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
Analysis of Autopilot Agent	80%	10%	45%	45%	Testing of current and future changes
Refine Collaborative Control	98%	5%	90%	5%	If necessary, develop new modes of collaborative control
Adjust Autopilot Agent(s)	20%	10%	50%	40%	Adjust system of local path decision to work with collaborative control
Create Highway Scenarios with ScenarioRunner	100%	5%	5%	90%	
Collaborative Overtaking	80%	20%	20%	40%	Work on blinking dots showing direction of lane change
Reactive Dashboard Refinement/Additions	95%	90%	5%	5%	Add mirrors to the side of the vehicle (this is nice to have, not a must have)

Tasks Accomplished

Analysis of Autopilot Agent:

The analysis of the autopilot was simpler than expected in some ways, but more complex in others. As described below in adjusting the autopilot agent, it was simple to understand how to make the agent roam randomly instead of going to a specific destination by making it just use its local planner algorithm. However, making the local planner cooperate with the user in collaborative control has proven to be more difficult.

Refine Collaborative Control:

The collaborative control schema as it was shown in Milestone 2, referred to henceforth as the split control schema, required additional changes to make it more usable. The first model created this milestone relied on passing control between the autonomous agent and the user, with the agent taking full control when there is no input from the user, and the user having full control whenever they give input.

From this schema and the split control schema, the final iteration of the collaborative control schema was created. This schema, referred to henceforth as the dynamic control schema, combined the ideas of the previous two schemas. When the user isn't giving input, the agent is given full control. When the user gives input, the control is split between the agent and the user. The key element is that the ratio between the agent's input and the user's input changes depending on the difference in input; if the agent gives input that opposes the user's (e.g. steering in the opposite direction or braking when the user is trying to speed up), the user gains control and the agent loses control. Conversely, if the agent gives control that matches the user's, the ratio returns to an even split. (The agent is never in more control than the user.)

Adjust Autopilot Agent(s):

The autopilot agent used for the collaborative control had two issues that needed to be addressed. Firstly, the agent would not adjust its route when collaborative control was used to deviate from its intended route. Secondly, the agent moved with an intended destination in mind. Initially, the team believed that the second problem caused the first. Fortunately, it turned out the agent *was* capable of moving without an intended destination using only a local route planner. The fix to the second problem was therefore easy; however, it did not fix the first problem. The local planner relies on a system of waypoints, and the waypoints aren't easy to update if the collaborative control changes the vehicle's direction.

The good news is that for the upcoming work for the end of Fall 2023 and start of Spring 2023, the team will be focused on using collaborative control on a long, straight highway. For this purpose, the autopilot agent will work fine. The finishing touches can wait until further down the line.

Highway Scenario with ScenarioRunner:

Several highway scenarios were implemented using scenario runner and later on collaborative control was incorporated into it. The first basic scenario implemented required no human input or intervention. We had to change the map to "Town4" which had an 8-lane highway. We had a stationary vehicle in front of our ego car (same lane) at a distance of approximately 100m and the ego vehicle is supposed to drive towards the stationary vehicle using the WaypointFollower class which by default follows forward waypoints indefinitely. Once the trigger distance (distance between the vehicles) is less than 30 a lane change request is initiated. Lane change is performed by following a waypoint plan to the target lane, the waypoint plan is calculated by a scenario helper function generate_target_waypoint_list_multilane(). We then had the stationary vehicle move but at a lower speed compared to our ego vehicle and the same results were obtained. Another scenario which used collaborative control was implemented. This scenario has a leading vehicle (in front of the ego vehicle) and three vehicles on the other lanes. The destination of the BasicAgent controlling our ego vehicle is set to the leading vehicle as we wanted the ego vehicle to always approach the leading vehicle. We also made a few changes so that the BasicAgent ignores vehicles thus collisions could occur. As the simulation runs, the distance between the ego vehicle and the leading vehicle is computed and once it is less than 30m, a beeping sound is activated, and a takeover request is displayed on the screen informing the user that they need to take control of the vehicle. Red dots are also displayed along the in-game steering wheel we implemented. These dots should blink towards the direction the user should perform the lane switch. The red blinking dots haven't been fully tested and is still work in progress. We later made the leading vehicle make an abrupt stop and the same takeover requests were observed.

Reactive Dashboard Refinement/Additions:

John's main goal throughout the semester has been making the third-person view of the vehicle first-person, with a "real" dashboard and steering wheel. Overall, it has been successful. In the beginning there were a few bugs that prevented the code from working properly, but it is now working in manual_control, and will be implemented into collab_control with the Logitech Steering wheel inputs as well. The team would like to implement some few "nice to have" items, like mirrors on the sides and a rearview mirror. There will also be flashing lights and a pop-up of some kind that will indicate when the human has to take control because the autopilot is unable to make a choice.

Member Contributions

John Vitali:

John continued to work on the dashboard implementation. In Milestone 2, it was restarted from scratch, and in Milestone 3 that same file was still used. It was fully implemented into “manual_control.py” and is now being implemented into “collab_control.py” with steering wheel inputs using the Logitech steering wheel. John also worked on adding the lights and pop ups on the screen when the user must take over. Although it does not seem like much, this milestone has been impactful to the “realism” in the simulator.

Brennan Pike:

The work with collaborative control during this milestone focused a bit less on the autopilot and a bit more on the last few modifications to collaborative control that were also necessary outside of the changes to the autopilot. Brennan’s focus was on alterations to the methods the collaborative control uses. Brennan created the two new schemas for collaborative control, as well as the change to make the autopilot wander. For the purposes of the plans for the upcoming milestone, these changes are sufficient, but more refinement of the autopilot will be required later.

Isaya Danice:

Isaya's contributions encompassed the implementation of all specified highway scenarios using scenario runner, including the creation of requisite XML files for scenario execution. Isaya also implemented an overtaking maneuver devoid of human intervention, ensuring a fully autonomous process without collaborative control. Isaya’s other tasks involved the integration of collaborative control into scenario runner so that collab_control.py uses the ego vehicle already created using scenario runner instead of spawning a new vehicle in a random spawn point. Additionally, Isaya implemented the necessary components for beeping and takeover requests within the collaborative control Python file.

Plan for Next Milestone

Task	John	Brennan	Isaya
Begin Real-World Testing (in simulator)	33%	33%	33%
Combine Collab_Control & Dashboard	45%	45%	10%
Setup Scenarios for Testing	5%	5%	90%

Discussion

Dates of Meetings with Client

October 31, 2023 - First Milestone 3 meeting
November 7, 2023 - Progress meeting
November 14, 2023 - Progress meeting
November 21, 2023 - Progress meeting & next steps

Faculty Advisor Feedback on each Task:

Analysis of Autopilot Agent:

Enter text here

Refine Collaborative Control:

Enter text here

Adjust Autopilot Agent(s):

Enter text here

Create Highway Scenarios with ScenarioRunner:

Enter text here

Collaborative Overtaking:

Enter text here

Reactive Dashboard Refinement/Additions:

Enter text here

Faculty Advisor Signature: _____ Date: ___**11-27-2023**_____

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: _____