Development of an Optical Tracking Based Teleoperation System with Virtual Reality

1st Yunpeng Su
Department of Mechanical Engineering
University of Canterbury
Christchurch, New Zealand
yunpeng.su@pg.canterbury.ac.nz

2nd Mohammad Ahmadi
Department of Mechanical Engineering
University of Canterbury
Christchurch, New Zealand
tedahmadi@gmail.com

3rd Christoph Bartneck
Human Interface Technology Laboratory
University of Canterbury
Christchurch, New Zealand
christoph.bartneck@canterbury.ac.nz

4th Frank Steinicke
Department of Informatics
University of Hamburg
Hamburg, Germany
frank.steinicke@uni-hamburg.de

5th Xiaoqi Chen
School of Materials Science and Engineering
Shanghai Jiao Tong University
Shanghai, P.R. China
xqchen@sjtu.edu.cn

***Abstract*—In this paper, we propose an optical tracking based teleoperation system with a virtual reality interface for mobile robots. The system allows the user to teleoperate a mobile robot using bare hands and the user can adjust the autonomy of the robot between two levels: direct control and autonomous navigation. A Leap Motion sensor based non-contact teleoperation method is developed to translate sensor messages into velocity commands to the robot in order to interact with a remote mobile robot in a natural manner. By incorporating HTC Vive virtual reality device, the user is fully immersed into the virtual space with visual feedback from the remote site. The system features cost-effective and extendable by leveraging commercial virtual reality devices and integrating it with open source robotic control middleware.**

***Index Terms*—Virtual Reality, robot control, human robot interaction**

I. Introduction

The development of teleoperation technology has revealed enormous potential in completing tasks in an interactive method between users and robots in complex environments. The continuous and operating and monitoring process is tedious and tiring for the users. It is highly beneficial if autonomous solutions take the workloads off them in terms of operating mobile robots, although developing mobile robots with fully autonomous capabilities which can navigate in unknown environments remains a challenging task[1]. Furthermore, the user should still be able to take over direct control of the robot in complex situations in which the autonomous approach is not feasible yet. In this paper, we propose an optical tracking based teleoperation system with which allow the user to accordingly adjust the autonomy of the robot between two levels: direct control and autonomous navigation.

Virtual Reality can be incorporated into teleoperation robotic systems to offer the user a more immersive and intuitive approach in terms of controlling robots at a distance. Based on a computer-generated virtual environment and multi-sensors on site, information from remote sites is transmitted back to users and the users serve as part of the control loop to manipulate the robots to accomplish certain tasks. Virtual reality allows humans and robots to enhance each others capabilities. Humans are able to supervise the robots from a distance and connect seamlessly to the robots working space[2]. Currently attempts to apply virtual reality with teleoperation have been made in remote Grasping, assembly task and complex manipulation and some other aspects in manufacturing[2][3][4]. Virtual reality is also used in mobile robots to improve the quality of control[5][6]. However, the virtual reality related teleoperation systems mostly rely on traditional contact control methods such as joysticks or controllers as input devices. It is an intuitive and effective approach to perform tracking of finger motions and hand gestures with a data glove[7][8]. However, a data glove has no capability of tacking hand trajectory without extra inertial and magnetic sensors embedded[9]. Non-contact teleoperation approaches are more suitable for an intuitive interaction method[10]. The control of the mobile robot movements in our teleoperation system is performed by utilizing a Leap Motion controller. The Leap Motion device features the hand motion tracking and hand posture recognition functionalities, which can be used to produce the control signals in order to provide a more natural and intuitive method for human-robot interaction[11][12][13].

ROS provides an architecture for robotic research with modularity, compatibility and extensibility, which is flexible and adaptable to enhance the functionalities of robotic teleoperation system and meet the needs of users[14][15]. Additionally, Unity is an effective platform for developing the human-robot interface in virtual reality and entirely compatible

978-1-5386-9490-9/19/$31.00 ©2019 IEEE

with certain commercial virtual reality devices such as HTC Vive. ROS features a Rosbridge module which can be used primarily for building the communication between ROS and non-ROS platform. Data via Rosbridge in JSON format is used as an interface to transmit messages in ROS topics. The implementation of Rosbridge as an intermediate integrates the communication seamlessly between ROS running on Linux and Unity on Windows system.

A number of Intuitive human-robot interaction systems for robotic teleoperation are developed. However, autonomous approaches are no incorporated into these systems to relieve the users mental and physical fatigue while performing intensive or repetitive tasks[16]. Adjusting the level of autonomy of teleoperation systems maximizes the capabilities of both the robots and the humans. It optimizes the decision-making capacity of humans in the human-robotic system and efficiently balances that with speed, repeatability and accuracy of robots[17]. Hence, the adjustable autonomy of control is integrated into the teleoperation system to provide the user options and optimize the performance of control. The user has access to two levels of autonomy of the robotic control: direct control and autonomous navigation.

In the direct control mode, the user operates the robot directly to move by giving gesture commands through Leap Motion. Therefore, human influence interferes with the robot's state deeply[12]. Autonomous navigation brings the robot operation efficiency and a certain reduction in the workload of the user. The user is able to switch to direct control if dexterous manipulation or human decision are required explicitly[1][18][2].

In this paper we introduce:

A new virtual reality and optical tracking based telerobotic architecture which features human user and robots virtual coexistence.

An effective method for interpreting hand gestures tracking data into velocity commands to a mobile robot.

A communication approach to facilitates the advantages of an immersive user interface and ROS robotic teleoperation systems with including different control levels.

II. System design

In order to integrate virtual reality and optical tracking into the system, we determine the following essential components: an HTC Vive virtual reality device, Leap Motion controller and A Microsoft Kinect sensor. A TurtleBot2 mobile robot works as the moving agent at the remote site and it is equipped with a Kinect 3D sensor and an onboard PC. Fig. 1 shows the main components of the mobile robot platform. A NUC mini PC and a Kinect sensor are set up on the mobile robot. The NUC PC with Ubuntu 14.04 and ROS Indigo installed is defined as the ROS Master to the entire system. It allows individual ROS nodes firstly to locate one another and enables the nodes to communicate.

In the virtual scene displayed with the head-mounted display(HMD), the user interacts with the virtual replica of the mobile robot and simultaneously receives a video stream from the remote site. The virtual scene the user can see as shown in Fig. 2. The virtual scene in HMD contains visual feedback from the robot for both eyes. The user feels fully immersive in the working space of virtual reality and the intuitive interaction makes the user feel like he is in the robots action. The head tracking devices have the capability to keep tracking the pose of the HMD accurately so that the user is able to walk around in the virtual scene, which immensely enhances the users spatial perception and the illusion of immersion. The virtual scene presenting, video stream rendering and data integration are accomplished on a workstation.

The general concept and main components of the proposed teleoperation system are illustrated in Fig. 3. At the mobile robot side, the Ubuntu computer running ROS Indigo is onboard the mobile robot and it is set up as the ROS Master for the entire system. The subsystem control and message communication are performed on the Ubuntu computer. A Kinect sensor resides on the mobile robot to capture real-time stream in direct control mode and perform scanning tasks in autonomous navigation mode. The Kinect sensor consists of both an RGB camera and a depth camera. The RGB camera on Kinect captures live video stream at the remote site and transfers it back to the HMD through the communication network in real time in direct control mode. Further, in autonomous navigation, the depth camera on the Kinect is used to provide a simulated laser scan for environment mapping. Both of the nodes on Kinect sensor and the mobile base of the robot register with the ROS Master on the Ubuntu computer and the communication between all the nodes can be carried out then.

At the user side, the Windows PC with Unity software installed serves as the core of the subsystem. HTC Vive HMD is linked to Unity on the PC to display the virtual scene generated in Unity. The virtual scene in HMD containing on-site visual feedback from the mobile robot is rendered into images for both eyes. The user feels fully immersive in the working space and feels like he is in the robots action. Head

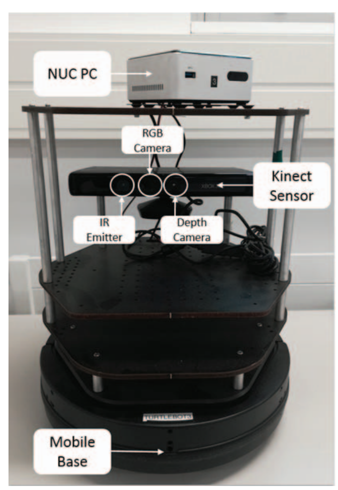

Fig. 1. The mobile robot platform equipped with a Kinect sensor.

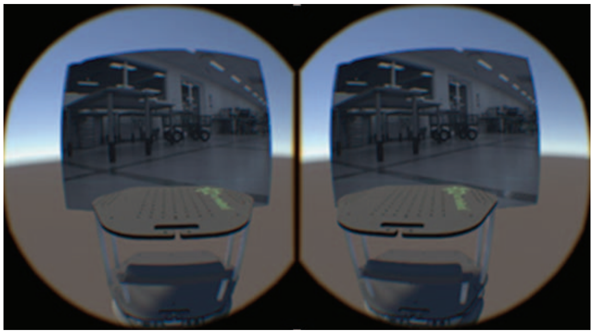

Fig. 2. The virtual scene in HMD containing visual feedback from the robot for both eyes.

tracking devices are connected to the PC and it tacks the pose of the users head in real time as the user performs teleoperation tasks. It has a significant influence on the users immersive perception in the virtual scene.

A Leap Motion controller is front-mounted on the HTC Vive display and separately connected to Unity on the PC to sensor the users hand gestures which are converted motion commands afterwards. The user has the ability to issue commands by presenting simple and intuitive hand gestures in the Leap Motion field of view. Leap Motion controller is a non-contact motion tracking device which primarily focuses on hand gestures and finger positions. Furthermore, the Ubuntu computer and the Windows PC communicate with one another wirelessly in WebSocket protocol on the Internet.

III. Implementation

A. Hardware

To completely implement the system design of virtual reality and optical tracking based teleoperation, the system is categorized into four entities: motion tracking and virtual reality at a client (User), a mobile agent in remote working space (Robot), virtual reality generating platform (Unity) and communication and integration middleware (ROS). On the hardware side, the virtual reality teleoperation system includes an HTC Vive HMD and base stations which tack the exact locations of the HMD, a Leap Motion hand tracking sensor is with front-mounted onto the HMD to track the motion of the users hand movements. And a TurtleBot2 equipped with a computer and a 3D visual sensor serves as a base platform.

The concept implementation is based on an HTC Vive virtual reality device, a TurtleBot2 open source mobile robot and a Leap Motion's hand tracking sensor. HTC Vive is an off-the-shelf virtual reality device which includes a high-resolution HMD, hand controllers and base stations. The locations of the HMD is tracked accurately the base stations. In this setting, the user is privileged to move around and fully engage with the teleoperation in virtual reality instead of staying stationary at the fixed spot.

A Microsoft Kinect 3D sensor is mounted on the robot to acquire visual information and mapping in direct control mode and autonomous navigation mode respectively. The direct control of TurtleBot is accomplished by a Leap Motion sensor and the movement of the robot is controlled by the user with bare hands. The function of the Kinect sensor is used to capture the images of the remote environment and send the streams back to the HMD to provide the user with fully immersive perception while interacting with the robot. Further, mapping about the environment around the TurtleBot can be done simultaneously while the robot is driven around by the user with the Leap Motion controller in direct control. It serves as a basis of the autonomous mode of the robot. Autonomous mode of the robot is significantly augmented by the 3D mapping conducted in real time. The mapping function accomplished by the Kinect sensor provides the TurtleBot with the capability of building a map of its surroundings, evaluating the potential paths, navigating to a predetermined position on the map autonomously.

B. Software

Robot Operating System (ROS) is incorporated to control the robot and accomplish the integration of modular functionalities. Unity is utilized to provide with a virtual reality interface. Rosbridge is introduced to connect the ROS and Unity in terms of communication across operating systems.

Most of the state of art virtual reality devices and relevant development platforms concentrate on Windows operating systems. Virtual reality modules are currently not well incorporated into ROS to directly utilize and empower its functions, despite the fact that ROS features numerous robust robotic control, simulation and visualization resources to fulfill different modular functions in robotic systems.

Unity is utilized to provide with virtual reality interface and render the live video stream. As a multipurpose development platform that supports scripting using C sharp, Unity has the capability of creating multiplatform 2D and 3D scenes and providing the user with interactive experiences. And it has built-in support for a number of off-the-shelf virtual reality devices, such as HTC Vive. The implementation of Unity is an effective approach to build interaction with the robot in teleoperation with virtual reality. Thus, an intermediary communication layer between Unity and ROS needs to be integrated into the system. Fig. 4 demonstrates the general schema of the implementation and the two-way communication among four essential entities which are the user, Unity, ROS and the mobile robot respectively. To communicate between Unity on Windows and ROS on Linux, Rosbridge suit is implemented as a feasible and effective communication layer. It allows the system to take advantage of both the convenience of virtual reality development platforms in Windows and the versatility of ROS modular functions. Once the communication between Unity and ROS is established, the user interacts with Unity directly by giving commands with Leap Motion controller and obtaining the virtual scene displayed in the HMD. Furthermore, the communication between ROS and the mobile robot is conducted through messages in different types of Topics.

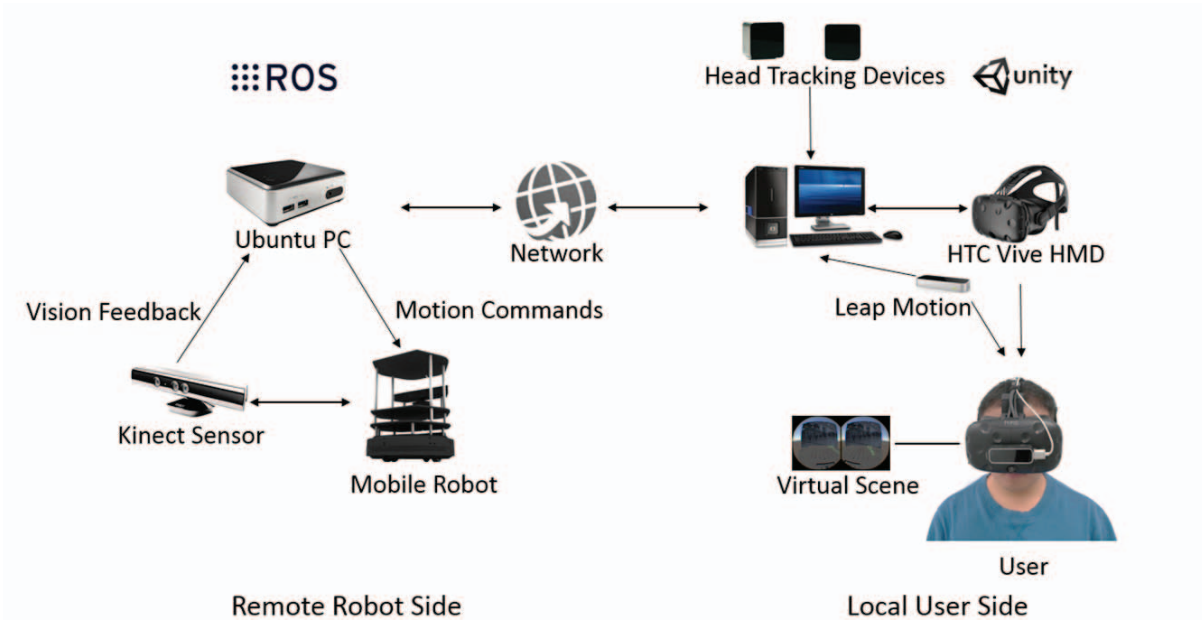

Fig. 3. A system concept and main components of the proposed teleoperation system.

When the communication between 3D development engine and ROS system is built up, an immersive user interface can be developed with the library and tools provided by the 3D engine and the different levels of robot control can be achieved by taking advantages of ROS system. It facilitates the implementation of virtual reality with ROS to integrate its modular robotic functionalities. Furthermore, in ROS the robot modeling is represented in Unified Robot Description Format (URDF), which is XML based robot description. When the connection between Unity and ROS is successfully built, the URDF model of Tutlebot2 is imported in Unity by publishing data to parameter server in ROS. Hence, the poses and movements of the virtual robot in Unity adapt to the ones of its real counterpart in the remote environment simultaneously.

C. Network configuration

ROS is given the capability to manage multiple computers to maintain communication and share and only one of the computers in the network can be the Master. An Intel NUC computer mounted on the top of TurtleBot and it can access to the Internet with Wi-Fi. The IP address of the NUC computer connected to the wireless network is assigned to ROS Master Address. Thus, the NUC computer resides on TurtleBot is defined as the ROS Master to a remote desktop computer. And on the remote computer ROS Master variable is accordingly set to the address of NUC computer. Once a network connection is accomplished, nodes on the remote computer can register with the Master on TurtleBot and the remote computer is able to communicate with commands to the NUC computer wirelessly.

IV. Levels of robotic control

A. Direct control

In our telerobotic system, a method is developed to convert Leap Motion tracking data to control input into Unity which are translated into velocity commands subsequently in ROS. A Leap Motion controller is connected to Unity in order to provide sensor messages and these messages are translated into /sensor msgs/joy message type sending to ROS via Rosbridge WebSocket[19]. A node in ROS converts the received messages from Leap Motion into geometry msgs/ Twist messages which are six-dimensional values representing the linear velocity along each axis and the angular velocity around each axis. Twist message type is used in ROS for publishing motion commands to a base controller node which combines the motor driver and PID controller.

TurtleBot is a type of differential drive robot moving only forward or backward along its horizontal axis and rotate only around its vertical axis. Hence, only the linear x component and the angular z component are required to portray its motion. In the case of the TurtleBot, Twist messages indicate the linear velocity of the Turtlbot along its forward x-axis and the angular velocity around its vertical z-axis (the other values remain zero). And the base controller node subscribes to the /cmd vel topic and translates geometry msgs/ Twist messages into motor signals which in reality drive the wheels. The robot's base controller node turns motion commands into real-world velocities of the mobile robot utilizing internal position data and PID control. Thus, the proposed method can successfully convert Leap Motion sensor messages into real-world velocities of the mobile robot in order to realize optical tracking based teleoperation. Fig. 5 shows a flowchart demonstrating the hand gesture based robotic control. Leap

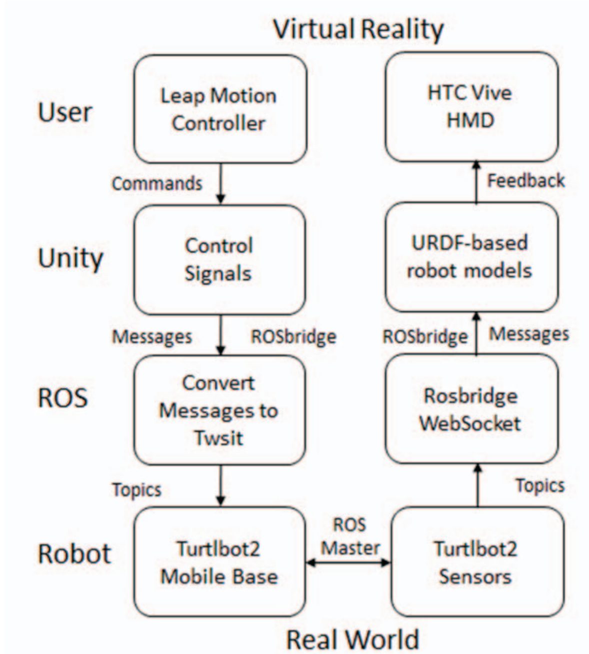

Fig. 4. A general schema of the implementation and the two-way communication among four essential entities.

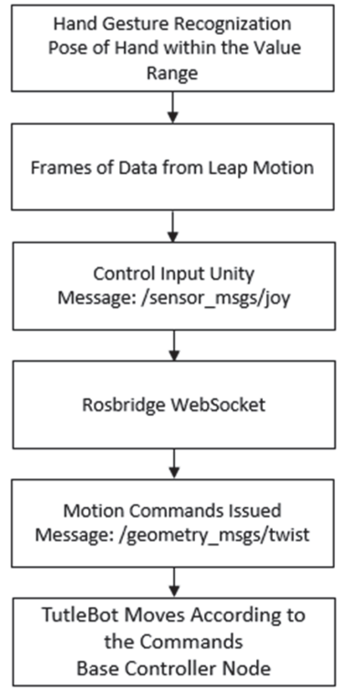

Fig. 5. A flowchart for hand gesture based robotic control.

Motion is used as the interface for hand gesture based robotic control. A set of specific range values of hand gestures are defined with regard to roll, pitch and yaw angles for hand gesture tracking and recognition. The Cartesian position and orientation of the users hand are retrieved from the Leap Motion sensor. Once the users hand is detected in the Leap Motion sensors field of view and the data generated are within the given range. The tracking data are returned by Leap Motion sensor in the form of frames and certain motion commands are issued to the motor drivers which are installed on the mobile base of the robot. Pitch angle values are changed into moving forward/backward. Roll angle values are converted into turning left/right. Thus, the robot is able to perform actions move forward, move backward, turn clockwise and turn anticlockwise accordingly. Table 1 shows the hand commands input to Leap Motion Controller and relevant robot movements. The pose of the palm is described with the pitch and roll range values (and along pitch and roll axes). The teleoperated TurtleBot publishes topics including /odom, /joint states and /camera/rgb/image raw/compressed to Unity and the relevant messages are synchronized to the URDF robot model in Unity. Thus, the user is able to control the remote TurtleBot to move by interacting with the virtual TurtleBot displayed in the HTC Vive HMD with bare hands only.

B. Autonomous Navigation

The capabilities of the vision system on TurtleBot are shown in Fig. 6. 2D colour images of the working space can be acquired from the RGB camera on Kinect. Point cloud of working space can be obtained with the depth camera on Kinect. In autonomous navigation, the vision system on TurtleBot will be used to create a map of the environment. The map is loaded into Rviz (ROS visualization tool) and the TurtleBot can autonomously navigate to a location selected on the map. Simultaneous localization and mapping (SLAM) approach is applied in teleoperation to provide prerequisites for autonomous navigation. SLAM is one type of approach for mobile robots to create or update a map of an unknown environment and simultaneously perform tracking the exact location and orientation of the robot in accordance with the environment. In particular, TurtleBot accomplishes the mapping by using the ROS gapping package based on OpenSlams GMapping, which is an exceedingly efficient Rao-Blackwellized particle filter algorithm to create grid maps according to the required laser range data.

The built-in CMOS sensor on Kinect continuously captures image data and the data are subsequently converted into depth information showing the distance that each infrared beam has moved[20]. Simultaneously the actual robot is communicating with a workstation which is running Rviz. The current configuration of the virtual replica robot model will be expressed

TABLE I
HAND POSE LIST AND THE RELEVANT MOVEMENT OF TURTLEBOT

Hand Gesture	Range Values	TutleBot2 movement
Hand pitch high	30 to 60	Move forward
Hand pitch low	-30 to 60	Move backward
Hand roll clockwise	50 to 80	Rotate to left
Hand roll anticlockwise	-50 to -80	Rotate to right

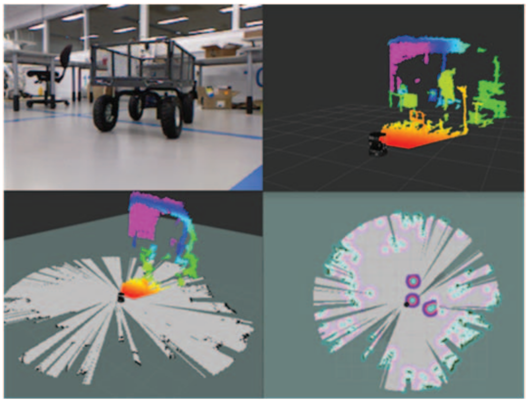

Fig. 6. 2D color images of the working space(upper left). Point cloud of the working space(upper right). Environment mapping in Rviz (lower left and right).

in Rivz. 3D sensor data from the Kinect sensor can be displayed in the form of point clouds which is a type of 3D representation of the depth image. Simulated laser scanning is provided by Kinect sensor and it serves as laser range data which is integrated to the laser-based SLAM approach. The slam gmapping node in ROS package accepts laser scan stream as input and converts it into the odometry frames.

While the robot is teleoperated around the working space, the gmapping node in ROS publishes the map topic to update the occupancy grid map (OGM) with the location and the surrounding environment of TurtleBot in compliance with the scanning data from the Kinect sensor. With the environment map constructed, the user is able to command TurtleBot to move from its current location to a given spot on the map. Thus, the user can manipulate the mobile agent in a comfortable and relaxed manner and has the capability to take over the immersive and intuitive direct control when necessary.

V. Conclusion

In this paper, we have developed an optical tracking based teleoperation system involving virtual reality. Specifically, a method is developed to convert Leap Motion tracking data into velocity commands of the robots mobile base. The system particularly combines the strengths of both ROS and Unity in network architectures across different operating systems. With modularity, compatibility and extensibility provided in ROS, the system has the capability to incorporate extra modular functionalities and to enhance and realize human-robot interaction with versatile robotic systems.

The system has the ability to immerse the human user into the working space of virtual reality with interactions that make the user feel like he is in the robots action and to ensure intuitive human-robot interaction based on hand gesture recognition and motion tracking. Two levels of autonomy control of the robotic teleoperation systems allow both intensive human intervention and autonomous capability.

In the future, we plan to enrich the user interface with the feedback from the robot and other essential functionalities.

References

[1] S. Muszynski, J. Stuckler, and S. Behnke, Adjustable autonomy for mobile teleoperation of personal service robots, in Proceedings - IEEE International Workshop on Robot and Human Interactive Communication, 2012, pp. 933-940.

[2] J. I. Lipton, A. J. Fay, and D. Rus, Baxters Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing, 2017.

[3] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel, Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation, 2017.

[4] E. Matsas and G.-C. Vosniakos, Design of a virtual reality training system for humanrobot collaboration in manufacturing tasks, Int. J. Interact. Des. Manuf., pp. 1-15, 2015.

[5] M. Mostefa, L. K. El Boudadi, A. Loukil, K. Mohamed, and D. Amine, Design of mobile robot teleoperation system based on virtual reality, Control. Eng. Inf. Technol. (CEIT), 2015 3rd Int. Conf., no. 1, pp. 1-6, 2015.

[6] J. Jankowski and A. Grabowski, Usability Evaluation of VR Interface for Mobile Robot Teleoperation, Int. J. Hum. Comput. Interact., vol. 31, no. 12, pp. 882-889, 2015.

[7] S. Park, Y. Jung, and J. Bae, A tele-operation interface with a motion capture system and a haptic glove, in 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2016, 2016, pp. 544-549.

[8] K. Kukliski, K. Fischer, I. Marhenke, F. Kirstein, M. V. Aus Der Wieschen, D. Slvason, N. Krger, and T. R. Savarimuthu, Teleoperation for learning by demonstration: Data glove versus object manipulation for intuitive robot control, in International Congress on Ultra Modern Telecommunications and Control Systems and Workshops, 2015, vol. 2015-Janua, no. January, pp. 346-351.

[9] B. Fang, F. Sun, H. Liu, and D. Guo, A novel data glove using inertial and magnetic sensors for motion capture and robotic arm-hand teleoperation, Ind. Robot An Int. J., vol. 44, no. 2, pp. 155-165, 2017.

[10] D. Bassily, C. Georgoulas, J. Gttler, T. Linner, T. Bock, and T. U. Mnchen, Intuitive and adaptive robotic arm manipulation using the leap motion controller, Isr Robot., pp. 78-84, 2014.

[11] S. Scheggi, L. Meli, C. Pacchierotti, and D. Prattichizzo, Touch the virtual reality: using the Leap Motion controller for hand tracking and wearable tactile devices for immersive haptic rendering, ACM SIGGRAPH 2015 Posters, p. 31, 2015.

[12] L. Peppoloni, F. Brizzi, C. A. Avizzano, and E. Ruffaldi, Immersive ROS-integrated framework for robot teleoperation, 2015 IEEE Symp. 3D User Interfaces, pp. 177-178, 2015.

[13] Y. Kim, P. C. W. Kim, R. Selle, A. Shademan, and A. Krieger, Experimental evaluation of contact-less hand tracking systems for teleoperation of surgical tasks, in Proceedings - IEEE International Conference on Robotics and Automation, 2014, pp. 3502-3509.

[14] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, ROS: an open-source Robot Operating System, Icra, vol. 3, no. Figure 1, p. 5, 2009.

[15] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex, Comparing Robot Grasping Teleoperation across Desktop and Virtual Reality with ROS Reality, in International Symposium on Robotics Research, 2017, pp. 1-16.

[16] D. Krupke, P. Lubos, G. Bruder, J. Zhang, and F. Steinicke, Natural 3D Interaction Techniques for Locomotion with Modular Robots, no. Hirose 1993, 2015.

[17] S. A. Green, M. Billinghurst, X. Chen, and J. G. Chase, Human-Robot Collaboration: A Literature Review and Augmented Reality Approach in Design, International Journal of Advanced Robotic Systems, vol. 5, no. 1. pp. 1-18, 2008.

[18] M. A. Goodrich, D. R. O. Jr, J. W. Crandall, and T. J. Palmer, Experiments in Adjustable Autonomy Robot Effectiveness Neglect, IJCAI Work. Auton. Deleg. Control Interact. with Intell. Agents, pp. 1624-1629, 2001.

[19] R. Codd–Downey, P. M. Forooshani, A. Speers, H. Wang, and M. Jenkin, From ROS to unity: Leveraging robot and virtual environment middleware for immersive teleoperation, in 2014 IEEE International Conference on Information and Automation, ICIA 2014, 2014, pp. 932-936.

[20] K. K. Biswas and S. K. Basu, Gesture recognition using Microsoft Kinect, Automation, Robotics and Applications (ICARA), 2011 5th International Conference on. pp. 100-103, 2011.