Autonomous Vehicles in Computer Engineering Program

Dr. Afsaneh Minaie, Utah Valley University

Afsaneh Minaie is a Professor and Chair of Engineering Department at Utah Valley University. She received her B.S., M.S., and Ph.D. all in Electrical Engineering from University of Oklahoma. Her research interests include gender issues in the academic sciences and engineering fields, Embedded Systems Design, Mobile Computing, Wireless Sensor Networks, Nanotechnology, Data Mining and Databases.

Dr. Reza Sanati-Mehrizy, Utah Valley University

Reza Sanati-Mehrizy is a professor of Computer Science Department at Utah Valley University, Orem, Utah. He received his M.S. and Ph.D. in Computer Science from the University of Oklahoma, Norman, Oklahoma. His research focuses on diverse areas such as: Database Design, Data Structures, Artificial Intelligence, Robotics, Computer Aided Manufacturing, Data Mining, Data Warehousing, and Machine Learning.

Benjamin Chambers

Benjamin Chambers is a Graduate Student taking courses from the University of New Mexico studying Electrical Engineering. He received his B.S. is Computer Engineering from Utah Valley University. His interests include Robotics, Embedded Systems, Space Systems, Video Game Design, Cybersecurity, and Artificial Intelligence.

Autonomous Vehicles in Computer Engineering Program

Abstract

The area of autonomous vehicles design has undergone tremendous growth in recent years. A major contributor of this growth has been the advances in sensor design, computational intelligence, and computer vision. The remarkable growth in autonomous vehicles design has given rise to a demand for engineers with experience in designing and implementing these systems. Automotive companies are focusing significant research and development efforts on these systems. They are recognizing the need for a large, well-trained workforce that can conduct these research and development projects. Autonomous vehicle design is currently not yet well represented in undergraduate academic programs.

In order to prepare our computer engineering students for the autonomous vehicle design experience which can be considered as a complex embedded systems design, we offer two courses on embedded systems. However, these two courses on embedded systems design are not enough to teach the students the skills that they need. In order to satisfy the ABET requirements students in computer engineering program are required to take a capstone course. The projects that students do in this capstone course are embedded projects. This paper will describe autonomous vehicle projects that the students have done in this capstone course.

Introduction

Wikipedia defines autonomous vehicle as " A self-driving car, also known as an autonomous vehicle (AV), connected and autonomous vehicle (CAV), driverless car, robo-car, or robotic car is a vehicle that is capable of sensing its environment and moving safely with little or no human input [1, 2, 3, 4]." "Self-driving cars combine a variety of sensors to perceive their surroundings, such as radar, lidar, sonar, GPS, odometry and inertial measurement units [1]. Advanced control systems interpret sensory information to identify appropriate navigation paths, as well as obstacles and relevant signage [5, 6, 7]." It can be said that autonomous vehicles are complex embedded devices.

The area of autonomous vehicles goes back to at least the 1920s where the first radio-controlled vehicles were designed. The dream of an autonomous vehicle has been there for almost a century. It can be said that the first "autonomous" vehicle was Stanford Cart which was built in 1961. The automated car concept was evolved truly in 1980s with Carnegie Melon University's NavLab and ALV projects which resulted in ALVINN which used a neural network to drive [8]. The automotive industries are conducting significant research and development efforts in the area of autonomous vehicles. The major automotive companies like Toyota, Ford, Mercedes Benz, BMW, Volvo have announced their plans for creating an autonomous car. In the race towards autonomous vehicles, Tesla and Waymo (from Google) are two of the most advanced companies. In 2014, Google introduced Google car with camera and lidar sensor. In 2015, Tesla introduced its autonomous car. Google and Tesla take different approach in their design. Google uses LIDAR (Light Detection and Ranging) and Tesla uses an array of cameras for computer vision. The autonomous cars can be fully autonomous without a need for a driver or can be driver-managed autopilot-enabled. Google SDC is an example of an autonomous car without a driver and Tesla Model S is an example of the second type.

The area of autonomous vehicle design has been gaining a tremendous growth in recent years. A major aspect of this growth has been advanced technology, a rich set of sensors and cameras, advances in computational intelligence and machine vision. The autonomous vehicle has attracted the researchers, the automotive industries, and robotics communities.

Autonomous vehicles are a hot topic in the world today. Almost every major car manufacturer and many new startup companies are racing towards a fully autonomous vehicle. These cars of the future will allow us to regain our lost drive time and perform other tasks during our daily commute to work, running errands around town, and any and every other time we have to hop into the car to go anywhere at all. Not only will autonomous vehicles give us back part of our day, they will also provide a much safer driving environment overall.

Autonomous vehicles are a well-recognized motivational tool for engineering education. It is an enjoyable topic for many students. They are great motivational tools for teaching different concepts in computer and electrical engineering programs.

Background Information

Utah Valley University (UVU) is a comprehensive regional university with over 40,000 students charged with serving Utah County, which is the second largest county in the state. UVU has a dual mission – that of a comprehensive university offering 91 bachelor’s degrees and 11 master’s degrees, and that of a community college offering 65 associate degrees and 44 certificates. To fill its community college mission, the institution maintains an open-enrollment policy. To facilitate academic robustness, UVU has implemented a structured enrollment policy that establishes requirements which students must meet before they can engage in all the courses of their major and provides additional access to advising and other resources. These additional preparatory course increase students’ time to graduation but helps them succeed. As a large public university UVU has a very high number of low-income students – the largest proportion in the state [9]. Around 35% of students are classified as non-traditional students (age 25 or older). Nineteen percent of the students have children under the age of five [10]. UVU’s students live at home or in off-campus housing, which makes it very difficult to organize activities for them. Many students do not have time to spend much time outside of class on campus, leading some to feel little connection with other students. About 80% of UVU’s students will remain in their communities and pursue employment in this region [11, 12].

Engineering and Computer Science Departments

To meet one of the region’s most pressing workforce needs, UVU initiated three new engineering programs in Fall 2018. The new bachelor’s degree programs in Electrical Engineering, Civil Engineering, and Mechanical Engineering have joined UVU’s established programs in Computer Engineering and Pre-Engineering in a new Department of Engineering. The new programs were immediately popular with students, with 300 students enrolling for Fall

2018. Currently, the new Engineering Department has more than 800 students in five programs which are housed in that department. Before forming the Engineering Department at UVU, Computer Engineering program was housed in the Computer Science department which offers a bachelor's degree in Computer Science with two areas of specialization – Computer Science (traditional) and Computer Networking. It also offers a Software Engineering degree. The Bachelor of Science in Computer Science program was one of the first Bachelor of Science programs implemented at UVU in 1993. The program's goal has been to provide a quality program that meets accreditation standards while providing the students with a skill set that allows them to succeed in computing careers. The Computer Science degree at UVU is accredited by Accreditation Board for Engineering and Technology (ABET) in 2002 and currently has more than 1200 students.

Computer Engineering Program's Senior Design Project Course

Our Senior Design Project Course serves as a project-oriented capstone course for computer engineering majors. This required course emphasizes major hardware and software co-design. This course satisfies the ABET (Accreditation Board for Engineering and Technology) requirements for providing students with a significant hands-on design experience [13]. Our senior design course is structured as a collection of open-ended independent student projects which are mutually selected by the faculty supervisor and student. It is shown that this type of student-driven, open-ended project requires a great deal of instructor's flexibility, deep familiarity with available components, and ready suggestions for potential projects. However, for instructors who are willing to take on the effort, a student-driven design project can provide significant experience for students in problem specification and engineering design. The typical design process experience includes problem definition and constraints, gathering information, concept generation, preliminary design, detail design, communication of results, and improvements [14]. Our capstone course is based on the Engineering Design Process which is outlined in Table 1[15].

Engineering Design	
Requirement Analysis	**Identify the problem and constraints** **Define goals and criteria**
Functional Analysis	**Research and gather data**
Design Synthesis	**Brainstorm: Develop Possible Solutions** **Analyze potential solutions** **Model and test candidates** **Select a promising solution** **Build a Prototype** **Test and evaluate prototype** **Implement** **Communication of Results**
System Analysis and Control	**Improve: Review and redesign as needed**

Table 1: Engineering Design Process [15]

Our Capstone course is offered every semester. The students in the Computer Engineering program take this course during their last semester. Students have the option of selecting their own embedded project or to work on a project that is given to them by their advisors. During the first week of the semester, students write a proposal to define problems and identify solution approaches for their project in addition to identifying the hardware and software that is needed for their project. After several iterations, the advisor approves their project. The faculty adviser will meet with each student individually on a weekly basis at a regularly scheduled, mutually agreeable time.   These meetings are considered mandatory for the students. Occasional conflicts are inevitable, but the students need to understand that a portion of their grade for participation is based on attendance at the weekly meetings. At each meeting, issues associated with the project will be discussed and a status report will be provided by the student to the advisor. Students will keep a daily journal/work log detailing the work that was done, how much time was spent that day, and any technical details that might be needed for later reference. The faculty advisor keeps notes of each meeting as well as action items to be accomplished for the next meeting. Reviewing the log sheet from the previous meeting is a great way for the faculty to prepare for the upcoming one and provides further evidence to the student of the meeting's importance. At the end of the semester, students turn in a final written report and final presentation which is evaluated by several faculty members from the department.

Integration of Autonomous Vehicles concepts in Computer Engineering Program at UVU

In order to prepare our computer engineering students for the embedded systems design experience, which serves a critical element of their education; we offer two courses on embedded system design. Embedded Systems I is a junior level course and Embedded Systems II is a senior level course. However, these two courses on embedded systems design are not enough to teach the students the skills that they need. The focus of our computer engineering capstone design class has been the design of embedded systems. By requiring an embedded design project in our capstone course, our students receive hands-on training in embedded systems that will enable them after graduation. Autonomous vehicle design is a complex embedded systems design. This paper presents the details of two projects that our computer engineering students have done in the area of autonomous vehicle in their capstone course.

First Project: Autonomous Robotics

The objective of this project was to build a robot to travel autonomously using a 360-degree lidar range finding system. The lidar sensor was used to detect objects around the robot, allowing the robot to navigate and avoid obstacles around it. An XBee radio chip was used to transmit data from the robot to a computer for both debugging purposes and map generation purposes. The massive amount of data points from the lidar system was divided into zones, and decisions about where to go and how to avoid obstacles depended on how close objects were in each zone around the robot.

A PhantomX AX Metal Hexapod Mark III kit (hexapod robot) [17, 18] was used for this project. Figure 1 depicts the final configuration of the hexapod robot.

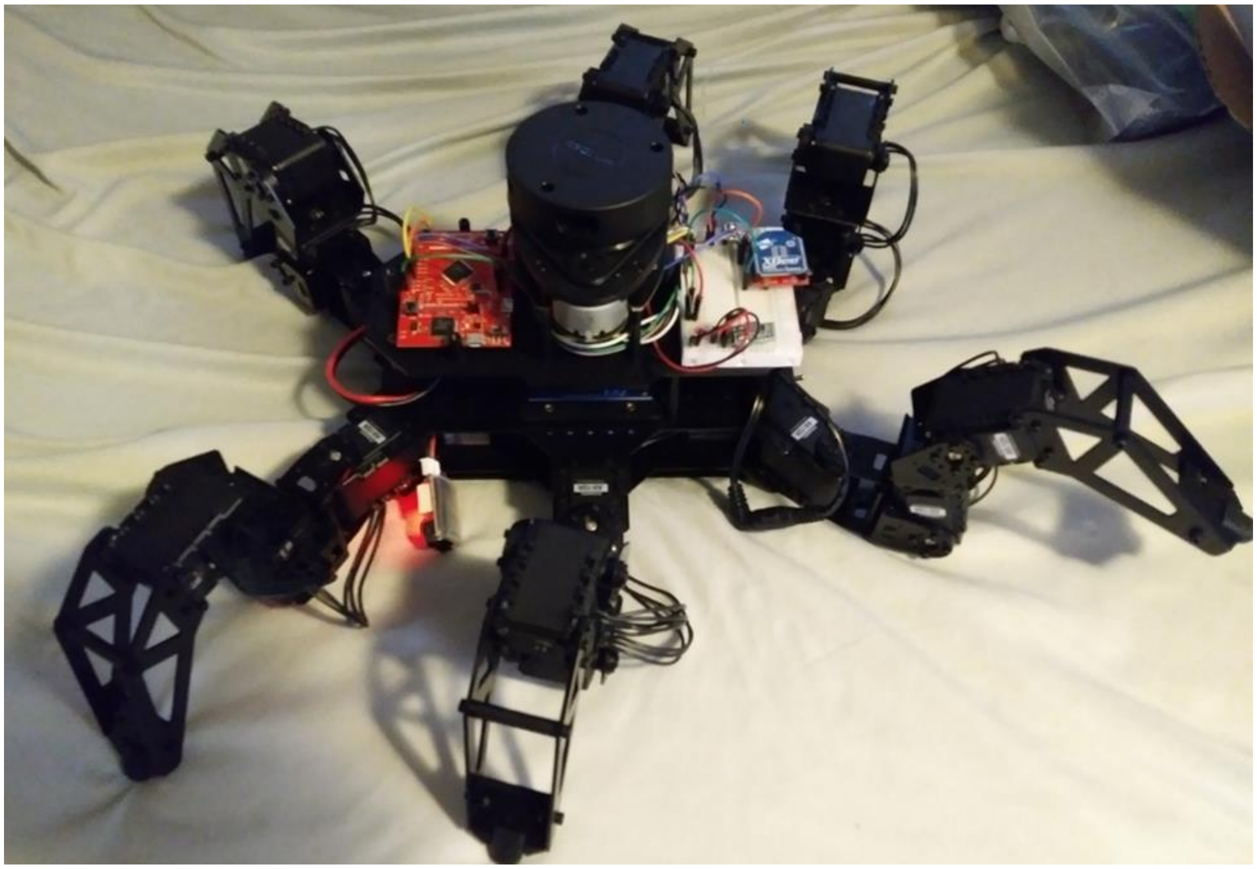


Figure 1: Final configuration of the hexapod robot

The EK-TM4C123GXL (TM4C) microcontroller (Figure 2) was used to parse through all the input data from the Lidar and do the actual autonomous route generation. The TM4C will then via UART serial send commands to the ArbotiX-M microcontroller to move the legs. The programming of the ArbotiX-M microcontroller was not changed from the kit’s defaults.

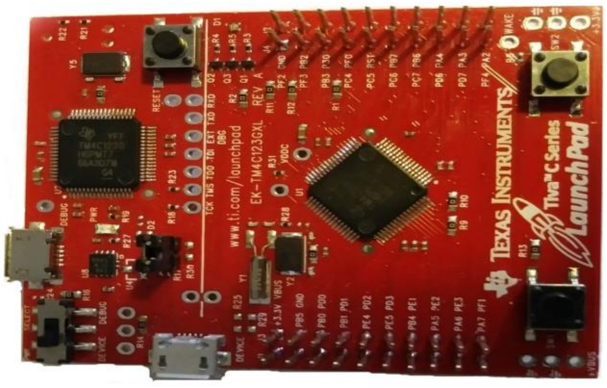


Figure 2: EK-TM4C123GXL Microcontroller

The system diagram for this project is shown in Figure 3.

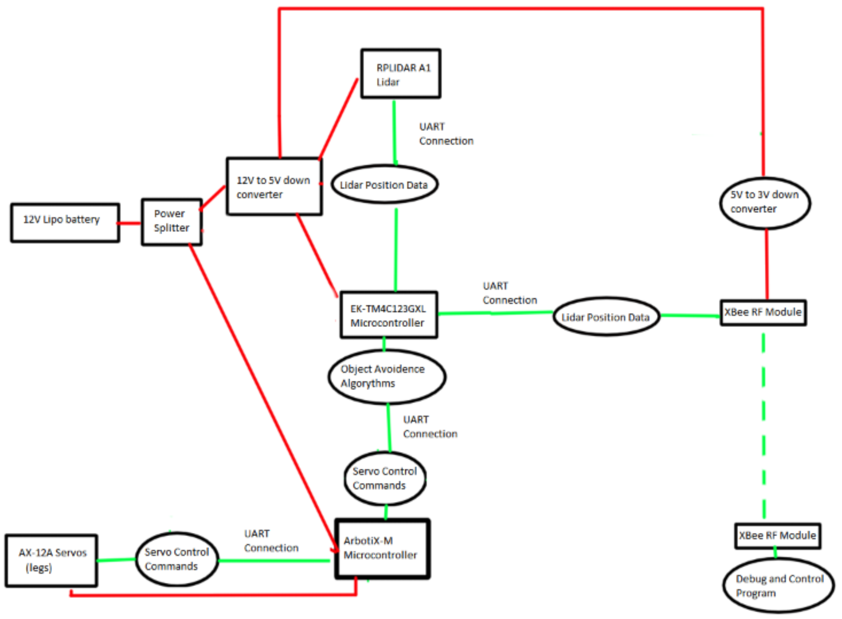


Figure 3: System Design Schematic

Data Flow

The data flows first from the RPLIDAR A1 in the form of a frame 5 bytes long through a UART connection line to the EK-TM4C123GXL microcontroller. This frame has the raw measurement data that the RPLIDAR A1 received from its sensors. The frame has both the angle and distance measured.

Once arriving at the EK-TM4C123GXL microcontroller the data is immediately copied and sent through a second UART bus to a connected XBee radio frequency module. This module will then send the raw frame over its radio connection to another XBee module connected to a computer running the debug and control program for analysis.

Once the copy is sent the EK-TM4C123GXL will then start to parse the data, throwing out bad frames and computing the direction the robot needs to travel in. It then formats a frame on how to move so that the ArbotiX-M microcontroller will understand [18], and sends in on a third UART bus connection to the ArbotiX-M. The ArbotiX-M will then direct the AX-12A servos in the legs of the robot exactly where to move.

Wiring diagram for the robot is shown in Figure 4.

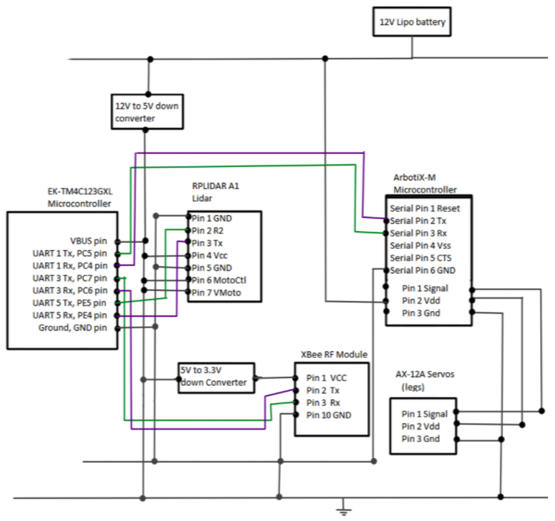


Figure 4: Wiring Diagram for the Robot

Object Avoidance Algorithm

The RPLIDAR A1 is designed to continually send data points once a scan has begun. This data is in the form of a 5-byte frame. First byte is for frame detection, second and third are for the angle measurement was made in degrees, and the last byte is for the distance measured in mm.

The first instinct is to have the microcontroller map out all the points in a two-dimensional grid and then use a path-finding algorithm on that data grid as seen in Figure 5 below, generated by the debug program on the computer itself. That is not a valid option though for the EK-TM4C123GXL microcontroller though as it would take up much time to parse such a large data set, preventing the robot from moving in real time.

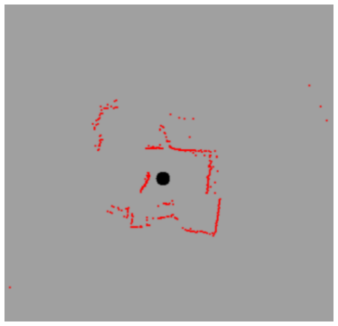


Figure 5: Room mapped out via lidar

Instead for this project a positioning algorithm to detect the closest object in 30-degree arcs around the robot is used. Each arc will both detect the closest object that it might need to avoid, and it will detect the average distance of objects in that arc. Both data points allow the robot to make intelligent decisions on where it should go. This limits the number of points that need to be iterated over to 12, rather than thousands it would take to measure a grid. Figure 6 below has this algorithm in action, showing by color how dangerous a zone near the robot is. That figure is from the debug and control program that the robot also sends data to.

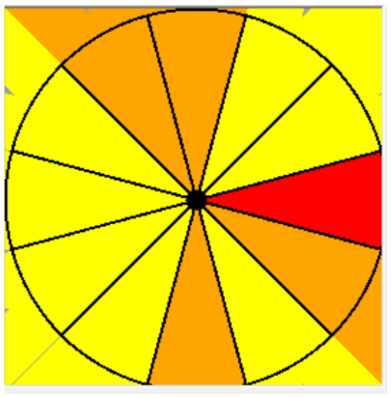


Figure 6: Zone danger levels graphics from debug tool

The debug program calculates these separately in order to show the visual above, but the EK-TM4C123GXL follows the exact algorithm in the debug program. When a change is made to one, it is made to both for consistency of the debug tool.

The danger levels in Figure 5 are calculated once every 360 valid data points received from the lidar system. The EK-TM4C123GXL then calculates the best probable path to move forward. The robot will move forward until it hits a barrier, avoiding objects along the way. At that point it hits a barrier it will turn until it finds a new path to go, before continuing onwards.

The robot also has a series of command characters that it can receive from the debug program. These characters allow more direct control of the robot. If autonomous mode is turned off both the debug program and the PhantomX controller that came with the kit can be used to directly control the robot’s motions. The debug tool then has a button to turn autonomous mode back on. Figure 7 shows a diagram of the basic algorithm’s steps that the EK-TM4C123GXL runs upon startup.

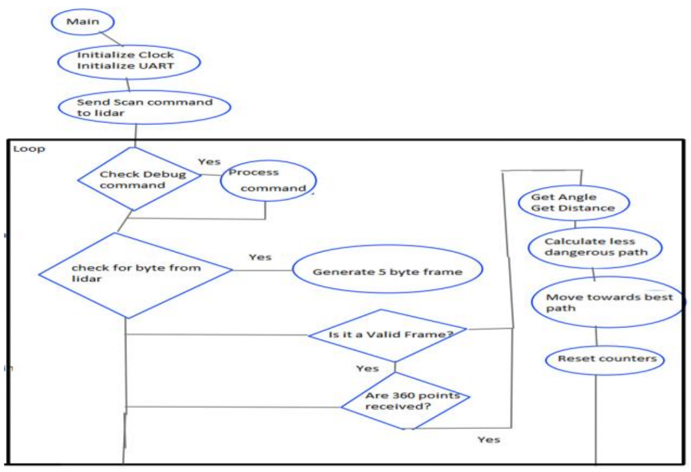
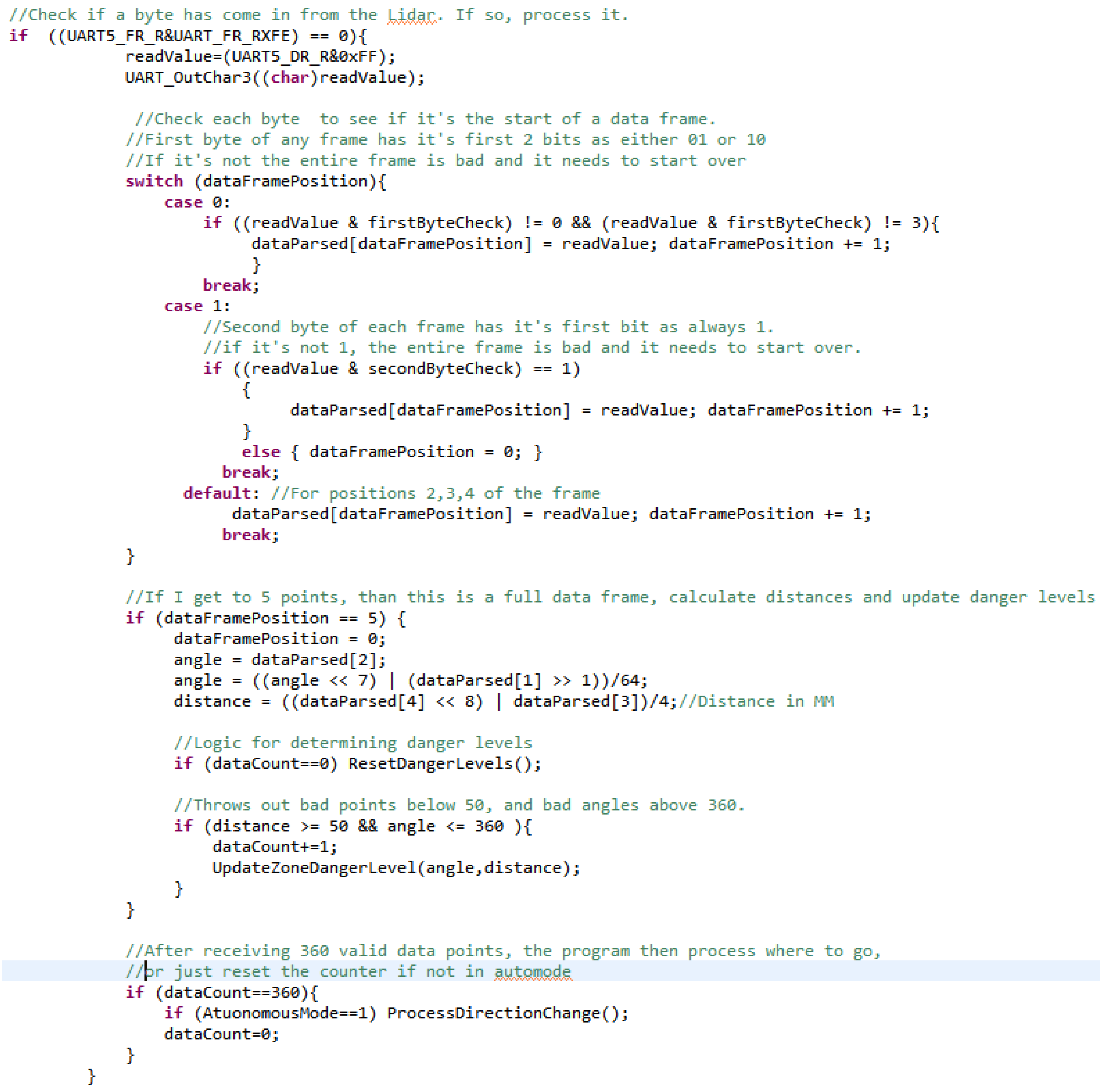


Figure 7: Basic Flow Chart

Figure 8 shows EK-TM4C123GXL's main code snippet.



```
//Check if a byte has come in from the Lidar. If so, process it.
if  ((UART5_FR_R&UART_FR_RXFE) == 0){
            readValue=(UART5_DR_R&0xFF);
            UART_OutChar3((char)readValue);

             //Check each byte  to see if it's the start of a data frame.
            //First byte of any frame has it's first 2 bits as either 01 or 10
            //If it's not the entire frame is bad and it needs to start over
            switch (dataFramePosition){
                case 0:
                    if ((readValue & firstByteCheck) != 0 && (readValue & firstByteCheck) != 3){
                         dataParsed[dataFramePosition] = readValue; dataFramePosition += 1;
                         }
                    break;
                case 1:
                    //Second byte of each frame has it's first bit as always 1.
                    //if it's not 1, the entire frame is bad and it needs to start over.
                    if ((readValue & secondByteCheck) == 1)
                        {
                             dataParsed[dataFramePosition] = readValue; dataFramePosition += 1;
                        }
                        else { dataFramePosition = 0; }
                      break;
                  default: //For positions 2,3,4 of the frame
                       dataParsed[dataFramePosition] = readValue; dataFramePosition += 1;
                      break;
            }

            //If I get to 5 points, than this is a full data frame, calculate distances and update danger levels
            if (dataFramePosition == 5) {
                 dataFramePosition = 0;
                 angle = dataParsed[2];
                 angle = ((angle << 7) | (dataParsed[1] >> 1))/64;
                 distance = ((dataParsed[4] << 8) | dataParsed[3])/4;//Distance in MM

                 //Logic for determining danger levels
                 if (dataCount==0) ResetDangerLevels();

                 //Throws out bad points below 50, and bad angles above 360.
                 if (distance >= 50 && angle <= 360 ){
                     dataCount+=1;
                     UpdateZoneDangerLevel(angle,distance);
                 }
            }

            //After receiving 360 valid data points, the program then process where to go,
            //or just reset the counter if not in automode
            if (dataCount==360){
                if (AtuonomousMode==1) ProcessDirectionChange();
                dataCount=0;
            }
        }
```

Figure 8: EK-TM4C123GXL main code snippet

Each 5-byte frame has 3 bits that are used to check if the frame is valid. The first 2 bits of the first byte are either 10 or 01. The first bit of the second byte must be a 1. If either of these conditions are false, then the frame is bad, and the program starts over on a new frame. Once a frame is verified and all 5 bytes are received, the program then pulls out the angle and distance data from the frame. Next the program will update danger levels in 30-degree arcs around the robot, called danger zones. Zones range from 0 to 12, with zone 1 facing directly forward, and zone 1 includes degrees from the lidar from 255 degrees to 285 degrees.

This project was completed successfully. The student was incredibly satisfied with his capstone project, stating that “I feel I learned quite a bit while doing this project. I learned the technical details about all the pieces that were used to put the robot together. I also learned how to deal with stress on large projects, through time management throughout the semester, and pushing myself to work on it even if it felt overwhelming in both scope and in some of the technical problems I ran into”.

Second Project: Donkey Car - Autonomous RC Car

The purpose of this capstone project was to explore current technologies related to autonomous vehicles. “As such, a practical implementation of the researched technologies was constructed, tested, and evaluated” [18]. The DK project was selected by this student as a basis for his research due to its form factor and primary goal of creating an inexpensive open source RC car. The form factor being an RC car, it is far less expensive to procure than a full-size vehicle, and much safer to conduct experiment with. Furthermore, the DK project makes use of the TensorFlow and Keras Python libraries to train a neural network that will drive the vehicle. This is an excellent example of how to create a simple autonomous vehicle, while the open source nature of the project allows undergraduate students to dive into the functionality of the machine learning aspects [18].

A Donkey Car (DK) is an open source RC platform that will be an autonomous RC car when fully assembled and programed. DK uses a Raspberry Pi 3B+, PWM controller, and Pi Camera. All of the documentation required to build a DK can be found online at www.donkeycar.com ("Donkey® Car"). The total cost of the project if all parts are purchased from the recommended sources is about $300.00. The following parts are required to build a DK [18]:

1. A compatible RC car - An Exceed Racing Desert Short Course Truck (1/16th scale) was used in this build
1. A Raspberry Pi 3B+
2. A PCA9685 PWM controller.
3. A microSD card, at least 8GB in capacity.
4. SanDisk Extreme 32GB microSDHC card
5. An external battery to power Raspberry Pi.
6. A Pi Camera with ribbon cable.
7. An external battery to power Raspberry Pi.
    a. Mobile device power banks typically come with a USB A to Micro USB cable, which will fit the micro USB port on the Raspberry Pi.
    b. An Anker Astro E1 power bank was used in this build.
8. Dupont female to female jumper cables.
    a. This is to connect the Pi to PWM controller board.
9. A Pi Camera with ribbon cable.
    a. A fisheye lens is recommended, for a wider world view on training data.
10. A framework to hold all the parts on the RC car body.
    a. 3D printed parts can be purchased directly from the DK project site
    b. 3D model files for the framework are open source and can be downloaded from the project site and printed on any available 3D printer.

Figure 9 shows front view of the assembled DK.

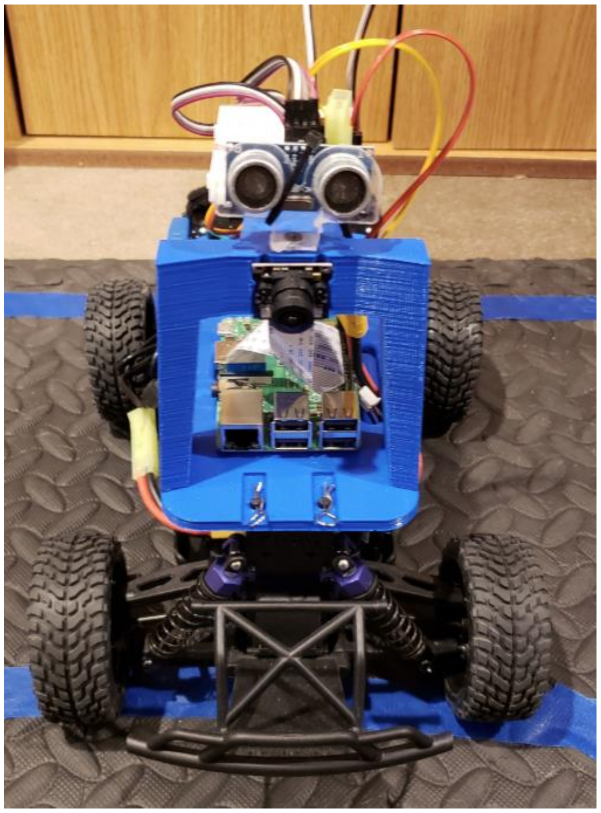


Figure 9: Front View of Assembled Donkey Car

The Donkey Car (DK) project has been proven to be a good platform to study machine learning (ML) and autonomous vehicles for undergraduate students. The DK project has also provided experience in assembling hardware and viewing a working ML training process and model for undergraduate students [18].

The Donkey Car (DK) base project encourages the addition of sensors, to make an autonomous vehicle that is better than the base project. In this project, an ultrasonic distance sensor was added to the DK, in order to give another data point in the training process [18]. This student commented after finishing his project that I had a lot of fun with project and it was an excellent learning experience.

Conclusion

Interest in autonomous vehicles are growing among tech giants, carmakers, and emerging startup companies. They are racing to first implement an autonomous vehicle. They are dedicating a lot of resources for research and development in the area of autonomous vehicles. The companies are realizing the shortage for well-trained workforce that is equipped with knowledge to do research and development in this area. The undergraduate students are realizing the benefits of having knowledge in this area which opens a lot of opportunities for them.

Senior capstone design courses remain an engaging aspect of undergraduate computer engineering education and fulfill many requirements set forth by the Accreditation Board for

Engineering Education and Technology (ABET). This paper presented senior design projects recently conducted in the area of autonomous vehicles. Our senior design course is structured as a collection of independent student projects. As our students are required to design, build, and troubleshoot a fully functional embedded project, they find this course both challenging and rewarding. The students' feedback and their final project presentation indicate that they have pride in their project accomplishments and have gained confidence in their engineering abilities.

References

1. Taeihagh, Araz; Lim, Hazel Si Min, *"Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks"*. Transport Reviews. 39 (1): 103–128, January 2019.
2. Maki, Sydney; Sage, Alexandria , *"Self-driving Uber car kills Arizona woman crossing street"*. Reuters, March 2018.
3. Thrun, Sebastian, *"Toward Robotic Cars"*. Communications of the ACM. **53** (4): 99–106, 2010.
4. Gehrig, Stefan K. and Stein, Fridtjof J., "*Dead reckoning and cartography using stereo vision for an automated car*, IEEE/RSJ International Conference on Intelligent Robots and Systems, 1507–1512, 1999.
5. Lassa, Todd *"The Beginning of the End of Driving"*. Motor Trend, January 2013.
6. *"European Roadmap Smart Systems for Automated Driving"*, EPoSS, 2015.
7. Lim, Hazel Si Min; Taeihagh, Araz, "*Algorithmic Decision-Making in AVs: Understanding Ethical and Technical Concerns for Smart Cities"*. Sustainability, **11** (20): 5791, 2019.
8. Fayjie, Abdur, et.al., "*Driverless Car: Autonomous Driving Using Deep Reinforcement Learning in Urban Environment*", 2018 15th International Conference on Ubiquitous Robots, 2018.
9. Annual Report on the State of Poverty in Utah, 2014, Community Action Partnership of Utah, http://caputah.org/images/poverty-reports-full/2014_Annual_Report_on_Poverty_.compressed_1.pdf , access on 1-2-2020.
10. Annual Report on the State of Poverty in Utah, 2012, Community Action Partnership of Utah.
11. Information and statistics provided by the UVU Office of Institutional Research and Information – IRI.
12. NSF Proposal document, "Strengthening Outcomes for Students in Computer Science and Engineering through Leadership, Engagement, Academic Mentoring, and Preparation (LEAP)", August 2013.
13. ABET, Inc. Criteria for Accrediting Engineering Programs, http://abet.org, 2013.
14. Dieter, George and Linda Schmidt, "Engineering Design", 4th edition, McGraw-Hill, 2009.
15. Prairie, Michael, et al., "Introducing Systems Engineering Concepts in a Senior Capstone Design Course", Proceedings of American Society for Engineering Education, 2012.
16. Trossen Robotics- Phantomx Hexapod. Accessed April 2019 from https://www.trossenrobotics.com/phantomx-ax-hexapod.aspx.
17. Trossen Robotics. Arbotix Commander Packet Calculator. Accessed April 2019 from https://learn.trossenrobotics.com/arbotix/arbotix-communication-controllers/168-arbotix-co .
18. Mathew J. Burnett, "Donkey Car: Autonomous RC Car", ECE 4800, Utah Valley University, Final Report, Spring 2019.