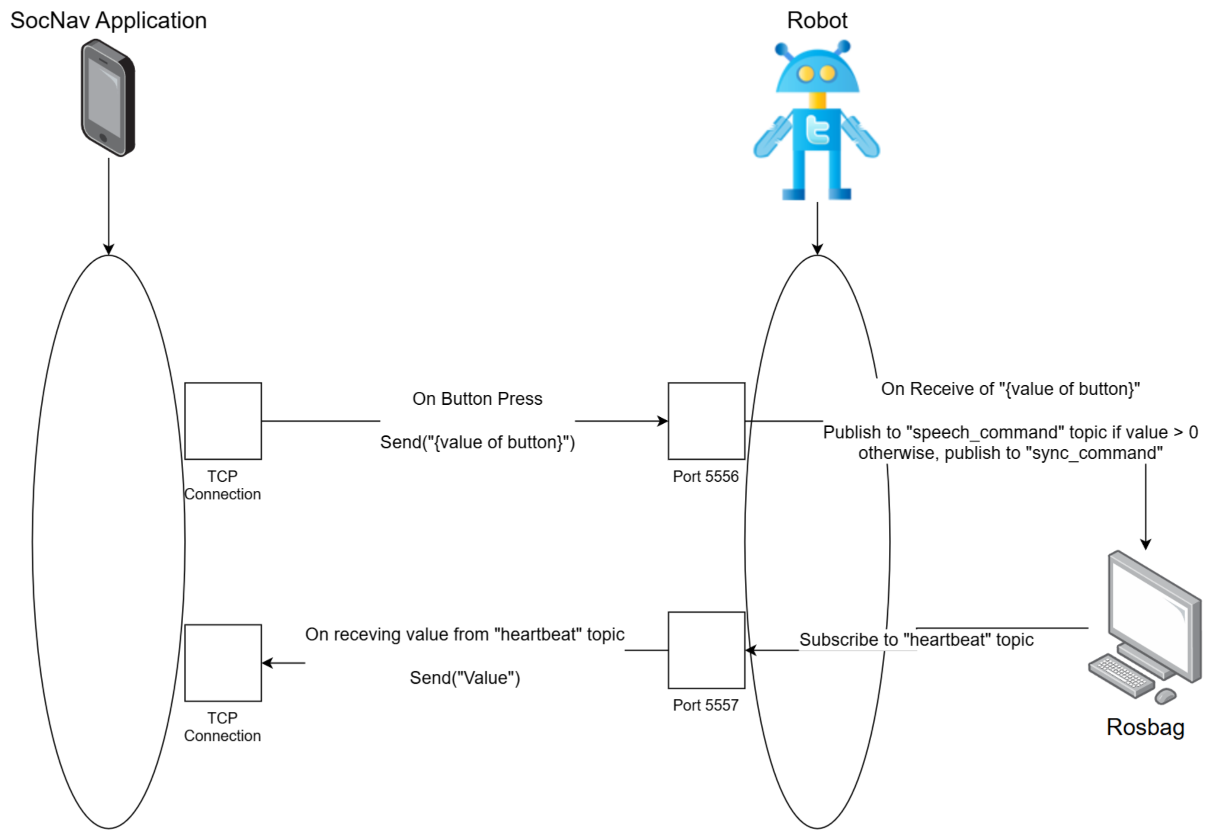
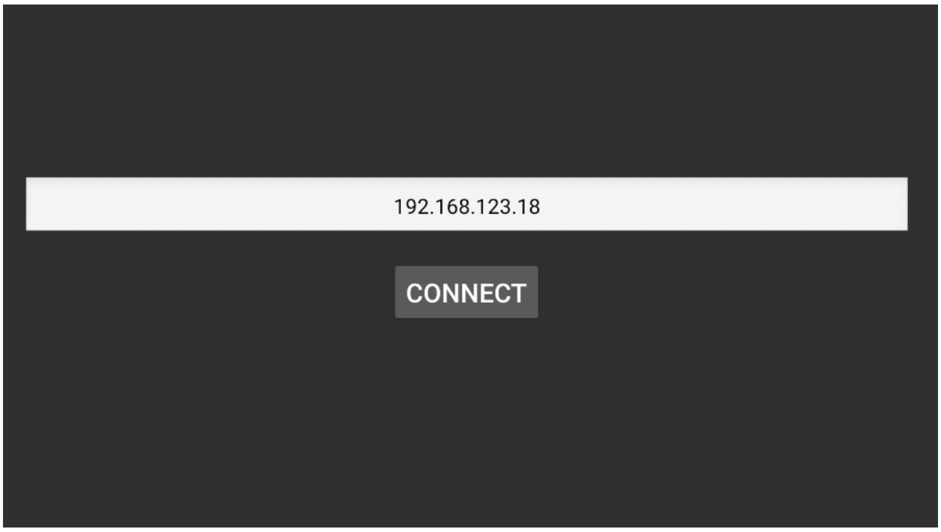
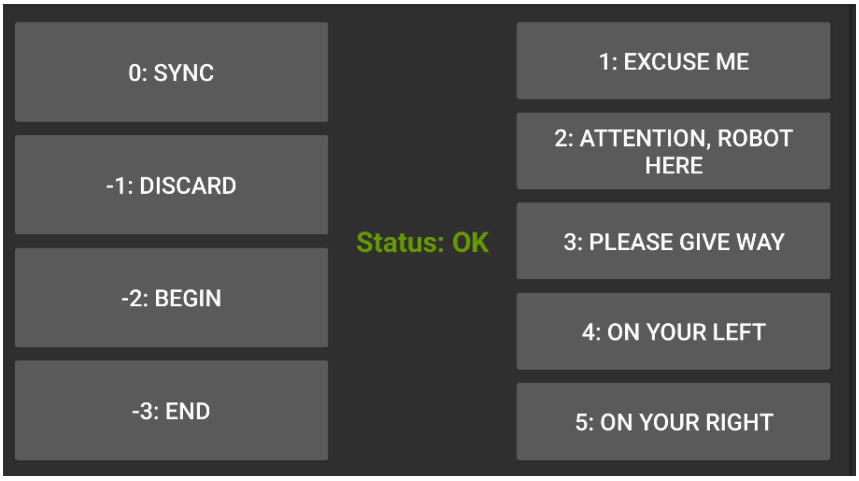

SocNav Application
Robot
TCP Connection
On Button Press
Send("{value of button}")
Port 5556
On Receive of "{value of button}"
Publish to "speech_command" topic if value > 0 otherwise, publish to "sync_command"
On receving value from "heartbeat" topic
Send("Value")
Port 5557
Subscribe to "heartbeat" topic
TCP Connection
Rosbag

The overview of the communication between the application and the robot is as shown above. First, the robot will wait for the incoming client connection in port 5556 and 5557. The smartphone must be connected to the Wi-Fi router of the robot to continue. Then the application will open 2 TCP connections concurrently with the robot, one for sending voice command output and marking trajectories, and the other for keeping track of any sensory failure from the robot. In addition, the Bluetooth speaker must also be connected to the smartphone, so that the output voice command of the application could be heard from the robot.



192.168.123.18

This is the main page of the application. Firstly, the operator will key in the IP address of the robot. Upon clicking the “CONNECT” button, the button is disabled and the application will try to establish 2 TCP connection to the given IP address and port 5556 and 5557 respectively. Depending on whether the application successfully established connection with the robot, it will show “CONNECTED” and change to the button layout, or “Error has occurred. Please try again” and re-enable the “CONNECT” button for the user to try again.



1. Synchronisation of Robot and BEV cameras (Sync (0) button)

This command is used to mark the point used to perform position synchronisation between the robot and BEV cameras. Upon tapping the “Sync” button, the assigned value (0) is published to the topic “sync_command” in the ROSbag. The positions of the BEV cameras will be estimated with respect to the image with a timestamp closest to this sync command.

2. Trajectory Marking (Begin (-2), End (-3), Discard (-1) buttons)

Before starting to record a trajectory, the teleoperator will tap on the “Start” button, which the assigned value (-2) will then be published to the topic “sync_command” in the ROSbag. At the end of recording for a particular trajectory, the teleoperator will tap on the “End” button, which the assigned value (-3) will then be published to the topic “sync_command” in the ROSbag. The particular trajectory will be implicitly defined by the teleoperator. The published value of each specific synchronisation messages would aid in slicing useful trajectories from the bag file and discarding the rest.

During a data collection, unforeseen events (someone standing right in front of the robot to take a picture, or the teleoperator executing a non-socially compliant manoeuvre by mistake/as a jerk reaction) may happen, and the teleoperator should mark such events to discard that section of the recorded data. The teleoperator can tap on the “Discard” button and the assigned value (-1) will then be published to the topic “sync_command” in the ROSbag. However, the “Discard” button should be only tap on after the “Start” button has already been tapped, and before the “End” button is tapped. These parts of the data will not be included in the dataset. Note that after tapping on the “Discard” button, the teleoperator should not tap on “End” button anymore.

3. Voice Command (Excuse me (1), Attention, robot here (2), Please give way (3), On your left (4), On your right (5) buttons)

During the recording of a trajectory, when the teleoperator encounter a scenario where they think that verbal command is useful, they may tap on one of the designated buttons on the interface. Then, the voice command from the robot will be played on the speaker mounted on the robot, allowing us to simulate natural human-robot interaction. The phrases that are said by the robots will be the same as shown in the button. Upon tapping on the any of the voice command button, the assigned value will then be published to the topic “speech_command” in the ROSbag. The published values are as follows: “Excuse me” (1), “Attention, robot here” (2), “Please give way” (3), “On your left” (4), “On your right” (5). The scenarios to use these commands will be dependent on the teleoperator.

For example, if the teleoperator identify that a pedestrian looks like they are not aware of the surrounding, then he could tap on the “Attention, robot here” button to alert them to be more mindful. The voice command used are generated from an online voice generator, to mimic a robotic voice that is also clear.

4. Status of Robot

To check the status of the robot, I have it displayed in between the buttons. The robot published a discrete value to the topic “heartbeat” for which the application subscribed to. The application will keep track of the timing of the last published “heartbeat” message. In the event that difference between the last published message and the current time is more than 10 seconds, we defined the robot to have a sensory failure, and hence we alert the teleoperator with an onscreen status showing “Not OK” in red. This would indicate to the teleoperator immediately during teleoperator that something has went wrong with the robot, so that they can immediately stop recording and check the issue of the robot, without wasting precious time on collecting data on a faulty robot.