Reusable Open-Source Virtual Twin of the 4-wheel SumitXL robotics platform

Abstract

Our contribution for the SofaAward is an open source virtual twin for the SummitXL, a 4 wheels robotics platform made by https://robotnik.eu/. The virtual twin is implemented following the “Prefab” principles to allow re-usability in third party projects.

Our virtual twin implementation key features:

- modeling of the differential control of the four wheel for realistic simulations.
- precise contact modeling for both the robot as well as the ground-to-wheels interaction for complex simulation scenarios.
- a Sofa.Controller to control the virtual twin using ROS2. This ROS2 node follows the official SummitXL API so the twin can be controlled in the same way as the real robots.

I. Introduction

Within the field of robotics, having realistic virtual twins for robots is important for several reasons. First it allows users to develops and test the behavior of the robots in simulation before going to real world for field experiments. Second it allows them to experiment with the robot's in situations that cannot be easily made in real-life (eg: simulating errors or mechanical problems). Finally having a virtual twin within realistics environments is more and more needed to generate training datasets needed for AI based/machine learning controllers.

In this work, our contribution is an open source implementation of a virtual twin for the differential 4 wheel SummitXL robot. The SummitXL is made by

[Robotnik](#) society. Common applications for the SummitXL are: R&D, remote control, surveillance, dangerous areas and military.

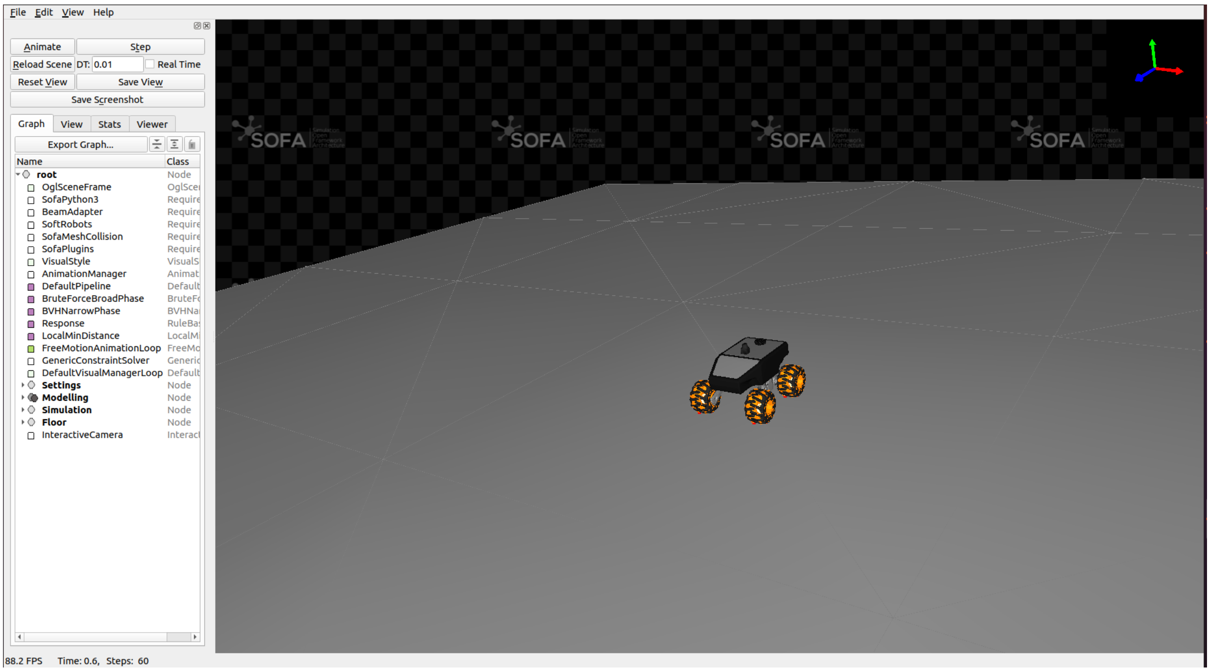
The SummitXL is shipped with a virtual twin implemented in Gazebo. This virtual twin has a ROS1 compatibility, a geometric movement model which does not reproduce the behavior of the differential wheel controller. In addition there is no wheel-to-ground contact model and the friction is not taken into account.

In order to

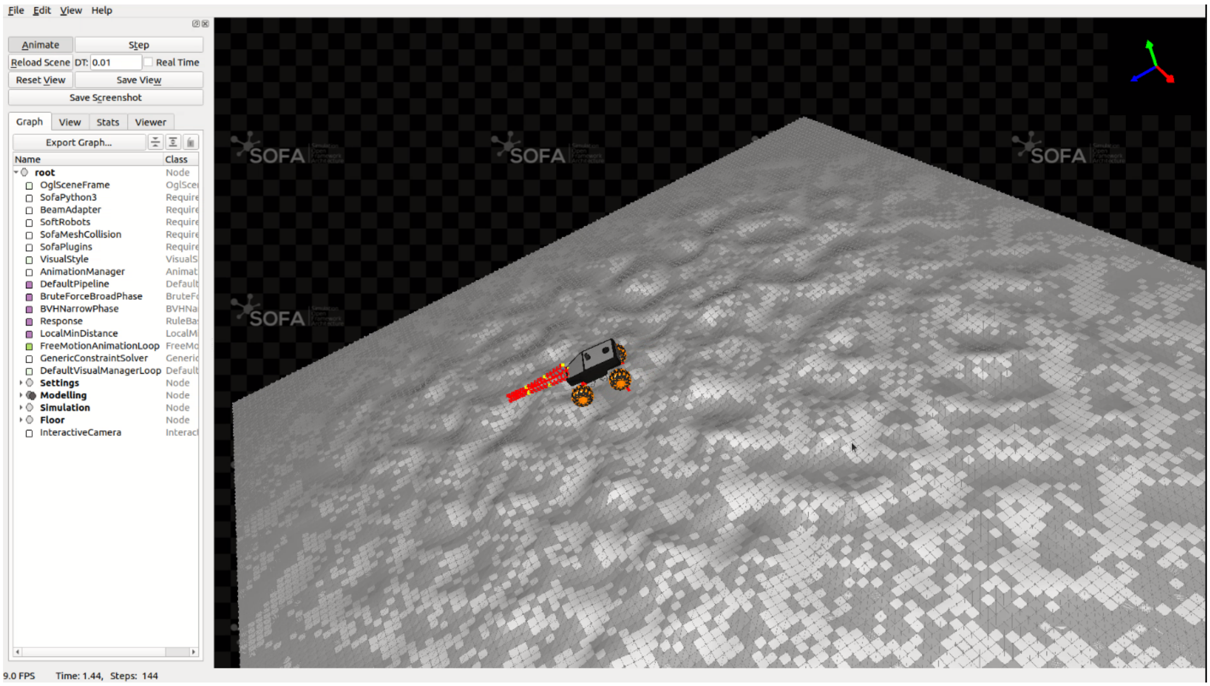
II. Using the robots

Thanks to the keyboard controller, we can control the virtual twin by sending it speed commands. Thus we can choose at which linear and or angular speed and in which direction the robot advances or turns. *The virtual twin incorporates a contact model of the wheels with the ground, which allows for a more realistic simulation and thus makes it possible to simulate several use cases :*

- *robot on flat ground*



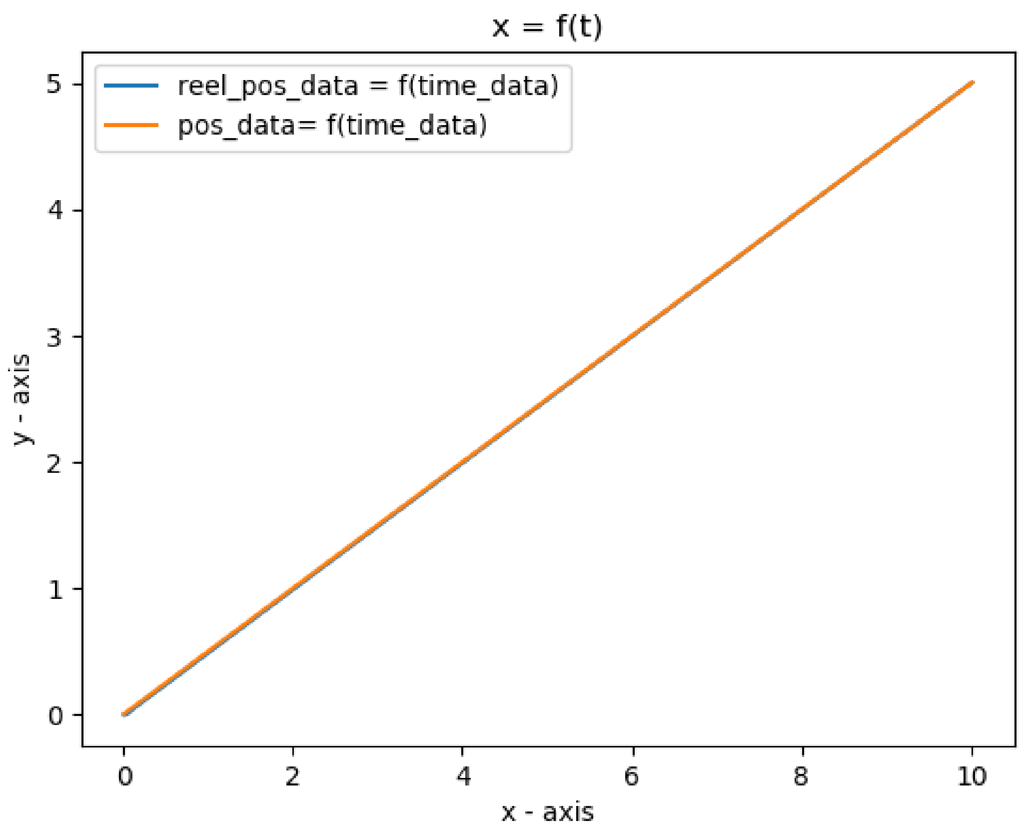
- *rocky ground*



III. Implémentation and simulation précision

For the implementation of the robot, we used STL files describing each part of the robot.
Since the robot consists of 4 omnidirectional Mecanum wheels, a kinematic model was used which indicates the relationship between the linear speed and the rotational speed of the robot with the rotational speed of the 4 wheels.
In order to validate the accuracy of the simulation, we tested 2 cases:

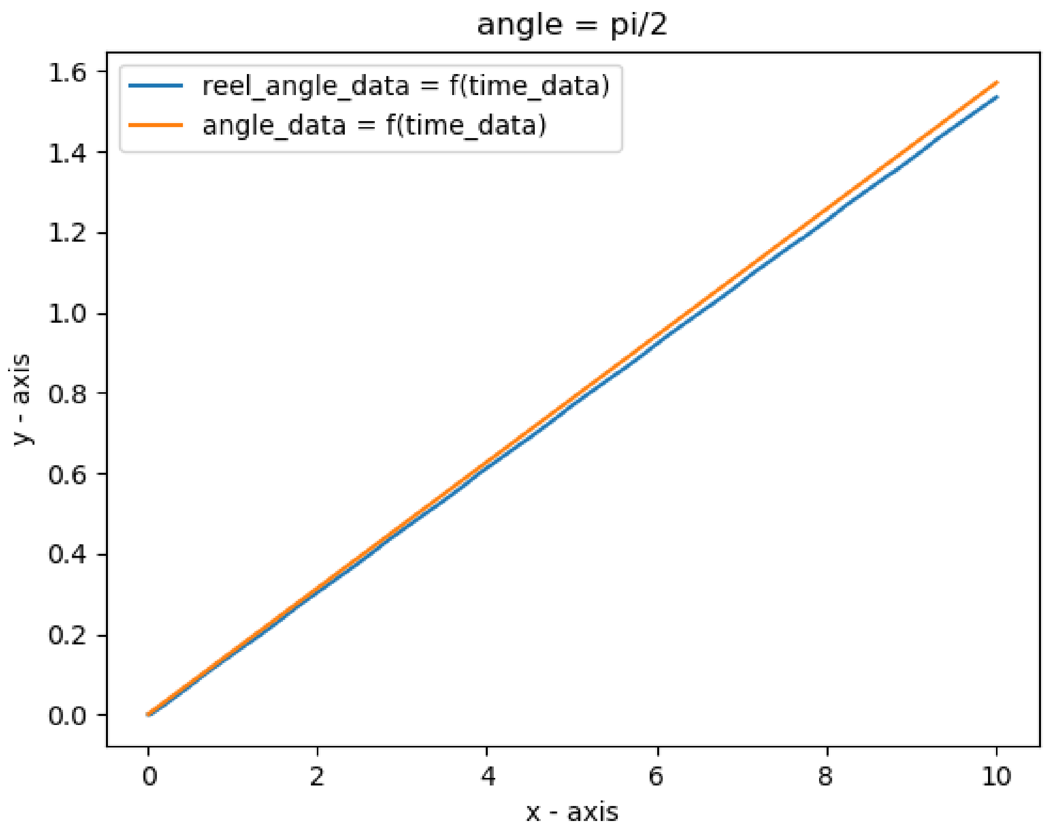
- With a speed of 0.5 m/s, moving the robot forward for 10s is supposed to cover a distance of 5m.



In blue the actual distance traveled by the virtual twin and in orange a distance of 5 m traveled in 10 s. We can see that the 2 curves coincide. We calculated the error between the 2 curves.
We have calculated the error between the final position of the virtual twin and the final value that the robot will have traveled with a speed of 0.5 m/s for 10 s and we obtain an error of 0.0004 which is rather low as shown in the figure below

```
final position =  5.0045864510056 traveled distance =  5.0049999999999155 Erreur =  0.0004135489943157822
```

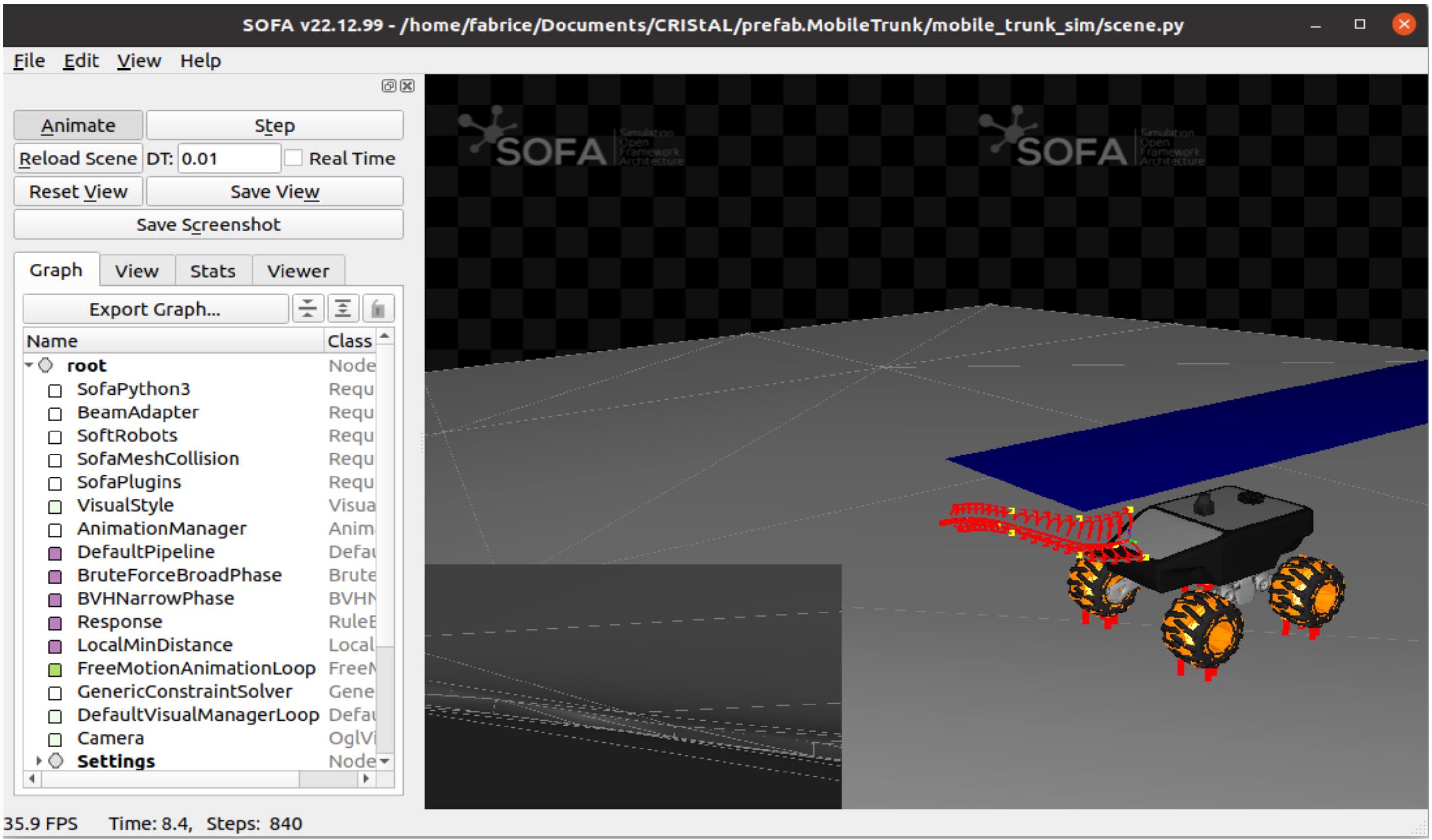
- With an angular speed of 0.15 rad/s and a linear speed of zero, the virtual twin is rotated for 10s so that it makes a rotation of pi/2. As before, we compare the curve of the orientation of the virtual twin with that of the angle of rotation which gives us the figure below:



In blue we have the orientation of the robot and in orange the value of the angle at each instant during these 10 seconds. We can see that the 2 curves are almost merged

IV. Re-using & extending the SummitXL

By following Sofa Prefab guidelines, we are able to attach other objects to the virtual twin quite easily. To do this, we just had to create a tie knot on the basis of the virtual twin on which we mechanically tie the trunk as shown in the figure below.



SOFA v22.12.99 - /home/fabrice/Documents/CRIStAL/prefab.MobileTrunk/mobile_trunk_sim/scene.py
File Edit View Help
Animate
Step
Reload Scene DT: 0.01 Real Time
Reset View
Save View
Save Screenshot
Graph View Stats Viewer
Export Graph...
Name Class
root Node
SofaPython3 Requ
BeamAdapter Requ
SoftRobots Requ
SofaMeshCollision Requ
SofaPlugins Requ
VisualStyle Visua
AnimationManager Anim
DefaultPipeline Defau
BruteForceBroadPhase Brute
BVHNarrowPhase BVHN
Response RuleB
LocalMinDistance Local
FreeMotionAnimationLoop FreeM
GenericConstraintSolver Gene
DefaultVisualManagerLoop Defau
Camera OglVi
Settings Node
SOFA
SOFA
35.9 FPS Time: 8.4, Steps: 840