Project Deep Learner

DESIGN DOCUMENT

Senior Design Team 10
Dr. Diane Rover
Jose Carlos Garcia
Benito Moeckly
Jazzlyn Jacobus
Caleb DeBoef
sdmay23-10@iastate.edu
https://sdmay23-10.sd.ece.iastate.edu

Revised: December 2, 2022/Version 4

Executive Summary

Development Standards & Practices Used

- IEEE 2830-2021: IEEE Standard for Technical Framework and Requirements of Trusted Execution Environment based Shared Machine Learning
- IEEE 2050-2018: IEEE Standard for a Real-Time Operating System (RTOS) for Small-Scale Embedded Systems
- IEEE 26531-2015: International Standard for Systems and software engineering
- IEEE 802.11: Standard for wireless LANs

Summary of Requirements

- Develop a DeepRacer Model capable of competing in AWS's DeepRacer League.
- Develop a test bed with the DeepRacer as the basis to implement into undergraduate courses in forms of labs to expand the average undergraduate ECPE student's knowledge and introduce machine learning in an embedded systems context.
- Allow for scalability of the project to be cost effective and rapidly prepared in a by-semester experience.

Applicable Courses from Iowa State University Curriculum

- Introduction to Embedded Systems (CprE 288).
- Computer Organization and Assembly Level Programming (CprE 381).
- Cybersecurity Fundamentals (CybE 230).
- Introduction to Algorithms (Com S 311).
- Operating Systems (CprE 308).
- Electronic Circuits and Systems (EE 230).
- Real-time Systems (CprE 458/558).

New Skills/Knowledge acquired that was not taught in courses

- Fundamental AI Knowledge.
- Fundamental Machine Learning Knowledge.
- Machine Learning Algorithms.
- Introduction to Neural Network Implementations.
- Reward Based Machine Learning Training.
- Fundamental Cloud Computing Knowledge (AWS).

Table of Contents

List of figures/tables/symbols/definitions

Figures

Tables

1 Team

1.1 Team Members:

Jose Carlos Garcia, Benito Moeckly, Jazzlyn Jacobus, Caleb DeBoef

1.2 Required Skill Sets for Your Project

- Embedded Systems Knowledge
- Operating Systems Knowledge (processes, memory allocation)
- Efficient Algorithm Development
- AWS Knowledge
- Machine Learning/AI Basics and Algorithms
- Circuit Systems Knowledge
- Real-Time Operating Systems Knowledge
- Networking Stack Knowledge
- Software Development Practices – version control, proper git etiquette

1.3 Skill Sets covered by the Team

- Embedded Systems Knowledge
  - Members: All Members
- Operating Systems Knowledge (processes, memory allocation)
  - Members: All Members
- Efficient Algorithm Development
  - Members: Jose Carlos, Jazzlyn, Benito
- AWS Knowledge:
  - Members: All Members
- Machine Learning/AI Basics and Algorithm:
  - Members: All Members
- Circuit Systems Knowledge
  - Members: Caleb
- Real-Time Operating Systems Knowledge
  - Members: Benito and Jose Carlos
- Networking Stack Knowledge
  - Members: Jose Carlos, Benito, Jazzlyn
- Software Development Practices – version control, proper git etiquette
  - Members: All Members

1.4 Project Management Style Adopted by the team

For this project, we have decided to adopt an Agile-Waterfall hybrid for this project. For the most part we can do many tasks in any order with some that we have intentionally put off before completing preliminary tasks.

1.5 Initial Project Management Roles

For the project, as we all have our specialties in knowledge, we decided to not have management roles and instead communicate efficiently with each other to get everything done well and in a timely manner. This group structure of nobody being "more important" than another allowed us to more efficiently get work done as there was never time wasted waiting for someone else to do their job as we were all responsible for the work. This also created a good power dynamic in the group where nobody felt bossed around and increased our group morale.

2 Introduction

2.1 Problem Statement

We were not given a problem but rather a topic to explore for our senior design project. The topic was machine learning for embedded systems. The goal of this topic exploration is to see if there is a way that we can integrate machine learning into the class CPRE 288. We decided to pick up an AWS DeepRacer unit to learn more about machine learning on an embedded system in hopes of finding a way to make this topic accessible to undergraduate students in the ECPE department.

2.2 Intended Users and Uses

The primary audience for this project will be undergraduate students in our department. These students will likely be second or third years in one of the engineering disciplines associated with the ECPE department (electrical, computer, software, and cyber security.) The secondary audiences will be the instructors and Tas for CPRE 288. Sent. All potential users will have similar needs. Because we can expect our intended users to be able to code and use coding environments. What the users will need is documentation on how to make a Python script to program the reward function, as well as how to train the DeepRacer to learn how to complete any given task.

2.3 Requirements & Constraints *

- Physical:
    - Track must be compact to preserve space as well as be able to be stored and rebuilt while still being within the specified width and length requirements. (**Constraint**)
    - Additional features to the car must be supported by the AWS DeepRacer framework and licensed by Amazon. (**Constraint**)
- Resource:
    - Software shall be stored in a method that is easy to access and to update efficiently, with version control.

    - The entire process must be well documented, with the added ability to easily append updates to the design for future use. This means high-level and low-level documentation as well as commented, readable code.

    - Implementation must be lightweight, to run on the DeepRacer hardware (embedded machine learning brings more constraints than regular machine learning that might have access to extremely powerful computers.)

        - Lightweight in terms of ensuring the OS meets its scheduling requirements effectively and predictably. (**Constraint**)

- Functional:

    - Functional algorithm that will teach a robot car to run a track.

    - Ability to work in the simulated environment and on a physical track with DeepRacer robot.

    - Must be able to complete a full lap on a track in the ASW DeepRacer League

- Aesthetic:

    - The DeepRacer can be modified to include additional sensors, however – it must not compromise the design in a method that interferes with the overall performance of the vehicle.

2.4 Engineering Standards

- IEEE 2830-2021: IEEE Standard for Technical Framework and Requirements of Trusted Execution Environment based Shared Machine Learning
    - With a potential fleet of multiple cars in the educational environment, this would effectively fall into the domain into shared machine learning. As a result, we will need to comply with the IEEE standard.
- IEEE 2050-2018: IEEE Standard for a Real-Time Operating System (RTOS) for Small-Scale Embedded Systems
    - With our project in the domain of a small-scale real-time embedded system with multiple sensors, we will also follow the IEEE standard to ensure that our device behaves properly in unforeseen circumstances – following the ideals of real-time operating systems and scheduling.
- IEEE 802.11: Standard for wireless LANs
    - We will be using the 802.11 ac Wi-Fi standard for our project. This is because machine learning applications to embedded systems almost always require a network connection to a more powerful computing devices in order to be able to process its data.
- IEEE 26531-2015: International Standard for Systems and software engineering
    - As code within a group can become very messy, we have decided to follow the process in this document to ensure that our code conforms to a basic standard to ensure readability and reliability.

3 Project Plan

3.1 Project Management/Tracking Procedures

We plan to use agile with a few waterfall aspects as we must continue in a semi-linear fashion. We plan on trying to work on code in parallel, however since some code will depend on others, we will need to waterfall into our end goal.

We also plan on using Discord for Communication and Git/GitHub for our version control software. Any meetings which will require us to see each other, such as our advisor and client, we will use Zoom instead of Discord.

3.2 Task Decomposition

- Learn the skills and concepts we need for this project
    - Finish Coursera modules
    - Finish AWS module
- Create Algorithm and Simulation testing
    - Create code for virtual simulation
    - Develop model for testing
    - Test in virtual simulation
    - Debug/revise code to improve performance
    - Tune any PID feedback loops
- Test physical car
    - Build track
    - Put together deep racer with sensors
    - Test on track
- Modify code for 288 labs
    - Change sensors to fit Embedded Systems lab goals if needed
    - Develop rough code for 288 labs to use as test/example
    - Test on physical track
    - Compare to CyBot performance
    - Determine what code will be given and what will be assigned for 288 students in lab

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestones:

- Learn Skills and concepts

    - Fully complete all work and videos for the Coursera Introduction to Embedded Machine Learning course (12 hours/person total)
    - Fully complete all work and videos for Amazons DeepRacer AWS training (10 hours/person total)
- Begin Work on Algorithm
    - Develop all base functions predicted to be required for simulation (Hardware controls, Bot tolerance testing, reward function) by 20 group hours
    - Create first trained model by 40 hours
    - Have simulation tested and any visual issues analyzed by 45 hours
    - Write up first debug report and changes to be implemented by hour 45
    - Have car complete a full lap without going out of bounds virtually by hour 60
    - Get time for lap down to 1:30 by hour 75
    - Have function ready to put into physical car by hour 80
- Test Physical Car
    - Get confirmation on track location by hour 5
    - Build track by hour 10
    - Have car complete full lap by hour 15
- Modify code for lab
    - Research sensors and choose which, if any, would be beneficial to our lab/project by hour 5
    - Develop new reward function for lab assignment
    - Complete code for lab assignment and compare to CyBot by hour 30

3.4 Project Timeline/Schedule

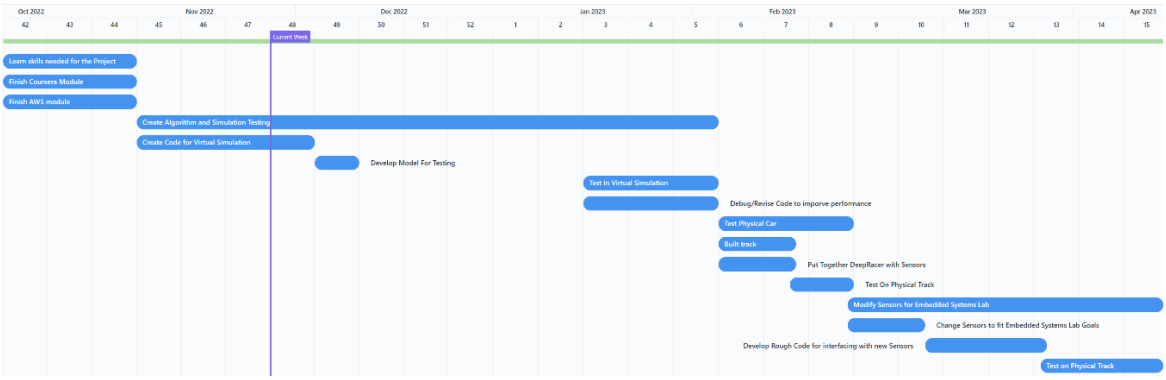

Figure 1: Gantt chart for our Project Timeline

3.5 Risks And Risk Management/Mitigation

Consider for each task what risks exist (certain performance target may not be met; certain tool may not work as expected) and assign an educated guess of probability for that risk. For any risk factor with a probability exceeding 0.5, develop a risk mitigation plan. Can you eliminate that task and add another task or set of tasks that might cost more? Can you buy something off-the-shelf from the market to achieve that functionality? Can you try an alternative tool, technology, algorithm, or board?

- Agile project can associate risks and risk mitigation with each sprint. The main risk of this project is not getting any of the deliverables done, which should not happen.
    - 5% chance, can be mitigated by following milestones closely and keeping team accountable
- Possibility that sensors/robot may not be as useful as ones already used in 288 to teach development techniques.
    - 3% chance, can be mitigated by researching sensors in advance of testing them
- Possibility that we might not be able to successfully add sensors to the DeepRacer bot
    - 20% chance, will still be able to fall back on default sensors
- Possibility that the robot breaks while testing or sensors become damaged.
    - 10% chance, will lose time dealing with Amazon returns/Customer support

3.6 Personnel Effort Requirements

Task	*Estimated hours*	*Explanation*
Initial Learning	*22*4 = 88*	*Coursera Course has about 12 hours of work listed and AWS course has about 10 hours listed*
Algorithm & Virtual Testing	*20*4 = 80*	*This process will involve writing the machine learning algorithm and testing it virtually via a simulation provided by AWS. It will also involve debugging and making the algorithm more efficient. We believe each team member will spend about 20 hours completing this task.*
Physical Testing	*20*	*Once the bot has been virtually tested, everything should transfer over to the physical model, leaving little or no issues to be resolved in Physical testing. We believe all members will spend about 20 hours in total completing the testing since there should be no issues.*
Test Bed Development	*35*	*For this task we will have to do all our own research on sensors and the process of adding them to the DeepRacer car as well as developing code to in the data. The purpose of this test bed is to develop potential labs for undergraduate courses.*

Table 1: Personnel Effort Requirements

3.7 Other Resource Requirements

- Documentation on the DeepRacer libraries in order to process data from its sensors and control it's actuators (this is provided on the AWS website.)
- Resources on machine learning algorithms and theory, specifically on reinforcement learning.
- Supplies for track to test (tape, large open space, obstacles such as pipes and rocks)

4 Design

4.1 Design Context

4.1.1 Broader Context

We are only designing this project for the students of ISU. It will primarily be an educational tool and only be used in the context of learning about machine learning or a competition.

Public health, safety, and welfare:

Our project will not be a public project. However, there may be some implications for safety as it pertains to the physical DeepRacer unit. It shouldn't be a huge safety concern but designing the algorithm to keep the car within the bounds of the track should help it be safer.

Global, cultural, and social:

Seeing as how we are looking to set the groundwork for a club/team at ISU, we are affecting the community here in the ISU engineering department.

Environmental:

There shouldn't be an environmental impact from our project, other than the materials required to make the deep racer or the energy needed to run our algorithm.

Economic:

We do need to consider the monetary cost of the deep racer. It is not a lot but if we want to integrate machine learning into CPRE 288 then we may want to consider a cheaper interface that can be bought for a whole class section to use at once.

4.1.2 Prior Work/Solutions

We are not making a product or following previous work. Using DeepRacer as a learning tool for a university appears to us as a novel idea. While it may be possible to create a similar product using off the shelf components, that would involve much more compatibility troubleshooting and would take away from the key purpose of the project which is teaching machine learning.

4.1.3 Technical Complexity

First, we will address the metric of utilizing mathematical and engineering principles. Most of the physical components are already made for us so the components in question are mostly software and mathematics. We will be using the reinforcement learning approach to machine learning, which is a popular choice for applying machine learning to autonomy. We will also be studying the documentation on the DeepRacer in order to utilize its current libraries for our own purposes. Second, the problem scope does match current industry standards. Amazon has a test bed used for teaching the DeepRacer to run laps around a track. We will be implementing our own version of this test bed that has a different application (navigating a maze.)

4.2 Design Exploration

4.2.1 Design Decisions

1. We need to figure out what kind of sensors we are going to use. We might choose to use the proprietary sensors from Amazon or utilize the sensors from CPRE 288. These decisions will rely on their compatibility as well as their application to learning and understanding the overall concept of machine learning.

2. Speed ratings and how we will control the bots slowing and turning into corners. While this can be done fully with machine learning, it also may be possible to utilize feedback controllers to minimize the effect speed has on our model which could lead to faster training times as there will be much less time from the model spent on controlling speed and more spent analyzing the track elements.

3. Performance metrics were a large concern in determining the success of the project. We needed to determine what was important and what the cutoff for a failure is, such as driving instability, lap times, and consistency to different environments. By deciding this, we then have a proper goals and metrics to use when comparing different sensors and trained models to determine what our final product will be.

4.2.2 Ideation

With our project being much more on the conceptual side now, we didn't have much to design in terms of the design as the car comes mostly pre-assembled aside from the potential add-ons we may include. For this we have a few ideas when it comes to the software design, track layout, and visual design choices when we make such expansions:

1. Making changes to the car's design:

   For this approach, we will implement a waterfall model as we will have to implement, test, decide, and adjust in different stages. This will allow us to narrow down our options – even taking a longer time when adding new sensors and capabilities to our vehicles. The DeepRacer has a smooth external shell we wish to maintain aesthetically – and in addition not have the design affect the performance.

2. Software Development Practices (machine learning):

   With our project falling into the domain of both machine learning and embedded systems, both coincide in attempting to maintain a low-profile OS/software to allow for quick and effective computations by our equipment. For this, we will have adopted an AGILE method of design, as we continue to build onto our software – we have to ensure that the writing of the data back to the cloud and pulling of this information is effective and does not affect the performance of the vehicle in addition.

3. Software Development Practices (embedded systems):

Additionally, we'll also have to implement effective OS at getting information about the vehicle's environment and processing this information is near constant O time to ensure we have the speed to write/read from the cloud in the training of this module. For this, we have decided an AGILE method would be both dynamic in developing an effective software piece – and allow us to make those quicker changes compared to the physical design of the car. And in addition to this, use a function-oriented design when building software to allow us a quick process of smaller functions combining for larger functions.

4 Track Design:

The track design is a more practical and software design choices we have to implement, as this won't be performed until the end and we have the option to train our model virtually – this will take on a more of a water fall design, in which we brainstorm – build – then test, repeating this cycle to make effective environments to test out vehicle. This will be a complete top-down design as we work down the requirements for a successful track and implement the finer details as we wrap -up the process.

5 Testing:

In terms of testing our software, we will implement a bottom-up approach testing nearly every variable to ensure both predictable and concise computing. Using automated test cases- we will focus on testing by function (see Ideation 3).

4.2.3 Decision-Making and Trade-Off

While there are pros and cons to all the ideated options, we decided to implement the 5 listed above in a modular sense as we must maintain different approaches with every element of our project. This project is unique in this design sense, in which there will be multiple aspects of the various portions of the project. However, to identify why we chose these – it comes down to personal experiences with the ideations. In the past, our group has participated in varying styles of organization and brainstorming – these were a collective effort in combining our personal experiences with where it would fit best.

4.3 Proposed Design

4.3.1 Overview

The main design here for the project is the testbed. AWS already has one for the DeepRacer League. The key components for this are:

- The AWS DeepRacer.
- The track (either a physical track or a virtual environment.)
- The reward function: the reward function is a function of the car's interaction with the environment. Something like leaving the boundaries of the environment will be worth 0 reward but reaching a goal or dodging obstacles could give the DeepRacer greater reward values. This is what our audience will be interfacing with the most.
- The Navigation algorithm. This will vary based on use, but our proposed usage is maze navigation like the final project in CPRE 288.

- AWS server, or another powerful computer connected to the car's Wi-Fi-network equipped with the ML software to modify the car's algorithm.

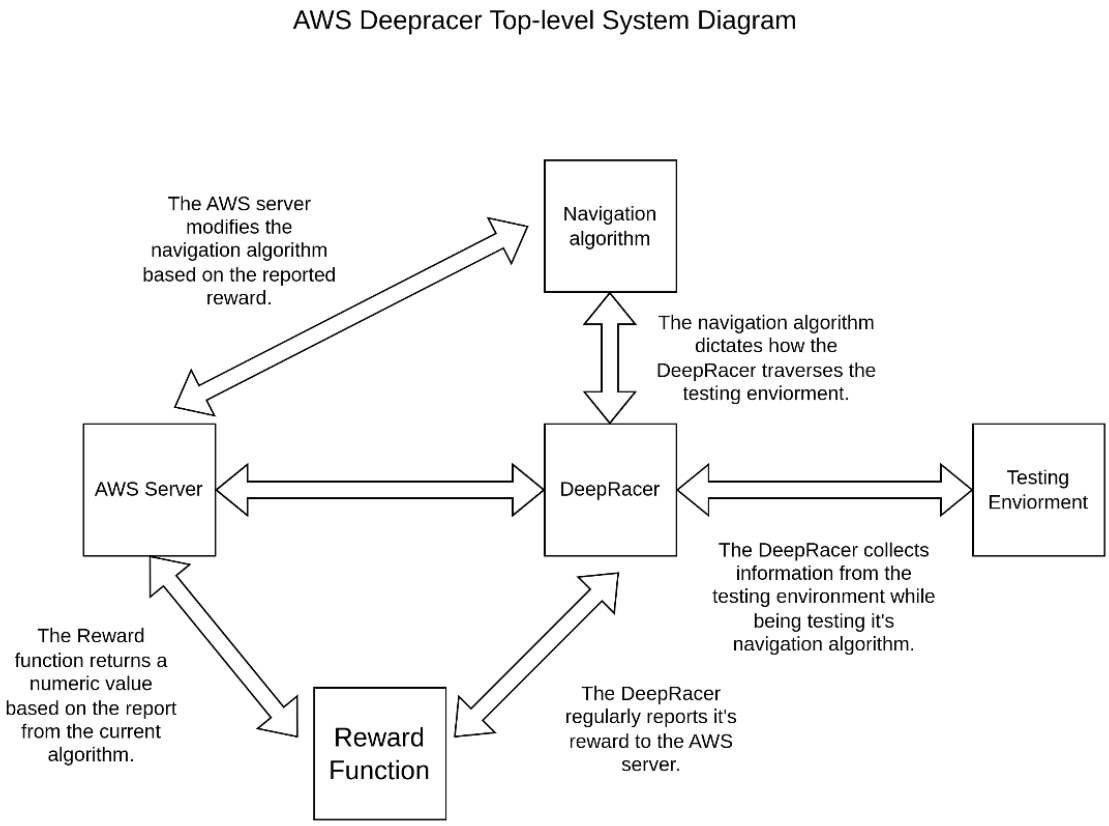

Figure 2: AWS DeepRacer Top-level System Diagram

4.3.2 Detailed Design and Visual(s)

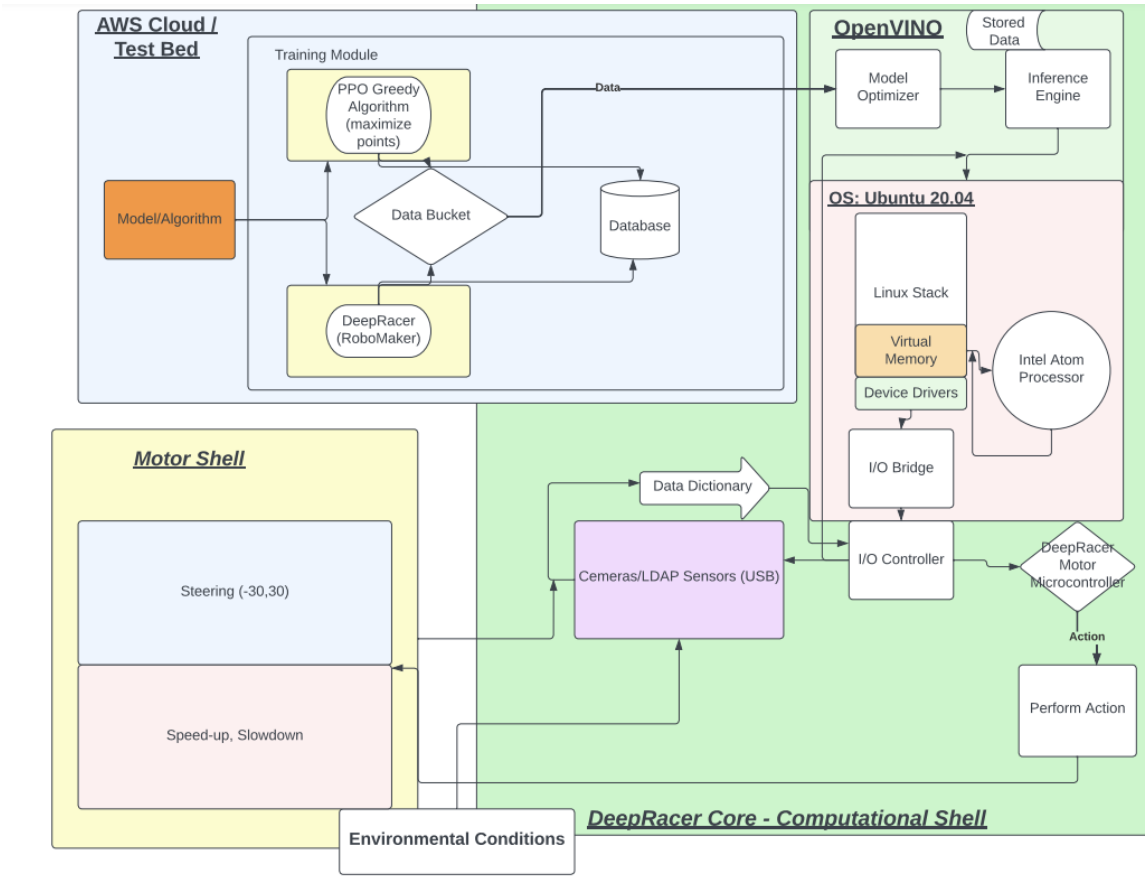

Figure 3: Detailed System Design

Shown above is a by-section overview of the DeepRacer bot and its components. As stated in other sections – the DeepRacer has the ability to be trained in a test bed – for most, AWS is used as grounds to train any model that is developed to run a course and gain faster times. For our purposes and our goals, we are aiming to develop our own test bed as well as have the ability to train models locally to offset costs associated with the standard DeepRacer league.

Starting off with the AWS Cloud/ Test Bed Section – the models, data, and other algorithms are allocated and trained using cloud computing as shown in the light blue section. The algorithms are trained via reward based PPO algorithms – therefore incorporating a greedy algorithm derivative in order to achieve optimum local performance. This test bed component is then used to reinforce what the real-world OpenVINO interface is performing. This is Intel's open AI solution – used here to make decisions for our robot.

Heading further down the software stack we find that our robot is powered by Ubuntu 20.04, naturally – much like the Android Stack, the DeepRacer core is hosted on the Linux Kernel in order to perform its processes, take input, and further establish the model. Heading further down the Linux stack – we arrive at two of note components – such as the I/O Bridge, our memory abstraction, and the CPU. This portion is data stream in which we have decisions being sent to the motor control microcontroller (essentially an op-amp circuit that either increases the coefficient of

the equation to relay whether the robot should steer one way or another (-30 to 30 degree range), or if it should be speeding up or slowing down. This simplifies the design plentifully as giving the model only two actions to perform allow it to become efficient in timing its steering and its speed to gain maximum points.

As mentioned, the highlight in this section is the AWS portion – as our project is mainly going to focus in this area – we are going to be developing a test bed that can be expanded exponentially in a local setting for undergraduate students.

4.3.3 Functionality

Currently, the plan will be that users will be able to connect to their robot via Wi-Fi through a PC on campus. They then will be able to use a simple UI in order to upload a Python script that they have created for the purpose of teaching the DeepRacer how to complete the given task (navigating a maze.) They can run it within the testing environment in a learning mode for as long as they want. When they are confident that it can complete the task, they switch to a performance mode and have the robot use the new algorithm shaped by its time spend training.

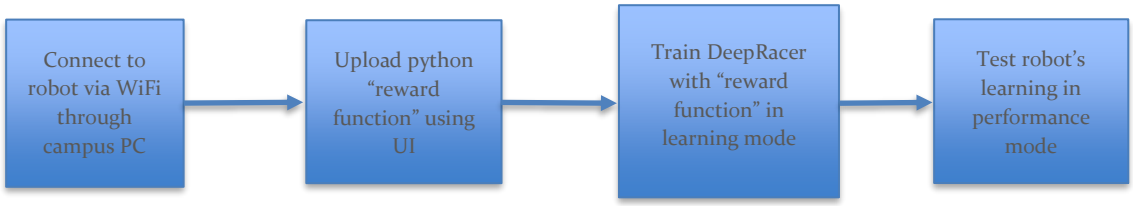

Figure 4: Visual Timeline of Functionality

4.3.4 Areas of Concern and Development

As our goal is to teach students about machine learning, we believe that our project will fulfil this very well. Labs are the best way to get students engaged in a topic so we believe that we will fulfil their learning goals well.

Our primary concerns are creating a lab that is both engaging and fun while not being too easy or difficult for the students. We need to strike a good balance to be sure that any student who wants to learn can enjoy themselves. Another concern is that we are able to adequately teach the subject using a lab as nobody has any real teaching experience. As we all have firsthand experience doing labs, we will not be going in blind however there is always a disconnect between making and doing a lab, so we hope any issues are found during our testing.

4.4 Technology Considerations

The largest technology used in our design is machine learning, which allows the system to compensate learn and compensate for many tasks given enough learning iterations. This is excellent for complex tasks such as the image processing done by the robot as it allows it to quickly reference what it sees to trained matrix and quickly compute its next action. This gives the system the strength of being resilient to changes in environment as it can simply learn from any mistakes and adapt. A downside of machine learning is the long and intensive computation required for training the model after every iteration, however thanks to cloud computing this is a non-issue.

The next technology is the bot itself, which utilizes an intel atom processor which interfaces with a camera and lidar sensor to take in images and information about its environment as well as an accelerometer and gyroscope to use for the machine learning algorithm. For different tasks, we would be able to use different sensors as the processor runs a version of Linux which can be modified to interface with more IO, so more sensors may be added to assist in our task.

The final technology used for our project is the AWS web services, which is a cloud computing platform which the DeepRacer uses to run simulations and train the machine learning algorithm. The downside of AWS is the requirement to pay hourly for training, however it is required for the DeepRacer.

4.5 Design Analysis

What has changed since the beginning of our project is scope. In the beginning we decided that teaching the robot to run a lap might be a semester long task. It proved to only be about a weeklong task after getting the robot. So, we had to expand the scope of our project to be a worthwhile task. This meant making our own application for the DeepRacer and designing our own reinforcement learning model.

6 Testing

5.1 Unit Testing

There are a two types of unit testing for us to carry out, the initial one focused on training the DeepRacer. For this, we have the DeepRacer core if training offline for custom projects and the Reward Function which contains a library of datapoints from our sensors to reward the robot for desired behavior.

The latter is testing individual experiments within our Test Bed approach (the end goal), as mentioned previously – we have open access to the DeepRacer core and its machine learning kits – allowing us to develop custom use cases for the DeepRacer bot. The specific unit testing in this case is editing a core class of the DeepRacer and running virtual tests locally to figure out performance and behavior predictably. One example of this would be getting the DeepRacer to identify traffic signs, this would require us to modify which objects our DeepRacer should be looking for – rather than other vehicles.

5.2 Interface Testing

We have 2 main interfaces: the mechanical side of the robot and the software side. The mechanical side will need to be tested for sensor calibration out of the box, bot speed consistency, ability to drive in a straight line when commanded, and movement thresholds. These will be tested as soon as the bot is available to us and will be done with the software provided by AWS for controlling the robot outside of a ML algorithm as well as with conventional testing methods such as finding deviation on a line for straightness testing and using a meter stick to measure its distance over a set time.

For the software interface, we will use the ML simulation to test our robot's functions provided we calibrated the bot correctly it should translate over 1-to-1 and provide a smooth integration of both interfaces. As there are always going to be discrepancies between simulation and real-world

performance, we will rely on our calibrations being correct enough for the machine learning algorithm to compensate for this.

A third interface we will encounter is interfacing our design with students and a classroom by creating labs and a club to allow EcpE students the opportunity to learn more about machine learning. This interface testing will be done by working directly with Dr. Rover and Dr. Jones as well as using our own experiences with the project to ensure students will get the best possible learning experience and come out of it with an appreciation for ML.

5.3 Integration Testing

There is only one unit that we are creating in this project which is the machine learning algorithm to reward and train the DeepRacer bot to run a track according to a racing strategy chosen by the team. Like previously mentioned, we will be designing and testing multiple algorithms which we will then integrate based on how they performed during testing and if they achieved the desired outcome. The integration of these algorithms will be tested the same way as the unit testing, using the DeepRacer simulation and training software. We will also be making sure that the algorithm still works with added sensors to the DeepRacer bot by making sure the bot can still complete multiple laps around a track.

We also want to integrate the DeepRacer into CprE courses. In order to test ways in which the DeepRacer could be included in specific CprE courses, we will be creating small lab assignments using the DeepRacer and machine learning topics that will be reviewed by professors and/or students. In order to test this, we will need to do these labs from scratch as if we were learning about them for the first time. After we conclude the labs make sense and are possible, we can begin to ask others in related majors major to test them and give constructive feedback. After this has been done, we can begin testing during the following semester in the CprE288 class and use the feedback we get from the students from the students to enhance it even further.

5.4 System Testing

All the tests will involve testing the system as a whole. This is because we require many trials/tests to be ran in order to teach the robot how to function. In this setting of testing, we'll have to ensure that the system we are documenting and presenting will be easily picked up by undergraduate students – if the system, for whatever reason, is unable to be easily delved into – then we will have to look at the system as a whole and append changes as we progress through the project.

In this portion of the testing, we'll incorporate focus groups within introduction to embedded systems and additional minor focus groups on campus – this will provide us the feedback needed to ensure that the educational aspect is on-par with other materials here at the university.

5.5 Regression Testing

As to train the robot it is required to run in a simulation, regression testing will be quite simple as we will only need to implement a new function and restimulate. This will allow us to quickly and easily see if a new addition to our code makes performance worse or interacts poorly with the older code and gives us a good measure for how to compensate for unexpected behavior. After we simulate, we will put the code onto the physical bot and see if it interacts with the new additions in

the same way and if not, we will be able to look at the differences between the simulated and physical bot and pinpoint the issue.

5.6 Acceptance Testing

A fully functioning DeepRacer bot will be able to navigate the maze from CPRE 288 and be able to stop in the given checkpoint. The integration of the DeepRacer bot and CprE courses will be accepted if our designed labs are accepted by professors and students as engaging and beneficial to the material being taught.

5.7 Results

While we cannot specifically provide tables for this portion of the document, as our project isn't spotted only in the engineering field – we will firstly have to approach this complicated embedded systems design and simplify aspects of it, with the intention of replication and education. With this in mind, there are two main goals:

1. The first main goal is for our DeepRacer bot to act in a predictable manner, and to perform with the intention of competing in the AWS DeepRacer League.
2. The other goal is for our DeepRacer bot to act as a platform for undergraduates to approach and learn more about AI, Machine Learning, and so forth – as there truly aren't very many experiences in which the average undergraduate can participate and learn about AI.

6 Implementation

This semester we created the first few iterations of our reward function to begin working with Reinforcement learning. We are currently ranked around 200 in the DeepRacer league. Next semester we plan to begin physical training/testing. Once finished with that, we will move on to our main focus which is creating the testbed for CprE course work. This will involve creating an example scenario similar to the 288 cybot final project. We will dive deeper into the embedded programming portion of our project.

7 Professional Responsibility

This discussion is with respect to the paper titled "*Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", International Journal of Engineering Education Vol. 28, No. 2, pp. 416–424, 2012*

7.1 Areas of Responsibility

Area of Responsibility	Definition	NSPE Canon	IEEE	Difference
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.	6. to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or	The IEEE code touches on also improving one's competence while the NSPE Canon only says to perform services they are competent in.

			after full disclosure of pertinent limitations	
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	4. to avoid unlawful conduct in professional activities, and to reject bribery in all its forms	The NSPE Canon specifies "employer or client" while the IEEE code is more general in stating when and where you should be ethical, which I would interpret as one should always be ethical not just when interacting with your employer or client.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	5. to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others	Both state that you should be honest when reporting your work or estimates. The IEEE code also touches on accepting and offering criticism and acknowledging errors while the NSPE Canon does not.
Health, Safety, Well Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	1. to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment	The egining of the IEEE code matches the NSPE canon word for word, but the IEEE code also touches on sustainability while the NSPE Canon does not.
Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	4. to avoid unlawful conduct in professional activities, and to reject bribery in all its forms; 5. To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others	The NSPE Canon is more general with there statement while the IEEE code is more specific with their statements on respecting the property, ideas, and information of others, such as properly crediting people ad accepting other's criticism.
Sustainability	Protect environment and natural resources locally and globally.		1. to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment	The IEEE code of ethics states one should ethically and sustainably design and develop while the NSPE code of Ethics does not touch on sustainability at all.

Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	2. to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems	The NSPE Canon says ones conduct should be to "enhance the honor, reptation, and usefulness of the profession" while the IEEE code of ethics does not which I would interpret as one should always be socially responsible with no agenda in mind.

Table 2: Areas of Responsibility

7.2 Project Specific Professional Responsibility Areas

Area of Responsibility	Importance to Project	Level of Performance
Work Competence	High: Our team wants to produce the best possible product since the goal of our project is to pass on our knowledge gained to future students and we want to provide them a good experience.	High: We have all completed the trainings provided to us or found through our research on DeepRacer in order to competently complete this project.
Financial Responsibility	Low: Our product will not be sold and will cost the end user nothing to use or participate in. The cost to create our project is only the bot and possibly some sensors which will all be bought from trustworthy sources.	High: Our team has only bought what we need for the project and the DeepRacer bot will be donated to either a class or a club at Iowa State at no cost.
Communication Honesty	High: It is important to communicate our progress on our project so that our advisor, professor, and TA will be able to provide assistance if needed and give feedback in order for us to create the best product.	High: Our team truthfully states our accomplishments and estimates to our advisor/client in each meeting or correspondence.
Health, Safety, Well Being	Low: We have little to no safety concerns with the DeepRacer bot as it will be functioning on track by itself and will cause little to no harm to our advisor/client if things were to go wrong.	NA: Our project should not have any negative effect on the health, safety, and well being of others
Property Ownership	Medium: We will most likely be using resources from Iowa State or the internet so we must make sure to credit everything properly and use resources ethically.	High: Our team has been respectful toward any of the resources given to us or found as well as respectful to any ideas presented to us by either our advisor or other team members.
Sustainability	Low: The DeepRacer bot should cause little to no harm to the environment and natural resources besides what was used to create the bot. We will not be buying multiple bots and if the bot were to be adopted into a club or course we would donate the bot to the purpose so it would be reused.	High: Our team has only bought what we need for the project and the DeepRacer bot will be reused in either a class or a club at Iowa State
Social Responsibility	High: The goal of this project is to benefit the students at Iowa State and possibly	Medium: We are still in the process of producing a product that will benefit the students at Iowa

	surrounding high schools by creating an opportunity for them to learn and develop Machine Learning knowledge and skills.	State, but we'll make sure that as we continue designing and developing our DeepRacer bot and code that its intent will remain as a product to enhance and expand the Machine Learning community at Iowa State in a fun and meaningful way.

Table 3: Project Specific Professional Responsibility Areas

7.3 Most Applicable Professional Responsibility Area

The most applicable professional responsibility area for our project is work competence, as being knowledgeable in the topic is vital for implementing it as a teaching tool in classes such as CprE288. Without work competence, the entire project falls apart as if we don't know how the project works, how will we plan on teaching it.

8 Closing Material

8.1 Discussion

Our DeepRacer team is currently ranked in the top hundreds of the AWS DeepRacer League in the world. While this was initially the goal, we have found a new direction to take Project Deep Learning in one that enables this foundation to become a potential test bed for several undergraduate courses.

8.2 Conclusion

With the progress we have made in the project itself, and the several end goal shifts and amendments - we've finally settled on developing a test bed for allowing for an expansion on embedded systems, AI, machine learning, and networking. While these topics may prove to be brash to introduce for a lab or two in an undergraduate course – there are several instances in which these topics can be deduced to simple, yet effective practices to expand the knowledge of the average undergraduate ECPE student at Iowa State University.

Thus far, we've been able to dive into AWS, learn about machine learning, and implement a system that employs embedded machine learning. In doing so, we've successfully developed a model that is ranked in the top 100 in the world after a few weeks of development. While our path so far has been mostly in the educational realm with a smaller number of technical/hands-on experiences, we have gained the respective perspective of AI and machine learning coupled with our personal undergraduate experiences we've been able to pivot of our initial goals of simply having a robot to race in the AWS DeepRacer league to building an experience that will assist our peers in discovering a different branch of computers.

With these developments, a few questions arise – the biggest being how will we arrive at our goals? With the varying target goals, will we be able to implement a successful testbed for several undergraduate courses? The truth of the matter is, while we recognize that there are courses dedicated to Machine Learning and AI, there aren't many experiences in embedded machine learning, nor do many students realize that these courses may be their passion. We are aiming to help push these students down the path of AI/Machine Learning regardless of whether it is with

embedded systems or other such systems – our plan to achieve this, sits on a few experiments. Firstly, we will need to gather more information on what is fully available as an undergraduate student. Secondly, we will need to compile all of the gathered information and see how we can scale the experiments we have been able to perform – so far, we are envisioning the DeepRacer being used similarly to the CyBot which are used in Introduction to Embedded Systems (Cpre 288).

8.3 References

"AWS deepracer - the fastest way to get rolling with machine learning." [Online]. Available: https://aws.amazon.com/deepracer/.

8.4 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc,. PCB testing issues etc., Software bugs etc.

8.4.1 Team Contract

Team Members:

1) Benito Moeckly 2) Jazz Jacobus

3) Caleb DeBoef 4) Jose Carlos Garcia

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    a. Mondays at 3 pm
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. Discord voice/text chat, and face-to-face.
3. Decision-making policy (e.g., consensus, majority vote):
    a. The majority vote with Dr. Rover to advise in the event of a tie.
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    a. Discord chat logs.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
    a. Everyone is expected to attend all team meetings except when not available.
2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

    a. Equal responsibility for all team members to fulfill team assignments, keep to timelines, and meet deadlines.
3. Expected level of communication with other team members:
    a. Communication is key, so all members should communicate at minimum weekly to keep everyone else updated.
4. Expected level of commitment to team decisions and tasks:
    a. Expect team members to attend weekly meetings and put in 4-6 hours in the project weekly outside of team meetings.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    a. Everyone is an equal team member and will have a hand in all parts of the project including client interaction, designing, and testing.
2. Strategies for supporting and guiding the work of all team members:
    a. Weekly meetings allowing members to discuss whatever they are working on or any concerns they have, so other members can provide feedback or help with whatever is needed.
    b. Set clear goals to help all team members understand their responsibilities
3. Strategies for recognizing the contributions of all team members:
    a. Keeping track of discord logs, code headers, GitLab commit logs.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
    a. Benito: Industry experience in unit testing, embedded systems, talking to clients, have already taken statistics 330.
    b. Jazz: Some experience with AWS, python, java. Has taken statistics 330 and math 207.
    c. Caleb: Electrical Engineering knowledge, MATLAB and Simulink experience
2. Strategies for encouraging and supporting contributions and ideas from all team members:
    a. Open team meetings to bring up and discuss any ideas with encouraging feedback
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
    a. Speak up. All criticism is accepted

Goal setting, Planning, and Execution

1. Team goals for this semester:
    a. Have a design that we are proud of ready for our senior design project. Learn about machine learning and embedded systems.
2. Strategies for planning and assigning individual and teamwork:
    a. All tasks developed and assigned at weekly team meetings
3. Strategies for keeping on task:
    a. Weekly updates in discord and through voice chat about our work, ask each other what we are working on.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    a. Verbal warning from team members after 1st offense
    b. Verbal warning and a meeting with Dr. Rover to discuss actions to correct
    c. Meet as a team with Dr. Fila to discuss moving forward
2. What will your team do if the infractions continue?
    a. Discuss options with Dr. Fila and/or TA
    b. Assign whatever is needed to available team members

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Jazzlyn Jacobus_____ DATE __12/2/2022_____

2) Caleb DeBoef_____ DATE __12/2/2022_____

3) Benito Moeckly_____ DATE __12/2/2022_____

4) Jose Carlos Garcia_____ DATE __12/2/2022_____

5) _____ DATE _____

6) _____ DATE _____

7) _____ DATE _____

8) _____ DATE _____