Project DeepLearner

Senior Design May '23

Team 10

Members: Benito Moeckly, Caleb DeBoef, Carlos Garcia, Jazzlyn Jacobus

Introduction

- **Problem**: ISU offers many classes on machine learning and AI but not much focus on machine learning for embedded systems.
- **Solution**: Create educational materials for introducing machine learning for embedded systems to current ISU embedded curriculum.
- Educational materials adapt the DeepRacer by Amazon Web Services, an electric car equipped for machine learning and creating autonomous AI models via reinforcement learning.

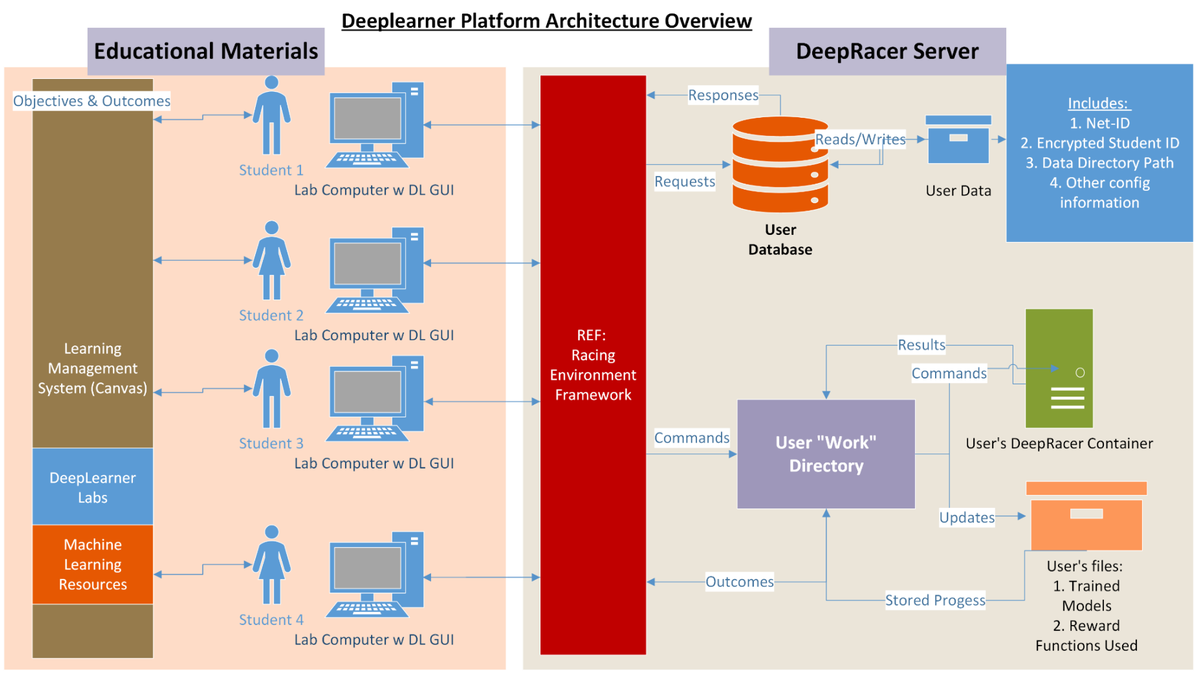
Overview

- Educational materials involve two aspects: documents and activities for labs and a platform for training AI models.
- Target users are ISU Students, Professors and TAs involved in embedded systems classes - specifically CprE 288.
- 2 Lab experiments which properly introduce machine learning to students
- We also created an infrastructure for training and evaluating AI models for the Amazon DeepRacer.
- Students are able to easily train models via reinforcement learning with their own reward function
- They can also view models as they train in a virtual environment and retrieve models for use in the physical DeepRacer

Methodolgy

- At first we investigated the DeepRacer by Amazon Web Services along with initial research on machine learning.
- We liked how easy DeepRacer was to use and it's interactive features.
- Due to mostly financial reasons it would not be practical to use in a classroom environment.
- We decided to focus a bulk of our project on creating a similar service that runs on a local network and researching reinforcement learning.

Implementation

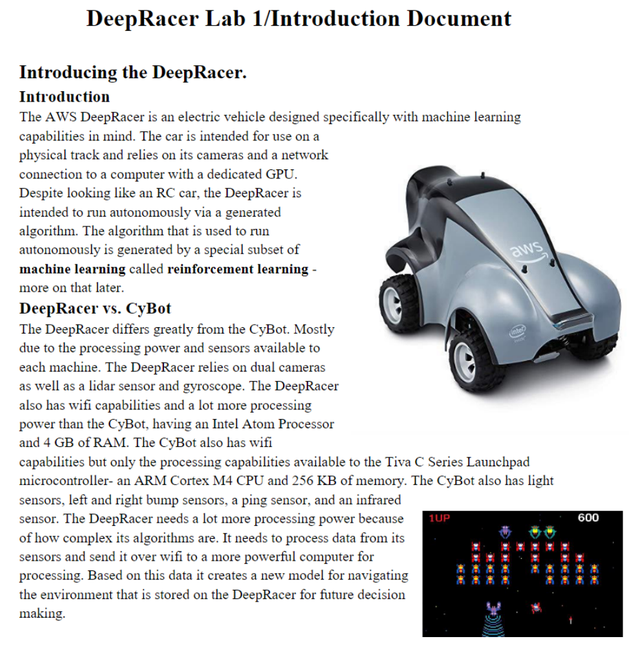


DeepRacer Server

- The DeepRacer Server contains necessary software for training AI models and managing multiple users
- Each user is given a Container which has all the software necessary for AI training.
- Container uses open-source software: Minio, RoboMaker, SageMaker
- Racing Environment Framework (REF) manages user's work directory.
- REF also communicates with the GUI to process user requests and store user information.

Educational Material:

- Labs give introductory explanation and experience with Machine Learning
- Keeps students engaged with clear and fun goals
- Provides additional resources for further learning
- Bridges the gap between low level hardware programs and ML algroithms



DeepRacer Lab 1/Introduction Document

Introducing the DeepRacer.

Introduction

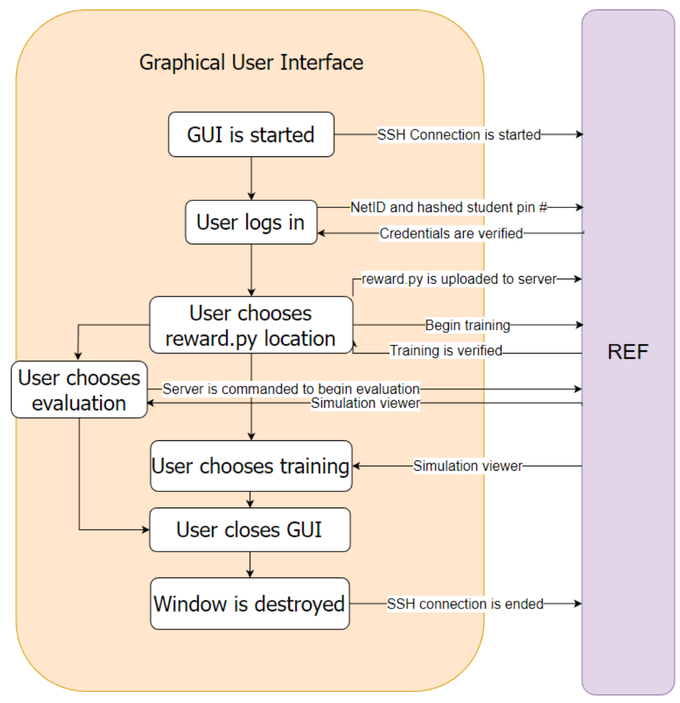
The AWS DeepRacer is an electric vehicle designed specifically with machine learning capabilities in mind. The car is intended for use on a physical track and relies on its cameras and a network connection to a computer with a dedicated GPU. Despite looking like an RC car, the DeepRacer is intended to run autonomously via a generated algorithm. The algorithm that is used to run autonomously is generated by a special subset of **machine learning** called **reinforcement learning** - more on that later.

DeepRacer vs. CyBot

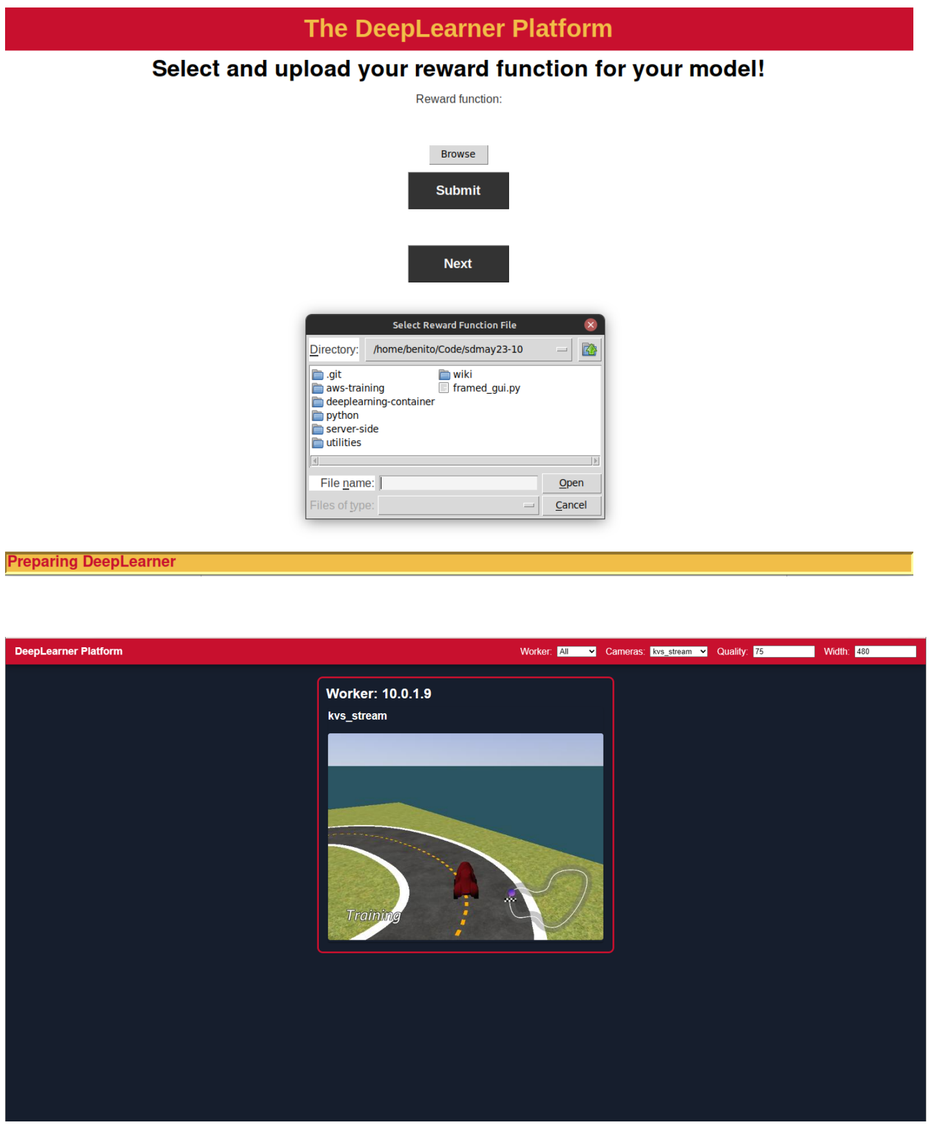
The DeepRacer differs greatly from the CyBot. Mostly due to the processing power and sensors available to each machine. The DeepRacer relies on dual cameras as well as a lidar sensor and gyroscope. The DeepRacer also has wifi capabilities and a lot more processing power than the CyBot, having an Intel Atom Processor and 4 GB of RAM. The CyBot also has wifi capabilities but only the processing capabilities available to the Tiva C Series Launchpad microcontroller- an ARM Cortex M4 CPU and 256 KB of memory. The CyBot also has light sensors, left and right bump sensors, a ping sensor, and an infrared sensor. The DeepRacer needs a lot more processing power because of how complex its algorithms are. It needs to process data from its sensors and send it over wifi to a more powerful computer for processing. Based on this data it creates a new model for navigating the environment that is stored on the DeepRacer for future decision making.

User Experience

- A GUI makes model training and evaluation easy for users.
- Limiting users to commands supported by the GUI also helps protect the backend infrastructure.
- Created using Python.
- Has user authentication to protect AI models and encrypts data sent to the REF.



Results



- **Front-end:** Able to upload custom reward functions, initiate and stop training, download trained models and view training from a web browser.
- Evaluation still is not fully implemented, but models can be imported to AWS for evaluation.
- **Server:** REF is up and working and the designed platform works on a server located in Coover Hall.
- Functionality includes training models, simulating the training environment, and also managing user resources.
- **Educational Tools:** 2 Lab docuemnts highlighting essential aspects of DeepRacer
- Highlights parameter estimation, the effectiveness of ML training, effect of time on training fidelity, and how the robot controls movements

Impact & Conclusion

- Similar projects are in production by AWS employees and people in the DeepRacer Employees, showing the demand for such platforms.
- Future projects could include bringing the platform to a custom robot, like the CyBot in order to make it a viable addition to CprE 288.