Motivating Computing and Engineering Undergraduate Research via Basic Utility Vehicle Dashboard Display

Bingrui Xiao[1] xiaob90@uwosh.edu

Polina Ibragimova[1] ibragp05@uwosh.edu

Kuan-Fu Chen[1] chenk75@uwosh.edu

Warren S. Vaz[2]* vazw@uwosh.edu

[1]Department of Computer Science, [2]Department of Engineering Technology
University of Wisconsin-Oshkosh, Fox Cities Campus
Menasha, Wisconsin, 54952
*corresponding author

Abstract

Undergraduate students at the University of Wisconsin-Oshkosh, Fox Cities Campus designed and built an all-electric basic utility vehicle (BUV). The vehicle was modeled after an electric golf cart. Some background on how the BUV project fits into the current landscape is provided, as well as the motivation in building it at the Fox campus. As part of a collaborative effort, students from the computer science club designed a dashboard-style display for the BUV. Presented are the complete design, acquisition, and assembly of the necessary display components. The creation of the display is described with all the technical challenges and solutions. The display is designed to connect with the motor controller and display information received. Modules use programming languages that are typically taught in introductory computer science courses (e.g. C/C++, Python, and Java). The application is created on a Windows 2-in-1 laptop and designed to automatically show collection of data. It was tested and found to work well. Challenges involved matching the speed at which data was updated as well as how the data was collected and stored between the software of the controller and the tablet. Future work will involve, in part, expanding the capabilities of the BUV and its onboard display. The presented project has the potential to be adapted to offer students flexible, enjoyable, and creative methods to collect real data to be used for a variety of research projects.

Keywords: basic utility vehicle, dashboard display, undergraduate research, C/C++ Java Python

1 Introduction

Climate change is at the top of the list of challenges facing humanity today, caused in large part from the burning of fossil fuels [1]. Electric vehicles are seen as the future of clean transportation. Thus, building an electric basic utility vehicle (BUV) is the perfect way to provide undergraduate engineering and technology students with a practical, cutting-edge project that also illustrates and teaches several pertinent scientific and engineering principles. There are several areas in the world that do not have the expertise or resources to build their own vehicles. Further, these areas have limited and only one or two of a specific type of energy resource (only diesel or only natural gas) and very little infrastructure. The BUV concept [2] was created for university campuses and other such institutions or organizations to partner with underdeveloped communities to help with their transportation needs. Transportation has been identified has one of the critical factors in helping people to escape poverty [3]. Thus, the idea is: an institution would work with an underdeveloped community to build a vehicle based on the local factors. For example, the proposed BUV would be suited to a place with poor roads or for off-road work, limited electricity but no petroleum-based fuels, and relatively modest income. In lieu of a real-world customer, the campus facilities and grounds staff would use the BUV for performing lawn care and other maintenance tasks. Building the vehicle instead of purchasing one allows for flexibility in design and room for future expansion. All this is performed in a team setting and part of a larger micro-grid research project that includes a 10-kW solar panel array and an atmospheric testing drone [4].

With the current pace of technology, the demand for skilled workers is only going to increase. Thus, it is critical to increase the number of students in areas such as computer science, engineering, and technology. Promoting and motivating students to excel in such fields as early as possible plays a big role in meeting the community's expectation and future job market needs. This can be accomplished by exposing students to stimulating projects that prepare students in enjoyable and effective ways [6]. The current paper focuses on a smaller portion of the overall project: creating a dashboard-style display for the BUV. The purpose of building and assembling a display for the BUV is to gain knowledge of how coding works, how to apply it to solve a real-world problem, and providing the BUV project with future design flexibly afforded by a native solution rather than a readymade one. The present work introduces the overall BUV project, discusses the design approach for the dashboard display, details the process by which the software was developed and integrated into the hardware, and presents some lessons learned.

2 Design, assembly, and performance

Figure 1 shows the completed BUV. It can seat two people, has a pick-up bed, and 4.8-kWh of available battery capacity. It is powered by a 10-hp electric motor controller by a 500-amp regenerative motor controller. This vehicle serves as a research and teaching testbed and has also been used in outreach and public education. It is intended to be used by the groundskeeping staff at the campus mainly as a means of transportation for these persons and their tools. While several components such as the electric motor, batteries, controller, etc. were purchased, several others were fabricated from stock metal and Unistrut. The frame, steering columns, pick-up bed, seat, etc. were designed and built in a modular fashion to afford design flexibility and versatility in the future. For example, the bed can easily be swapped for a second row of seats.

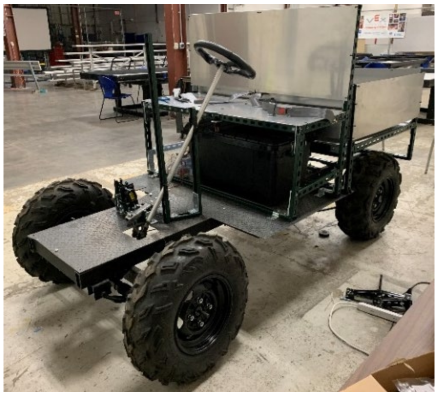


Fig. 1: Assembled basic utility vehicle.

2.1 Dashboard display design considerations

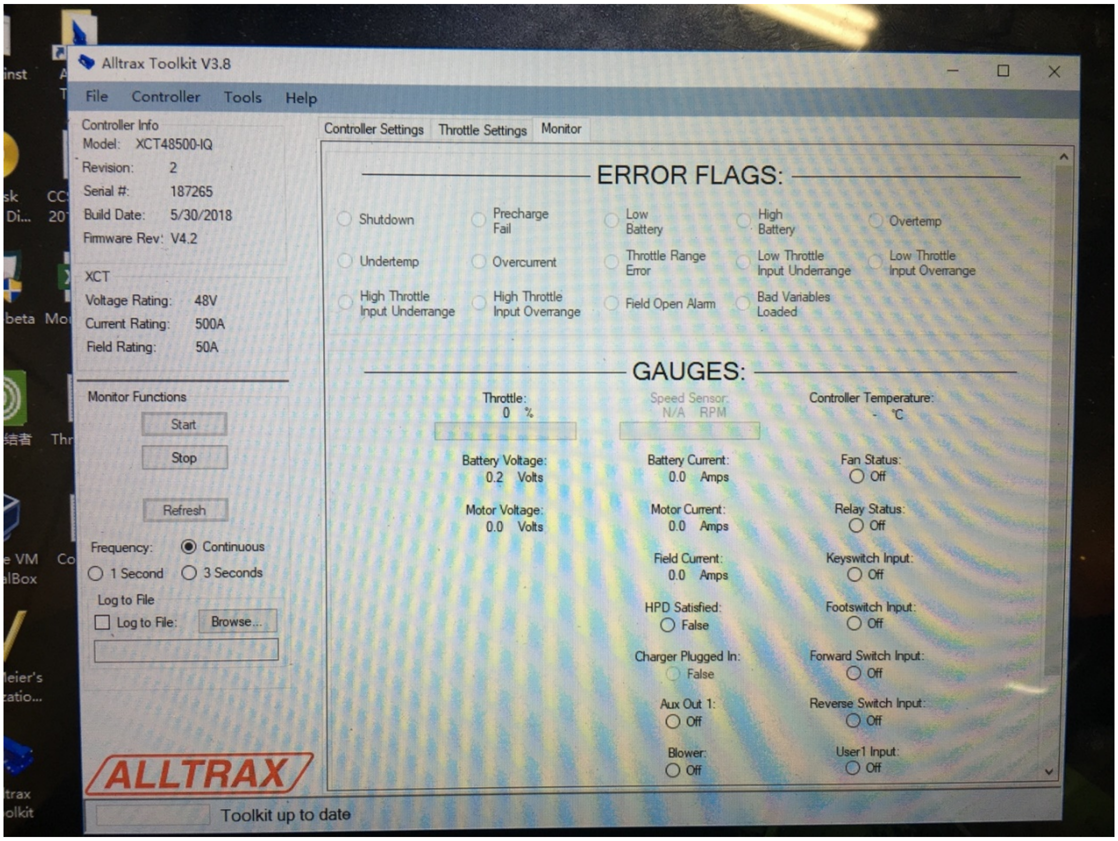
This section explains the design and building aspects of the display and some of the considerations. The section gives a rough idea about some of the key components as well.

The dashboard-style display is a critical part of the BUV as it is the main method to keep the driver informed of the vehicle status: speed, battery state of charge, etc. In order for the students to practice STEM skills and make this project work, there are two ways to display vehicle information. One is to buy different kinds of sensors and collect the data for display. Raspberry Pi would be used for this option. The display would show the speed, temperature, motor speed (RPM), battery capacity or charge level (state of charge), distance traveled, etc. Too many sensors and wires are required with this option, and it has no much educational significance for students since it is only needed to connect wires and produce simple coding. In this case, this option was rejected for the project. The other option is connecting the display with the motor controller that is used to control the current and signal to the electric motor. Based on the affordable technologies and devices that are used for building BUVs, the easiest way to create an accurate dashboard is to connect it directly with an the Alltrax controller. The students then focused on the necessary programming and problem-solving to realize this project.

The Alltrax XCT-48500 IQ shunt controller is the one connected with the display. This controller is Alltrax's newest line of regenerative, separately excited motor controllers. It offers many advantages over the older DCX line. The controller has a USB 2.0 Programming Port to make programming the Alltrax controller easier and it interfaces with a computer via the Toolkit program. It supports connections via Windows XP to Windows 10 and has a simple user interface. The software uses a standard USB printer cable to connect to the XCT controller, which allows simple changes to configure the controller to a specific situation, and monitors output with a laptop and saves the data to hard drive. Once the Alltrax Toolkit connects with the electric motor, the motor information (current, speed, etc.) can only be read to the Alltrax Toolkit software. Figure 2 shows the software in operation. As shown in the figure, the controller has all information that is needed to build the display software, only the formatting needs work.

Accommodating the student's experience and training, designing a car dashboard from scratch was a big challenge. For the current situation, especially low cost but providing a challenge for students, a Raspberry Pi would be the best choice. This is further supported by the versatility of the Pi unit [5]. Students at Fox have experience working with it via workshops and club work. Since Alltrax Toolkit has all the information that was needed and it only works on a Windows operating system, Windows was loaded on a Pi and the software was installed. The Pi proved to be unable to handle Windows in a practical manner; it would take over five minutes to start displaying information, far too slow for a vehicle display. The idea was scrapped.

The students tried to obtain the application programming interface (API) directly from Alltrax, including libraries, frameworks, toolkits, and software development kits. Organizations are increasingly providing their internal data on the web through public APIs [6]. APIs could allow one to bypass the Windows operating system; one can get data directly from the motor controller. The controller and Toolkit software are designed by the Alltrax company, which owns the intellectual property rights for their invention. For this reason, it was not possible to obtain the API. Thus, running Alltrax Toolkit on a Windows operating system was the only option for taking data from the motor controller and having a display.



Controller Info
Model: XCT48500-IQ
Revision: 2
Serial #: 187265
Build Date: 5/30/2018
Firmware Rev: V4.2

XCT
Voltage Rating: 48V
Current Rating: 500A
Field Rating: 50A

Monitor Functions
Frequency: ● Continuous
○ 1 Second ○ 3 Seconds
Log to File
☐ Log to File:

ERROR FLAGS:
○ Shutdown ○ Precharge Fail ○ Low Battery ○ High Battery ○ Overtemp
○ Undertemp ○ Overcurrent ○ Throttle Range Error ○ Low Throttle Input Underrange ○ Low Throttle Input Overrange
○ High Throttle Input Underrange ○ High Throttle Input Overrange ○ Field Open Alarm ○ Bad Variables Loaded

GAUGES:
Throttle: 0 %
Speed Sensor: N/A RPM
Controller Temperature: - °C
Battery Voltage: 0.2 Volts
Battery Current: 0.0 Amps
Fan Status: ○ Off
Motor Voltage: 0.0 Volts
Motor Current: 0.0 Amps
Relay Status: ○ Off
Field Current: 0.0 Amps
Keyswitch Input: ○ Off
HPD Satisfied: ○ False
Footswitch Input: ○ Off
Charger Plugged In: ○ False
Forward Switch Input: ○ Off
Aux Out 1: ○ Off
Reverse Switch Input: ○ Off
Blower: ○ Off
User1 Input: ○ Off

Toolkit up to date

Fig. 2: Alltrax Toolkit motor controller software.

2.2 Final decision

For a mobile (vehicle) application, it was decided to use a tablet as the hardware. The engineering team provided a list of desired outputs to be displayed along with a priority list. Running Toolkit on the tablet would allow data to be obtained from the motor controller and another control application would collect and display the data in a suitable format. A lot of memory is required for this, so that is another reason why the Pi was not used. Due to university contract restrictions, a Dell Latitude 3190 2-in-1 was purchased. The laptop obtains an Intel® Pentium™ N5000 Quad Core Processor, 8 GB memory, 128 GB Solid-State Hard Drive, Wired and Wireless Network Connectivity (including Bluetooth), and built-in keyboard.

3 Hardware design

The laptop must be mounted on the vehicle close to the steering wheel in order to replicate the experience that a normal vehicle would provide. The laptop is flipped backward to be used as a monitor. A Unistrut structure is used to support the steering column and this was also used to support the display as well. A custom plastic case was designed in SolidWorks and 3D-printed to house the display and protect it. The model is shown in Figure 3. The display was powered by connecting to the main vehicle 48-V battery bank via an adapter (110 V). It connects to the motor controller via a USB-A to USB-B cable. All the cables were safely mounted behind the steering support.

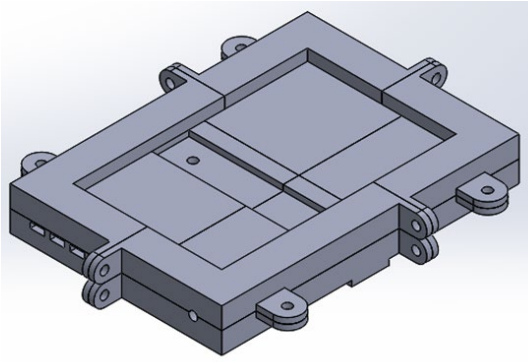


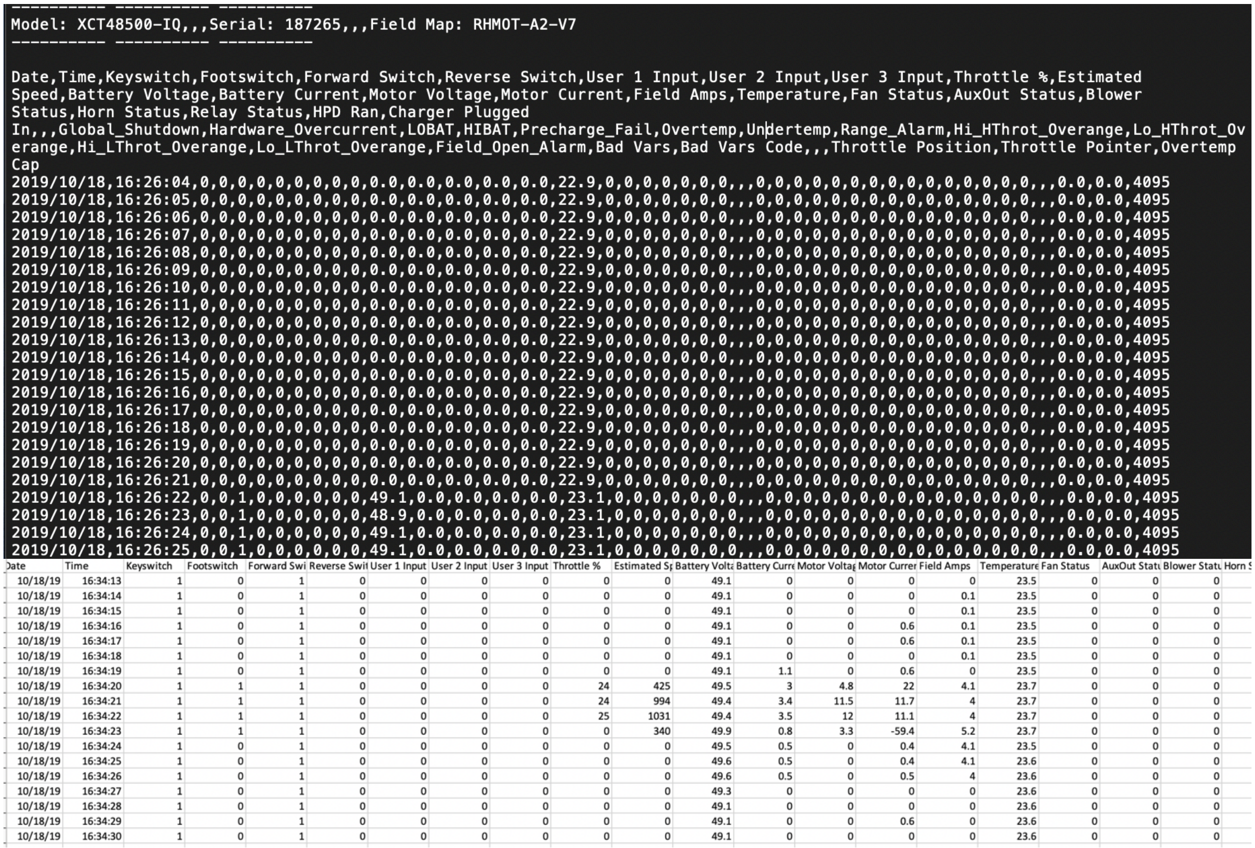
Fig. 3: Custom 3D-printed case for dashboard display.

4 Software design

The club members designed an application that took the data from the Alltrax controller that had all the data that was needed. C# and the Windows Forms library were used for front-end development. Furthermore, several members of the team had some experience in developing graphical user interfaces (GUI) in the form of applications. The first challenge was accessing the data directly from the application. Alltrax did not provide any solution and in the absence of source code, the data was saved to a text file. The team had to learn this from scratch.

It was easy to develop the basic layout of the GUI. After Excel files proved to be too slow, text files were used. String arrays with a capacity of 48 elements were used. Raw data was separated by commas, so the process was relatively straightforward. The file was read every quarter of a second and the values represented were labeled. The raw data and processed Excel file are shown in Figure 4. However, there was another problem. A virtual button on the Toolkit GUI had to be manually pressed every time new data was needed. To automate the process, the control program accessed the data via a button press every 250 ms (a timer was used). This included the creation of a new text file and so on. The button was programmed to be pressed automatically and was hidden from the user to avoid confusion. Thus, a fresh set of results was obtained from Toolkit every quarter second.

There was a lot of data that the controller provided and only the most critical was displayed: BUV speed (converted from rotational speed using electric motor rotational speed, rear differential ratio which was 12.1:1, and tire radius), battery state of charge (subtracting the current consumed by the controller from the fully charged state, 100%), power consumption (current multiplied by voltage), energy usage (power consumption integrated over time), and distance traveled (BUV speed integrated over time). Data such as the rotational speed and temperature were not displayed. To make the application more user friendly, Math.Round method to round up the decimals was used.



```
---------- ---------- ----------
Model: XCT48500-IQ,,,Serial: 187265,,,Field Map: RHMOT-A2-V7
---------- ---------- ----------

Date,Time,Keyswitch,Footswitch,Forward Switch,Reverse Switch,User 1 Input,User 2 Input,User 3 Input,Throttle %,Estimated
Speed,Battery Voltage,Battery Current,Motor Voltage,Motor Current,Field Amps,Temperature,Fan Status,AuxOut Status,Blower
Status,Horn Status,Relay Status,HPD Ran,Charger Plugged
In,,,Global_Shutdown,Hardware_Overcurrent,LOBAT,HIBAT,Precharge_Fail,Overtemp,Undertemp,Range_Alarm,Hi_HThrot_Overange,Lo_HThrot_Ov
erange,Hi_LThrot_Overange,Lo_LThrot_Overange,Field_Open_Alarm,Bad Vars,Bad Vars Code,,,Throttle Position,Throttle Pointer,Overtemp
Cap
2019/10/18,16:26:04,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:05,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:06,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:07,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:08,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:09,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:10,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:11,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:12,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:13,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:14,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:15,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:16,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:17,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:18,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:19,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:20,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:21,0,0,0,0,0,0,0,0,0,0.0,0.0,0.0,0.0,0.0,22.9,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:22,0,0,1,0,0,0,0,0,0,49.1,0.0,0.0,0.0,0.0,23.1,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:23,0,0,1,0,0,0,0,0,0,48.9,0.0,0.0,0.0,0.0,23.1,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:24,0,0,1,0,0,0,0,0,0,49.1,0.0,0.0,0.0,0.0,23.1,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
2019/10/18,16:26:25,0,0,1,0,0,0,0,0,0,49.1,0.0,0.0,0.0,0.0,23.1,0,0,0,0,0,0,0,,,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,,,0.0,0.0,4095
```

Date	Time	Keyswitch	Footswitch	Forward Swi	Reverse Swi	User 1 Input	User 2 Input	User 3 Input	Throttle %	Estimated Sp	Battery Volta	Battery Curre	Motor Voltag	Motor Curren	Field Amps	Temperature	Fan Status	AuxOut Statu	Blower Statu	Horn S
10/18/19	16:34:13	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0	23.5	0	0	0	
10/18/19	16:34:14	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0.1	23.5	0	0	0	
10/18/19	16:34:15	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0.1	23.5	0	0	0	
10/18/19	16:34:16	1	0	1	0	0	0	0	0	0	49.1	0	0	0.6	0.1	23.5	0	0	0	
10/18/19	16:34:17	1	0	1	0	0	0	0	0	0	49.1	0	0	0.6	0.1	23.5	0	0	0	
10/18/19	16:34:18	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0.1	23.5	0	0	0	
10/18/19	16:34:19	1	0	1	0	0	0	0	0	0	49.1	1.1	0	0.6	0	23.5	0	0	0	
10/18/19	16:34:20	1	1	1	0	0	0	0	24	425	49.5	3	4.8	22	4.1	23.7	0	0	0	
10/18/19	16:34:21	1	1	1	0	0	0	0	24	994	49.4	3.4	11.5	11.7	4	23.7	0	0	0	
10/18/19	16:34:22	1	1	1	0	0	0	0	25	1031	49.4	3.5	12	11.1	4	23.7	0	0	0	
10/18/19	16:34:23	1	1	1	0	0	0	0	0	340	49.9	0.8	3.3	-59.4	5.2	23.7	0	0	0	
10/18/19	16:34:24	1	0	1	0	0	0	0	0	0	49.5	0.5	0	0.4	4.1	23.5	0	0	0	
10/18/19	16:34:25	1	0	1	0	0	0	0	0	0	49.6	0.5	0	0.4	4.1	23.6	0	0	0	
10/18/19	16:34:26	1	0	1	0	0	0	0	0	0	49.6	0.5	0	0.5	4	23.6	0	0	0	
10/18/19	16:34:27	1	0	1	0	0	0	0	0	0	49.3	0	0	0	0	23.6	0	0	0	
10/18/19	16:34:28	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0	23.6	0	0	0	
10/18/19	16:34:29	1	0	1	0	0	0	0	0	0	49.1	0	0	0.6	0	23.6	0	0	0	
10/18/19	16:34:30	1	0	1	0	0	0	0	0	0	49.1	0	0	0	0	23.6	0	0	0	

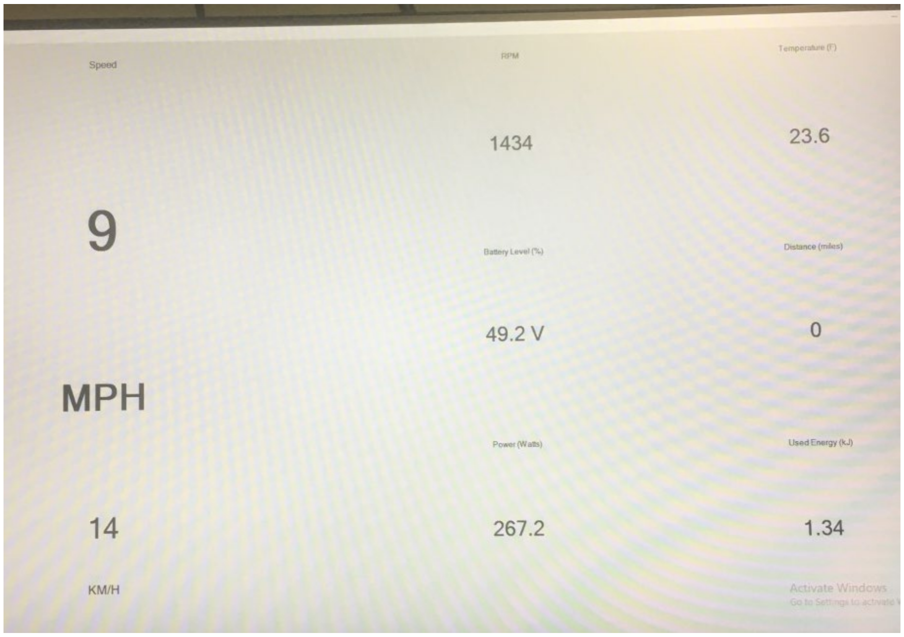
Fig. 4: Data from Alltrax Toolkit in a text file and an Excel file.

The display was operational, but the program did not have any capacity to store data between runs. As such, every time the user shut the display down, it would have to be manually restarted and all the variables were reset. This was not an issue for most of the data. However, for the battery state of charge and the distance travelled, one would prefer to know what the values are across multiple trips. The battery is not typically at full charge at the start of each trip and the vehicle does not start from a “zero” distance. The team experimented with a global variable, but finally a FormClosing event was used to save distance and energy usage into a separate text file when the application was closed. The application then referred to these values when it was restarted.

After running the application connected to the BUV, adjustments to the display font size and layout were made. Debugging of the software was done, especially the hidden button press.

5 Performance and Design Modification

The prototype display was connected and can be seen in Figure 5. The updating frequency of 4 Hz or once every 250 ms worked well. The data was expected to be more accurate than using external sensors as this was pulled directly from the motor controller. More tests will be conducted when electrical and mechanical testing of the BUV is done in the future.


Speed

9

MPH

14

KM/H

RPM

1434

Battery Level (%)

49.2 V

Power (Watts)

267.2

Temperature (F)

23.6

Distance (miles)

0

Used Energy (kJ)

1.34

Fig. 5: Prototype BUV display.

6 Closing Remarks

This paper details the design and development of a dashboard-style display for the BUV being built at the University of Wisconsin-Oshkosh, Fox Cities Campus. The design decisions were outlined along with some challenges and solutions. The display was successfully completed and worked well on the BUV, pulling data directly from the motor controller and displaying it after storing and processing it via the 2-in-1 laptop on board. Since the data is also recorded, it can be later retrieved and used for other studies on the BUV performance and future modification.

This project exposed freshmen and sophomore engineering and computer science students to a variety of disciplines and challenges as part of a fun, stimulating exercise with real-world applications. It is the type of project that has an impact on the future of the students and can be expected to continue to grow out of this BUV project in the future.

7 Future Work

The students will continue to refine and improve the display, including the type of information that is shown and possibly displaying some of it graphically. As the BUV project continues to grow, the amount of data that is measured and monitored is expected to increase. The display will double not only as a dashboard for the driver, but also to monitor this data. Most probably, there will be different screens that the user can flip through so that the driver is not overwhelmed with extraneous information. The code can be improved to be more reliable, efficient, speedy, and stable. From a practical standpoint, automatic starting is something that can be implemented in the future, obviating the need for the hidden button press. The desired operation would be for the display software to automatically connect to Alltrax Toolkit when the vehicle is switched on, perform its function, and shut down when the vehicle is switched off. At the moment, the display is a laptop that must be manually operated as does the software. The BUV project is expected to continue to provide interesting challenges for students in the areas of clean transportation, automation, controls, etc. Future projects are expected to involve remote-controlled operation and autonomous driving as well as trip planning and optimization [7].

8 Acknowledgement

The authors are grateful to funding received from various programs in the University of Wisconsin-Colleges and University of Wisconsin-Oshkosh. They also acknowledge the work of the members of the Fox Engineering Society, particularly David Juckem, Griffin Tedlie, Andrew Zillges, and Roque Matos Martinez. Contributions from the students in the UW-Platteville Engineering Partnerships program under the supervision of Prof. Fang Yang (Electrical Engineering) are also acknowledged. The authors are grateful to Professor Ahilan Sivasamy and the FoxBytes Computer Science Club for developing the open-source code that was used.

References

[1] IPCC, 2019, “IPCC Special Report on the Ocean and Cryosphere in a Changing Climate”, H.-O. Pörtner, D.C. Roberts, V. Masson-Delmotte, P. Zhai, M. Tignor, E. Poloczanska, K. Mintenbeck, A. Alegría, M. Nicolai, A. Okem, J. Petzold, B. Rama, N.M. Weyer (eds.), in press.

[2] Institute for Affordable Transportation (IAT), retrieved from <http://www.drivebuv.org/>, 2020.

[3] Department for Transport. A new deal for transport: better for everyone. United Kingdom National Archives white paper < https://www.gov.uk/government/organisations/department-for-transport>, 1998.

[4] McDaniel, E.S., Juckem, D.A., Vaz, W.S. Utilizing a Raspberry Pi on a Hexacopter Drone to Visualize Atmospheric Pollution. 18th UW System Symposium for Undergraduate Research and Creative Activity, University of Wisconsin- Green Bay, April 26, 2019.

[5] Foster, S.W., et al. A Low-Cost, Open-Source Digital Stripchart Recorder for Chromatographic Detectors Using a Raspberry Pi. Journal of Chromatography A 1603, 396-400, 2019.

[6] Myers, B., Stylos, J. Improving API Usability. Communications of the ACM 59 (6), 62-69, 2016.

[7] Vaz, W., Nandi, A.K., Landers, R.G., Koylu, U.O. Electric vehicle range prediction for constant speed trip using multi-objective optimization. Journal of Power Sources 275, 435-446, 2015.