ENPM662- Introduction to Robot Modelling

Project 1- CAD Modelling and Simulation using Gazebo

- **Objective**: Designing a specific Robot model with provided constraints and specifications in Solidworks, simulating the model in the Gazebo in a map, and using teleop to navigate the robot by supplying input to controllers and transmission.

- **Steps followed**:-
  1. Building a CAD Model using Solidworks on citrix workspace with given specifications for the robot such as chasse length*width as 38*20 inches and thickness as 3 inches and wheel diameter as 8 inches.
  2. After designing the parts and building the assembly, export the assembly file as a urdf folder using the inbuilt URDF Exporter tool in citrix workspace. While exporting the URDF we the following steps were performed:
     a. Mating all the parts to the base_link (chasse) and the base_link origin to the main assembly origin.
     b. Adding base_link as Parent link and introducing four child links, namely Front_Peg_Link_Left, Front_Peg_Link_Right, Back_wheel_Link_Left and Back_wheel_Link_Right. The Front two wheels will be added as child links to Front Peg Left and Right respectively.
     c. Providing angular constrains for steering of front wheels which were -0.5 Radians to +0.5 Radians which is approximately -/+30 degrees.
     d. Defining the motion of the steering joints as revolute and other joints as continuous.
     e. Previewing and exporting the URDF Folder.
  3. Moving on to the next step which is exporting the folder into src file of catkin_ws and build all the necessary packages inside the source directory by giving the command catkin_make.
  4. In order to proceed make sure if the URDF folder defines all the links in the order intended by using following command check_urdf sfinal.urdf in the sfinal/src/urdf.
  5. Once the intended results are obtained from previous step, we modified the urdf file. A dumy_link and a dumy_joint are created in the robot inside urdf file in order to run the model in the Gazebo world.
  6. The next step is to launch the model in an empty gazebo world to make sure the links and joint are aligned perfectly and the model is at desired position and orientation. The following command is used to launch the model in Gazebo:- roslaunch sfinal gazebo.launch.
  7. The model is now perfectly aligned and placed in the gazebo environment. Now in order for in to move in the simulated world, we provide transmissions to the joints, such as velocity commands for the wheels and effort_controllers for steering. These transmissions are added inside the urdf file. For the controllers to work we need to install controllers package in the Ubuntu system.
  8. Add the Lidar and .dae filed in the urdf folder in the src of project_package.
  9. After carefully editing the urdf file and adding the necessary data, create a xacro file by using following command: - rosrun xacro xacro –o sfinal.urdf xacro_file.xacro.
  10. The next step is to write one python code using the provided template to control the robot in the environment using teleop and another code for publisher and subscriber data.
  11. The final step is to launch the robot model in the given simulated Gazebo environment and navigate it from the origin to a point through a path using the teleop. While performing the teleop we can see the subscriber and publisher receiving and sending data respectively on the terminal.

- **Challenges**:

  - The first major challenge we face was during the exporting of the URDF file. When the origin of the chassis was not mated with the origin of the assembly file as we discovered later when spawning the model in gazebo resulted in a broken disassembled format.
  - The second major issue we faced was the editing of the XACRO file. Defining the paths and various attributes of the ydlidar urdf and ensuring the parameters that were passed were correct. As the combined urdf file had errors that if the paths were not set correctly.
  - The third major issue we faced was the launching of our model "sfinal" in the given world. This had various issues from not spawning the model to not loading entirely. We solved these issues by ensuring that we catkin_cleaned && make was executed and then sourced. Our initial iterations of the model had problems with the joint types that were defined as we had not set limits to the front steering wheels. Which was corrected in later iterations.
  - This fourth major issue was implementing the controllers to the transmission blocks that were defined. The major problem that we faced was the front-wheels steered independently and thus required inverted commands to ensure the model could steer correctly.
  - The last issue we faced was the tuning of the controls in the yaml file. As the initial values enter for the PID gains were not set correctly. This led to driving the model to its destination an arduous task, as it would either over or understeer. We corrected this through trial and error and found ideal values for the PID gains.

- **Contribution**:
  - I edited the urdf file that was exported from SolidWorks, the addition of dummylink and the addition of the transmission blocks to control the model.
  - I worked on editing the XACRO file and ensured the attributes of the ydlidar urdf and paths were set correctly, and the exported file had all the correct attributes and parameters.
  - I worked on creating the python script that controlled the robot. The mapping of the correct inputs to the correct publisher to control the robot.
  - I worked on debugging the model to remove errors during launch.
  - We created the publisher and subscriber scripts to display the relationship between the nodes.

Video Links:

https://drive.google.com/file/d/1GT0-SFKPvV2CSbJA8AugnRGKLKXAMA6I/view?usp=sharing

https://drive.google.com/file/d/1kCI_yuIfTEN0OudXjzvMXwwci5s1bL-m/view?usp=sharing