Proceedings of the ASME 2021
International Mechanical Engineering Congress and Exposition
IMECE2021
November 1-5, 2021, Virtual

IMECE2021-69308

DESIGN OF AN EFFICIENT, LOW-COST, STATIONARY LIDAR SYSTEM FOR ROADWAY CONDITION MONITORING

Jarod Bennett, Mather Saladin, Daniel Sizoo, Spencer Stewart, Graham Wood, Thomas DeAgostino, and Christopher Depcik*

Department of Mechanical Engineering, University of Kansas, Lawrence, KS, United States
*Corresponding Author

ABSTRACT

Light Imaging Detection and Ranging (LiDAR) systems generate point cloud imagery by using laser light to measure distance to a surface and then combine numerous points to create a three-dimensional (3-D) image. Since early adaptations, LiDAR is now common in aerial and subterranean geographical surveying and autonomous vehicle operations. The transportation industry uses LiDAR to monitor roadway quality, which can allow hazardous roadway corrosion to be spotted and repaired before endangering drivers. However, a leading issue with LiDAR availability is the respectively high price point for effective systems, therefore preventing widespread usage.

Previous work at fabrication of a low-cost LiDAR system generated high resolution 3-D imagery but was faulted by limited portability and a long run-time while also finding issues with gimbal translation and C++ programming. This effort improves the prior work by combining a touchscreen Graphical User Interface (GUI) with a rangefinder (Garmin LiDAR-Lite v3HP) powered by Raspberry Pi 4 Model B hardware. The rangefinder is housed in a 3-D printed gimbal mount that translates via two stepper motors and driver board. The system runs via a Python script that allows the user to select varying levels of resolution on the GUI prior to data collection onto a Secure Digital card or a file accessible through an internet connection. Like the previous work, data output is in Cartesian coordinates through a .xyz file format with a MATLAB script used to create a point cloud and two-dimensional image with a depth gradient. Overall, a more efficient, easier to use, and accurate LiDAR system was created that offers various resolution levels for under the cost of $500.

Keywords: LiDAR, rangefinder, point cloud, transportation, three-dimensional mapping, inexpensive

NOMENCLATURE

2-D	Two-Dimensional
3-D	Three-Dimensional
GUI	Graphical User Interface
LiDAR	Light Imaging Detection and Ranging
MATLAB	Matrix Laboratory
SCL	Serial Clock Line
SDA	Serial Data Analyzer
SRAM	Static Random-Access Memory
USB	Universal Serial Bus

1. INTRODUCTION

Transportation plays a vital role in society. Whether driving personal vehicles, transporting goods, or leisurely activities like riding a bike, people depend on transportation daily. As safety technology is advancing in automobiles, the associated infrastructure is aging. In the United States, accidents caused by road conditions were estimated to cost $217.5 billion [1]. Studies have shown that potholes alone cost American drivers approximately $3 billion in property damage per year [2]. Building an affordable system that can quickly detect poor road conditions can help reduce the amount of roadway accidents, lower economic losses, and potentially save lives. Analogously, newer cars use sensors that survey the surrounding area to quickly warn drivers of any hazards and even act in their place (e.g., Automatic Emergency Braking [3]); thereby, increasing the driver's safety. Although there are existing commercial systems mapping roadways, minor road defects may go unnoticed. If smaller and cheaper systems could be implemented into existing road safety fleets, a higher percentage of road defects may be detected.

One such potential system, Light Imaging Detection and Ranging (LiDAR), came to relevance in the 1980's after initially being applied to aerospace systems in the 1960's. Since these early adaptations, LiDAR systems are now common in aerial and subterranean geographical surveying [4]. Like radar systems, laser rangefinders can collect distances to a point using light wave lasers. By combining these recorded distances with rotation of the system, data are combined into a point cloud image. This provides a three-dimensional (3-D) detailed image of the area recorded. In today's applications, ground and airborne LiDAR systems are used for numerous applications including measurements of atmospheric carbon dioxide absorption, mapping forests and agricultural land, population distribution through city mappings, and characterization of autonomous navigation [5-8]. Currently, LiDAR systems are being utilized on the hardware side of autonomous systems [8]. While these commercialized systems are efficient, they rely on equipment that may be too expensive for consumers or vehicle fleets.

The high cost threshold of LiDAR, specifically for transportation-based activities revolving around safety, is evident after reviewing the Transport Research International Documentation database. In this database, numerous projects were found involving the use of LiDAR to evaluate lane width estimation in work zones [9], assess the risk of landslides [10, 11] and natural gas pipelines [12], plan seal coats [13], capture highway elevation data for stormwater runoff [14], survey railroad grade crossings [15], scan concrete surfaces to determine the degree of warping [16], determine rail crossing roughness [17], and evaluate traffic data at intersections [18]. All these projects employed relatively expensive commercial hardware (est. Velodyne VL-16 $8k, Velodyne HDL-32E $13k used, Velodyne HDL-64E $100k, RIEGL VZ-400 $30k, Trimble Tx5 3D $20k used, five SICK LMS 511 $3.5k each, Leica ScanStation C10 $17.45k used, and Optech Lynx SG1 unknown). One project did indicate the measurement of particulate matter using a constructed LiDAR system for ambient air quality; however, no information on the cost of this system was found and it was not constructed for mobile activities [19]. Similarly, in the Transportation Research Board's Research in Progress database, LiDAR is mentioned many times. For example, it has been used for sinkhole detection [20] and measuring bridge clearance heights [21]; yet, it appears these projects also involve commercial systems. One effort did construct a low-cost LiDAR system tasked with tracking freight trucks [22], but since it only measured a single point in space it had to be combined with a camera for vehicle identification [23]. Perhaps what is most interesting is a current project by the University of Kentucky that is investigating the benefits and costs of using LiDAR technology for transportation while stating that "the expense of data collection and post-processing may outweigh the overall benefits" [24]. Companies, such as Waymo, who began selling their Laser Bear Honeycomb LiDAR sensor on the market for $7,500, are currently finding it difficult to cut costs [25]. Therefore, developing an affordable yet efficient LiDAR system will allow the technology to move into industries that can apply its benefits quicker.

As a result, this effort focuses on construction of a relatively inexpensive LiDAR system for monitoring roadway surfaces. Previous work in this area at the authors' institution included the fabrication of two working prototypes. The first was assembled using a Garmin LiDAR-Lite v3 device as the range sensor and an Arduino Mega 2560 v3 as the microprocessor [26]. This system was able to generate 3-D point clouds with similar accuracy as many commercial systems with a final cost of less than $300. However, it had limited portability since it required a hard connection to a large Direct Current power supply and needed a significantly longer amount of time (e.g., 130 minutes for a 700,000 data point cloud) than commercial systems. The second prototype upgraded the hardware to a Garmin LiDAR-Lite v3HP rangefinder to increase update rates while lowering current consumption and enhancing the microprocessor to a Raspberry Pi 4 Model B. This lowered system runtime by increasing clock frequency and Static Random-Access Memory (SRAM) capacity. This second prototype also included a redesigned housing for the LiDAR rangefinder and two 28BYJ-48 Stepper Motors. While the housing was assembled with primarily 3-D printed components, after testing it was found to have stability deficiencies and the point clouds generated contained offset points due to the housing being inadequate to support the weight of the motors.

This effort continues the development of a low-cost stationary LiDAR system by modifying key components to address previous issues. The stepper motors were tested to ensure system stability during runtime, while a new Graphical User Interface (GUI) was added to increase usability while offering multiple levels of point-cloud density. The system software was rewritten in Python instead of C++ providing added capabilities for users. Finally, the entire system was packaged together to prevent any wired components from becoming disconnected to complete an instrument that should cost less than $500 while producing varying densities of point clouds in an efficient manner.

2. MATERIALS AND METHODS

LiDAR systems detect point distances by calculating the time between emitting a light beam and receiving the signal back from an object. To develop a 3-D point cloud, a laser rangefinder must be combined with several components to scan objects or surfaces. To create 3-D point clouds, the final configuration of the LiDAR system includes a laser rangefinder to collect distances, two stepper motors with driver boards to provide rotation in the horizontal and vertical directions, and a microprocessor to execute the code and store data.

2.1 Component Selection

At the end of the rangefinder analysis, a Garmin LiDAR-Lite v3HP optical sensor was selected after comparison to other options. Since low-cost is a key project outcome, the Garmin product provides sufficient performance for the LiDAR system's

application at a reasonable cost ($150) in comparison to the other options: TF03 Long-Distance LiDAR Module ($230) and TeraRanger Evo 60m USB ToF Rangefinder ($118). Key characteristics of the rangefinder for the LiDAR system were researched and compared in Table 1.

TABLE 1: COMPARISON OF THE DIFFERENT RANGEFINDERS CONSIDERED

Specifications	Lidar Lite v3HP	TF03 Long-Distance	TeraRanger Evo
Dimensions (W x D x H)	24.5 x 53.5 x 33.5 mm	44 x 43 x 32 mm	29 x 29 x 22 mm
Mass	34 grams	77 grams	12 grams
Range	1 m to 40 m	0.1 m to 180 m	0.5 m to 60 m
Current Consumption	85 mA	180 mA	210 mA
Update Rate	> 1000 Hz	1 to 1000 Hz	240 Hz
Cost	$150	$230	$118

The Garmin LiDAR Lite v3HP provides the fastest update rate while requiring the lowest current, along with being packaged at a reasonable size and weight to ensure smooth translation during data acquisition. Although it is not the least expensive rangefinder available, it is the ideal component for this application as it provides accurate distance readings (+/- 2.5 cm at distances > 2 m) within its range for a relatively low cost.

For both the vertical and horizontal motors, HiLetgo 28BYJ-48 stepper motors were chosen due to their low cost and high precision. Three other motors were considered, a SureStep Nema 8 Bipolar stepper motor, and both STEPPERONLINE short and SureStep full bodied Nema 17 stepper motors. These four motors are compared in Table 2.

TABLE 2: REVIEW OF RESPECTIVELY LOW-COST MOTORS AVAILABLE FOR RANGEFINDER TRANSLATION

Specifications	28BYJ-48	Nema B	Short Body Nema 17	Full Body Nema 17
Manufacturer	HiLetgo	SureStep	STEPPERONLINE	SureStep
Mass	50 grams	60 grams	140 grams	360 grams
Steps/Revolution	4096	200	200	400
Cost	$2.80	$22.69	$10.99	$17.95
Torque	3.5 Ncm	1.6 Ncm	13 Ncm	48 Ncm
Voltage	5 V	24 V	3 V	3 V

The 28BYJ-48 is capable of 64 steps per revolution, which itself does not outpace the other motors (200, 200, and 400 steps per revolution, respectively). However, coupled with a gear reduction ratio of 64, the 28BYJ-48 can produce a total output of 4096 steps per revolution, outperforming the others significantly. In addition to its higher precision, the 28BYJ-48 motors are unipolar; whereas, the other motors considered are bipolar; meaning that switching direction is more complicated and requires an H-bridge logic board, like a HiLetgo STMicroelectronics L298N. Due to the fact that the system oscillates horizontally while capturing data, the ability to switch direction quickly and easily is important. An H-bridge logic board would also require an additional 6 VDC battery supply, which increases the size and cost of the system. The 28BYJ-48 motors require a HiLetgo ULN2003 driver board; however, this board is capable of powering itself from the microcontroller and it is designed to have a simple plug connection to the motor, eliminating the chance for a wiring mistake. The 28BYJ-48 stepper motors also have the benefit of being less expensive that the others. The drawback of this motor is a limited torque output of 3.5 N·cm, which still outperforms the Nema B at 1.6 N·cm, but falls short of the Short Body Nema 17 and Full Body Nema 17 motors, at 13 N·cm and 48 N·cm, respectively.

To run the code to control the motors, rangefinder, and store data in an efficient manner, a relatively powerful microcontroller is needed. After researching cost-effective microcontrollers, three options were analyzed: Raspberry Pi 4B, Rock Pi 4 Model C, and Odroid-XU4. Their key specifications are compared in Table 3.

TABLE 3: ANALYSIS OF MICROCONTROLLERS CONSIDERED

Specifications	Raspberry Pi 4B	Rock Pi 4 Model C	Odroid-XU4
Dimensions (W x D x H)	88 x 58 x 19.5 mm	85 x 54 x 22 mm	82 x 58 x 22 mm
SRAM	4 GB	4 GB	2 GB
Clock Frequency	1.5 GHz	1.4 GHz	1.5 GHz
WiFi Connectivity	2.4/5.0 GHz	2.4/5.0 GHz	2.4/5.0 GHz
Cost	$62	$59	$58

While the Rock Pi 4 and Odroid-XU4 offered similar performance as the Raspberry Pi at a nearly identical price, the Raspberry Pi had the highest combination of processing speeds (clock frequency) and available SRAM. A marginally larger size and slighty higher cost were not significant enough factors to justify selecting another processor with slightly less performance capabilities. Another Raspberry Pi strength is its library of programming languages. For instance, Raspberry Pi's support includes many different languages, such as Scratch, Python, C++, and JavaScript. The programming language chosen is discussed in Section 2.3.

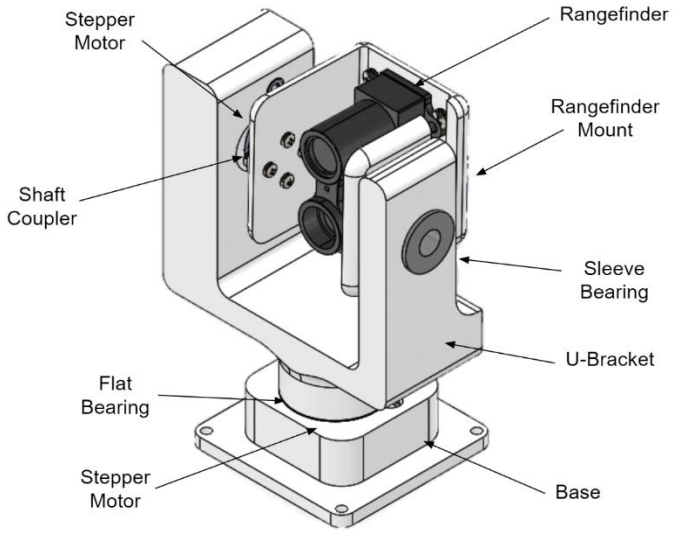

FIGURE 1: ASSEMBLY OF LIDAR HOUSING SHOWING COMPLETE SYSTEM

2.2 System Setup

Once the stepper motor model was decided, the decision was made to utilize previous custom 3-D printed components produced from a Stratasys Mojo 3-D printer (layer thickness:

0.018 cm). The components were designed specifically for the motors selected. Placed between the horizontal motor and the component containing the rangefinder is a needle-roller thrust bearing (McMaster-Carr model number: 5909K34). This allows for a U-Bracket to be equally supported on all sides. A multipurpose flanged sleeve bearing (McMaster-Carr model number: 7815K24) is used to support the end opposite from the motor shaft by being placed through the rangefinder mount and the U-Bracket. Both items help minimize any undesirable interference of vibrations or tilt without prohibiting full rotation range in both the vertical and horizontal directions. A detailed Computer Aided Design drawing of the assembly can be seen in Figure 1.

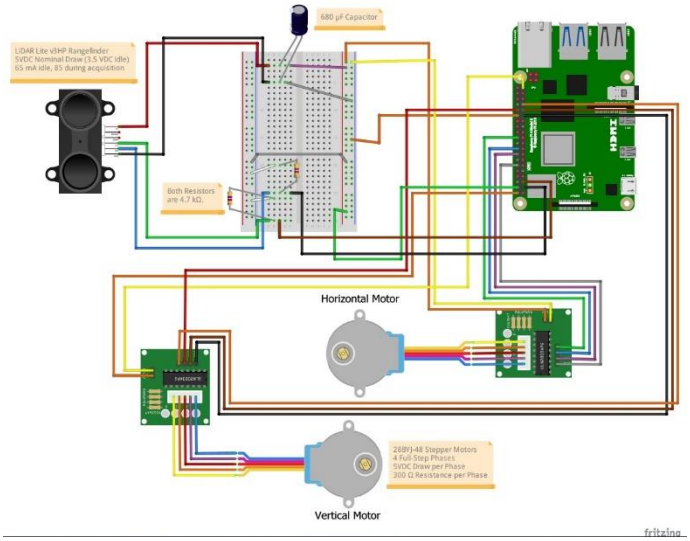

FIGURE 2: WIRING SCHEMATIC FOR LIDAR SYSTEM

The overall wiring diagram is shown in Figure 2. The rangefinder utilizes Serial Clock Line (SCL) and Serial Data Analyzer (SDA) wires. The SCL acts to synchronize all data transfers over an Inter-Integrated Circuit connection, whereas, the SDA is responsible for transfer of actual data. Both connections allow for communication between the rangefinder and the Raspberry Pi 4B's general purpose input/output pins. The SCL and SDA connections are paired with pull-up 4.7 kΩ resistors to restore the SCL and SDA signals to high when the Raspberry Pi is not transmitting a low signal. The pull-up resistors also ensure that the connections are given a well-defined voltage. The entire system is powered by an external battery source (SlimThin 10000 mAh) that is connected to the system with a Micro-Universal Serial Bus (USB) to USB cord.

2.3 Software

The C++ language was first selected because of the amount of control and precision it can provide. However, with no previous understanding of basic coding in this language, there was a struggle trying to edit and adjust code functionality. Ultimately, Python ended up being chosen for two primary reasons: close similarity to MATLAB that is taught in the authors' curriculum and ample online resources regarding the use of stepper motors. A basic knowledge of the format and syntax based on a MATLAB background proved to be beneficial as iterative testing of the code began.

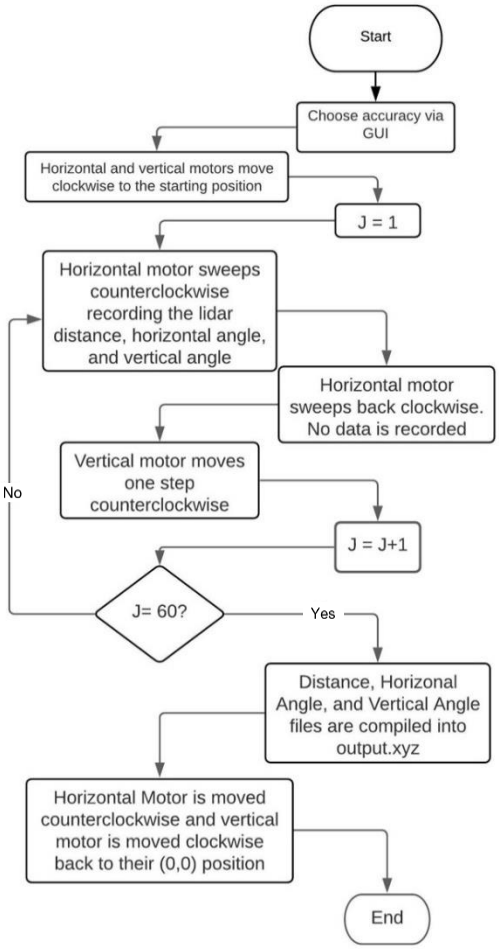

FIGURE 3: FLOWCHART OF CODE EMPLOYED IN THE LIDAR SYSTEM

Python 3 was used to run the two 28BYJ-48 stepper motors in conjunction with the Garmin LiDAR-Lite v3HP. At the start of the code, four .txt files are opened and cleared of any data that was previously recorded. The horizontal motor is then called to move a set angle clockwise (half of a sweep) and the vertical motor moves the LiDAR to a set angle counterclockwise (up). This is the starting position. Using nested 'for' loops, the system sweeps horizontally counterclockwise to a set angle, back horizontally clockwise, and then vertically down 0.71035 degrees. This loop will continue for as long as the initial 'for' loop indicates (`for j in range(_)`). After each motor step, the rangefinder records the distance of the object. A correction on the distance (i.e., distance*cosine(horizontal angle)*cosine(vertical angle)) is used to get an actual depth to account for the continual orientation change of the rangefinder lens. The code outputs four .xyz files: distance, horizontal angle,

vertical angle, and output. The output file combines the previous three files into a single .xyz file that is loaded into MATLAB to produce three images (see Section 2.5). A flowchart explaining the code's processes is shown in Figure 3.

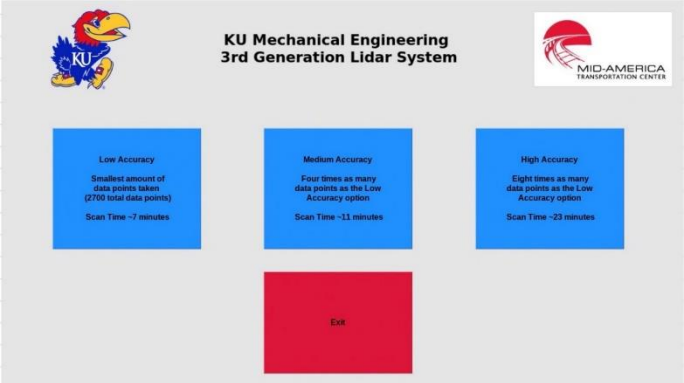

FIGURE 4: SCREENSHOT OF THE GRAPHICAL USER INTERFACE

2.4 Graphical User Interface

A GUI was created to provide user interaction with the system. The GUI was created in Python 3 on the Raspberry Pi to be used in conjunction with an official Raspberry Pi 7" Touch Screen Display. The GUI displays a visual menu with four interactable buttons: Low Accuracy, Medium Accuracy, High Accuracy, and Exit as seen in Figure 4. When a button is selected on the touchscreen, a command is sent to the Raspberry Pi 4 to perform the desired outcome. The three accuracy buttons (low, medium, and high) have embedded text to describe their intended purpose. Pressing each accuracy button will start the LiDAR system and run the code to allow for motor rotation and scanning to create a 3-D point cloud. These buttons from left to right increase the time it takes for a scan to complete while improving the accuracy of the 3-D point cloud. The accuracy is enhanced by increasing the number of horizontal data points taken per degree that creates a denser 3-D point cloud with less space in-between data points. It should be noted that the three options share the same number of vertical data points taken and only vary in the horizontal direction. The amount of vertical data points remains constant since the perceived slight enhancement in image quality on the vertical axis was deemed initially inconsequential to the significant increase in scan time that would occur. However, this can be revisited in the future if better vertical data resolution is needed.

The left button results in the lowest number of data points, while the middle button increases the number of horizontal data points scanned by four times, and the right button increases the number of horizontal data points scanned by eight times. These options produce 2700, 10800, and 21600 total data points, respectively, when the default values for degrees swept in the horizontal (32.16°) and vertical (42.19°) direction are used. The approximate time to complete a scan for the buttons from left to right are seven minutes, eleven minutes, and twenty-three minutes, respectively.

2.5 MATLAB Analysis Code

MATLAB code was employed to visualize the resulting data by plotting the points in a 3-D point cloud, 2-D point cloud, and 2-D contour plot. This code contains colored points based on the distance, creating a point cloud image. The code automatically generates three figures that show the point cloud from two different angles that can be rotated as desired and a third that shows a filled in 2-D contour plot of the data, making it easier to see edges. In the images provided in Section 3, the units for the X-, Y-, and Z-directions are cm, horizontal degrees, and vertical degrees, respectively. The LiDAR, GUI, and Matlab code can be found online at https://depcik.ku.edu/lidar.

2.6 Device Packaging

The system was designed to place all components in a centralized and compact location. All system components are located within the touchscreen case, or at the base of this case as seen in Figure 5. A SmartiPi Touch 2 was used as this touchscreen case that connects the Raspberry Pi 4B and Raspberry Pi 7" Touch Screen Display. This allows for the Raspberry Pi 4B to be mounted on the open backside of the case, and the touchscreen display internally mounted in the case with the display being accessible on the outside.

FIGURE 5: RASPBERRY PI 4B MOUNTED TO TOUCH SCREEN DISPLAY

The mount locations on the case allow for the Raspberry Pi to adequately connect to the touchscreen, as well as the other components attached at the base. This case has two hinges on the base that permit the angle of the touchscreen display to be changed to the user's desire. The solderable breadboard, motor driver boards, and LiDAR system are mounted on a custom 3-D printed part that can be seen in Figure 6 with the full system assembly illustrated in Figure 7. This part is connected to the base of the touchscreen case by two bolts located at the corners behind the touchscreen display. The bolt location utilizes four holes, one at each corner of the case, that come preinstalled on the base of the case so that no drilling is required to connect the custom part. In addition, the 3-D mounting plate provides holes for the breadboard, driver boards, and the LiDAR housing to be

secured, creating a single solid system. Overall, the purchased components cost for this system was $482.49.

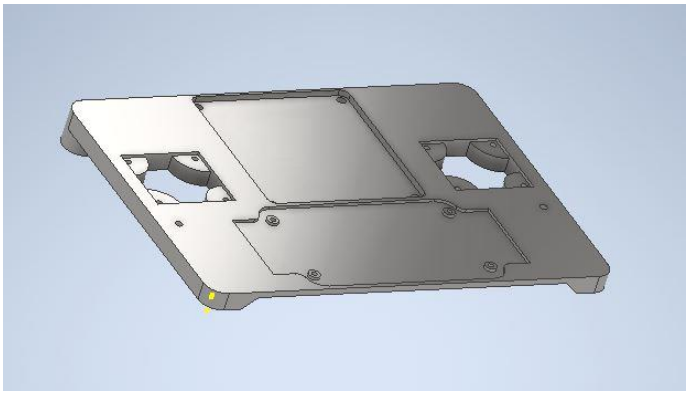

FIGURE 6: MOUNTING PLATE FOR SYSTEM COMPONENTS

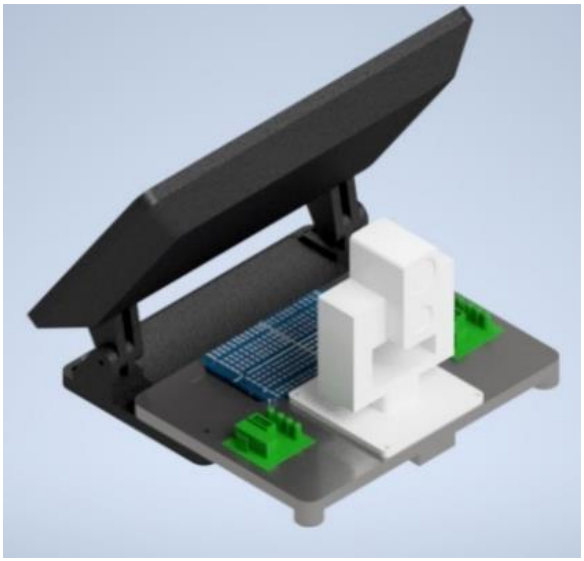

FIGURE 7: FULL SYSTEM ASSEMBLY INCLUDING TOUCH SCREEN, LIDAR HOUSING, AND SYSTEM COMPONENTS ON MOUNTING PLATE

3. RESULTS AND DISCUSSION

To run a complete test, connecting the power cable to the external battery turns on the Raspberry Pi and touchscreen. After selecting the desired accuracy, the Python code will then run. Once the system is finished scanning, the output .xyz file needs to be transferred to a different computer capable of running the MATLAB code. This can be done by either emailing the code from the Raspberry Pi using WIFI, or with a USB drive, if no internet is available. As indicated prior, the MATLAB code will produce three plots: a 3-D point cloud, 2-D point cloud, and 2-D contour image.

As an initial test of the system, a bag of sugar with noticeable crinkling was placed on a chair in front of a wall in Figure 8. Scanning the image took 7.24 minutes and a 3-D point cloud, 2-D point cloud, and 2-D contour plot can be seen in Figure 9, Figure 10, and Figure 11, respectively.

FIGURE 8: BAG OF SUGAR ON A CHAIR IN FRONT OF A WALL

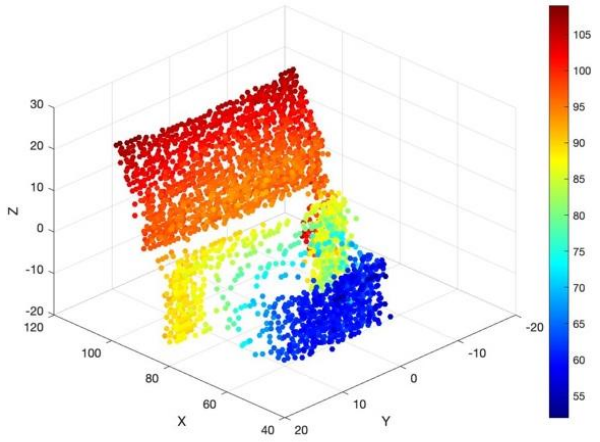

FIGURE 9: THREE-DIMENSIONAL POINT CLOUD OF THE BAG OF SUGAR IN FIGURE 8

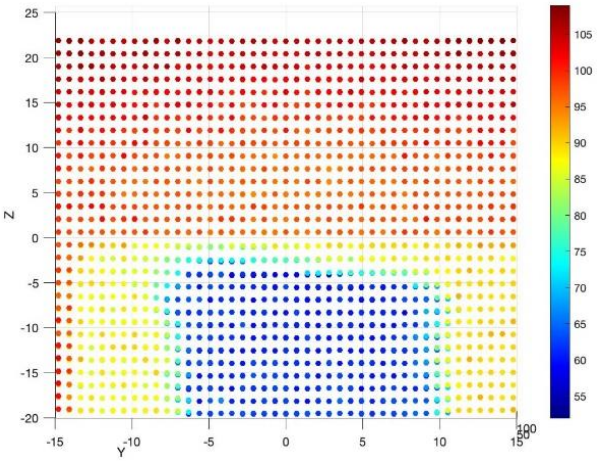

FIGURE 10: TWO-DIMENSIONAL POINT CLOUD OF THE BAG OF SUGAR IN FIGURE 8

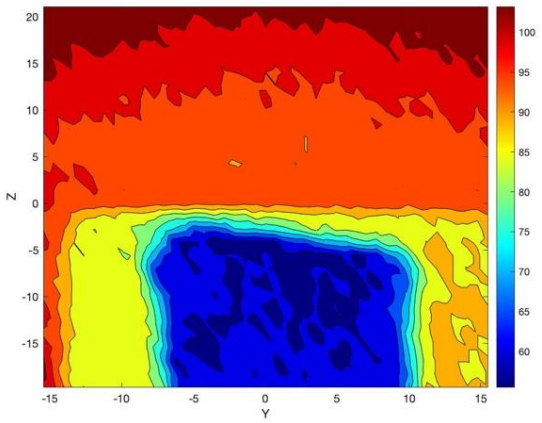

FIGURE 11: TWO-DIMENSIONAL CONTOUR PLOT OF THE BAG OF SUGAR IN FIGURE 8

While the outline of the sugar bag and chair can be easily seen, there were issues with the system detecting a wall in the background. The different red colors indicate dissimilar depth distances, which should not occur as the wall is a uniform distance away from the system. The 2-D point cloud also had too much space in between each point, limiting the resolution of each plot produced. Although the image was relatively good in a short amount of runtime, sacrificing time to achieve a more accurate point cloud became more ideal.

Since the rangefinder is rotating, it effectively moves away from the wall (as a function of its starting point) during data collection. Thus, to correct the depth issue, the distance measurement from the LiDAR was multiplied by the cosine of the horizontal angle as well as the cosine of the vertical angle. By taking both angle measurements into account, the correct depth was now recorded. An object with more defined edges and unique shapes was used for the final test. Figure 12 shows a shoe with uneven parts and edges on top of a box. Figures 13, 14, and 15 are the 3-D point cloud, 2-D point cloud, and 2-D contour plot created from the MATLAB code with the new data taken, respectively. The total scan time for this test was 7.26 minutes.

FIGURE 12: SHOE ON TOP OF BOX USED FOR TESTING

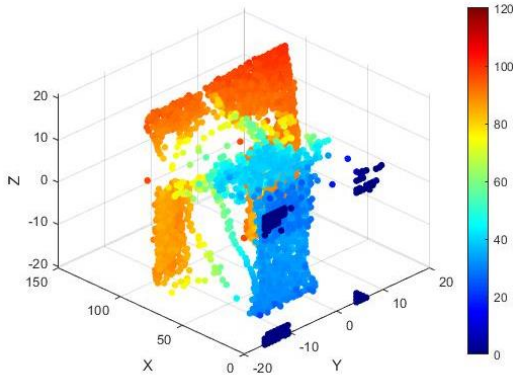

FIGURE 13: LOW ACCURACY 3-D POINT CLOUD

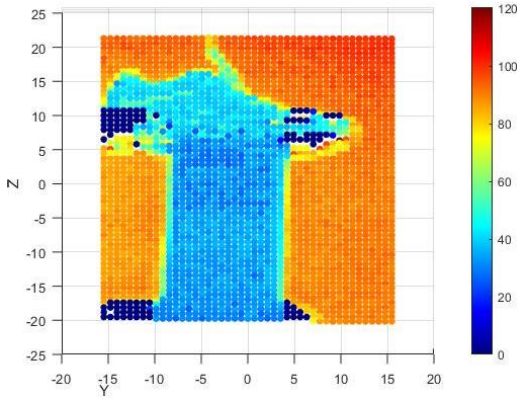

FIGURE 14: LOW ACCURACY 2-D POINT CLOUD

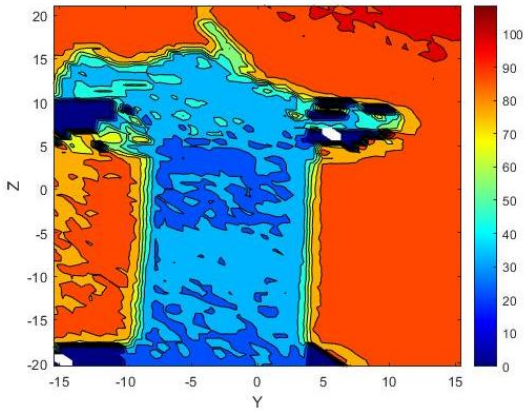

FIGURE 15: LOW ACCURACY 2-D CONTOUR PLOT

Overall, the outline of the shoe is evident along with a correct wall depth when looking at the 2-D images; however, the 3-D point cloud does not provide for a noticeable image. Thus, the next step was to increase the number of data points recorded to improve the accuracy. This was accomplished two different ways: full stepping and half stepping. By full stepping, data are recorded each full step of the motor whereas before, data was only recorded at the end of the entire motor step. This led to four times as many data points and a more accurate 3-D point cloud,

2-D point cloud, and 2-D contour plot of the same shoe and box setup in Figures 16, 17, and 18, respectively, with an overall run time of 11.27 minutes.

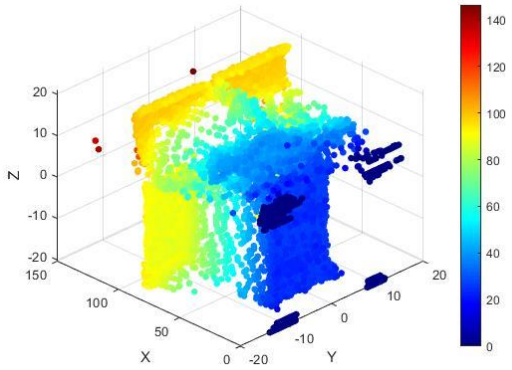

FIGURE 16: MEDIUM ACCURACY 3-D POINT CLOUD

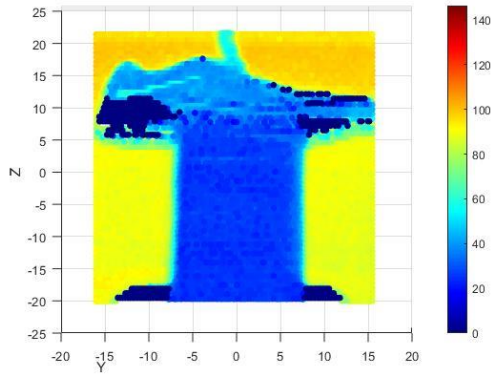

FIGURE 17: MEDIUM ACCURACY 2-D POINT CLOUD

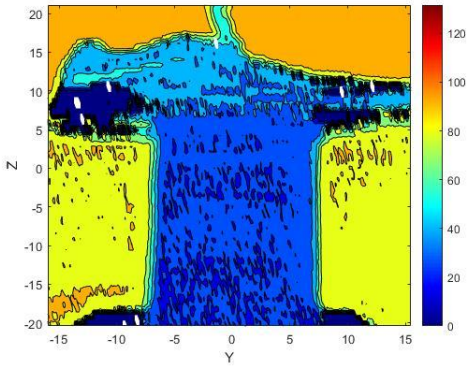

FIGURE 18: MEDIUM ACCURACY 2-D CONTOUR PLOT

This resulted in a more noticeable shoe in the 3-D point cloud (e.g., heel, facing, and toe cap) with the 2-D images illustrating more ridges along the shoe. To increase the number of data points again, half stepping was used where an additional step of data collection occurred in between the full steps. Half stepping produces eight times as many points as the low accuracy code and two times as many as full stepping. The same shoe on box setup was used and the 3-D point cloud, 2-D point cloud, and 2-D contour plot results are seen in Figures 19, 20, and 21, respectively. In all three figures, the outline of the shoe becomes more evident along with a distinct heel, sole, collar, facing, tongue, and toe cap. Interestingly, the logo of the shoe becomes more noticeable as it starts to reflect from the laser. The run time for this experiment was clocked at 22.58 minutes.

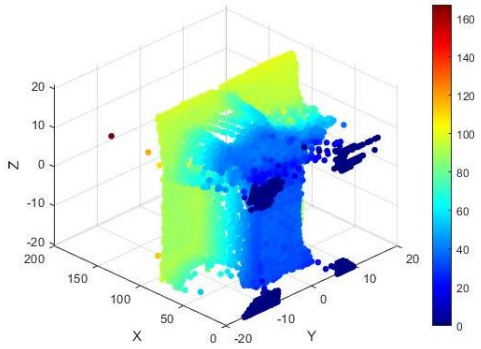

FIGURE 19: HIGH ACCURACY 3-D POINT CLOUD

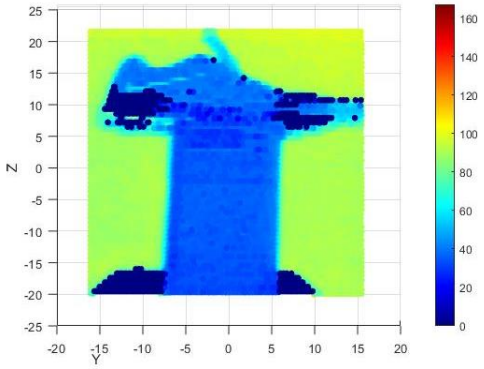

FIGURE 20: HIGH ACCURACY 2-D POINT CLOUD

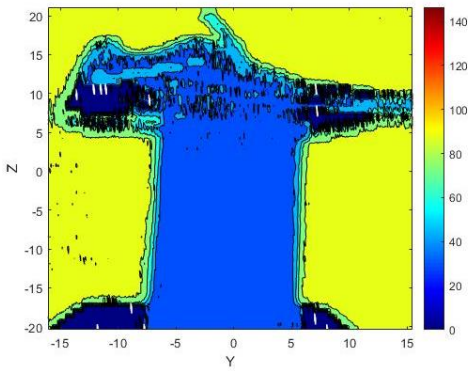

FIGURE 21: HIGH ACCURACY 2-D CONTOUR PLOT

3.1 Recommendations

The LiDAR system created can efficiently obtain reasonably accurate point clouds of objects that are relatively close to the rangefinder. Testing illustrated that the farther the object, the more distorted the point cloud became. Further investigation is needed into the six different rangefinder

configurations and their accuracy as a function of object distance. The choice of rangefinder configuration can be added as an option to the GUI. Additional GUI options, such as selection of the horizontal and vertical degrees swept, can be added to provide the user with more control over the LiDAR system.

Replacing the 28BYJ-48 stepper motors with motors that can move in smaller angle increments (i.e., microstepping) will allow the system to record more data points, providing a higher accuracy point cloud. Another option is to add a specific motor driver (e.g., Easy Driver Stepper Motor Driver [27]) that can accomplish this microstepping in Python [28]. Here, the driver must be synchronized to be of unipolar type to work with the unipolar 28BYJ-48 motor. Another option is to convert this motor to be bipolar to work with bipolar motor drivers, which seems readily possible [29]. Due to a recording issue with the clockwise sweep using the horizontal motor or due to possible gear backlash as the motor direction reversed, data was only able to be accurately recorded during counterclockwise sweeps. A solution to this issue will cut the run time of the system in half. Another method to improve image accuracy is to sweep the clockwise and counterclockwise direction with the horizonal motor at different increments while at the same vertical location. Doing so would double the number of points taken, creating a higher-density point cloud. A third methodology to enhance image accuracy is to post-process the images using upsampling. This could provide more accurate images without a need to increase LiDAR system run time. Finally, a comparison of the efficiency and speed to other commercial products mentioned in the introduction would help to justify (positively or negatively) the cost-effectiveness of the created system.

4. CONCLUSIONS

As technology advances in the automotive sector, the infrastructure has slowly deteriorated. Although there are commercial systems monitoring roadway conditions, the development of a low-cost LiDAR system could increase the percentage of roadway defects found. This would enhance driver safety, lower economic losses, and potentially save lives. This effort endeavored to create such a system using commercially available hardware (i.e., rangefinder, stepper motors, and microcontroller). Overall, three different levels of measuring accuracy were generated, able to produce point clouds of 2700, 10800, and 21600 points, resulting in images of low, medium, and high accuracy, respectively. The scan times are relatively short, with the most accurate at slightly over 23 minutes, the medium at approximately 11 minutes, and the low accuracy at a little over 7 minutes. In addition to being able to capture images quickly and efficiently, the system is also cheaper than many alternatives with a total final cost of less than $500. The system also remains respectively simple to use as the additions of a GUI, clean packaging, and an external battery allow users to operate the system at any location. The combination of all these factors results in an ideal starting place for others to enhance the output and create an inexpensive system that could be widely utilized. Subsequently deployment within the transportation sector will aid in the recognition of low-quality or hazardous roadways and surfaces helping to create a safer environment.

ACKNOWLEDGEMENTS

The research described in this paper is funded, in part, by the Mid-America Transportation Center via a grant from the U.S. Department of Transportation's University Transportation Centers Program, and this support is gratefully acknowledged. The contents reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein, and are not necessarily representative of the sponsoring agencies. The authors would like to acknowledge Afnan Elhamshari, Zachary Hall, Justin Lewis, and Brandon Ly for their efforts in fabricating the second prototype that served as a base for this effort.

REFERENCES

[1] Zaloshnja, E., and Miller, T. R., 2009, "Cost of Crashes Related to Road Conditions, United States, 2006," Ann Adv Automot Med, **53**, pp. 141-153.

[2] Insurance Journal, 2016, "Study: Pothole Damage Costs U.S. Drivers $3B a Year," https://www.insurancejournal.com/magazines/mag-features/2016/03/21/401900.htm.

[3] U.S. Department of Transportation, 2020, "Automatic Emergency Braking," https://www.safercar.gov/Vehicle+Shoppers/Safety+Technology/aeb-1/.

[4] National Ocean Service, 2021, "What is Lidar?," https://oceanservice.noaa.gov/facts/lidar.html.

[5] Abshire, J. B., Riris, H., Allan, G. R., Weaver, C. J., Mao, J., Sun, X., Hasselbrack, W. E., Kawa, S. R., and Biraud, S., 2010, "Pulsed Airborne Lidar Measurements of Atmospheric CO_2 Column Absorption," Tellus B: Chemical and Physical Meteorology, **62**(5), pp. 770-783. doi: 10.1111/j.1600-0889.2010.00502.x.

[6] Kelly, M., and Di Tommaso, S., 2015, "Mapping Forests with Lidar Provides Flexible, Accurate Data with Many Uses," California Agriculture, **69**(1), pp. 14-20. doi: 10.3733/ca.v069n01p14.

[7] Qiu, F., Sridharan, H., and Chun, Y., 2010, "Spatial Autoregressive Model for Population Estimation at the Census Block Level Using LIDAR-derived Building Volume Information," Cartography and Geographic Information Science, **37**(3), pp. 239-257. doi: 10.1559/152304010792194949.

[8] Moras, J., Rodríguez, F. S. A., Drevelle, V., Dherbomez, G., Cherfaoui, V., and Bonnifait, P., "Drivable Space Characterization using Automotive Lidar and Georeferenced Map Information," Proc. 2012 IEEE Intelligent Vehicles Symposium, pp. 778-783. doi: 10.1109/IVS.2012.6232252.

[9] Habib, A., Lin, Y.-J., Ravi, R., Shamseldin, T., and Elbahnasawy, M., 2018, "LiDAR-Based Mobile Mapping System for Lane Width Estimation in Work Zones." doi: 10.5703/1288284316730.

[10] Olsen, M. J., Butcher, S., and Silvia, E., 2012, "Real-Time Change and Damage Detection of Landslides and Tech Report Earth Movements Threatening Public Infrastructure." https://rosap.ntl.bts.gov/view/dot/24404.
[11] Shan, Y., Bangaru, S. S., Li, J. Q., and Yang, X., 2017, "Assessing the Risk of Landslide on Oklahoma Highways Using LIDAR." https://rosap.ntl.bts.gov/view/dot/36330.
[12] Gong, J., 2016, "Mobile Hybrid LiDAR & Infrared Sensing for Natural Gas Pipeline Monitoring." https://rosap.ntl.bts.gov/view/dot/32121.
[13] Goehl, D., Gurganus, C., Crockford, W., and Ravipati, D., 2018, "Planning the Next Generation of Seal Coat Equipment: Guidelines and Implementation." https://rosap.ntl.bts.gov/view/dot/36954.
[14] Toran, L., Eisenman, S., Caplan, J., van Aken, B., McKenzie, E., Nyquist, J. E., Ryan, R., Kelley, J., Traver, R., Tu, M.-C., Schmidt, N., and Calt, E., 2017, "Storm Water Control Management & Monitoring." https://rosap.ntl.bts.gov/view/dot/35094.
[15] Omar, T., 2016, "The Federal Railroad Administration's LiDAR-Based Automated Grade Crossing Survey System: Research Results." https://rosap.ntl.bts.gov/view/dot/37038.
[16] Ceylan, H., Gopalakrishnan, K., Kim, S., Taylor, P., Alhasan, A., and Yang, S., 2016, "Impact of Curling and Warping on Concrete Pavement." https://rosap.ntl.bts.gov/view/dot/36924.
[17] Souleyrette, R. R., Wang, T., Lau, D., Peng, X., Aboubakr, A., and Randerson, E., 2015, "3D Methodology for Evaluating Rail Crossing Roughness: Vehicle Dynamic Modeling." https://rosap.ntl.bts.gov/view/dot/30862.
[18] Tarko, A. P., Ariyur, K. B., Romero, M. A., Bandaru, V. K., and Liu, C., 2014, "Stationary LiDAR for Traffic and Safety Applications – Vehicles Interpretation and Tracking." https://rosap.ntl.bts.gov/view/dot/36884.
[19] Iftekharuddin, K. M., Elbakary, M., Afrifa, K., Cetin, M., Rakha, H., and Abdelghaffar, H., 2017, "LiDAR for Air Quality Measurement." https://rosap.ntl.bts.gov/view/dot/36625.
[20] Zhang, S., Baros, S., and Bogus, S., 2019, "Karst Sinkhole Detecting and Mapping Using Airborne LiDAR," https://digitalcommons.lsu.edu/transet_pubs/40
[21] Rister, B., 2019, "Using LIDAR for Bridge Clearance Heights," https://rip.trb.org/View/1524210.
[22] Asborno, M. I., Burris, C. G., and Hernandez, S., 2019, "Truck Body-Type Classification using Single-Beam Lidar Sensors," Transportation Research Record, **2673**(1), pp. 26-40. doi: 10.1177/0361198118821847.
[23] Blankenau, I., Zolotor, D., Choate, M., Jorns, A., Homann, Q., and Depcik, C., 2018, "Development of a Low-Cost LIDAR System for Bicycles," SAE Technical Paper 2018-01-1051. doi: 10.4271/2018-01-1051.
[24] McIntosh, D., 2020, "Utilization of Lidar Technology — When to Use It and Why," https://trid.trb.org/view/1638641.
[25] Lienert, P., and Nellis, S., 2019, "Cheaper Sensors Could Speed More Self-Driving Cars to Market by 2022," https://www.reuters.com/article/us-autos-autonomous-lidar/cheaper-sensors-could-speed-more-self-driving-cars-to-market-by-2022-idUSKCN1TD2MY.
[26] Wiklund, T., Heim, M., Halberstadt, J., Duncan, M., Mittman, D., DeAgostino, T., and Depcik, C., 2019, "Design and Development of a Cost-Effective LIDAR System for Transportation," ASME 2019 International Mechanical Engineering Congress and Exposition, Salt Lake City, UT. doi: 10.1115/IMECE2019-11279.
[27] Schmalz, B., 2021, "Easy Driver Stepper Motor Driver," https://www.schmalzhaus.com/EasyDriver/.
[28] pyeasydriver, 2021, "Python Driver for Stepper Motors Controlled with the SparkFun EasyDriver Motor Controller Board." https://github.com/ECE-492-W2020-Group-6/pyeasydriver.
[29] Benchoff, B., 2014, "Changing Unipolar Steppers to Bipolar," https://hackaday.com/2014/07/29/changing-unipolar-steppers-to-bipolar/.