Gait Controlled Treadmill

ECE 445 Project Design Document- TA: Anthony Caton
Team 43 - Jacob Pruiett, Charles Leonard Lewis IV and Jiangtian Feng

1- Introduction

1.1- Objective

Indoor running on treadmills plays a significant role in our daily life. However, running on a treadmill can be an uncomfortable affair. This is due to users having to match the speed of the treadmill, rather than moving naturally at their own pace. It is also an common sight to see gym patrons holding on to the rails of the treadmill in order to get a sense of safety, since this unnatural pace can at times feel dangerous. Although there are a suite of protective features, including safety clips, stop buttons and handrails, it is still difficult for users assuage their fears. Solving this problem would serve to greatly improve user experiences with treadmills.

Our solution to this problem is to build a treadmill that automatically matches the gait of the runner. Upon start-up, the belt will have a slow initialization speed, the treadmill will then naturally control the belt speed according to runner's speed and position on the belt. Specifically, we divide the belt into three area: at the front area, the control system will increase the belt velocity; if the runner is at the rear area, the belt will slow down in response. The center of the belt will be the zero position, indicating the current speed is comfortable for runners. The system also adjusts the velocity of the belt based on the runner's velocity relative to the belt, accelerating and decelerating in response to a difference. This natural speed control system will prevent unexpected injury because the system will increase the belt speed in pace with the runner which allows them to warm-up, and to not be forced to run at a predetermined pace, allowing for a run that feels natural and safe.

1.2- Background

Historically indoor vs outdoor running has been a subject of debate on a variety of different fronts. While the debate of whether one is more challenging or better than the other remains in contention, the fact that the two differ in execution is not up for debate. One major fact is that outdoor running tends to be considerably more challenging, since the forces that need to be generated by the runner in order to accommodate velocity changes causes more wear on the runner when on pavement. On the other hand, the presence of the electromechanical devices forces the user to tamper with the controls on the treadmill whilst trying to run. This can be a deterrent for a hardcore runner, and may be a reason why so many exercise enthusiasts avoid using treadmills. Despite this disparity between elite athletes and the average runner, this does not diminish the popularity of this piece of equipment. According to the Consumer

Report Safety Commission, over 50 million Americans use a treadmill for activity needs [1]. This being the case, there is clearly a large interest in the treadmill as a viable piece of exercise equipment among the fitness community.

In addition to the popularity of the treadmill there is, more importantly, the risk of injury associated with its use. There is a plethora of articles and information concerning treadmill related injuries, such as there being over 70,000 mechanical exercise based injuries between the years 2007 and 2011 [1]. While the numbers are not as high as say automobile related injuries, there is still reason for concern especially considering the easy access to young children. While there are a number of safety mechanisms embedded in the electromechanics of a current day treadmill, such as those mentioned in the objective statement, they are mostly mechanical in nature and require user input. As such, the need for safety concerns is not eliminated, and automatic safety mechanisms are still in need.

1.3- High-Level Requirements

- Sensor subsystem must have greater than 95% distance accuracy at all ranges and speeds, and must report this data back with a sampling rate of at minimum 12Hz, as we want to collect data at about three times human reaction speed.
- Outside of a 9” center zone, the control system must begin responding to changes in user speed within half a second of that change occurring on the user end, and must be able to adjust the belt velocity to any value between 3mph and 10mph in at most 3 seconds.
- System must stop within 3 seconds depending on current velocity when nothing is detected by sensors, when the system has slowed down to a halt as a result of decreasing user speed, or when the stop button is pressed.

2- Design

A traditional treadmill needs three central components to function: a power supply, a control unit, and a motor. Treadmills typically take in power via a wall outlet and pass it through an AC/DC converter that supplies the controls unit with 5V, and a variable amount to the motor based on input from the control unit. In the control unit, we process data from between four and seven different sources to produce a single output signal that will adjust the velocity of the motor. Finally the motor will take in a range of voltages from the power supply and drive the treadmill, whilst feeding back its current velocity data to the control unit.

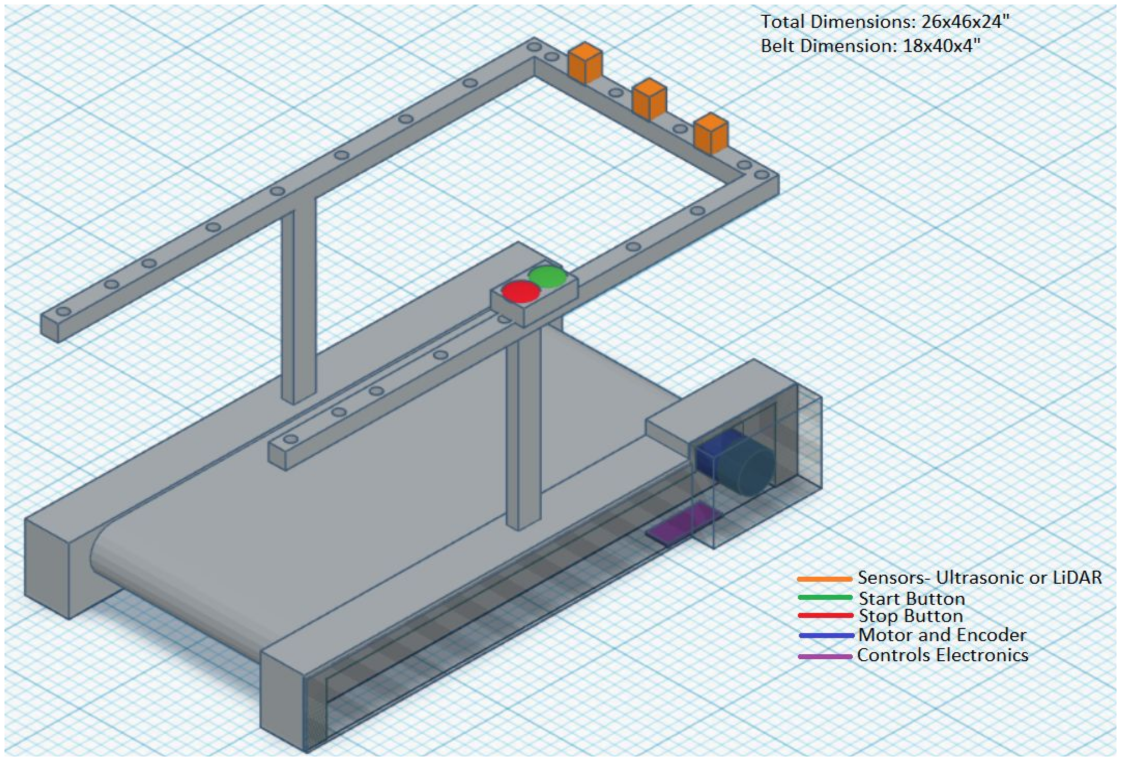


Figure 1. Physical Diagram

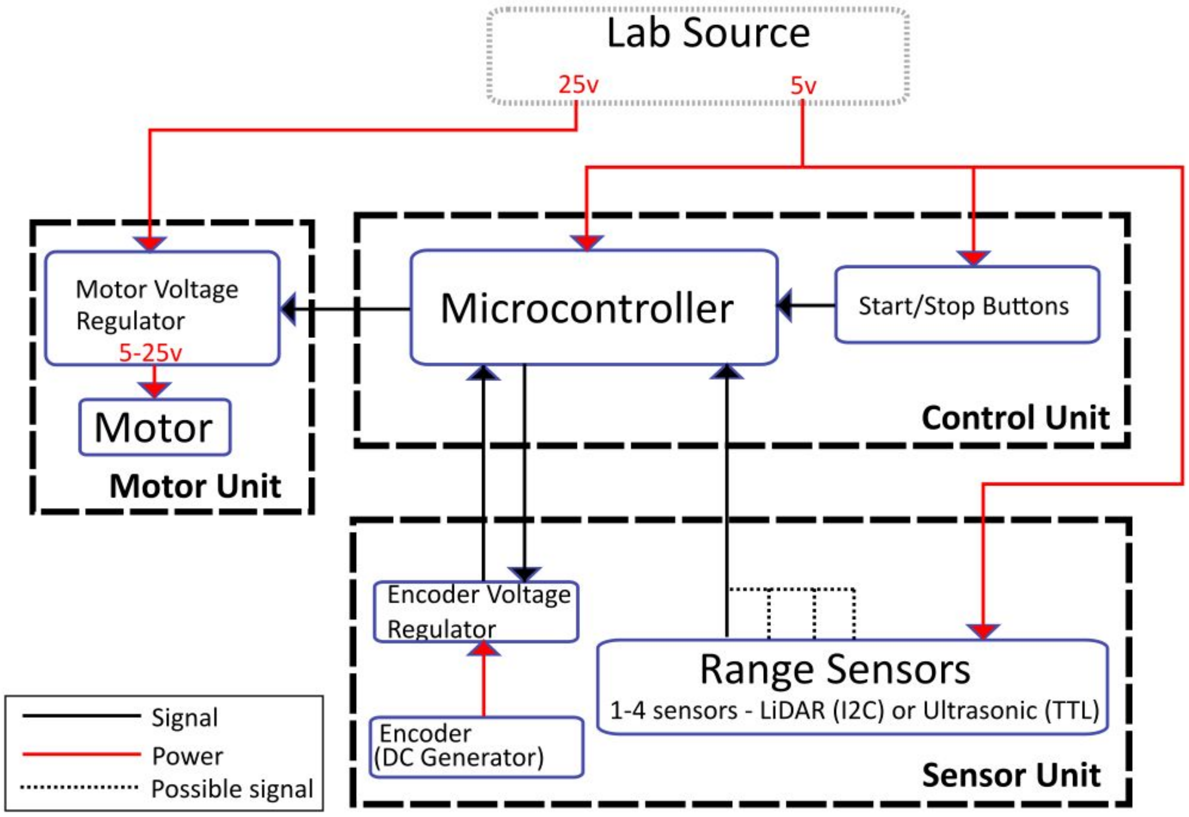


Figure 2. Block Diagram

2.1- Motor Unit

The motor unit will take in voltage from a lab source, and a signal from the microcontroller to drive the treadmill belt at a range of desired speeds.

2.1.1- Motor

The motor will take in voltage values from the motor voltage regulator, and speed up or slow down in response. It will drive the treadmill belt, meaning it is vital to the function of this system.

We will be using a Minertia small size DC r-series (r02m) servo motor we will not need to use a driver. The motor will be rated at a max voltage of 25.5V with a max torque of .5N*M at 3000rpm. We wish to use a high RPM low voltage motor as they tend to be cheaper and it is relatively easy to control the speed of them.

Requirements	Verification
Motor must be able to adjust to a range of input voltages between 5 and 25v within a timeframe of at most 1s. Motor must drive the treadmill at a max speed of at least 10mph.	a. Attach motor to the treadmill, such that it is driving the belt. b. Attach input terminals of motor to a DC power supply. c. Attach output of encoder to the microcontroller, and attach that to a PC to read total output. d. Vary the value of the power supply from 5v to 25v, changing the step size from 1v up to 5v, 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 1s. e. Once the input voltage is at 25v, check the velocity of the belt using the speed given by the encoder and the circumference of the roller the motor drives. Make sure that value is at least 10mph.

2.1.2- Motor Voltage Regulator

The motor voltage regulator will need to be a DC/DC converter that steps up voltage to meet operational voltage levels in the DC motor (in our case up to 25.5v for our current motor) based on input from the microcontroller. Although, amplifiers and boost converters are inherently unstable and dangerous to work with so we will instead build a buck converter that will take in voltage from an external source and buck it down to operational voltage levels of the motor depending on the PWM signal running gate of the active switch.

Requirements	Verification
Output current must be no more than 22A.	a. Attach the regulator to a 26V power supply. b. Connect an output from the microcontroller to the regulator, and power it. c. Make sure it outputs a PWM signal with a duty cycle of (25/26)*100% = 96%. d. Connect a multimeter to the outputs of the regulator. Make sure the current the regulator outputs is 22A or less.

2.2- Control Unit

The control unit will take in external user input as sensor data and be used as input into a unity gain closed feedback loop that will be used in addition to a PID controller taking in a sum consisting of both the sensory input minus the motor feedback info being relayed by the encoder. The microcontroller will allow us to set the PID gain constants in order to fine tune the transfer function performance.

2.2.1- Microcontroller

The microcontroller will be the heart of this unit, and will be where all of the input data is fed into, and where the output signal will originate from. We will use an ATmega328p since we do not need much memory to perform the operations we will be making, and it is inexpensive, which will allow us to put more money into other facets of the design. Its functionality will allow us to set and adjust the PID control parameters (connected through LabView via USB) as well as adjust the PWM signal going to the power supply of the DC motor.

Generally a basic treadmill PWM signal is controlled by an adjustable potentiometer, but our microcontroller will instead utilize a CCP (capture compare PWM) methodology for easily adjustable PWM control. In addition, the sensors will need to be fed into the control unit as an input signal to set the appropriate timer adjustments in the CCP modules in order to adjust the PWM signal (determined by the position control system algorithm) to the gate of the buck converter and out to the DC motor which in turn will adjust the speed (given by the voltage input level) and give feedback, via the encoder, back into the microcontroller and control circuit for processing.

Requirements	Verification
Data processing algorithm must interpret sensor data so that it can produce a velocity value with 5% accuracy.	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 5V. d. Upload program to microcontroller. e. Attach range sensors as appropriate for the implemented algorithm, and power them with the 5V power supply. f. Attach sensors to the treadmill in desired configuration. g. Place test car at end of treadmill, then run it at a constant 3mph. h. Compare velocity values given by the microcontroller to the known velocity. Make sure it does not differ from the actual velocity by more than 5% i. Repeat steps g and h at 5mph, and then again at 10mph.

2.2.2- Start/Stop Buttons

The start and stop buttons are a simple user interface for the sake of safety and ease of use. If the start button is pressed the system should begin moving and responding to user movement, and if the stop button is pressed the system should immediately begin slowing down to a halt.

2.3- Sensor Unit

2.3.1- Range Sensors

The sensors will be vital to the function of our system. In order to most accurately measure on the position of our user, we will test a number of configurations and type of sensors, namely ultrasonic and Lidar sensors. These configurations and how we interpret this data to reduce error is further extrapolated upon in our tolerance analysis.

In this section, we are most concerned with the qualifications of each individual sensor. The sensors will be connected to the I/O pins of the microcontroller, and will be used to determine both the position of the user, and their velocity relative to the treadmill. If we use Lidar sensors, they will communicate to the microcontroller via an I2C protocol. Our ultrasonic sensors communicate with our microsensor via TTL signals.

Requirements	Verification
Sensor must have at most 5% error up to 6 feet away.	a. Fix the sensor 24” above the ground, connect its data pins to a microcontroller’s I/O ports, and connect a 5V power supply to the sensor’s power inputs. b. Setup an object with a width of at least 1’, 6’ 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 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.

2.3.2- Encoder w/ regulator

The encoder will measure the angular velocity of the motor as it's running, and output that data to the microcontroller. This is important so that our control algorithm can know what the current speed of the treadmill is. For this purpose we will be using a preinstalled Yaskawa 3vc tachometer generator, which outputs a voltage signal that correlates to the rotational velocity. Because it is a DC tachometer generator, it does not require any power supply, and outputs a voltage value in real time according to the speed of the motor shaft.

However, this then means that we need to make sure that the output voltage can be read by the microcontroller without harming it, so we run the output values of the encoder through a buck converter to step down the voltage to levels the microcontroller can tolerate. The buck converter will be sent a constant PWM signal from the microcontroller to drive it, and the signal will have a duty cycle of 23.8%, since that is the ratio from 21V to 5V, which is the highest output voltage of our encoder, to the highest input voltage of our microcontroller.

Requirements	Verification
Output voltage can be no more than 5V, and the output current must be no more than 200mA.	a. Attach the motor to a power supply, with the encoder/regulator subsystem attached. b. 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%. c. Connect a multimeter to the outputs of the regulator. d. Set the input voltage to the rated motor voltage, 25V. e. Make sure the voltage the regulator outputs is 5V or less, and the current is 200mA or less.

2.4- System wide requirements

Requirements	Verification
The system must be able to respond to changes in the velocity of a user completely within 3 seconds after they leave the 9” long zone at the center of the belt, and begin the shutdown process after slowing down below the minimum velocity.	a. Attach all modules together, as shown in our circuit schematic and block diagram. b. Place test car on the treadmill in the center of the belt. c. Press the start button and turn on the treadmill. d. Input voltage to the car such that it moves at 3mph. It should remain stationary. e. Increase the speed of the car to 4mph, after it leaves the equilibrium zone, the system should at minimum match that velocity within 3 seconds. f. The belt should become slightly faster than the car, and push it back to the equilibrium zone. g. Repeat this process from 4mph up to 6 mph, 6mph down to 3mph, and from 3mph up to 10mph, then slowly step down this speed 1mph at a time until the car is below 3mph, after which the treadmill should begin the shutdown process.

2.5- Tolerance analysis

The most important tolerance to keep track of in our system will be in our sensors. We will be comparing different sensors in different configurations to determine a setup that best suits our needs. The two types of sensors we wish to test are LiDAR and ultrasonic sensors, and there are three different configurations we will test with in mind. A single sensor at one of the rear corners, three sensors at the front, and four sensors, one in each corner. The single sensor setup will be with our LiDAR sensor, whilst the three and four sensor setups will use our ultrasonic sensors. Because the data we actually need is how far the user is from the rear of the treadmill, we will take the data from the sensors and put them through a few equations:

For a front sensor:

$$Distance = SensorData \qquad \text{Eq.1}$$

For a corner sensor:

$$Distance = \sqrt{SensorData^2 - (9")^2} \qquad \text{Eq.2}$$

Then average the different distance values:

$$FinalDistance = \sum_{1}^{i} \frac{Distance_k}{i} \quad \text{i = \# of sensors, k = current sensor} \qquad \text{Eq.3}$$

We will then do an error analysis with real world measurements:

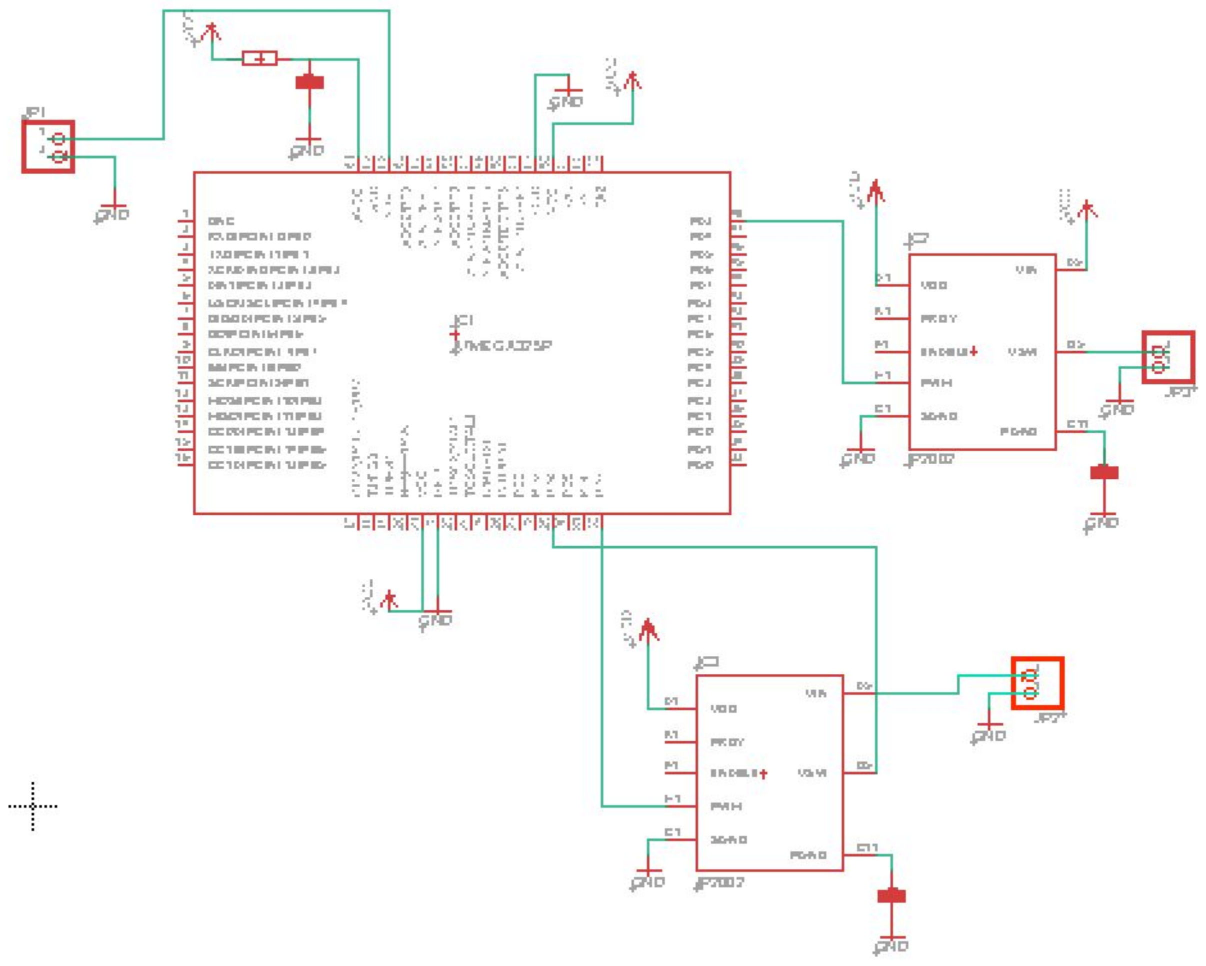
$$Error = \frac{RealDistance - FinalDistance}{RealDistance} * 100\% \qquad \text{Eq.4}$$

Getting these values to be accurate is important because any errors they have compounds through the rest of our system. An inaccurate distance reading will lead to an inaccurate velocity reading, which will in turn cause the microcontroller to send the wrong voltage signals, which finally will cause the motor to accelerate to the wrong velocity, leading to the treadmill to match a velocity the user is not running at. The issue with this is that the user will then have to slightly adjust their speed to match, which will in turn elicit more speed change from the controller, and form a feedback loop based on this error that gets worse with time.

To counteract these error issues, we will run those data through an algorithm in our microcontroller before passing those to the main motor control algorithm. Our algorithm depends on the specific sensors configuration.

Here is a breakdown of our algorithm. To clarify, any mention of “abnormal” indicates any data more than five inches larger or smaller than the data entry it is being compared with, since at a sampling rate of 60Hz that indicates a velocity of over 17mph, which is very improbable for the average human. The first step after receiving data from the sensors is to check whether these are the initial data entries. If they are, these values are immediately stored for future use. On the other hand, if these data values are not the initial data entries, then we compare them with previously stored data in order to eliminate abnormal sensor data. After storage of data, our algorithm differs depending on the number of sensors in our configuration. For the two LiDAR sensor configuration, if one of the sensors is abnormal, we use the previous averaged data, adjusted for the current user velocity. However, for the four ultrasonic sensors configuration, if there is one sensor with abnormal data value, we just average the other three and ignore the abnormal one. If there are two or more sensors with abnormal values, we use the previous averaged data, adjusted for current user velocity. After processing out abnormal data, we then proceed to average together all of the data as per Eq.3, then push that data to memory and use it in our control algorithm. Our algorithm is displayed in separate flow charts on the next two pages.

2.6- Circuit Schematic



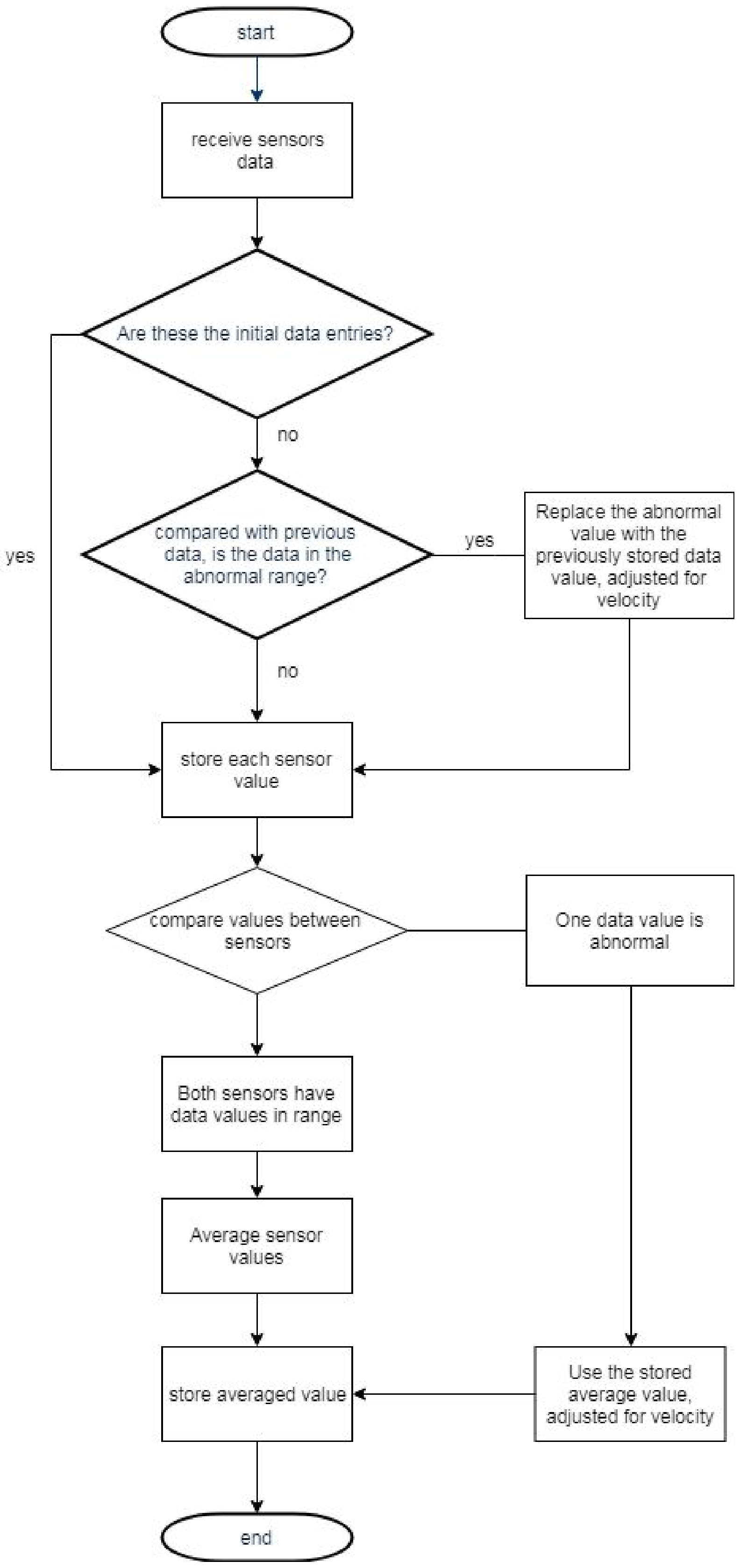

start
receive sensors data
Are these the initial data entries?
yes
no
compared with previous data, is the data in the abnormal range?
yes
Replace the abnormal value with the previously stored data value, adjusted for velocity
no
store each sensor value
compare values between sensors
One data value is abnormal
Both sensors have data values in range
Average sensor values
store averaged value
Use the stored average value, adjusted for velocity
end

Figure 3. Tiny LiDAR Sensor Configuration(2) Processing Algorithm

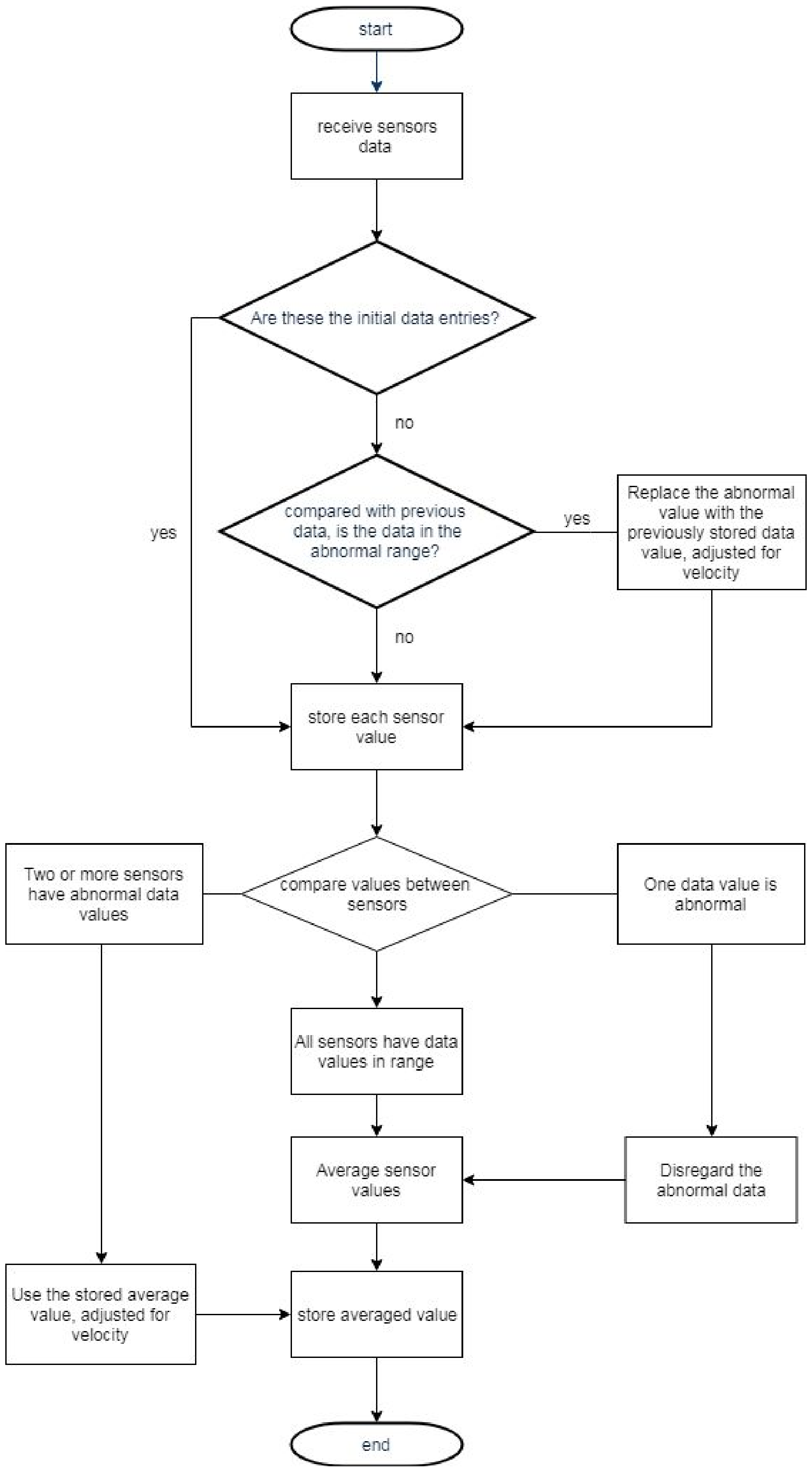

start
receive sensors data
Are these the initial data entries?
no
yes
compared with previous data, is the data in the abnormal range?
yes
Replace the abnormal value with the previously stored data value, adjusted for velocity
no
store each sensor value
Two or more sensors have abnormal data values
compare values between sensors
One data value is abnormal
All sensors have data values in range
Average sensor values
Disregard the abnormal data
Use the stored average value, adjusted for velocity
store averaged value
end

Figure 4. Ultrasonic Sensor Configuration(4) Processing Algorithm

3- Costs

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

$$3 \cdot \frac{\$32}{hr} \cdot \frac{8hr}{wk} \cdot 8wks \cdot 2.5 \;=\; \$15,360 \qquad \text{Eq.5}$$

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- 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	Begin testing the sensor and control algorithms on microcontroller
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- Ethics and Safety

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 [...]” [2]. 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 our 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, we will ensure that safety rails to help catch the user and an emergency stop button is installed to help prevent it. In addition we will use our sensors 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. In addition, the testing procedure would be incredibly dangerous if we were to use a live subject, so for this reason we have opted to use a scaled down treadmill and an RC car to do testing, since no live subject will be needed that way.

Treadmills are also very large and difficult to move around, and since we will be making ours in a an environment shared by our peers, this could be seen as violation of IEEE code of ethics #10: “to assist colleagues and co-workers in their professional development [...]” [2], since we would be disrupting the work environment of our peers and possibly inhibiting their ability to do their work properly. As such we have decided to use a scaled down model of a treadmill to prototype our design, since this would greatly reduce the footprint we have 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” [2]. 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 we scale it up without testing, it may bring harm to someone, and if we lie about the test results or the scalability of our system on a conceptual level, we would be violating those codes. As such we have decided to use an algorithm that uses positional tracking and velocity measurement to drive our control system, as that is a more linearly scalable system than if we were to use pressure or force detection.

References

[1] Graves JM, Iyer KR, Willis MM*, et al*
Emergency department-reported injuries associated with mechanical home exercise equipment in the USA
Injury Prevention 2014;**20:**281-285.

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