Avrora Unior Car-like Robot in Gazebo Environment

Ksenia Shabalina[1], Artur Sagitov[1], Kuo-Lan Su[2], Kuo-Hsien Hsia[3], Evgeni Magid[1]
[1]Laboratory of Intelligent Robotic Systems (LIRS), Intelligent Robotics Department, Higher Institute for Information Technology and Intelligent Systems, Kazan Federal University, 35 Kremlyovskaya street, Kazan, Russia Federation
[2]Department of Electrical Engineering, National Yunlin University of Science and Technology, Taiwan
[3]Department of Electrical Engineering, Far East University, Taiwan
E-mail: ks.shabalina@it.kfu.ru, sagitov@it.kfu.ru, sukl@yuntech.edu.tw, khhsia@mail.feu.edu.tw, magid@it.kfu.ru
http://kpfu.ru/robolab.html

Abstract

Experiments are valuable tool of robust control algorithms design, but experiments tend to be expensive. In order to conduct thousands of complex experiments with autonomous car navigation algorithms it is safer and cheaper to start algorithm verification within a simulation, relying on a proper robot model, which preserves physical properties of underlying objects. In this paper we present the design of Avrora Unior mobile robot model, which is a Russian car-like robot with Ackermann steering geometry, and describe the process of modeling its kinematics and dynamics. Robot model was designed within open source robotics framework ROS for Gazebo simulator.

Keywords: car-like robot, non-holonomic robot, simulation, Avrora Unior, Gazebo, ROS.

1. Introduction

In the past decades a broad use of simulations became an essential part in robotics research field. We highlight the following reasons of using simulation in robotics: real experiments with robots are expensive while simulations are cheap; simulation experiments take less time than real experiments and are always safer than experiments in real world; simulations allow testing novel concepts and algorithms even if required hardware is not available for a user; simulations help to quickly detect and correct gross errors in algorithms; finally, simulations could provide fairly reasonable testing of experimental setups. Those reasons apply to all robot types and tasks.

A simulator became a very progressive tool that can be used to reproduce complicated environments (e.g., water); they can use user-defined physics with dynamic changes, which makes it possible to construct and use a robot model behaving very close to a real robot. Gazebo is the one of the most popular 3D simulator of robots, which was successfully used to simulate UAVs[1] and UGVs[2,3]. It could be used as an environment for performing various experiments: testing basic robot motions, path planning and collaboration with other robots[4], manipulation[5], modeling USAR scenarios[6] etc. Other specialized simulators (e.g., UWSim[7]) could help simulating water environments for AUV robots.

In our research we develop autonomous car navigation and locomotion algorithms for mobile robot Avrora Unior (Fig.1), and use Gazebo simulator to construct the robot model and to test control algorithms before performing experiments with the real robot.

2. Avrora Unior Car-like Robot

Avrora Unior robot is a car-like mobile robot that was created by Russian company Avrora Robotics (Fig. 1). Originally, this robot was designed for students' training in the field of autonomous driving, and such purpose clearly influenced technical design of the robot. Due to a small size and relatively low weight (~43,5 kg) of the robot, it is a safe solution for testing and validating

algorithms for autonomous driving both in indoor and outdoor environments. As it is a small robot, it could be easily moved around the laboratory room by two students. Moreover, Avrora Unior provides Ackermann drive mechanism that is used in real cars. Table 1 and Table 2 show linear dimensions and weight parameters of the robot, respectively. The robot consists of body panels (robot shell), chassis (metal backbone), power source (lead acid battery), sensors and control unit (laptop Acer TravelMate P2). Another significant advantage of this robot is an open source software and off-the-shelf hardware.

Hardware could be easily replaced, upgraded or new elements could be retrofitted (e.g., sensors). The robot has preinstalled Linux Ubuntu 16.04 LTS operating system with ROS Kinetic. The vanufacturer also provides open source steering and drive control packages.

Fig. 1. Avrora Unior car-like robot.

Table 1. Robot dimensions.

Dimension	Dimension (m)
Rim diameter	0.204
Wheel diameter	0.26
Wheel width	0.11
Lateral length of the entire robot	1.112
Robot height	0.57
Robot width	0.65

3. Robot Operating System Environment

Robot Operating System (ROS) is a popular open source framework for collaborative robot software development. The framework is composed of atomic units (packages), which aim to solve particular problems in domain or provide particular functionality for a certain robot(s). All existing packages in ROS could be improved by a user and a new package could be added and described in a special knowledge base.

Table 2. Weight of Avrora Unior robot parts.

Element	Weight (kg)
Wheel	0.9
Shell of the robot	8.5
Cover	3.0
Hokuyo weight	0.16
Kinect weight	0.45
The entire robot	43.5

Gazebo provides an open source 3D simulator integration with ROS that can properly simulate a various types of existing robot models (including AUVs, UGVs, manipulators, etc.) and allows creating custom robots. Gazebo evaluates world physics and a robot model interaction. Developers can test their algorithms with a proper robot model in a virtual environment a desired number of times, modify the environment (e.g., add or remove obstacles) to explore algorithm's strengths and weaknesses before conducting field experiments with expensive equipment. In addition, Gazebo provides customizable plugins that simulate various types of sensors. For our robot model we used standard sensor plugins after a number of necessary adjustments. Gazebo supports robot model in SDF and URDF formats and allows importing STL and DAE meshes for objects.

4. Gazebo Robot Model

This section describes the process of modeling. We present the architecture of the model package, discuss key elements and their characteristics, cover joint geometry of the model and sensor specifications.

4.1. *Architecture*

Using *avrora_ros* name for our metapackage we decomposed the model into several individual packages: *avrora_description, avrora_gazebo, avrora_control*. This decomposition of the model provides a convenient and effective division of main functional parts of the robot model. Package *avrora_description* contains general description of the model for ROS framework: robot URDF (Universal Robotic Description Format) representations, component meshes and materials of the model. Package *avrora_gazebo* contains configurations of the model that should be loaded into Gazebo simulator. Package *avrora_control* contains configuration of the model joints' settings and geometry: joint type, PID controller values and ROS controller settings.

Our project uses URDF model representation format, which is a XML format file that describes a robot model with a help of special tags. The main tags, which are used for describing virtual structure of a robot, are link and joint tags. Links (robot elements) are connected with a joints, generating a tree-structured representation.

4.2. *Core elements*

We selected several link elements (i.e., core components) to represent our model: *base_link, right_steer_link, right_steer_wheel, right_drive_wheel, left_steer_link, left_steer_wheel, left_drive_wheel*. In order to make the model visually similar to real Avrora Unior robot we applied meshes for those links (the meshes are the courtesy of Avrora Robotics company) with corresponding inertial parameters (Fig. 2).

Link *base_link* represents the outer shell (or the body) of the robot. Links *right_steer_link and left_steer_link* describe the parts of Ackermann steering mechanism actuating the front wheels (*right_steer_wheel and left_steer_wheel*, respectively). Links *right_drive_wheel* and *left_drive_wheel* represents the rear driven wheels.

All links in the model were scaled to proportions and dimensions of the real robot. To adapt inertial characteristics of the robot to the model, we calculated simplified inertial tensor for all links. We substituted compound meshes of a *base_link* with a solid cuboid object, steer wheel links, rear and front wheels as a solid cylinder to calculate approximated link inertia tensors.

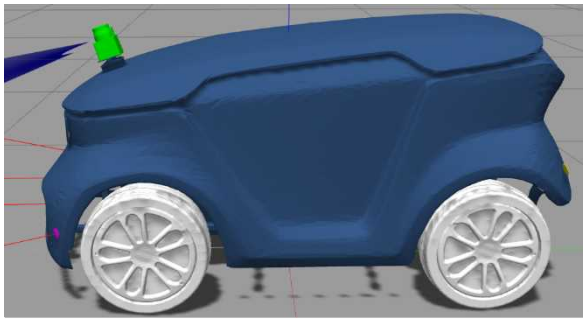

Fig. 2. Avrora Unior model in ROS/Gazebo environment.

4.3. *Joints, motors and transmission*

URDF format provides several types of joints: prismatic, revolute, continuous, floating and fixed. In our model we used three type of joints: fixed, revolute and continuous. Fixed joints attach rigidly two links with each other and in our model we used these joints to rigidly attach robot sensors to the main body. Continuous type allows rotation of a joint around a specified axis without upper and lower limits on the angle of rotation. Those type of joints in the model (*left_steer_wheel_joint, left_drive_wheel_joint, right_steer_wheel_joint, right_drive_wheel_joint*) were used to connect the wheels to the chassis. Revolute joints allow constrained rotation around a specified axis within upper and lower limits. We used revolute type joints for steering joints (*left_steer_joint* and *right_steer_joint*), to implement Ackermann steering geometry for model steering control (Fig. 3). The axis of rotation in both joints is Y-axis. Such joint configuration is a base for implementation of Ackerman drive mechanism for Avrora Unior model.

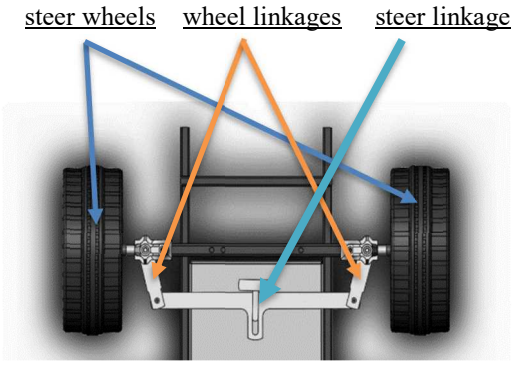

Fig. 3. Avrora Unior chassis scheme.

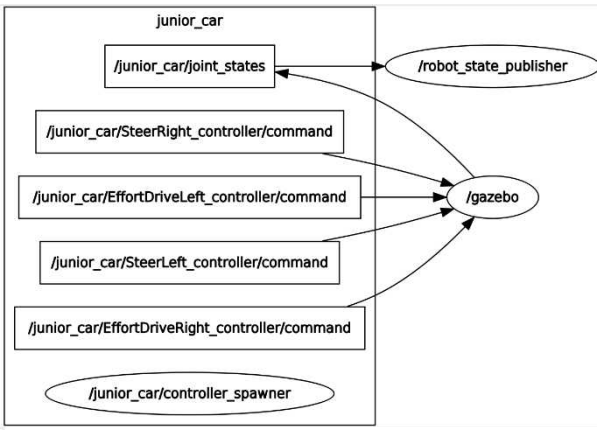

Fig.4. *rqt_graph* of Avrora Unior model topics (sensor's topics are not visualized).

We approximate Ackermann geometry by moving the steering pivot joints steering to angles (between

steering kingpins and the centre of the front axle) that are calculated from a desired steer angle as Eq. 1:

$$\begin{cases} \delta_{1l} = c_\delta * \delta_{st}, \delta_{1r} = arc\cot\left(cot(\delta_{1l}) - \frac{2B}{l_1}\right) when\ \delta_{st} > 0 \\ \delta_{1r} = c_\delta * \delta_{st}, \delta_{1l} = arc\cot\left(cot(\delta_{1r}) + \frac{2B}{l_1}\right) when\ \delta_{st} < 0 \end{cases} \quad (1)$$

where δ_{st} is the input angle, c_δ = 0.5, B is a half of the wheelbase, l_1 is the wheel linkage length. As a result, we have four continuous joints providing wheel rotation in X-axis direction (linear motion on the plane) and two revolute joints to replicate steering mechanism of Ackerman drive system.

Since there are only two motor driven wheels in Avrora Unior and a motor actuated steering mechanism, we configured the following ROS control joint controllers: position controllers *SteerRight_controller* and *SteerLeft_controller* (ROS *JointPositionController*), effort controllers *EffortDriveRight_controller* and *EffortDriveLeft_controller* (ROS *JointEffortController*). Controllers' topic visualization is showed in Figure 4.

4.4. *Sensor Simulation*

Table 3 presents the list of sensors of the real robot, their implementation in ROS framework and Gazebo environment. The model sensors were placed in the same positions as the real robot sensors in order to further match real sensor data. All Gazebo sensor plugins were configured to fit specifications of real robot sensors. We visualize sensor data with Rviz software.

5. Conclusions and Future Work

In this paper, we presented car-type Avrora Unior robot modeling. We used ROS environment and Gazebo robotics simulator. To construct the model, we used URDF format, which is supported by Gazebo. Core elements of Avrora Unior model were imported from mesh files provided by the manufacturer. Appropriate joint controllers for Ackermann steering and sensor simulations were added. As part of our future work, we are preparing a proper setup of the PID parameters of joint controllers and developing an improved Ackermann drive system. After successful modelling the cooperative path finding tasks will be applied in simulated within Gazebo urban environments[8] prior to real world testing.

Table 3. Sensors and plugins.

Sensor	ROS package	Gazebo plugin
LIDAR	hokuyo_node	laser_controller
Microsoft Kinect	freenect_stack	camera_plugin
GPS	ur_nmea_driver	novatel_gps_sim
Proximity	ur_rangefinder_driver	Sonar
IMU	myahrs_driver	imu_plugin

Acknowledgements

Part of the work was performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

1. A., Sagitov and Y., Gerasimov, Towards DJI Phantom 4 Realistic Simulation with Gimbal and RC Controller in ROS/Gazebo Environment. In *Int. Conf. on Developments in eSystems Engineering*, (2017), pp. 262-266.
2. I., Afanasyev, A., Sagitov, E., Magid. ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In *Int. Conf. on Advanced Concepts for Intelligent Vision Systems*, (Springer, Cham, 2015), pp. 273-283.
3. M., Sokolov et al., Modelling a crawler-type UGV for urban search and rescue in Gazebo environment, *In IEEE Int. Conf. on Artificial Life and Robotics (2017).*
4. W., Yao et al., A simulation system based on ros and gazebo for robocup middle size league, *In IEEE Int. Conf. on Robotics and Biomimetics,* (2015), pp. 54-59.
5. W., Qian et al., Manipulation task simulation using ros and gazebo, *In IEEE Int. Conf. on Robotics and Biomimetics,* (2014), pp. 2594-2598.
6. S., Kohlbrecher et al., Robocuprescue 2014-robot league team hector Darmstadt (Germany), *RoboCupRescue* (2014).
7. O., Kermorgant, A dynamic simulator for underwater vehicle-manipulators, In *Int. Conf. on Simulation, Modeling, and Programming for Autonomous Robot*s (Springer, Cham, 2014), pp. 25-36.
8. A. Andreychuk and K. Yakovlev. Applying MAPP Algorithm for Cooperative Path Finding in Urban Environments. In *Int. Conf. on Interactive Collaborative Robotics*, (2017), pp. 1-10.