Collaborative Control of Autonomous Cars

Team Members:
John Vitali - jvitali2020@my.fit.edu
Brennan Pike - bpike2020@my.fit.edu
Isaya Danice - inyangira2020@my.fit.edu

Faculty Advisor:
Tom Eskridge - teskridge@fit.edu

Client:
Tom Eskridge, affil. Florida Institute of Technology

Progress Matrix

Task	Completion	John	Brennan	Isaya	To Do
Collaborative Control Python File	90%	5%	90%	5%	Make human inputs override AI inputs
Implement Reactive Dashboard	90%	90%	5%	5%	Implement CARLA libraries for speed calculation & display
Analyze "ScenarioRunner"	85%	5%	5%	90%	Fix ego vehicle not found issue, fix tensorflow module errors.
Implement Overtaking Scenarios.	30%	5%	5%	40%	Implement overtake python file and its corresponding xml file required for scenario runner.
Analyze Autopilot Agents	50%	5%	85%	10%	Improve the autopilot agents to respect road safety and find their destination
Create a New Viewpoint of Driver's Seat	100%	90%	5%	5%	None

Tasks Accomplished

Collaborative Control Python File:

Milestone 1's main purpose was learning the ins and outs of CARLA so that we could create the actual collaborative control file. This file takes input from the user and input from an autopilot agent to determine the car's actual actions. Presently, the way it does this is simple: it takes both inputs and averages them, with an additional caveat that braking cancels out any application of throttle. The collaborative control will eventually be more complex; the code is built so that the combination of control is easy to change without causing unintended issues.

CARLA has two agents for autopilot. One is built to travel to specific destinations, and isn't directly wired into the CARLA vehicles; the other is built to travel aimlessly and is more adaptable because of it. The first agent, due to not being directly wired into the CARLA vehicles, is easier to use for different purposes (e.g. the collaborative control program), so that is the one that the collaborative control currently uses. However, it has its own issues, and we may look into using the other agent to alleviate those issues.

Implement Reactive Dashboard:

In Milestone 1, the plan was to just overlay a simple PNG of a dashboard and have lights pop up. Upon further investigation and thinking, we decided it would be better if the dashboard was "reactive." The thought was to create a first person view of inside the vehicle, which would show a steering wheel, speedometer, etc. So, the original dashboard was scrapped and version 2 was started on. There were a few bumps in the road; the CARLA system uses its own library for vehicle, or "actor" data, so creating the dashboard right in the CARLA main program proved difficult. We decided to take a step back, and go back to the basics. We created a reactive dashboard in pygame on its own file, which can then be either implemented to the CARLA main function *or* can be run as a separate python file like "generate_traffic.py" in the PythonAPI files.

Analyze ScenarioRunner:

In this milestone, we aimed to utilize Scenario Runner, a tool designed for testing autonomous driving agents within the CARLA simulator environment. Unfortunately, we encountered issues during the installation and operation of Scenario Runner. We later realized that the versions were not compatible, we had 0.9.13, which was not compatible with our CARLA version 0.9.14. So we had to adjust our CARLA version to achieve compatibility and that meant installing Carla 0.9.13 and building it from scratch. The primary goal of scenario runner was to help us implement and test the overtaking feature which was one of the requirements we aimed at accomplishing during this milestone.

Analyze Autopilot Agents:

As mentioned in the section on collaborative control, CARLA has two different agents used for autopilot. The first agent is imported from a folder of rudimentary autopilot agents. It is built to travel to a specific destination in a very specific manner. Because of the agent's dedicated `run_step()` method, it is much easier to use and manipulate for collaborative control. However, its intended route is hardwired, meaning it cares more about staying on its path than it does about following road laws. As such, this agent will either need to be heavily modified or replaced.

The most likely candidate for another agent is CARLA's other autopilot agent, which is hard-wired into the `carla.Vehicle` data type. Because it's hard-wired into the vehicle, it doesn't have a dedicated step-running method like the other agent does, making it harder to use for collaborative control. (It may also need editing to focus on a destination instead of wandering aimlessly, but if this is the case, it is something that will be considered later in the project.) We haven't chosen which of these paths we will pursue yet.

Create a New Viewpoint of Driver's Seat:

We were also tasked to create another point of view from within the vehicle. This was to try and enhance the experience of "driving" the car. The default views were the side fender of the car, on the front bumper of the car, or a 3rd person view of the car. It was fairly simple, all that had to be done was to create another view within the code and find the correct x, y, and z coordinates to get the view that was wanted.

Member Contributions

John Vitali:

John's primary task was to work on the dashboard implementation. In Milestone 1, it was also his main task, but was unable to complete due to some complications. In Milestone 2, John continued to work on the dashboard, but decided to restart from scratch. In the original version, there was an error where the dashboard would open in a new Pygame window, and prevent the rest of the code in the main function from running. Although an easy fix, it was not to the standards that it should have been at. The dashboard was restarted from scratch and created in its own Python file which can then be implemented to the main function or run as its own separate entity like some of the other files that are used in CARLA.

John also focused on creating a new viewpoint for the driver. It was fairly simple. All that had to be done was find the code where the viewpoints are made, add a new line for a new viewpoint, and then find the correct x, y, z coordinates that would position the camera where it was supposed to be within the car. The car chosen for the viewpoint was the Ford Crown Vic taxi car because upon further investigation, it was the car that had the most complete interior because the team is unable to edit the vehicles, or "actors."

Brennan Pike:

Brennan's primary focus was on creating the collaborative control program, with analysis of the autopilot agents as a secondary goal necessary for completing the collaborative control program. The creation of this program was simple, but required careful edits to already-existing programs to combine the necessary components. Work on collaborative control will continue into the next milestone, but the focus will be pivoted to analysis (and likely alteration) of the autopilot program.

Isaya Danice:

Isaya was tasked with analyzing and installing the Scenario Runner. Once the installation and compatibility with the CARLA environment was achieved his role was to implement the overtaking maneuver and create scenarios that would test its efficiency. A first basic approach would be to utilize carla.TrafficManager and pass a parameter of 100 to its keep_right_lane_percentage(actor, perc) so that the agent always stays on the right lane. Get waypoints based on the agent's current location using Carla's inbuilt function get_waypoints() found in the carla.Map class. Waypoints hold information about the specific lane that they are in, thus we can check if a lane change based on the current waypoint is allowed using waypoint.lane_change(). Check if the left lane marking is a broken line then use radar sensors mounted on the agent to check for any obstacles to prevent collision then implement the overtaking maneuver. This should have been implemented and tested during this milestone but unfortunately, we haven't completed it since most of our time was spent in fixing compatibility and technological issues stemming from the installation phase.

Plan for Next Milestone

Task	John	Brennan	Isaya
Finish Dashboard	Implementation and demo (80-90%)	Testing, plus interface with collaborative control, if necessary (5-15%)	Testing (5%)
Finish Analysis of Autopilot	Analysis of hard-wired agent (20%)	Analysis of general agent (40%)	Analysis of general agent (40%)
Refine Collaborative Control	Testing and demo, plus additional implementation if necessary (40-50%)	Implementation and testing (40-50%)	Testing (10%)
Adjustments to Autopilot Agent	Testing and demo (15%)	Implementation and testing (50%)	Implementation, including overtaking (35%)
Fix Haptic Feedback Issue with Logitech G29	Testing and demo (15%)	Testing (5%)	Implementation and testing (80%)

Discussion

Dates of Meetings with Client

October 3, 2023 - Milestone 1 meeting.
October 17, 2023 - Meeting to show progress on Collaborative Driving & go over a few necessary programs needed (ScenarioRunner).
October 24, 2023 - Meeting to demo Collaborative Driving & Dashboard. Provide updates on ScenarioRunner research.

Faculty Advisor Feedback on each Task:

Collaborative Control Python File:

Enter feedback here

Implement Reactive Dashboard:

Enter feedback here

Analyze “ScenarioRunner”:

Enter feedback here

Analyze Autopilot Agents:
Enter feedback here

Create a New Viewpoint of Driver’s Seat:
Enter feedback here

Faculty Advisor Signature: ________________________ Date: _______________

Evaluation by Faculty Advisor

- Faculty Advisor: detach this page and return to Dr. Chan or email scores.
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

John Vitali	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Brennan Pike	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Isaya Danice	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: ________________________________ Date: ______________