Information technologies for development of educational resources in robotics

D S Chikurtev¹, A A Chikurteva¹ and N R Spasova²

¹Institute of Information and Communication Technologies, Bulgarian Academy of Science, acad. G Bonchev str., block 2, Bulgaria
²Technical University of Sofia, Department of Microelectronics, Faculty of Electronic Engineering and Technologies, Bulgaria
e-mail: dchikurtev@gmail.com

Abstract. Digital learning resources are a major part of distance learning in education. In the engineering sciences, where the training is applied and it is often necessary to conduct practical exercises, it is necessary to create both standard digital training resources and specific ones. Specific digital educational resources aim to provide a reproduction of real experiments, through digital and virtual tools. The article discusses ICT for the development of standard and specific digital resources. Software tools and applications for creating various digital resources with application in robotics are presented: Gazebo, Robot Operating System, Arduino. The process and the technologies used to create a digital resource (simulation model) through a Gazebo simulator for working with robots and sensors are described. The created digital resource could be used in various engineering fields such as robotics, embedded systems, software engineering, sensor systems, electronics and others. The resource can be distributed and shared over the Internet (Github), as well as upgraded (modified) by teachers and students.

1. Introduction

The application of information technologies in the learning process enables teachers to develop interactive educational resources to stimulate better learning of learning content and independent activity of students in the learning process [1, 2]. In today's technological world, digital educational resources are widely applicable and are a universal tool for organizing engineering training [3, 4].

Robotics is a multidisciplinary field that includes a number of other engineering subjects such as mechanics, electronics, sensors, hardware and software. In recent years, robotics has established itself as a useful and popular discipline, both in engineering higher education and in high school, and even in primary education [5, 6, 7]. More and more specialties in universities study robotics as a basic discipline, which allows the application of the learned theoretical knowledge in real problems of engineering practice.

However, in order to perform real tasks with robots, it is necessary to have a real robot in a learning environment. This problem can be solved only by purchasing educational robots and the necessary equipment: sensors, actuators and electronic devices. An alternative method of dealing with the problem is through simulations [8]. With the advancement of ICT, a number of applications and tools already exist for creating simulation models and simulation experimentation. The use of simulations has a number of advantages such as easier experiments, always ready environment for experiments, there is no danger of damaging any part and others [9]. A major problem with simulations is that basic models
of robots and sensors must be created. These models can be used directly so that students can focus on solving basic tasks - programming, algorithms, experiments [10].

In this article we consider the features and approaches for creating and working with simulation models in the Gazebo robot simulation environment and the robot operating system (ROS). With the help of the mentioned software applications, simulated robots can be created, controlled and tasted. The only equipment needed to work with simulations is the presence of a computer, which facilitates the work and contributes to better distribution and development of robotics. The created models and simulations can be freely distributed and executed on different computers.

The structure of the article is as follows: section 2 describes some of the commonly used applications for design and modeling, programming and control, and for robot simulation. Section 3 describes the creation and simulation of a mobile robot in the Gazebo application. Section 4 describes the development of simulation model control programs. Section 5 presents some experiments and the results achieved. Finally, a conclusion is presented.

2. Software tools and applications for creating robotics digital resources
In the process of creating educational resources for simulation and control of robots, we identified three types of applications: modeling, programming and simulation. Based on the types of applications, we can define three steps for creating and working with simulation models: modeling of robots and simulation environment, development of programs (software) for control and tests in simulation environment.

2.1. Modelling applications
Modelling applications are those which are used for building of 3D models. Examples of such applications are: Blender, SolidWorks, Autodesk 3ds Max, Autodesk Maya and others. Even the Gazebo application offers tools for modelling and creating 3D models of both robots and buildings or objects. But with quite basic capabilities for modelling complex shapes.

To create a complete 3D model of a robot, it is necessary to create drawings of all its moving components (links). The more accurate the models are made, the more realistic the simulation will be. Then these components must be connected to each other, and an important role in the connection stage is played by the formation of dependencies between the joints.

To achieve our goals, we use a ready-made model of a laser sensor downloaded from GitHub [11], which is pre-drawn using 3D software. The built-in Gazebo Model Editor is used to form the links of the robot. Through the editor we can set specific parameters such as mass, inertial characteristics, dimensions and others.

2.2. Software frameworks and development environments/ Programming applications
The robot operating system is among the most popular environments for robot control and programming. It offers a wide range of ready-to-use robot control packages. These packages provide algorithms for autonomous navigation, localization, data processing from various sensors - cameras, lasers, IMU, trajectory planning, control and more. In this way, everyone can integrate the necessary package into their robot. ROS also offers integration with the Arduino platform and a number of other hardware platforms used in robotics projects.

The principle of operation of ROS is described in a number of publications [12, 13]. In general, the main programs are called nodes, and the way they communicate is through topics. Some nodes publish topics, while others subscribe to these topics and process the data obtained. The main programs are developed in the programming languages C ++ and Python. In order to develop robot control, a program must be written to generate signals to drive the robot's motors. It usually takes several nodes to control a robot.

2.3. Simulation applications
Simulation environment are preferably used for testing the algorithms and methods that are subsequently
incorporated into a real robot. And because the simulation allows quick testing and a change in the environment, it's very convenient for quick experiments. Popular robot simulation platforms are Gazebo and Webots. Both platforms can work with ROS, the first even being installed as part of it. The article considers Gazebo platform.

In order to start the simulation of a robot, it is necessary to have a fully completed model of it. This means that all moving links of the model are connected to each other through joints. An important feature is that there should always be a basic link to serve as a basis.

The Gazebo application provides the ability to insert multiple items, entire rooms, apartments and even buildings. In this way, the robot can be placed in an industrial or home environment. There is a possibility to add external factors such as the presence of wind, sunlight and others. During the simulation we can observe the data that are processed, as well as visualize the topics from the sensors. We can visualize planned trajectories, positions of joints and units and other parameters.

3. Development of robot model
In this study, we will look at the development of a mobile robot with two active and two passive wheels. The robot will be equipped with a camera and a RPLidar A3 laser sensor. The robot will be located in a simulated room.

To create a model of a robot, we need to make a complete description of its components and the relationships between them. In order for ROS to understand this model description, the information is described in a file with the extension ‘xacro’. Usually three files of this type are needed - one to describe a robot, one to describe the materials and one to describe the specific characteristics of the units. An additional ‘gazebo’ file is added to represent the simulation model. This file describes some simulation parameters, adds controllers, sensors and all the characteristics of the robot. The programming language for the robot description in all files is XML. All these four files are placed in one folder, as the main file with the description of the robot launches the others and is used to start the robot in the simulation and in ROS.

The essential and important files are the one for the robot description and the one for the simulation. In the first file we have to describe the parameters and the location of the robot platform, the wheels, the camera and the laser scanner. It should be noted that only the physical parameters of the robot are described here. Initially, each part of the robot is described, and in the code below is the description of its basis.

```xml
<link name='chassis'>
  <pose>0 0 0.1 0 0 0</pose>
  <visual name='chassis_visual'>
    <origin xyz="0 0 0" rpy="0 0 0"/>
    <geometry>
      <cylinder radius="0.2" length="0.05"/>
    </geometry>
    <material name="orange"/>
  </visual>
</link>
```
As can be seen from the code, the link is named first, then its absolute position is set, because this will be the main link (parent link). The other parameters are visual appearance, inertial characteristics and collision. In order not to describe in detail the shapes of the object and all these parameters, we can take advantage of the option to insert an already created 3D model of the laser scanner, as mentioned above. This saves a lot of lines of code and creates more detail to the model.

```
<link name="laser">
  <collision>
    <origin xyz="0 0 0" rpy="1.5707 0 4.71"/>
    <geometry>
      <mesh filename="package://mybot_description/meshes/rplidar.dae" scale="0.001 0.001 0.001" />
    </geometry>
  </collision>
  <visual>
    <origin xyz="0 0 0" rpy="1.5707 0 4.71"/>
    <geometry>
      <mesh filename="package://mybot_description/meshes/rplidar.dae" scale="0.001 0.001 0.001" />
    </geometry>
  </visual>
  <inertial>
    <mass value="1e-5" />
    <origin xyz="0 0 0" rpy="1.5707 0 4.71"/>
    <inertia ixx="0" ixy="0" ixz="0" iyy="0" iyz="0" izz="0" />
  </inertial>
</link>
```

Once all the units are formed in a similar way, the joints between them must be described. Joint type elements have two attributes, name and type. The name must be unique for each joint, and the type can be: revolute, continuous, prismatic, fixed, floating and planar. Joints are also described by a number of parameters. The code presents the description of the joint between the robot base and one of the driven wheels (parent link).

```
<joint type="continuous" name="left_wheel_hinge">
  <origin xyz="0.0 0.2 0" rpy="0 0 0"/>
  <child link="left_wheel"/>
  <parent link="chassis"/>
  <axis xyz="0 1 0" rpy="0 0 0"/>
  <limit effort="100" velocity="100"/>
  <joint_properties damping="0.0" friction="0.0"/>
</joint>
```

After describing the physical parameters of the robot, the parameters for the simulation model such as the drive and control of the platform, the characteristics of the camera and the laser scanner must also be described. Various control and drive plug-ins can be used in the Gazebo. We will use the plugin for differential drive. To add the plugin, we use the following code.

```
<plugin name="differential_drive_controller" filename="libgazebo_ros_diff_drive.so">  
  <updateRate>$30</updateRate>
  <leftJoint>left_wheel_hinge</leftJoint>
</plugin>
```
In order for the plugin to work properly, a lot of parameters need to be set, and this is important in order to generate the correct drive on the mobile platform. What topic is needed for this plugin to work have to be considered. In this case, it is subscribed to the topic ‘cmd_vel’. This means that in order to move the robot, there must be a node that publishes commands in this topic.

In a similar way, the sensors are added using different parameters. The following code presents a method for describing the RPLidar. Here the sensor publishes data in the ‘scan’ topic, i.e. we must have a node to subscribe to this topic in order to be able to process the received data.
Once the description files for the robot and the sensors have been created, a new file must be created in which the simulation model is started, as well as the surrounding environment. If the robot model starts on its own, it will be in an empty environment. A file with the extension ‘world’ must be created for this purpose. This file stores the information about the created simulation environment. We can save such a file directly from the Gazebo menu at any time, when the simulation environment is ready to be used. Figure 1 shows the model of the robot in a simulation environment.

![Developed robot model in simulation environment.](image)

Figure 1. Developed robot model in simulation environment.

When the necessary simulation files are already available, a file must be created that starts the simulation environment first, then the robot model itself.

4. Control of robot model in ROS

When we start the robot and environment simulation, we have actually already started the ROS master core and the nodes and topics defined in the robot model. In order to control the mobile platform, a node must be started to publish the relevant commands in the ‘cmd_vel’ topic. Usually each node is created in a separate folder, in this folder an additional folder named ‘src’ is created, in which the program code file is placed. In order to start a node, a launch folder must be created in the main node folder, where files with parameters for starting the node are placed.

Because ROS also offers many ready-made programs and nodes, we will use a keyboard control node to test whether the created model can move. This node is set to send messages on the topic cmd_vel. It is part of the basic ROS packages and can be used directly without having to modify or adjust.

The connections between the nodes and the topics can be visualized by opening the rqt_graph application. Through this application we can visualize all the nodes and topics and how they are connected to each other. Figure 2 shows the relationship between the keyboard control node and the Gazebo simulator node via the ‘cmd_vel’ topic. In this case, the teleop_twist_keyboard node publishes the commands in the topic.

On the right side of the Gazebo node are the topics it publishes and the nodes that have subscribed to these topics. In this case, the laser_scan node receives the data from the laser scanner through the ‘scan’ topic. Camera_node has subscribed to the topic ‘camera/image_raw’ and receives the data from the robot's camera.
The keyboard control node is written in python code. The principle of operation of the node is to read whether the corresponding keys on the keyboard are active / pressed, to generate movement data and to publish messages in the respective topic. The method that publishes in a topic is as follows:

```python
self.publisher = rospy.Publisher('cmd_vel', Twist, queue_size = 1)
```

Even if another program is written to generate commands to drive the robot, this method will be the same. The important part of the code is the one in parentheses, where the topic name (cmd_vel), the message type (Twist) and queue_size are entered. 'queue_size' is the size of the outgoing message queue used for asynchronous publishing.

The other important element of the programming of nodes in ROS is the subscription to topics. In our case, Figure 2 shows two nodes that subscribe to different topics. Subscribing to a topic is similar to publishing, and in each program, it is done by the same method, where the name of the subscribing node must be initialized. In the specific example, a method for subscribing to the topic from the laser scanner is presented, which publishes messages in the LaserScan format.

```python
rospy.init_node('laser_scanner')
self.subscriber = rospy.Subscriber('/scan', LaserScan, callback)
```

ROS libraries offer a wide range of different message formats. This allows the most appropriate ones to be used according to the needs of the management. The documentation for each of the ready-made packages describes the setting parameters, the topics used and the types of messages. When we have a ready model of a robot, it is very easy to apply different packages for localization, navigation, control and others.

5. Experiments and results
Because we describe and research the development of educational materials and training resources in the field of robotics, we conducted basic experiments. The aim is to confirm that the developed model and environment for mobile robot simulation operates properly and can be used to train students.

The first experiment was to control the robot via a control node via the keyboard. This will confirm that the robot is moving and that any node that publishes the topic of motion will be able to control the robot.

The second experiment is to visualize a video stream from the robot's camera. This will show if we have a functioning simulation camera. Camera data can be used to recognize objects.

The third experiment involved tests with a laser scanner. The data from the sensor must outline the contour of the environment and can be used to implement localization and navigation.

During the first experiment, the robot successfully executed the given commands to move in all directions. The movements of the mobile robot are smooth, without any deviation. This means that the positions and dimensions of the components and joints of the platform are set correctly. Figure 3 shows a list of published commands from the topic to the robot.
First experiment: controlling the robot by sending commands through cmd_vel topic.

After the tests of the second experiment, a video stream from the robot's camera was successfully visualized. Figure 4 shows the robot in the simulation environment, and in an additional window on the right is the video from the camera.

Second experiment: visualizing the video stream from the robot's camera.

In the tests for the third experiment, the points from the laser scanner were successfully visualized and the contour of the room in which the robot is located was outlined. Figure 5 shows the simulation environment on the left and the data from the laser scanner topic on the right.

The obtained results show that the described methods and programs for creating and working with simulation models in a simulation environment are applicable and workable. These methods can be used
to create any model of robots - mobile platforms or robotic arms. Using the described methods, different sensors can be simulated so that a real robot can be completely recreated and simulated.

Figure 5. Third experiment: visualizing the laser scanner topic data.

6. Conclusion
The created model of simulation environment and mobile robot is formed in one package, which can be executed on any of the current ROS distributions installed on any computer. The development of ready-to-use simulations of fully equipped robots allows for the widespread use of robotics in educational environments. The easy use and modification of the simulation models allows the teachers to lead highly effective and useful courses. With the advancement of ICT, these teaching methods will develop significantly and will become standard practices.

Creating simulation models of robots and sensors in the Gazebo environment is a relatively easy process for people involved in robotics, design or programming. However, in order to achieve widespread use of simulation models of robots, methods for working with simulators and software need to be promoted. The methods presented in the article can be successfully used for these purposes. As a result of the research and the applied methods in the article the following models were created:

- Differential drive mobile robot with two castor wheels;
- Two sensor types: BGR camera and a LIDAR laser scanner;
- Simulation environment – room, where the robot will operates.

Acknowledgement
The research is supported by the Bulgarian National Science Fund under the contract № KP-06-M27/1-04.12.2018. The research is supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", financed by the Ministry of Education and Science.
References

[1] Toskova A T, Doychev E H, Toskov B P An Idea for Extension of the Virtual Educational Space for Lifelong Learning Computer science and communications 5 (4), 35-38

[2] Ramon Pontes Araújo, Francisco Petrônio Alencar de Medeiros 2020 Requirements Elicitation and Specification for Educational Technology Development: A Systematic Literature Mapping” 2020 15th Iberian Conference on Information Systems and Technologies (CIStI)

[3] Colvard Nicholas B, C Edward Watson and Hyojin Park 2018 The impact of open educational resources on various student success metrics International Journal of Teaching and Learning in Higher Education 30 no 2 262-276

[4] Santos-Hermosa Gema, Núria Ferran-Ferrer and Ernest Abadal 2017 Repositories of open educational resources: An assessment of reuse and educational aspects International Review of Research in Open and Distributed Learning: IRRODL 18 no 5 84-120

[5] Yen Air Caballero González, Ana García-Valcárcel Muñoz-Repiso 2017 Educational robotics for the formation of programming skills and computational thinking in childhood International Symposium on Computers in Education (SIIE)

[6] Hugo Costelha, Carlos Neves 2018 Technical database on robotics-based educational platforms for K-12 students IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)

[7] Joel M Esposito 2017 The State of Robotics Education: Proposed Goals for Positively Transforming Robotics Education at Postsecondary Institutions IEEE Robotics & Automation Magazine vol 24 issue 3

[8] Lucas M do Nascimento, Davi S Neri, Thiago do Nascimento, Erika A. Yanaguibashi, Sarah Sá, Luiz Marcos Gonçalves 2019 sBotics: Simulation Applied for the Practical Component of the Brazilian Robotics Olympiad Latin American Robotics Symposium (LARS)

[9] Abraham Prieto García, Gervasio Varela Fernández; Blanca María Priego Torres, Fernando López-Peña 2011 Educational autonomous robotics setup using mixed reality 7th International Conference on Next Generation Web Services Practices

[10] A Toskova, B Toskov, S Stoyanov, I Popchev Genetic Algorithm for a Learning Humanoid Robot Comptes rendus de l’Academie Bulgare des Sciences 72 (8) 1102-1110

[11] RPLidar A3 model: https://github.com/husarion/rosbot_description/tree/master/src/rosbot_description

[12] Gatesichapakorn, Sukkpranhachai, Jun Takamatsu, and Mitit Ruchanurucks 2019 ROS based autonomous mobile robot navigation using 2D LiDAR and RGB-D camera In 2019 First international symposium on instrumentation, control, artificial intelligence, and robotics (ICA-SYMP) IEEE pp 151-154

[13] Koubâa Anis ed 2019 Robot Operating System (ROS) vol 1 (Springer)