective, the overall system is comprised of three main modules. The motor unit encompasses both an electric DC based brushed motor in addition to a motor voltage regulator (or speed control circuit), which dictates the appropriate voltage level across the motor, and thereby regulates the speed of the motor. The motor unit block is fed both a 21 V DC signal from an external source as well as a varying square wave pulse width modulated (PWM) signal from a microcontroller to allow for acute speed control.

The sensor unit is comprised of a series of range sensors, a motor encoder (encased within an additional unpowered motor attached to the axle of the working motor), and an encoder voltage regulator as a means of translating feedback from the motor to the microcontroller while remaining within the safe operating voltage range of the microcontroller I/O pins, and lastly a 5 V input power signal to the sensors from an external source. One of the changes necessitated by moving from a velocity controlled system to a positional one was that both the encoder and the encoder voltage regulator became unnecessary. The motor feedback was originally needed to make comparisons between the current treadmill belt velocity and with velocity data that is determined by the microcontroller using the positional data procured by the range sensors. Since the algorithm within the microcontroller was modified to analyze and translate purely positional feed from the sensors, the motor speed was not needed, thus the encoder became irrelevant in the final implementation of the system. Additionally, the range sensors that were ultimately used consisted of one LiDAR as opposed to two, as two did not add

any additional accuracy even with the additive effects of the positional error control algorithm implemented in the control unit. Also, the LiDAR sensors were chosen over the Ultrasonic sensors, as the Ultrasonic sensors lacked the accuracy that the LiDAR exhibited by default when performing sensor performance verification tests, even with the additive effects of the control unit error analysis algorithm.

The final module is the control unit. The original concept for the control unit consisted of just the microcontroller in addition to two buttons. Both the positional as well as the velocity controlled system required a method of starting and stopping. The final implementation does not include the buttons as the focus was placed on integration of the vital functions with reference to the speed control and motor unit, as that is far more important to the function of the design. The control unit algorithm within the microcontroller, as suggested previously underwent modification as the focus of the software processing evaluated sensor information on a purely positional basis as opposed to the rate of positional changes over time. Although this did not incur modifications of the control unit block on a schematic viewpoint, the changes elicited a number of critical changes that affected the manifestation of the electronics on a functional level. In particular the Atmega328pb was swapped for an Arduino Uno. The move from the Atmega328PB Xplained Mini to the Arduino board stemmed from a number of technical changes that occurred due to a disparity between hardware when attempting to integrate. These discrepancies will be discussed later in section 2.3.1.2.

In this document the various subcomponents that necessitate base functionality of an autonomous positional/movement controlled treadmill are analyzed. The various design considerations are mapped out from both a macro and micro perspective. In addition the design verification procedures used to define the standards set for success or failure of components, modules, and the various subsystems are also discussed. After this discussion, an analysis of the costs of labor and parts in order to layout the foundation for future marketing considerations is performed. Lastly, the decisions regarding the final product as a function of the successful and unsuccessful implementations as well as the detailed rationale behind those closing decisions are discussed. A particular focus is placed on the reasons behind the move from velocity to positional control in addition to what led to failure of integration within the control unit.

2 Design

2.1 Motor unit

2.1.1 Motor

The Decision behind choosing the brushed DC motor as opposed to a brushless DC or induction motor stems from the relative simplicity of control offered by brushed DC motors. The brushed DC motor does not require the supplementary components needed to drive more robust motor types (such as three phase power input, closed loop motor control with the addition of hall sensors or power electronics for finite control over winding excitation). Under rated speeds brushed DC motors require no closed loop control or the presence of a controller for error control as long as there are no varying torque loads to be concerned with. For a prototype such as the one designed here, closed loop control is unnecessary, as test sequences do not incur significant deviations from the torque of the belt itself. However, in real life circumstances closed loop control with error correction is required in order to maintain speeds due to additional loads on the belt. Additionally, brushed DC motors are known to wear more frequently than other motor types as their construction is composed of parts that contact more often and therefore require periodic maintenance; thus, the additive complexity of using more sophisticated electric machinery in order to account for the extra maintenance may be warranted for future iterations.

Initially the motor used was the Minertia small size DC r-series (R02MA). The peak rated torque met the specifications for just the load on the belt, but did not include the load of the belt itself. Thus it proved too weak for the torque load of the treadmill, and was replaced by the Minertia Motor P-series (P12H), which has a higher peak rated torque. After further testing it was realised that the motor could only barely move the belt at its max rated torque, and that the current it ran at was over the range of safe operation for the motor without cooling. To fix this requires acquiring a new treadmill belt and a gear reduction on the motor output. There was not enough time to do either, so instead the motor was ran without the belt in the final design.

2.1.2 Motor voltage regulator

For the motor voltage regulator, it was initially designed to be a buck converter, but was switched to a quarter H bridge because the addition of power electronics to manipulate voltage reduces efficiency as well as incorporated unneeded, possibly volatile, electronics that add unnecessary complexity to the overall system. H-bridges are a universally standard circuit and is sufficient to run many types of motors, including three phase; therefore for future iterations the H-bridge may be an acceptable method for running an alternative motor type with three phase power.

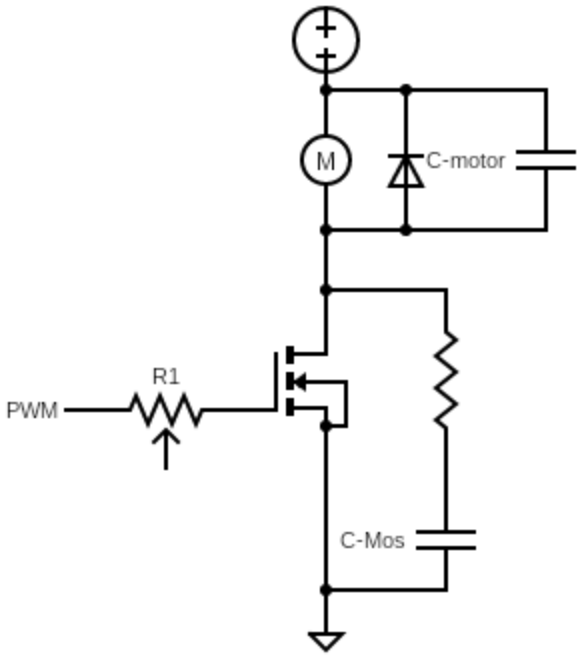

Figure 2: Voltage regulator/quarter H-bridge used to drive motor at varying speeds and voltages.

A quarter of the H-bridge is, used as it is sufficient for linear operation in one direction, and there is no concern with braking or being capable of changing direction of motor angular velocity. The circuit consists of a pull down resistor of 11 Ω at the gate of the MOSFET in order to burn off additional energy from excess current being drawn during turn on sequences; this is particularly important with the inclusion of logic circuits (such as a microcontroller). Additionally, the motor is connected in parallel to a passive 1N4003 recovery diode and a small $1\mu F$ capacitor to allow for safe current passage and to mitigate voltage spikes during operation, considering the motor is modeled as an inductor in series with a resistor, and that voltage spikes will occur in order to maintain current into the motor during switching sequences.

Additionally a bleed circuit is included in parallel with the MOSFET to further account for voltage spiking due to its inherent parasitic induction and capacitive effects. The bleed circuit consists of a series resistor and a capacitor of values 10 Ω and 10 μF respectively. The values for the components were designated by testing based off of the expressions 2.1 through 2.3.

$$R_{mosfet} > R_{ds-on} \tag{2.1}$$

Where the MOSFET resistor that is chosen must be larger than the turn on resistance (determined to be at a max of 7.1 mΩ[9])

$$R_{mosfet} = \frac{V_{s-min}}{I_{pk}} \tag{2.2}$$

Additionally the bleed circuit resistor consists roughly of the minimum voltage that is expected applied to the MOSFET drain over the peak current expected to be drawn

$$C_s = \frac{1}{2\pi R_s f_{ring}} \tag{2.3}$$

Lastly, the series capacitor value that is chosen is proportional to the inverse of the product of the resistive value that we choose into the frequency of the ringing that we wish to diminish

These expressions were used as a base stepping point to test for optimal values in order to limit transients and ringing at the mosfet. An example of the successful attenuation of the ringing is included in figure 3.

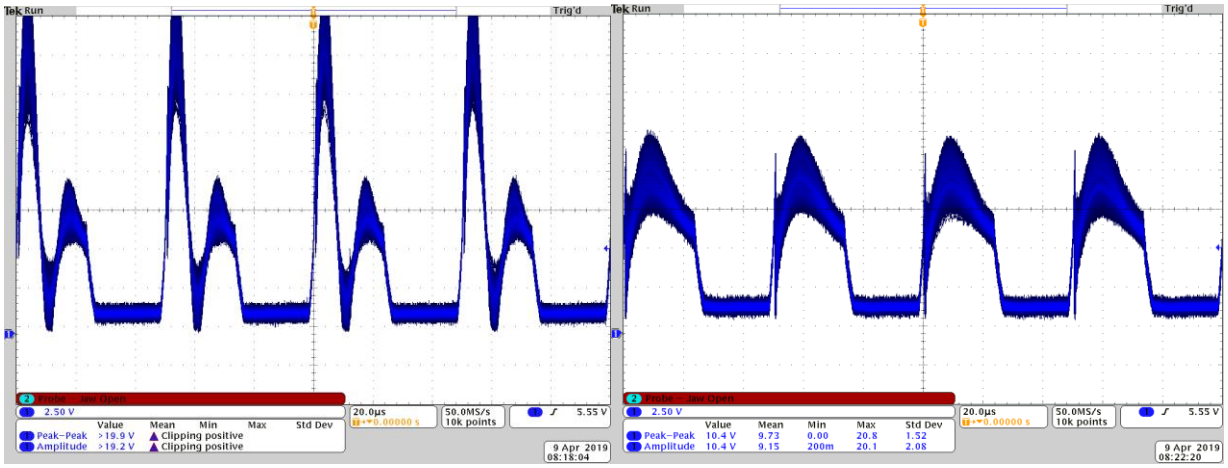

Figure 3: Waveforms depicting the before (right) and after (left) MOSFET drain voltage waveforms during operation at a 50% duty cycle, 10 V input. Before circuit implementation, ringing exceeded 9 V.

2.2 Sensor unit

2.2.1 Range sensors

The purpose of this unit is to determine and then transmit data to the microcontroller the current velocity of the belt, and the current position of the user. To determine the current position, both LiDAR and ultrasonic sensors were tested in multiple configurations. The LiDAR sensor is the TinyLiDAR ToF Range Finder Sensor, and the ultrasonic sensor is the Ultrasonic Ranging Module HC-SR04. These were chosen because whilst the LiDAR sensor is expensive but accurate, and the ultrasonic sensor is inexpensive and less accurate, both met the minimum distance requirements according to their data sheets, and as such are good candidates to test the separate configurations with. For the verification procedure, please refer to appendix A.

After verification, it was found that the ultrasonic sensor did not meet the minimum accuracy requirements, so the LiDAR sensor is used. After implementation of the raw data processing algorithm and control algorithm, it was found that the sensor was more accurate by itself rather than in a multiple sensor configuration, so only a single LiDAR sensor is used in the final implementation.

2.2.2 Encoder and Encoder Voltage Regulator

For determining the velocity of the belt a DC tachometer generator, which is a type of encoder that transmits velocity data using voltage, is used. This type of encoder was chosen because it changed voltage value in real time with the speed of the motor shaft. In contrast, the other type of encoder to be used would be an optical encoder, which sends a signal for each rotation or fraction of a rotation of the shaft, so data must be collected over some time step before speed can be determined. This causes an inherent delay before the signal is output, so we use the DC tachometer generator.

The encoder was initially determined to be the encoder native to the R02M, and so a regulator to step down the voltage outputs to a level that the microcontroller could handle was devised. However, upon testing the motor, it was found that the voltage output of the encoder at different rotational velocities was different from the expected values, and that the microcontroller handles these values natively, so the voltage regulator is unnecessary. For the verification procedure, please refer to appendix A. To account for the discrepancy in data values, the specifications of the new encoder were determined by hand for use with our velocity control algorithm. However, the current control algorithm is position based, so in this implementation the encoder is solely used for testing purposes.

2.3 Control unit

2.3.1 Microcontroller

2.3.1.1 Hardware

The specific microcontroller chosen for this project was the ATmega328PB. The ATmega328PB was chosen over other alternatives because the initial algorithm as envisioned would require a large enough amount of separate timers (2 8-bit timers & 3 16-bit timers) and I/O pins to implement data collection with multiple sensors and output a PWM signal. In addition, there were some concerns with memory. In the initial outlines for the algorithm a large amount of data would need to be stored for velocity, so a microcontroller with a large memory was chosen to counterbalance this concern.

In the implementation, the ATmega328PB was initially used in conjunction with the ATmega328PB Xplained Mini, which from now on will be referred to as the devboard, for program testing. The devboard is able to output an appropriate PWM signal with the desired frequency by manipulating the internal timer. But when testing the ultrasonic sensors, the voltage output from the echo pin of the HC-SR04 is below the value required to trigger a high signal on the board, even with the use of a pull-up resistor. When compared with the value from an arduino uno board, the echo pin outputs a value between 2-3.5 V, whilst the devboard outputs a value near 7 mV when activated. On the other hand, due to the fact that LiDAR sensors transmit data by I2C protocol, which is too complicated and time-consuming to implement. However, Arduino supports complete I2C library In the interest of time, and because the implementation of the algorithm was deemed more important than the device used to implement it, the final implementation was done using the Arduino UNO.

2.3.1.2 Software

In the initial design of the software, a velocity based algorithm was chosen because it provided greater accuracy in matching user speed, as it directly measures and then matches that speed, rather than approximating the relative speed based on distance from a center point. As such, the velocity based algorithm was theoretically more comfortable for the user than position based control. However, a velocity based control algorithm is more difficult to implement and test than a position based algorithm. As such, in the interest of time and because the best method of verifying the velocity based control was no longer available due to mechanical failure, position based control was implemented instead, with the intention of using velocity based control after integration with position based control was completed. Since integration was not completed, the process of implementing velocity control was never started.

For the data processing algorithm, it is as described in the flow charts in Appendix A.

The velocity based control algorithm that was envisioned during the design process is as follows: The positional data from the data processing algorithm is stored for a number of points. Initially a storage size of ten data points is used, but that would be subject to change with testing, as accuracy increases with more data points, but also increases processing time for each point used. After the ten points are filled, the difference across these data points is taken, and then divided by the time step that each data point is taken at, then averaged. The current belt speed of the motor is determined using equation 3.1 (found in section 3.4), and the current voltage across the encoder. The current belt speed and the averaged user speed are added together, translated into rpm, and then used as inputs to equation B.1 (found in appendix B), which outputs the duty cycle that the motor is then changed to, to match user speed. Duty cycle is then set to that value, and then slightly higher or lower if in the forward region or the rear region, each of which are defined by their position relative to the central neutral zone. The reason the value is set slightly higher or lower is to cause the user to move back into the neutral zone. Once the neutral zone is reached, Duty cycle is set to the original value, and then remains unchanged until the user leaves the neutral zone again, in order to account for natural variation a user will have in running gait.

The position based control algorithm is as follows: The current position output from the data processing algorithm is taken, then compared with the threshold values for the neutral zone. If the value is within the neutral zone, the duty cycle is kept the same. If the value is below the threshold, duty cycle is increased since that means the user is close to the sensors. Finally, if the value is above the threshold, duty cycle is decreased since that means the user is away from the sensors.

2.3.2 Stop/Start Buttons

For the stop and start buttons, there was the choice between using push buttons or throw switches. A push button maintains a signal so long as pressure is applied to it, whilst a throw switch constantly maintains the signal until it is switched back. In the controls algorithm the signals sent through the buttons are used once, and the system would not change upon repeated presses of the same button. But from a usability perspective, the pushbutton makes more sense, since the throw switch may cause confusion in the user, because the typical use of a throw switch as a start button turns off the system after being switched back from the on position. For these reasons, the push buttons are chosen for the stop and start buttons.

In terms of implementation, these buttons were never integrated into the system. Since there was no longer a belt on the treadmill, safety became less of an issue, and the focus on implementing all other portions of the design was prioritized over the implementation of the buttons.

3. Design Verification

3.1 Motor

The requirement set in place for optimal motor functionality given our particular standards was designated as having the capability of operating rotor angular velocity at specific values. These values translate to linear speeds between three and ten miles per hour. The motor also must be able to settle to a steady state speed within one second regardless of voltage step size. This gave a reasonable foundation for smooth transitions when varying speed of the motor, as any deviations beyond one second settling times would need to be factored in had we implemented a velocity based control schema, due to varying load on the shaft.

The maximum speed for the motor was designed to be 10 mph. The average finish time of an American marathon goer in 2016 was 4.72 hrs[11], which translates to an average speed of 5.56 mph, so a value roughly double that was chosen to grant a better range of speeds that represent what an athlete may realistically run at. Upon testing, the motor was found to reach this speed at 21 V. In addition, the treadmill's minimum speed was put at 3 mph, and upon testing the lowest duty cycle given the speed control circuit components turned out to be a duty cycle of 12%.

Figure 4 depicts the results of the verification process that was done in order to confirm that the requirements for the motor were met. The testing procedure is as described in appendix A. The results are best shown on the red trend line, which is below one second for every step size. In addition, the calculated net average step size is .798s, which shows that the requirements for our motor are met.

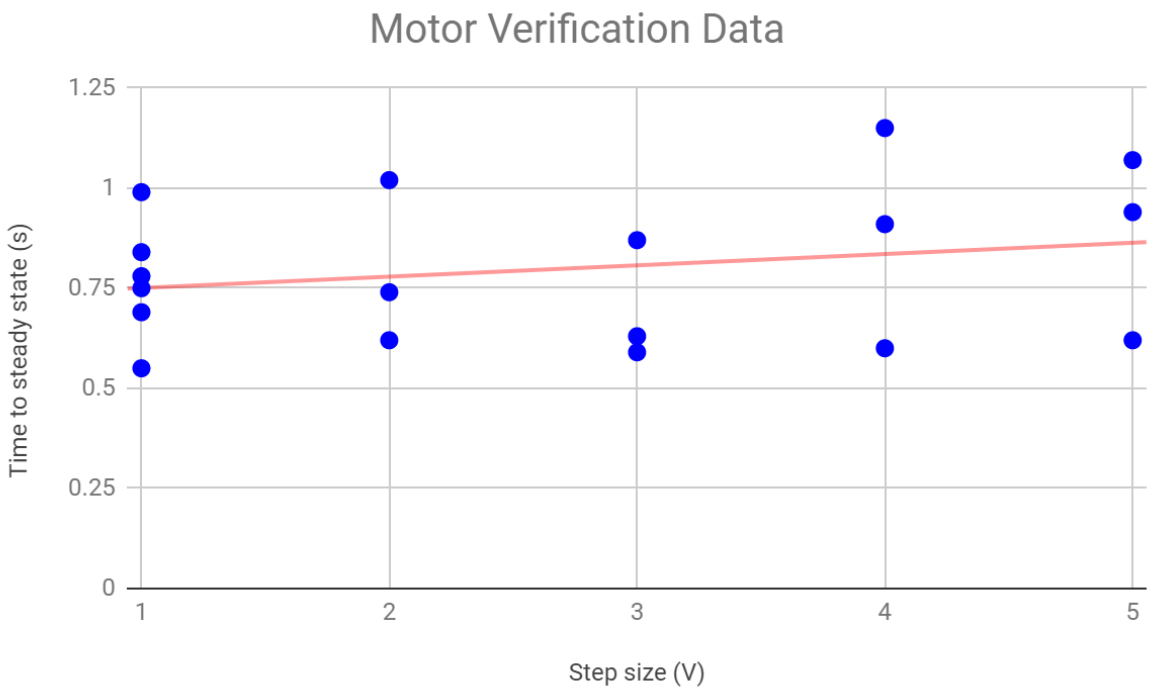

Figure 4: Graph depicting the verification procedure of the motor functionality under the guise of testing speed changes under a given voltage step size.

3.2 Motor voltage regulator

The requirement of the regulator is defined in a way such that the safe operating parameters of the electric DC servomotor that is used is never in danger of being exceeded. Since the machine lacked the presence of a belt which would have added a load on the rotor of the motor, the current draw was less than 1 A which was below the 22 A limit. Additionally, the 25 V maximum could be input to the motor and speed control circuit could be ran safely while varying the duty cycle of the PWM signal to the gate of the MOSFET without concern for voltage spikes exceeding the limitations set in place for the motor, given the speed limitation of 10 mph (see section 2.1.2 for speed control voltage transient details). The testing procedure is as described in appendix A.

3.3 Range sensors

For the range sensors, the setup is placed as detailed in appendix A, and the data is taken for both the LiDAR and the ultrasonic sensors. The original margin for error was 5%, and was determined to be such based on estimation. The reason to test for this margin of error is because, in the velocity based control algorithm, any error will be fed into the motor, which will in turn cause the error to grow larger as the velocity is changed to the wrong value, and will grow larger and larger over time. In order to counteract this, error must be minimized, but in order to determine the maximum margin of error that can be made before the effects of error quickly propagate through the system, testing with the velocity based control needs to be done. Since the velocity control was never completed, the exact margin of error that is required was never determined, so the value remained at 5%.

The data in figures 5 and 6 show that the LiDAR sensor is significantly more accurate than the ultrasonic sensor. In addition, the ultrasonic fails the verifications at the extremes of the measurement range, indicating that the effect of feedback and the bounce from the 15° cone it sends its signal at causes it to read increasingly incorrect data values up close and afar.

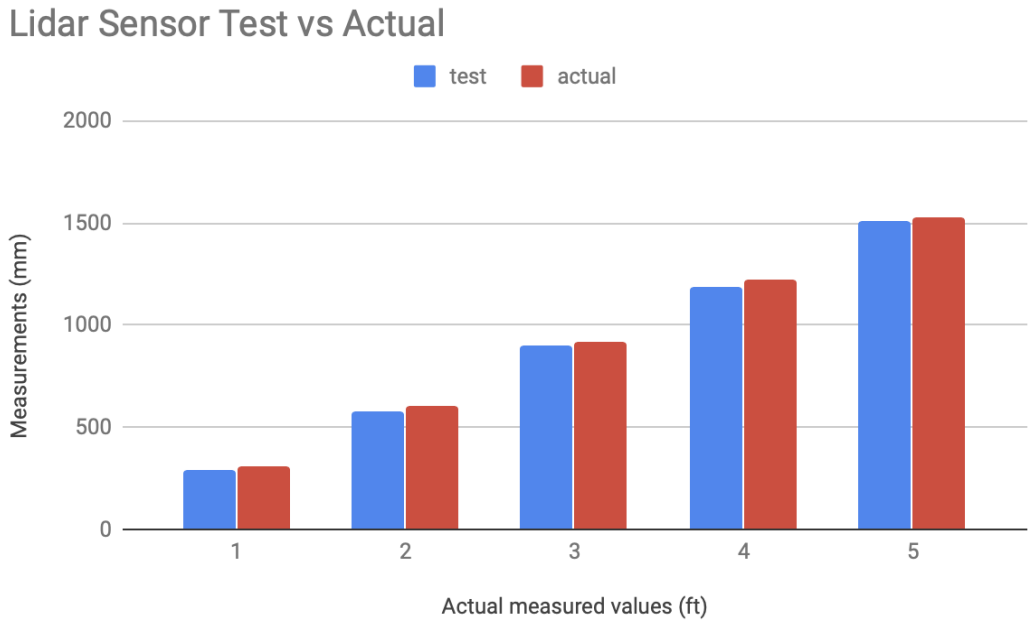

Figure 5: Verification test results for the LiDAR sensor- blue bar is sensor data, red bar is actual value

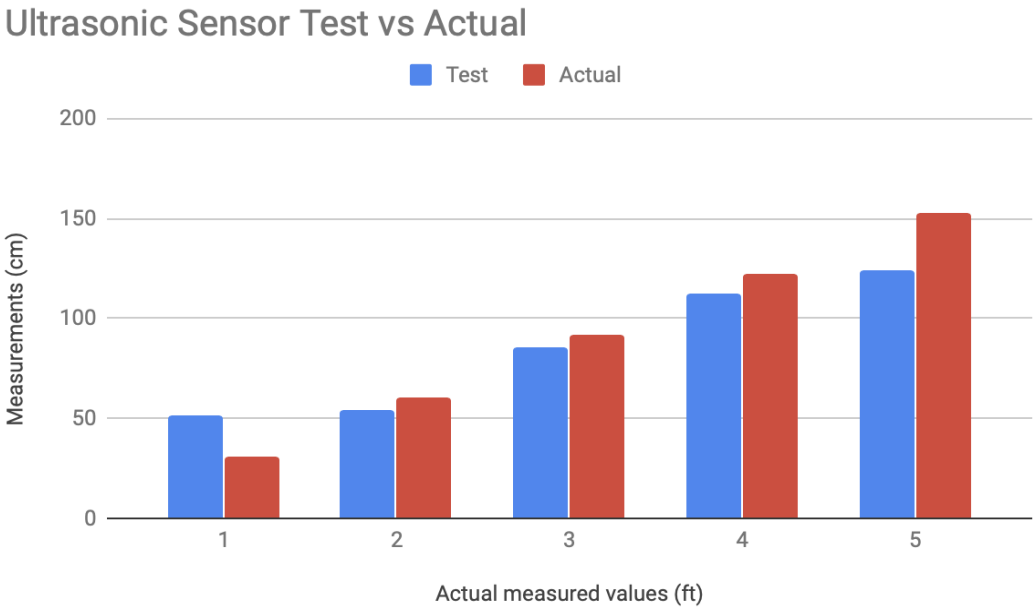

Figure 6: Verification test results for the Ultrasonic sensor- blue bar is sensor data, red bar is actual value

3.4 Encoder with regulator

The encoder requirement condition was established in order to stay within the voltage specifications of the Atmega I/O ports. The testing procedure consisted of testing the encoder combined with the voltage

regulator to ensure that, at the top speed for our design, the voltage and current thresholds were not exceeded. The encoder output values remained at and below 3.1 V and produced no more than 100 mA of current without the use of a regulator, so the encoder met these requirements natively.

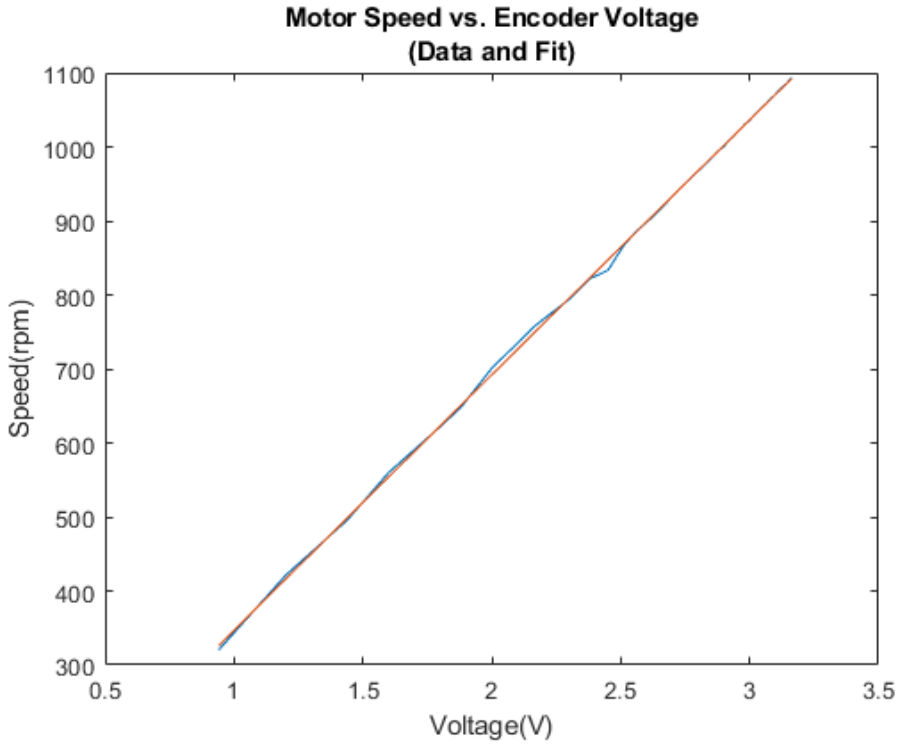

Figure 7: New encoder Voltage-Speed characteristics

$$Y = -.54x^2 + 346x + .25 \quad (3.1)$$

where Y = Duty Cycle, x = Motor Speed

3.5 Microcontroller

By the initial standards and requirements set in the design, the microcontroller fails requirements on the algorithm level, because the algorithm implements positional based control rather than velocity control as was initially envisioned. The reason that the algorithm was switched to a positional based control was due to the fact that mechanical portions of the treadmill no longer work, so there was no longer a reliable and objective method to quantitatively test the algorithm via a test car. A new verification was established to use a test function that inserts values into the algorithm, but it was realized that this procedure relies on real time data being constantly recorded from the encoder to accurately compare data, which would require a significantly larger amount of time to prepare than anticipated, so in the interest of time, a positional algorithm was implemented first, which later could be changed into a velocity based algorithm.

For testing the position based control algorithm, the system was configured in the same manner as was used in the verification procedure found in appendix A. But instead of measuring data based on predefined data values from a test file, the values were measured using the sensors directly, and the method to determine that the algorithm functioned correctly was by using a large flat surface, such as a piece of cardboard, and placing it within each region of activation, and determining that the PWM output signal of our microcontroller changes as described in section 2.3.1.2. That is, in the forward region the duty cycle increases, in the neutral region the duty cycle remains the same, and in the rear region the duty cycle decreases. When tested under this criteria, the microcontroller meets requirements for the positional control algorithm.

4. Costs

Make sure that any tables of costs are numbered, given titles, and cited directly in the text.

4.1 Parts and Labor

For labor costs, the estimate a fixed wage of $32/hr, 8 hours/week for three people to complete this project.

$$3 \cdot \frac{\$32}{hr} \cdot \frac{8hr}{wk} \cdot 8wks \cdot 2.5 \ = \ \$15{,}360$$

Table 1: Component costs

Component	**Cost**
Machine shop mechanical build quote	$175
Motor w/ Encoder [*;Minertia Motor R02MA203 w/ tg-3vc]	$150
Microcontroller [Digikey; ATmega328p]	$2.14
LiDAR sensor [RobotShop; tinyLiDAR ToF Range Finder Sensor]	$24.95
Ultrasonic sensor [Mouser; sparkfun SEN-13959]*4	$15.80
Buck converters [Banggood;Geekcreit 5A XL4005]*2	$5.20
Start/Stop Buttons [Amazon; HONBAYSPST Latching Type Push Button Switch] *2	$0.70
Assorted resistors/capacitors/transistors [Digikey; est.]	$10.00
Total Cost of Components:	$383.79

*It appears this motor is no longer in production, so price is based off of similar models on Ebay, and the advice of the power electronics department, who we are borrowing the motor from.

In total, our cost as a whole adds up to:
$15,743.79.

4.2 Schedule

Table 2: Weekly schedule

Week	Jacob	Charles	Jiangtian
2/25	Test motor, encoder and sensors. Procure and test buttons. Procure RC car that can fit test standards.	Assemble and test motor and microcontroller buck converters on breadboard	Buy sensors and microcontroller and get familiar with those
3/4	Consult with machine shop on finalizing design of treadmill. Work on mounting for sensors and buttons, and how to wire through the machine	Finalize functionality of buck converters making sure both function within tolerances	Test the basic sensor functionality using Arduino board
3/11	Work with Chas on PID for feedback control	Solder and put together buck PCB's. Begin work on control schema for feedback control	Keep testing and optimizing sensor and control algorithms to minimize the error rate
3/18	Spring Break	Spring Break	Spring Break
3/25	Test and finalize PID feedback control	Finalize functionality of feedback control	Finalize the sensor control algorithm
4/1	Make sure each part finishing the individual testing procedure and integrate together	Operate feedback control with other modules	Integrate with other parts and begin testing together
4/8	Combine and finalize optimization of all modules	Optimize functionality of remaining modules	Keep testing and optimizing in system level, make sure the treadmill is stable
4/22	Final Demo	Final Demo	Final Demo

5. Conclusion

In conclusion, the original manifestation of the system we wished to implement underwent a number of technical and physical iterations that deviated from the prototype that we had originally envisioned. We may consider the technical alterations that occurred to be primarily a direct result of parts specifications being chosen improperly, particularly the lack of a motor gear reduction as well as an appropriate belt and the associated accessories needed to turn the belt (e.g. the bearings); however, the main basis for the alternative design choices were a direct result of the methodology used for design. From the perspective of a designer, we could consider the method used to be a bottom up approach whereupon a top down approach would have been a more desirable methodological approach for design. In particular, the focus began by analyzing and attempting to implement low level components and work up to high level implementation as iterative designs did not work out. The idea stemming from this was originally the perspective taken in terms of how the project was evaluated; in a piecewise fashion. The modules and components that comprise the modules were analyzed as opposed to the totality of the system and the interconnectedness of the modules and components with one another. As the adage goes, “The whole is greater than the sum of the parts”. Once modules combine to become a system, operations may not function in the way that one would expect; that is, functional connectivity is not necessarily a linear process, and that was a lesson the team learned well.

5.1 Accomplishments

The main accomplishments attained during this project were: the implementation of the positional control schema, speed control circuitry for the motor, and the integration of the range sensors with the microcontroller. These accomplishments may be able to be used as a platform for future iterations of the project. The final version deviated from the velocity controlled architecture that had been originally envisioned; however, positional control can be used as a springboard for velocity control implementation insomuch that velocity is the ratio of positional change per unit time change, therefore positional control is inherently mandatory for the construction of a velocity control methodology.

5.2 Uncertainties

As mentioned in the conclusion, part of reason for the deviation from velocity to positional control stemmed originally from the lack of a belt; therefore, it is natural to begin with an analysis discussing the mechanical changes that needed to be implemented early on in order to promote the capability to quantitatively test velocity control. The belt spec’d for our purposes was far too stiff as its original intended purpose was for it to be used for conveyors. A more conscientious method of approach for properly speccing the belt would be to first choose from a list of belts used specifically for this purpose then to develop or research tests for determining max load as a function of the adjustments needed for both belt tracking and tightening. Additionally, the proper bearings would have needed to be chosen in such a way that the front and back drums of the treadmill would spin with the same amount of torque. In our case, the non-motor drum was mounted in bearings that were far too stiff; therefore the belt would buckle when torque was applied via the motor mounted on the opposite end.

In addition to the mechanical considerations, there was also the performance of the Atmega chip on the printed circuit board (or lack thereof). The inclusion of a surface mounted chip has the propensity to

perform better under nuanced situations, but for our purposes a through hole version of the chip would have been a considerable convenience in terms of PCB design as well as programmability. One of the issues was the chip testing was dependent purely on the PCB's ability to output the necessary functionality of the total electronic system. In our case, the Atmega was incapable of receiving the uploaded program from the programmer. It is theorized that upon multiple attempts at soldering and desoldering the chip to the board that the chip may have been damaged or a short could have occurred between the center ground pin and one of the critical pins (such as the power VCC pin); given a through hole component, a socket could have been soldered to the board and the chip could have been removed at will to be programmed on a breadboard for testing.

The final set of points to be made concerned the module integrations. A considerable barrier towards consolidating the sensors with the Atmega chip thereby warranting the move towards the arduino was the ultrasonic range sensors' failure to connect with the chip's I/O pins and output a signal within the chip's readable range. The sensor's echo pin output a signal less 7 mV; however, upon testing the Atmega in tandem with the Arduino while both were connected to an Ultrasonic sensor, the Arduino took in a data from the echo pin of its range sensor while the delivering a voltage range between 2.5 and 3.3 V which is within the readable range of the chip's I/O pins. Essentially the Atmega hardware somehow caused the output signal of the range sensor to become stifled during reads. Even with the presence of pull up resistors being engaged, the signal still was not sufficient enough to be read by the chip's I/O pins. Due to the devboard failing to read the echo pin signal of the ultrasonic sensors, we did not go through the effort of developing the I2C protocol of the tiny LiDAR sensors to test. We theorize that the ultrasonic sensors' recommended frequency and read times were not optimized and therefore caused severe interference when readings were occuring at the I/O pins of the Atmega. See appendix B, Table 4, for optimal operational frequency and read times for ultrasonic sensors.

In terms of integration, the final obstacle that needed to be overcome was the assimilation of the motor and the arduino (meaning the varying PWM signal output from the chip to the speed control circuit). The main issue concerning this problem was upon outputting the signal and attempting to use it to drive the FET of the motor driver circuit, the signal would become "washed out" (or flatten) at the drain when the motor was given power from an external power source despite the gate of the FET being ran with the same optimal signal from a function generator. It may be that the gate resistance was not sufficiently high enough to limit the current draw demand at the gate of the FET during turn on sequences. Logic circuits that are used to drive MOSFET's sometimes require a resistor of sufficient size to mitigate instantaneous current demands that cannot be met by logic circuitry or a controller.

5.3 Ethical considerations

Treadmills pose a number of safety and ethical risks. The most readily apparent risk is the fact that people can easily get hurt on a treadmill, violating IEE code of ethics #1: "to hold paramount the safety, health, and welfare of the public [...]" and #9: "to avoid injuring others [...]" [1]. On a typical treadmill people must match the speed of the treadmill, if the speed is too quick for them they will lose their footing and fall. The objective of this design is to help alleviate this issue by designing a prototype

treadmill that will match the speed of the user, rather than forcing the user to match the speed of the tread. However in the event that a fall still occurs, safety rails to help catch the user and an emergency stop button will be installed in the final iteration to help prevent it. In addition the sensors can be used to automatically stop the treadmill if it is sensed that the user is at a standstill or is no longer in view of the sensors. Finally, the testing procedure for velocity based control would be incredibly dangerous if we a live subject were to be used, so for this reason a scaled down treadmill and an RC car were intended to do testing, since no live subject would be needed that way.

Treadmills are also very large and difficult to move around, and since the project is in a an environment shared by peers, this could be seen as violation of IEEE code of ethics #10: “to assist colleagues and co-workers in their professional development [...]” [1], since this would be disrupting the work environment of our peers and possibly inhibiting their ability to do their work properly. As such it was decided to use a scaled down model of a treadmill to prototype our design, since this would greatly reduce the footprint it has in the shared workspace. However, scaling down the treadmill may pose a problem with both IEEE code of ethics #1 and #9 as stated above, and in IEEE code of ethics #3: “to be honest and realistic in stating claims or estimates based on available data” [1]. This breach in ethics is caused by the fact that a treadmill at a smaller size such as this may not scale up to a human size and function correctly still, which means that if it is scale it up without testing, it may bring harm to someone, and if the test results or the scalability of our system on a conceptual level were to be fabricated, it would be violating those codes. As such it was decided to use an algorithm that uses positional tracking and velocity measurement to drive the control system, as that is a more linearly scalable system than if pressure or force detection were to be used.

5.4 Future work

Future considerations for a more optimal build would be based off of the assurity that components are fully optimized mechanically; first and foremost. In addition, the methodology towards implementing a functional velocity controlled schema would first require a fully developed and tested positional control processing arrangement that is capable of taking in and differentiating between “good and bad” data within the predefined confines of the engineer. These boundaries would be defined such that the system will be able to match the user’s speed within a small percent error that will not noticeably inhibit impetus of the user when transitioning between velocities.

When developing the construct for velocity control, positional data from the range sensors will need to be optimized in the controller such that rates of positional change per static unit of time will be capable of yielding a duty cycle that varies as a gradient; that is, rates of velocity changes will need to be translated in a way that changes the rate of speed of the belt in conjunction with the rate of speed change with the user.

References

[1] Ieee.org, "IEEE IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html [Accessed: 7-Feb-2019]

[2] ELEC Freaks, "Ultrasonic Ranging Module HC-SR04," [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

[3] Robotshop, "tinyLiDAR Reference Manual," [Online]. Available: https://www.robotshop.com/media/files/pdf/tinylidar-tof-range-finder-sensor-datasheet.pdf

[4] Arduino I2C master library(originally by Wayne Truchsess), [Online]. Available: https://github.com/rambo/I2C

[5] MicroChip, "Atmega328PB Complete Datasheet," AVR® Microcontroller with Core Independent Peripherals and PicoPower® Technology, [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/40001906C.pdf

[6] MicroChip, "Atmega328PB Xplained Mini," [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/50002660a.pdf

[7] Allan G. Weber, "Using the I2C Interface on the ATmega328P and MC908JL16," Ming Hsieh Department of Electrical Engineering, EE459 Lx - Embedded Systems Design Laboratory, [Online]. Available: http://ee-classes.usc.edu/ee459/library/documents/I2C.pdf

[8] Yaskawa, 'Minertia Motor P Series: Small Size DC Servomotors', [Online]. Available: http://energy.ece.illinois.edu/files/2015/06/Yaskawa-MinertiaMotor-P12H-DB11.pdf [Accessed: 7-Feb-2019]

[9] International Rectifier, HEXFET Power MOSFET IRL8113, IR WORLD HEADQUARTERS, June 2004.

[10] Yaskawa, “Minertia Motor P Series,” P12H DC motor, June 1994

[11] Vania Nikolova, 'American Runners Have Never Been Slower (Mega Study)', Dec 2018. [Online]. Available: https://runrepeat.com/american-runners-have-never-been-slower-mega-study [Accessed: 7-Feb-2019]

Appendix A Requirement and Verification Table

Table 3: Requirements and verification table

Requirement	Verification	Verification status (Y or N)
1. Motor Motor must be able to adjust to a range of input voltages between 5 and 25 V within a timeframe of at most 1 s. Motor must drive the treadmill at a max speed of at least 10 mph.	1. Motor a. Attach motor to the treadmill. b. Attach input terminals of motor to a DC power supply. c. Attach output of encoder to a multimeter. d. Vary the value of the power supply from 5 v to 25 v, changing the step size from 1 v up to 5 v, and record via data from the encoder how quickly the motor reaches a steady state value, and make sure the maximum time frame any step requires to adjust speed is at most 1 s. e. Once the input voltage is at 25 V, check the drive velocity using the speed given by the encoder and the circumference of the roller the motor drives. Make sure that value is at least 10 mph.	Y
2. Motor Voltage Regulator Output current must be no more than 22 A.	2. Motor Voltage Regulator a. Attach the regulator power inputs to a 25 V power supply. b. Connect an output from a function generator to the regulator, and power it. c. Set the function generator to a 2.4 V peak to peak, 1.2 V offset square wave at 5 kHz. Set the duty Cycle to 99%. d. Connect a multimeter to the outputs of the regulator. Make sure the current the	Y

	regulator outputs is 22 A or less.	
3. Range Sensor Sensor must have at most 5% error up to 5 feet away.	3. Range Sensor a. Fix the sensor 18" above the ground, connect its data pins to a microcontroller's I/O ports, and connect a 5 V power supply to the sensor's power inputs. b. Setup an object with a width of at least 1' and height of at least 18", 5' away from the front of the sensor. c. Connect the microcontroller to a pc and display in real time the input values of the microcontroller. d. Turn on the sensor and average the data received over 100 values. Make sure the value it reads has at most 5% error. e. Repeat steps b-d, moving the object 1' closer each time, until you are 1' away.	Y
4. Encoder Output voltage can be no more than 5 V, and the output current must be no more than 200 mA.	4. Encoder a. Attach the motor to a power supply, with the encoder/regulator subsystem attached. b. Connect a multimeter to the outputs of the regulator. c. Set the input voltage to the rated motor voltage, 25 V. d. Make sure the voltage the regulator outputs is 5V or less, and the current is 200 mA or less. e. If the encoder alone does not meet these requirements, connect an output from the microcontroller to the buck converter, and power it. Make sure it outputs a PWM signal with a duty cycle of 23.8%.	Y

	f. Make sure the voltage the regulator outputs is 5 V or less, and the current is 200 mA or less.	
5. Microcontroller Data processing algorithm must interpret sensor data so that it can produce a velocity value with 5% accuracy.	5. Microcontroller a. Implement the data processing algorithm, as described in our Tolerance analysis. b. Append to this algorithm an algorithm to interpolate the positional data that is created, and generate a velocity based on the stored position data. c. Plug in microcontroller to a power supply and set it to 5 V. d. Implement a test subroutine that inputs a range of values that match the sensor values going from 3 to 10 mph. e. Compare velocity values given by the microcontroller to the known velocity. Make sure the algorithm outputs a proper duty cycle to appropriately match that speed.	N

Appendix B Essential Figures, Graphs, and Equations

Table 4: Ultrasonic sensor characteristics

Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

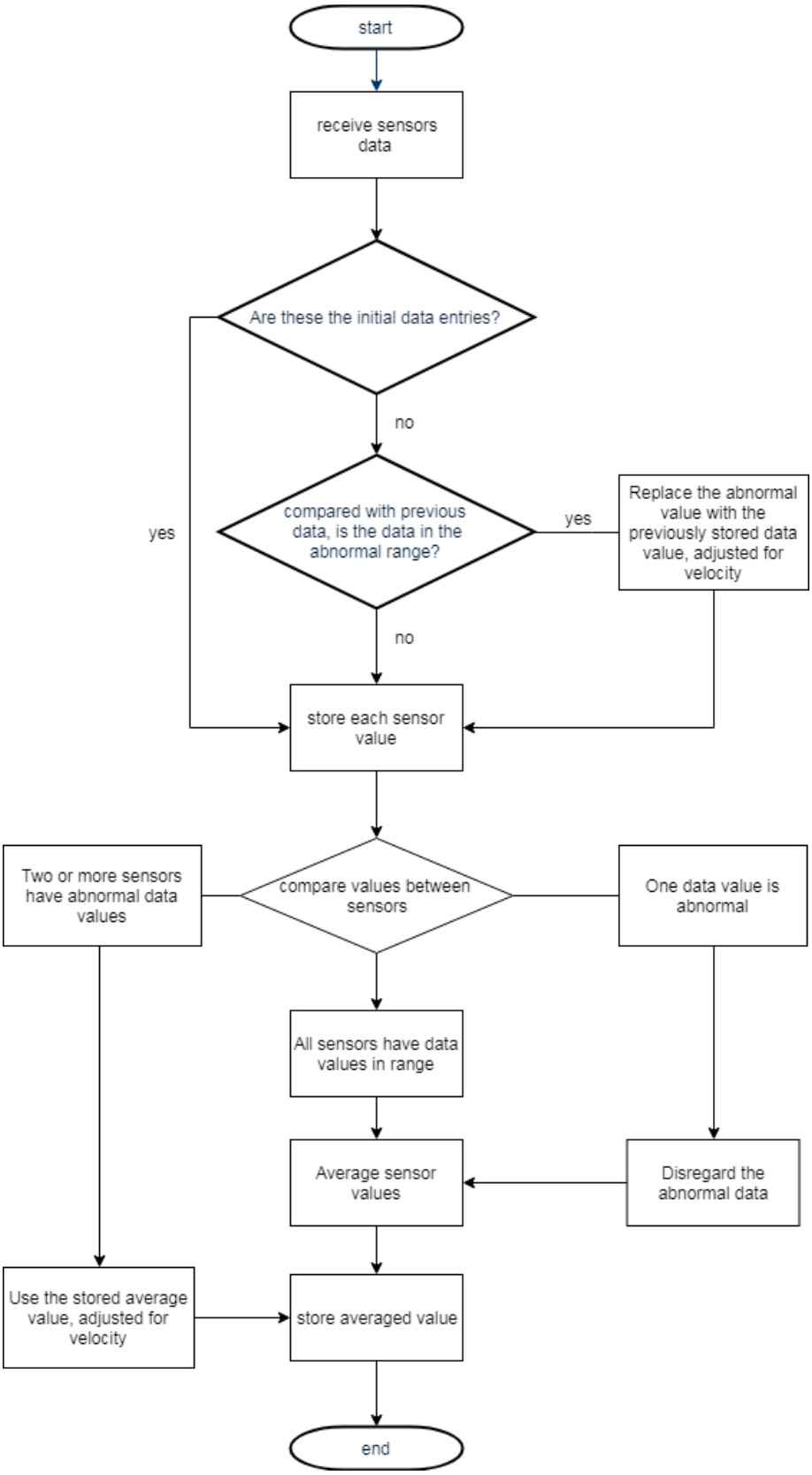

Figure 8: Four sensor data processing algorithm

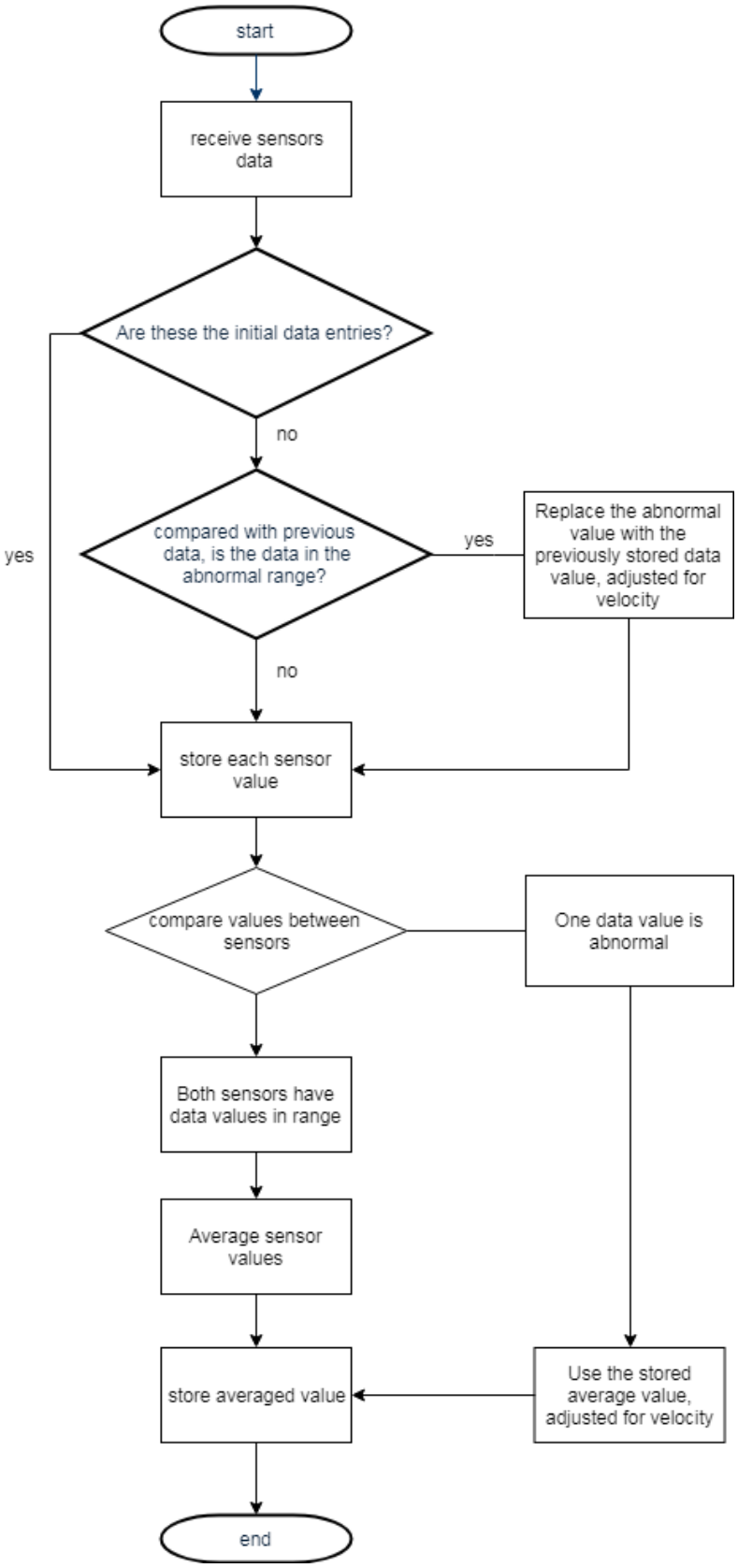

Figure 9: Two sensor data processing algorithm

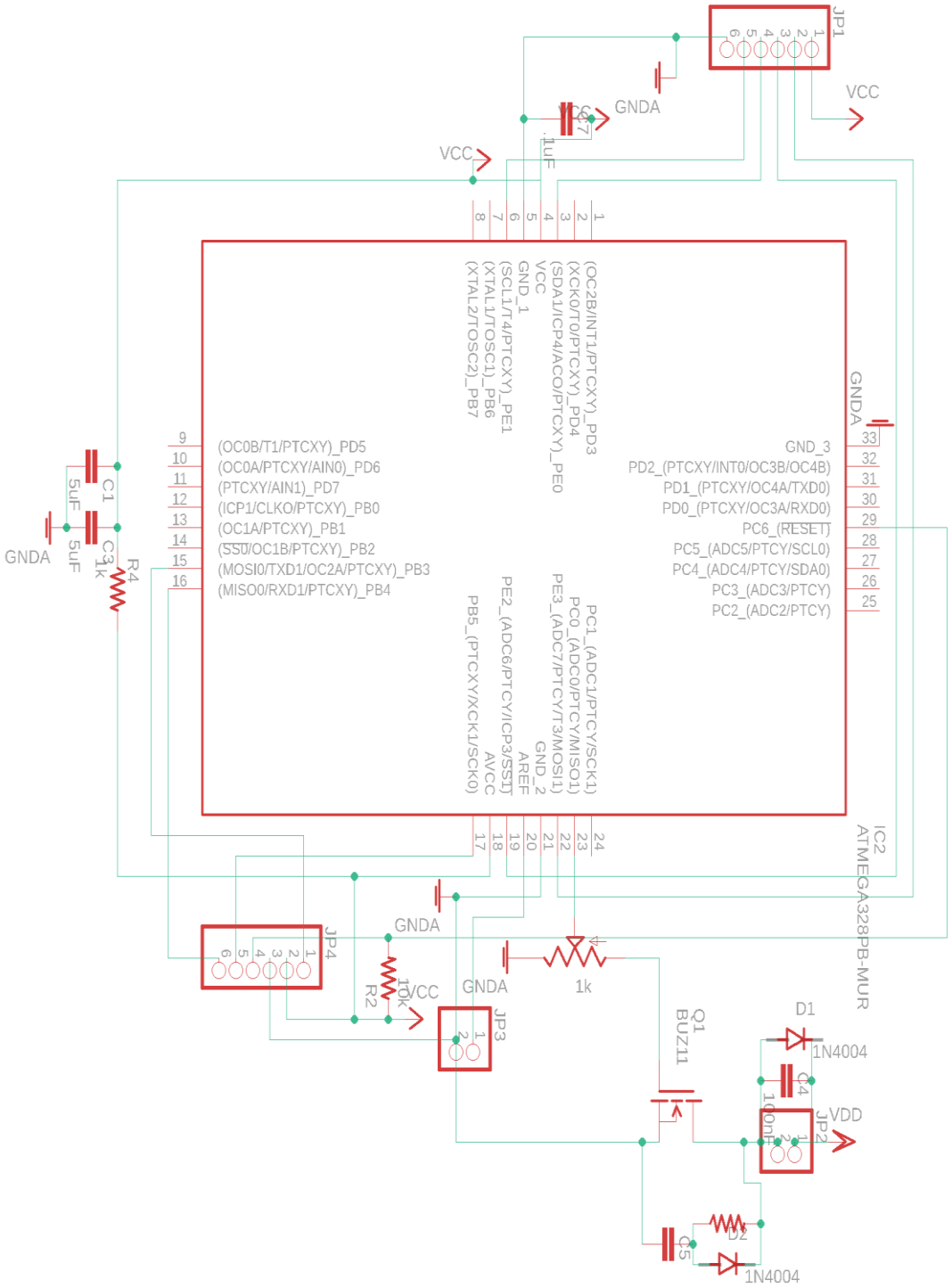

Figure 10: Circuit schematic

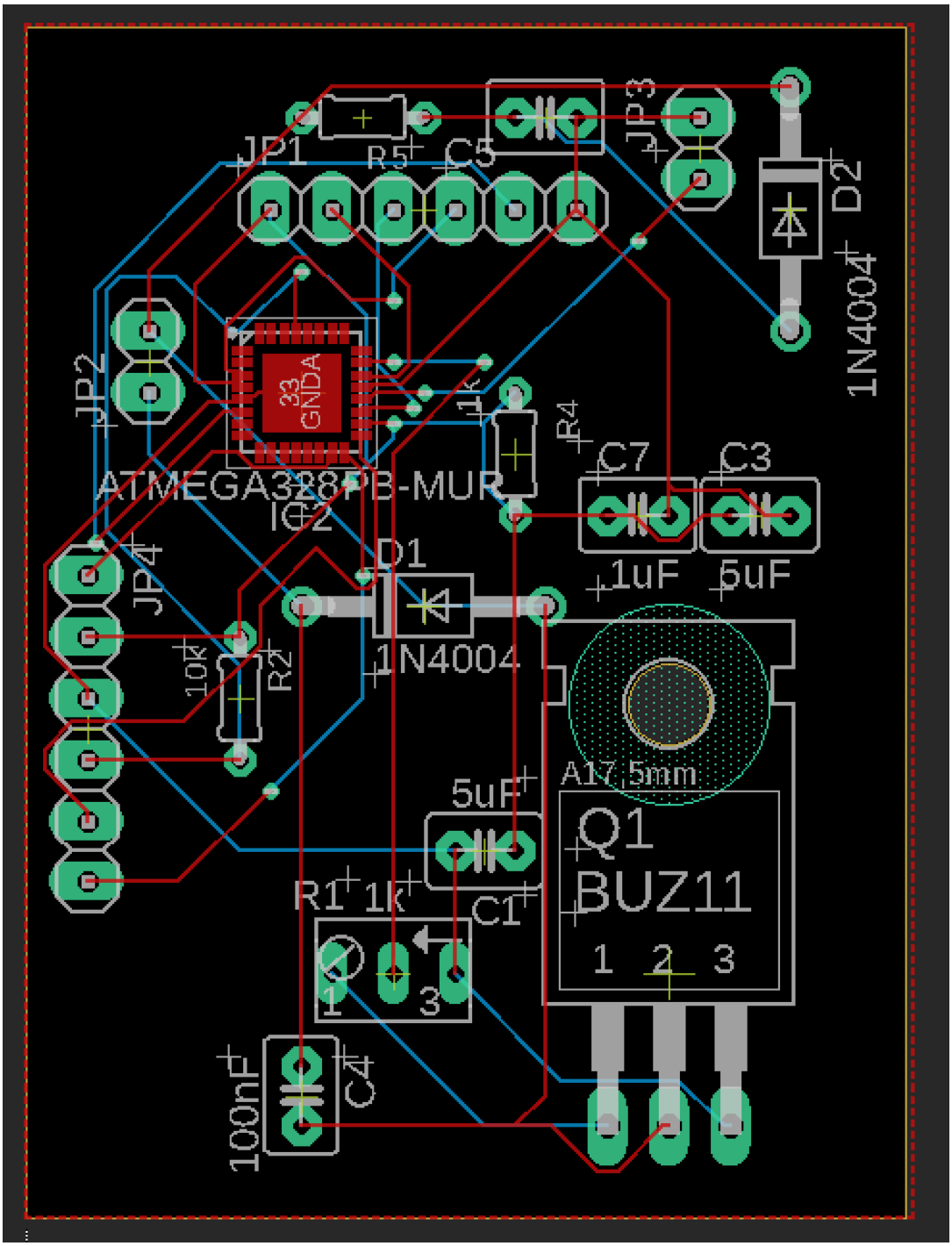

Figure 11: Circuit on PCB drawing

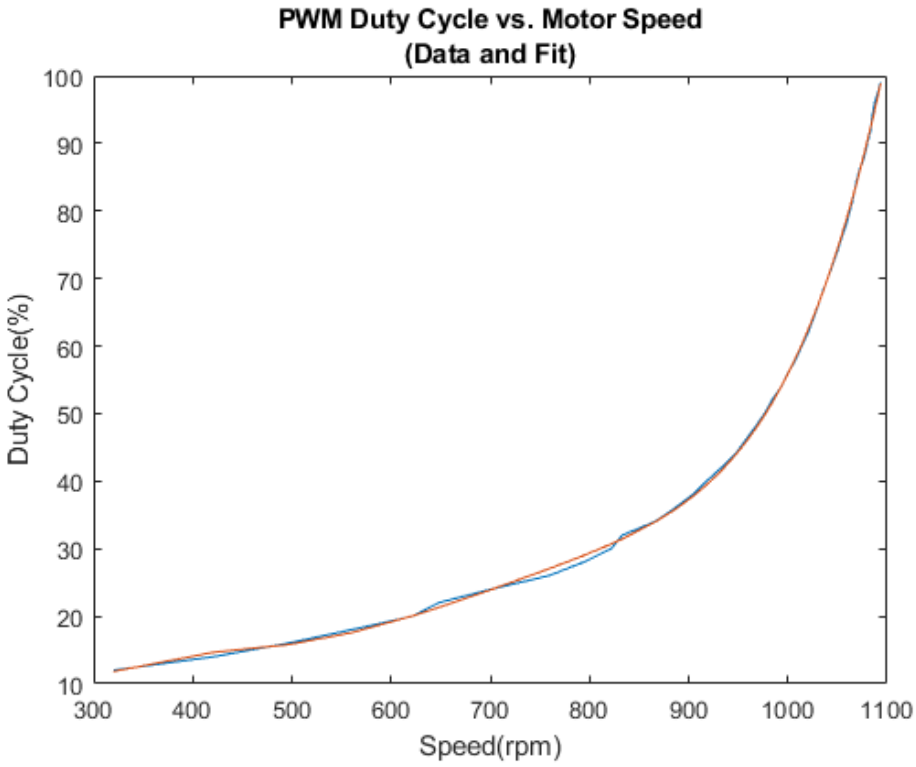

Figure 12: Duty cycle vs. motor speed characteristics

$$Y = 2.98 * 10^{-12}x^5 - 9.28 * 10^{-9}x^4 + 1.13 * 10^{-5}x^3 - 6.6 * 10^{-3}x^2 + 1.9x - 199 \quad \text{(B.1)}$$

where Y = Duty Cycle, x = Motor Speed