OSPP Summer 21 Final Report

Project Description:

Project Name : Assisted Teleop(210910596)

Organization: ROS

Name: Anushree Sabnis

Scheme Description:

In mobile robot and autonomous vehicle navigation, there are situations where a human driver is required to intervene to get the vehicle out of a sticky situation. This can be both as a backup in case of autonomy failure as well as the primary function of the robot (e.g. telepresence robots). [Link]

Project Summary:

Time Planning - Scheme Progress:

Pre-OSPP:

Set up ROS2 and Navigation2 (Galactic) on a local machine and try out various suggested examples to get familiarized with the code and the processes. It was completed.

July 1 - July 15:

Requested mentor to set up periodic meetings to guide throughout the project. Steve graciously arranged a weekly meeting to discuss projects, issues and next goals. My next focus was to research various ways Assisted Teleop could be implemented. I proposed this to that end :

- Proposal 1
- Proposal 2

July 16 - July 30:

Upon presenting the second proposal, My mentor opened up a ROS Discourse Discussion on how to implement Assisted Teleop. There were two ways which were discussed : pure collision rejection and pure collison avoidance. The former would reduce velocity till the robot was always **N** seconds away from collision while the latter was finding new collision-free velocities in the general direction of the

command. The former method was preferred by the community . Link to proposal 3

July 31 - August 15:

Prepared some demo code for footprint projection. This helped me determine which frame the projection should be carried out from, and also to test out the projection formulae.

Video demo : Link

August 16 - August 31:

Starting discussions on how exactly assisted teleop should be integrated in the Navigation2 framework. As advised by my mentor, Assisted Teleop could be integrated as a Controller / Recovery and as an Independent Server. To that end, I implemented a templated library for Assisted Teleop.

September 1 - September 15:

Carrying ahead with previous work, I focused my attention on adding Assisted Teleop as a Recovery Plugin to Navigation2. This was successful. Upon review by mentor, fixed some bugs and improved code readability.

September 15 - September 30:

Navigation2 planned to move from **Recovery Servers** to **Behaviour Servers**, which would entail working with Recovery-exposed Action Servers. I refactored my code to implement a separate Action Server for Assisted Teleop, getting rid of cyclic update functions as used previously to

gain more control. In the last week, I merged the server class and the Assisted teleop class into a single class/object and kept periodically testing the code, suggesting improvements such as adding more parameters to provide more user control.

September 30 - October 22:
2nd Evaluation Phase

The Accomplished Work - Project Output:

Recovery Server:

- Added a separate action server for assisted teleop
- Removed the cyclic onCycleUpdate and OnRun functions
- Moved the Server and Recovery Plugin within the same **Assisted Teleop Class**
- Thus, developed a Recovery-exposed Action Server
- Added timeout functionality to Action Server based on defined goal in Assisted Teleop Action

Assisted Teleop Action:

8 lines (8 sloc) | 172 Bytes

```
1  #goal definition
2  builtin_interfaces/Duration time
3  ---
4  #result definition
5  builtin_interfaces/Duration total_elapsed_time
6  ---
7  #feedback
8  builtin_interfaces/Duration time_left
```

Goal : Time to run Assisted Teleop
Feedback : Time elapsed since starting Assisted Teleop
Result : Time left

Assisted Teleop Plugin:

- Created plugin to carry out Assisted Teleop Behavior
- It currently takes a Pure Collision Rejection approach to Assistive Teleop, as mentioned in the linked discourse topic.
- In this, a teleop input velocity command is tested for collisions by projecting its footprint till N seconds in the future
- If a collision is detected within N seconds, velocity is scaled down so the robot again remains N seconds away from a collision
- Added four new parameters : projection_time, linear_velocity_threshold, input_vel_topic and cmd_vel_topic
- projection_time : Maximum allowed time to collision (N)
- linear_velocity_threshold : Upon collision detection within projection_time, if the scaled-down velocity of the robot is below this threshold, the robot is stopped.
- cmd_vel_topic : Assisted Teleop publishes velocity to this topic.
- input_vel_topic : Assisted Teleop subscribes to this topic to receive velocity commands

Problem and Solution:

- **Unfamiliar Codebase:**

Navigation 2's Codebase was very different from the other projects I had done before. While I had used ROS and the Navigation stack extensively before, taking a dive into the code behind the scenes was a challenge. It took me a week or two to understand the flow and good coding practices that should be followed while writing code. I practiced and tested the already implemented code to familiarize myself with the code.

- **Hardware Limitations**:

Running Gazebo on my laptop often caused performance issues, which slowed down testing/debugging. Nevertheless, the required tasks were completed in time

Development and Quality:

Navigation2 is a vast organization with passionate Maintainers who have a great zeal for it. Watching them work motivated me to give my best. In these 3 months, I have worked towards achieving the best output possible. I have tried to keep the codebase as simple as possible, and I plan to add further User guidelines

Future Work :

1. Update README to include Assisted Teleop as a Recovery
2. I need to create a Behaviour Tree Node for Assisted Teleop

3. I need to potentially implement more ways for carrying out Assisted Teleop, exploring a mixture of Pure Collision Avoidance and Rejection approaches.

Communication and Feedback:

My mentor Steve Macenski was very supportive and helped me throughout the period. I could reach him anytime through the community channels and he also engaged all mentees for a weekly sync meet. He was always available to clear any doubts and elaborate on any implementation. We were encouraged to attend the Technical Steering Committee meets of Navigation2, to widen our horizons. I learnt a lot about the Navigation stack, robotic software develeopment and most of all, team work and open source collaboration under his guidance.

Source Code:

https://github.com/ros-planning/navigation2/pull/2575