ROS-Based Multi-Robot System Simulator

Zhengguang Ma*, Lin Zhu, Peng Wang, Yongguo Zhao
Institute of Automation, Qilu University of Technology(Shandong Academy of Sciences)
Shandong Provincial Key Laboratory of Robot and Manufacturing Automation Technology
Jinan, China
*mazhg@sdas.org

***Abstract*—Recently, modelling, simulation and control of coordination behavior has become an important subject in the field of complex system science. For the seamless transition between simulation, semi-physical simulation and full hardware experimentation of coordination behavior, an efficient simulator based on Robot Operating System(ROS) and Qt are provided in this paper. ROS and Qt are both powerful tools which can be used in the field of robotics and Graphical User Interface (GUI) development. The ROS-based simulator can be used to verify the coordination protocols of multi-robot system(MRS). The simulator runs under the ROS framework can both simulate and control the real robots with minor modification, therefore, the ROS-based MRS simulator is much more efficient for theoretical verification of coordination protocols of MRS.**

***Index Terms*—MRS, GUI, ROS, Qt, Simulator, Coordination Behavior**

I. Introduction

In recent years, there is a growing interest in replacing a single complex robot with a team of robots working cooperatively that can achieve the same goals or even exceeds the simple summation of the performances of all individual robots [1]. Additionally, the group behaviors have the characteristics including flexibility, robustness, decentralized characteristics and self-organized [2], which show the advantages of the group coordination behaviors. In most of the literatures, the theory results of group behavior are verified by conducting numerical simulation in MATLAB [1], [3]–[5]. The position errors, trajectories and control inputs of robots are shown with MATLAB to illustrate the effective of the proposed protocols [6].

In parallel with these developments, a new flexible framework for robot applications has emerged in the last few years, taking a strong position in the academic world: the Robot Operating System(ROS) environment. As an open-source robot operating system, ROS provides a communication layer and a data and programming structure. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms. This system is being widely adopted by the scientific community, which allows for an unprecedented direct access to state-of-the-art developments in the robotics field [7].

This work is supported by Youth Science Funds of Shandong Academy of Sciences, Grant No.2019QN0015, The Key R&D Program of Shandong Province, Grant No.2016ZDJS02A14.

In practice, there are few simulators developed to test and validate the simulation of MRS coordination. MobileSim is software for simulating mobile robots, for debugging and experimentation with ARIA and relating software [8], while, it only supports ActivMedia/MobileRobots platforms and a usage license is needed to install the software. Liu introduces a simulation system that can be used to do fast simulation for consensus protocols of MRS and the simulation results are shown with MATLAB figures [9]. Similar to the simulation with MATLAB, the simulation code cannot be implemented into the real robots directly as well. The authors of [10] develop a ROS-based Graphical User Interface (GUI) for the controlling of an autonomous surface vehicles. The GUI provides shortcuts for easy initialization and killing of processes as well as gathering and viewing the vehicle data such as diagnostics, sensor information and live camera feed, all at one place. While, the GUI only support the controlling of one single vehicle. Kelpie, a simulator for water surface and aerial vehicles, is developed based on ROS framework in [11]. However, there is no easy-to-use graphical user interface. In [7], an ROS-based open-source simulator for multiple Ar.Drone quadrotors are provided. The simulator shows the functionality by implementing a formation control architecture and the step response and trajectory following test are shown as well. However, the simulator has no GUI to integrate all the setting, input and output information.

In this paper, an effective ROS-based MRS simulator with a simple and easy to use yet powerful GUI are developed. The GUI is developed using Qt as it has built-in support for ROS libraries and the software modules which link the GUI to the various parts of simulation are designed according to the ROS framework. The simulator is ideal for simulation of coordination behavior of MRS as it can port directly to real robots, requiring little change to run real life testing.

The rest of this paper is organized as follows. Some preliminaries which will be used are presented in Section II, and the problem to be considered are described. In Section III, the MRS simulator and the features of the simulator are provided. Section IV provides a formation controller to verify the effectiveness of the simulator. And the simulation results are given in Section V. Finally, the paper is concluded in Section VI.

Authorized licensed use limited to: University of Ottawa. Downloaded on January 08,2022 at 20:58:23 UTC from IEEE Xplore. Restrictions apply.

II. Backgound

In this section, some preliminaries related to MRS are presented to best illustrate the simulator.

Let $N = \{R_1, \ldots, R_n\}$ be the set of n robots with positions $z_i(t) = (x_i(t), y_i(t))^T, i = 1, \ldots, n$ and the kinematic model of each robot can be represented as follows:

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} cos(\theta_i) & 0 \\ sin(\theta_i) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{1}$$

Where(x_i, y_i, θ_i) is the position and orientation of $i-$th robot, v_i and ω_i are the linear and angular velocity of the center point of the wheelbase of the $i-$th robot respectively. While, the center point of the wheelbase has no tangential velocity, a point $\alpha_i = (p_i, q_i)$ outside the wheels axis is considered as the reference center as is shown in Fig.1.

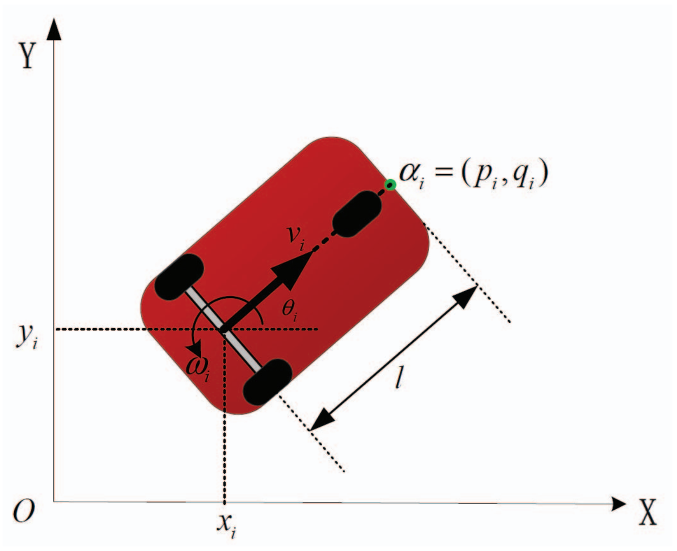


Fig. 1: Kinematic model of robot i

Take α_i as the reference center of robot i, the kinematics of robot R_i can be rewritten as

$$\dot{\alpha}_i = A_i(\theta_i) \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{2}$$

where

$$A_i(\theta_i) = \begin{bmatrix} cos(\theta_i) & -lsin(\theta_i) \\ sin(\theta_i) & lcos(\theta_i) \end{bmatrix}. \tag{3}$$

If there exist a control law $u_i(k) = f_i(\alpha_i(k))$ for each robot R_i such that:

$$\lim_{k\to\infty} (\alpha_i(k) - \alpha_i^*(k)) = 0, i = 1, \ldots, n \tag{4}$$

i.e. the desired position of the robot R_i with respect to the robot R_{i+1} is achieved, then we can say that the MRS satisfy the desired formation configuration [6].

III. The MRS Simulator

The MRS simulator is developed based on the open source ROS environment. As is shown in Fig.2, the simulation system consists of three layers including display, application function and data & communication. Parts of the features of the simulator are shown as follows:

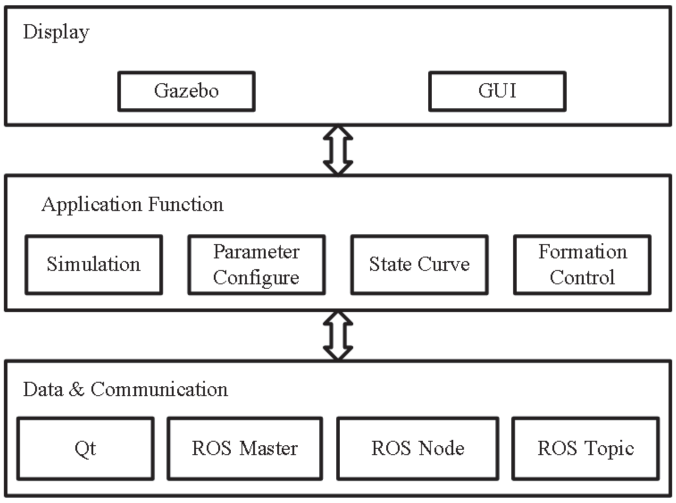
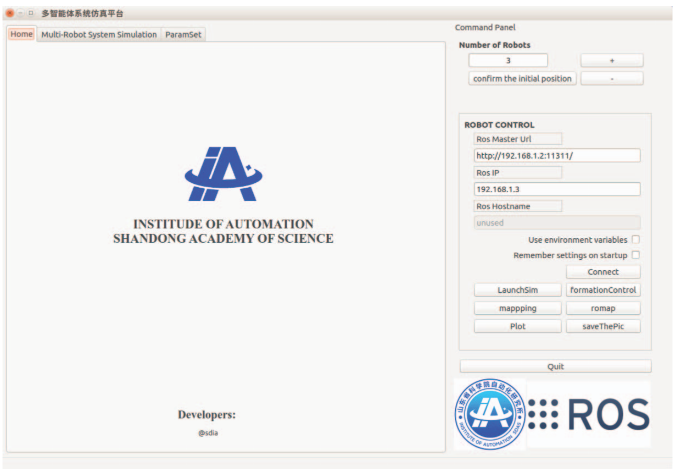


Fig. 2: The architecture of MRS simulator

1) Display

In this layer, all of the simulation process are displayed with Gazebo and GUI. Gazebo is a powerful 3D physics simulation platform with a powerful physics engine, some of the messages of robots are published through the open-source gazebo_ros package. And most of the simulation processes are presented in GUI and user can interact with the simulation system with GUI which is shown in Fig.3.



Number of Robots: 3

Ros Master Url: http://192.168.1.2:11311/

Ros IP: 192.168.1.3

Ros Hostname: unused

Use environment variables ☐

Remember settings on startup ☐

Fig. 3: The GUI of MRS simulator

2) Application function

The application layer contains almost all of the function of the simulator. The simulation of MRS can be executed through the shortcuts and the simulation process can be presented in the GUI as is shown in Fig.4. The trajectories of all the robots

Authorized licensed use limited to: University of Ottawa. Downloaded on January 08,2022 at 20:58:23 UTC from IEEE Xplore. Restrictions apply.

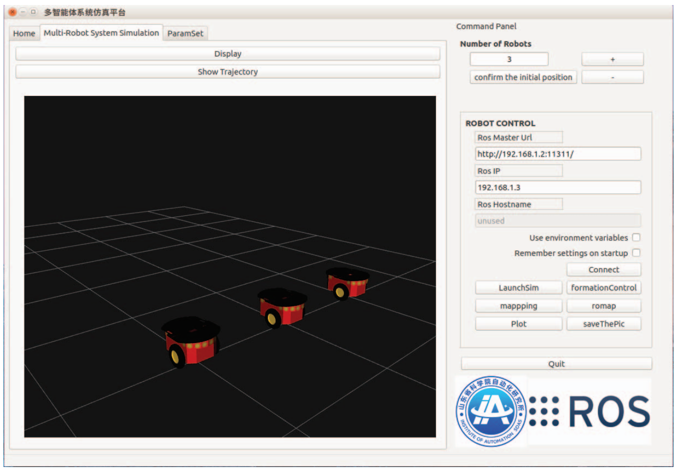


Fig. 4: MRS simulation

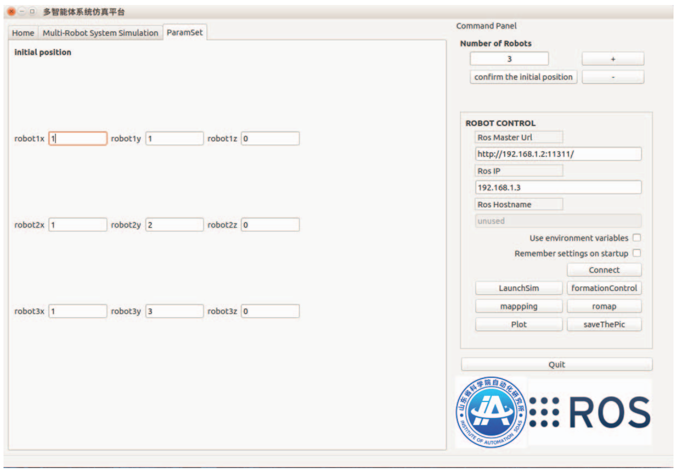


Fig. 5: Parameter configure

can be set to display or not. The number of robots in the MRS simulation can be configured directly with GUI. And the initial position of robots can be edited and the number of text edit widgets is dynamic change according to the number of robots(Fig.5). The state curves of each robots can be shown and the simulation results can be saved as figure using shortcuts. And the Formation controller can be executed through shortcuts which will be introduced in the next section.

3) Data & communication

The bottom layer is the fundamental part of the simulator which is responsible for the data transmission and communication. As the core of the bottom layer, the ROS Master can track publishers and subscribers to topics as well as services and enable ROS nodes to locate one another and communicate with each other peer-to-peer(Fig.6).

IV. Formation Controller for MRS

The cooperative behavior include consensus, formation and flocking, while, simulation of consensus are not feasible in practice due to the size limition of robots. Therefore, the multi-

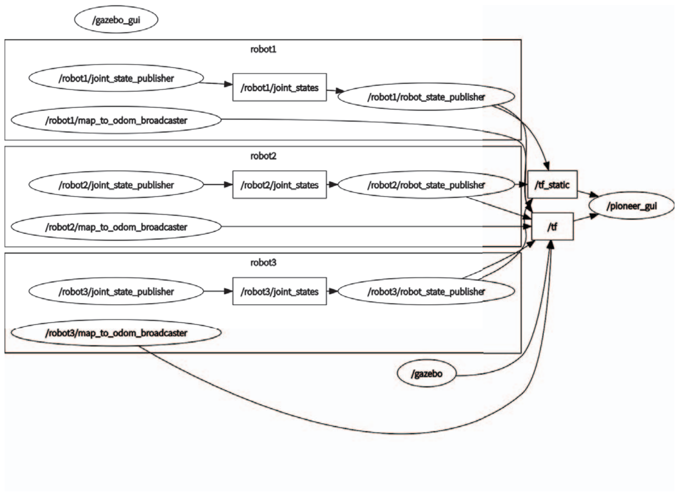

/gazebo_gui
robot1
/robot1/joint_state_publisher
/robot1/joint_states
/robot1/robot_state_publisher
/robot1/map_to_odom_broadcaster
robot2
/robot2/joint_state_publisher
/robot2/joint_states
/robot2/robot_state_publisher
/robot2/map_to_odom_broadcaster
robot3
/robot3/joint_state_publisher
/robot3/joint_states
/robot3/robot_state_publisher
/robot3/map_to_odom_broadcaster
/tf_static
/tf
/pioneer_gui
/gazebo

Fig. 6: Data transmission and communication

robot formation simulation is defined for a group of robots to verify the functionality of the simulator.

For the sake of simulation simplicity, the model (2) is discretized with Euler approximation which is given by:

$$\alpha_i^+ = \alpha_i + T\dot{\alpha}_i \tag{5}$$

where $T > 0$ is the sampling period.

Lemma 1 [6]: Consider a group of robots with the dynamic (5), the control law is shown as

$$\begin{bmatrix} u_i \\ \omega_i \end{bmatrix} = -\frac{1}{2T}k_i A_i^{-1}(\theta_i)\left(\frac{\partial \gamma_i}{\partial \alpha_i}\right)^T, i = 1, \dots, n, \tag{6}$$

in which $\gamma_r = ||\alpha_i - \alpha_i^*||^2, i = 1, \dots, n$ is the Atractive Potential Function and $0 < |k_i| < 1$. Then, it holds that $\lim_{k\to\infty}(\alpha_i(k) - \alpha_i^*(k)) = 0$.

With the discrete-time formation control law defined in Lemma 1, the MRS formation simulation is proceeded using the pioneer_control ROS node. The calculateControlLaw() function is defined to calculate the control law of each robot. The input variables are the serial number and position of all the robots. The output variables are the conrol inputs of all the robots and the calculateControlLaw() function is defined as follows:

```
/*@brief:   calcuate the control law
 *          The Core code, modify this function
 *          according to the control law.
 * @input   std::map<int, Position> pos_map;
 *          int:Robot's No.
 *          Position: Robot's position(x,y,theta)
 * @output  vector<geometry_msgs::Twist> cmd_vec;
 *          cmd_vec[int]: No.int Robot's control
 *          input
 */
void PioneerControl::calculateControlLaw()
{

}
```

However, for the sake of simplicity, some researchers take the robots as mass points when they do some theoretical

Authorized licensed use limited to: University of Ottawa. Downloaded on January 08,2022 at 20:58:23 UTC from IEEE Xplore. Restrictions apply.

researches in MRS coordination problems [8]. When they want to verify their theory results in the simulator, similar to [12], we can transform the mass point kinematic model into a kinematic model for a fixed point off the center of the wheel axis. Then the control variables of the mass point kinematic model can be transformed into the control input of each robots (v_i, ω_i) as following

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & \sin(\theta_i) \\ -\frac{1}{d_i}\sin(\theta_i) & \frac{1}{d_i}\cos(\theta_i) \end{bmatrix} \begin{bmatrix} u_{xi} \\ u_{yi} \end{bmatrix}, \tag{7}$$

where the d_i is the $i-$th robot's distance between the mass point and the fixed point off the center of the wheel axis, θ_i is the $i-$th robot's orientation .

V. The Simulation Result

The performance of the simulator is evaluated by implementing a MRS formation controller with the model presented in Section IV for $n = 7, l = 0.05, k_i = 0.02$. The initial position of all robots are given by $[\alpha_{1x}, \alpha_{1y}, \theta_1] = [10, -5, 0]$, $[\alpha_{2x}, \alpha_{2y}, \theta_2] = [5, -2, 0]$, $[\alpha_{3x}, \alpha_{3y}, \theta_3] = [2, 8, 0]$, $[\alpha_{4x}, \alpha_{4y}, \theta_4] = [-5, 12.3, 0]$. The desired formation of the MRS is a straight line and the separation of each other is $c = 2$.

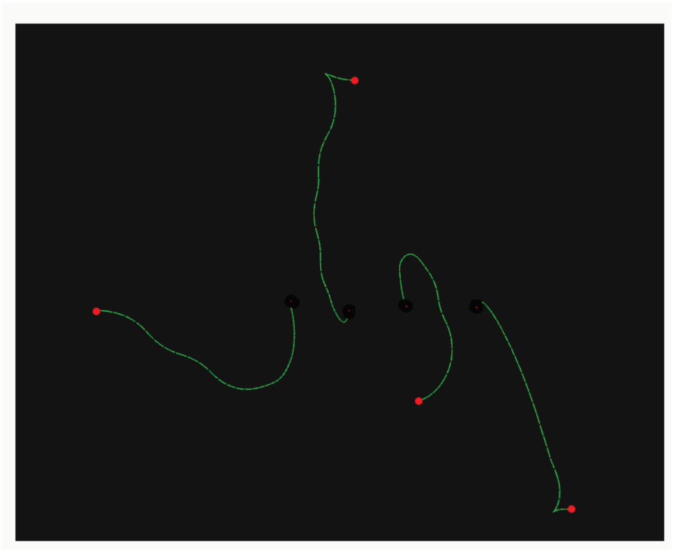


Fig. 7: The simulation result

Then the simulation result is shown in Fig.7, in which the red points are the initial position of the robots and the dash green lines are the trajectories of all the robots. It is obviously that the MRS finally form the desired straight line formation.

Fig.8 shows the x-axis state curves of all robots and the state curves converge to constant values with the separation of each other is 2. Fig.9 shows the y-axis state curves of all robots and all the state curves converge to a constant value which means they form a straight line formation parallelling to x-axis.

The control inputs of all the robots including linear and angular velocities are shown as Fig.10.

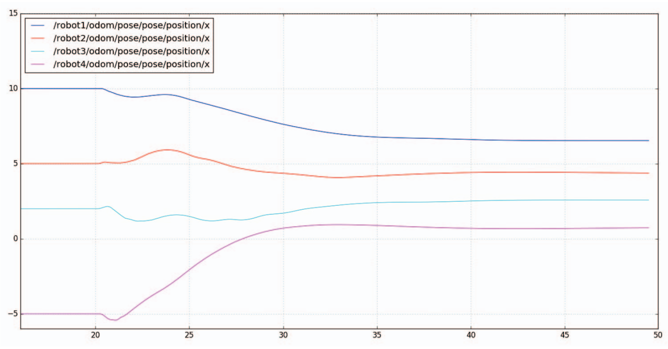


Fig. 8: The x-axis state curves

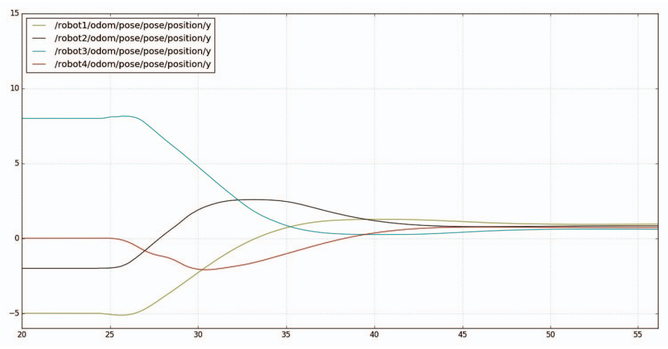
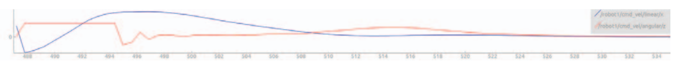
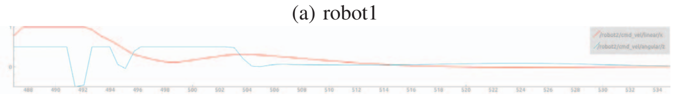
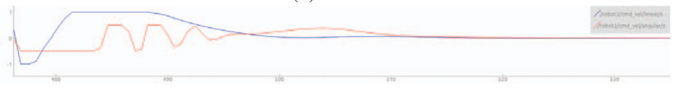


Fig. 9: The y-axis state curves


(a) robot1


(b) robot2


(c) robot3

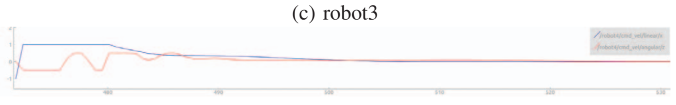

(d) robot4

Fig. 10: The control inputs of all the robots

VI. Conclusion

In this article, an open-source simulator for MRS based on Qt and ROS framework is addressed. The GUI of the simulator with the ROS environment is similar to the open source library interacting between ROS and the real robot. Therefore, the simulator reduces the gap between numerical simulation and full hardware experimentation. It will be an effective simulation tool for the theoretical researchers.

Some features of the simulator presented in this paper need to be further refined. The interaction topology of the MRS in this paper are predefined, however, the topology should be editable within the GUI. Also, the robot model should be choosen as needed including differenitial or omnidirectional

Authorized licensed use limited to: University of Ottawa. Downloaded on January 08,2022 at 20:58:23 UTC from IEEE Xplore. Restrictions apply.

drive robot, manipulator and quadrotor. Different coordination behavior including flocking and marching will be considered as well. We will consider these extensions as our future work.

REFERENCES

[1] H. Yamaguchi, N. Ai, and A. Kawakami, "Control of two manipulation points of a cooperative transportation system with two car-like vehicles following parametric curve paths," *Robotics & Autonomous Systems*, vol. 63, pp. 165–178, 2015.

[2] D. Z. Cheng and H. F. Chen, "From swarm to social behavior control," *Science & Technology Review*, vol. 22, no. 8, pp. 4–7, 2004.

[3] G. Wen, Y. Zhang, Z. Peng, Y. Yu, and A. Rahmani, "Observer-based output consensus of leader-following fractional-order heterogeneous nonlinear multi-agent systems," *International Journal of Control*, pp. 1–17, 2019.

[4] Q. Zhang, Z. Gong, Z. Yang, and Z. Chen, "Distributed convex optimization for flocking of nonlinear multi-agent systems," *International Journal of Control, Automation and Systems*, vol. 17, no. 5, pp. 1177–1183, 2019.

[5] W. Zhang, D. W. C. Ho, Y. Tang, and Y. Liu, "Quasi-consensus of heterogeneous-switched nonlinear multiagent systems," *IEEE Transactions on Cybernetics*, no. 99, pp. 1–11, 2019.

[6] D. E. Hernandez-Mendoza, G. R. Penaloza-Mendoza, and E. Aranda-Bricaire, "Discrete-time formation and marching control of multi-agent robots systems," in *International Conference on Electrical Engineering Computing Science & Automatic Control*, 2011, pp. 1–6.

[7] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[8] Z. G. Ma, Z. X. Liu, and Z. Q. Chen, "Modified leader-following consensus of time-delay multi-agent systems via sampled control and smart leader," *International Journal of Control, Automation and Systems*, vol. 15, no. 6, pp. 2526–2537, 2017.

[9] Z. X. Liu, Y. F. Wei, Y. Cao, and Z. Q. Chen, "Design of a simulation system for consensus protocols of multi-agent system," *Computer Simulation*, vol. 28, no. 1, pp. 6–9, 2011.

[10] D. Patil, S. S. Velamala, and X. Ming, "Development of ros-based gui for control of an autonomous surface vehicle," in *IEEE International Conference on Robotics & Biomimetics*, 2018, pp. 628–633.

[11] R. Mendona, P. Santana, F. Marques, A. Loureno, J. Silva, and J. Barata, "Kelpie: A ros-based multi-robot simulator for water surface and aerial vehicles," in *IEEE International Conference on Systems*, 2013, pp. 3645–3650.

[12] R. Wei and N. Sorensen, "Distributed coordination architecture for multi-robot formation control," *Robotics & Autonomous Systems*, vol. 56, no. 4, pp. 324–333, 2008.

Authorized licensed use limited to: University of Ottawa. Downloaded on January 08,2022 at 20:58:23 UTC from IEEE Xplore. Restrictions apply.