Virtual Submerged Floating Operational System for Robotic Manipulation

Qin Zhang, Jialei Zhang, Ahmed Chemori, Xianbo Xiang

WILEY | Hindawi

Research Article

Virtual Submerged Floating Operational System for Robotic Manipulation

Qin Zhang [iD],[1] Jialei Zhang,[2] Ahmed Chemori,[3] and Xianbo Xiang [iD][2,4]

[1]*State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China*
[2]*School of Naval Architecture and Ocean Engineering, Huazhong University of Science and Technology, Wuhan 430074, China*
[3]*LIRMM, CNRS - University of Montpellier, 161 rue Ada, 34392 Montpellier, France*
[4]*Shenzhen Huazhong University of Science and Technology Research Institute, Shenzhen 518057, China*

Correspondence should be addressed to Qin Zhang; qin.zhang@mail.hust.edu.cn and Xianbo Xiang; xbxiang@mail.hust.edu.cn

In this work, a virtual submerged floating operational system (VSFOS) based on parallel and serial robotic platforms is proposed. The primary aim behind its development lies in carrying out simulated underwater manipulation experiments in an easier and safer way. This VSFOS is consisted of a six-degree-of-freedom (6-DOF) parallel platform, an ABB serial manipulator, an inertial sensor, and a real-time industrial computer. The 6-DOF platform is used to simulate the movement of an underwater vehicle, whose attitude is measured by the inertial sensor. The ABB manipulator, controlled by the real-time industrial computer, works as an operational tool to perform underwater manipulation tasks. In the control system architecture, software is developed to receive the data collected by the inertial sensor, to communicate and send instructions. Furthermore, the real-time status of the manipulator is displayed in this software. To validate the proposed system, two experiments have been conducted to test its performance. In the first experiment, the test is carried out to check the communication function of VSFOS, while in the second one, the manipulator is intended to follow the movement of the parallel platform and perform simulated operational task in the space. The obtained results from these two experiments show clearly the effectiveness and the performance of the proposed VSFOS.

1. Introduction

As the ocean attracts great attention on environmental issues and resources, the need for and use of underwater operational systems become more and more demanding [1–3]. With significant advances of different types of mobile robots and manipulators in a bunch of application fields [4–9], marine robots including unmanned surface vehicles (USVs) [10–12], remotely operated vehicles (ROVs) [13, 14], and autonomous underwater vehicles (AUVs) [15–18] have been well developed in last decades [19–22]. Recently, submerged floating operational systems (SFOS), represented by underwater vehicles equipped with manipulators, have evolved from laboratory engineering prototypes into mature field tools for underwater operational applications [23–27]. As illustrated in Figure 1, SFOS can replace a human diver to perform underwater robotic tasks in harsh and dangerous environments, such as underwater salvage, and underwater pipeline inspection, ship maintenance [18, 28–31].

Owing to its great application perspectives, research and investigations concerning SFOS have already been launched all over the world [32, 33]. The Deep Submergence Operations Group developed the Jason ROV at the WHOI Deep Submergence Laboratory in 1989 [34]. Equipped with a custom-designed 5-axis robotic arm, this ROV is able to perform a wide variety of scientific tasks including temperature measurements and underwater deployment, as well as sampling of sediments and biological organisms. Within the European project TRITON, a manipulator installed on the autonomous underwater vehicle was used to open/close a valve [35]. The Nereus vehicle designed by Deep Submergence Lab at WHOI, USA, in 2008 was carrying a six-degree-of-freedom (DOF) electrohydraulic robot arm to conduct the benthic survey operations [36, 37]. The Polar-ARV designed by the
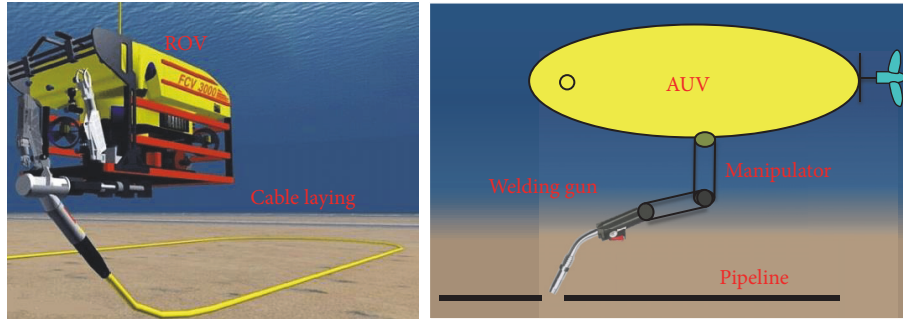
FIGURE 1: Illustration of a submerged floating operational system (SFOS). Left: a remotely operated vehicle (ROV) equipped with a manipulator laying the cables on the sea floor. Right: an autonomous underwater vehicle (AUV) carrying a torch for underwater welding.
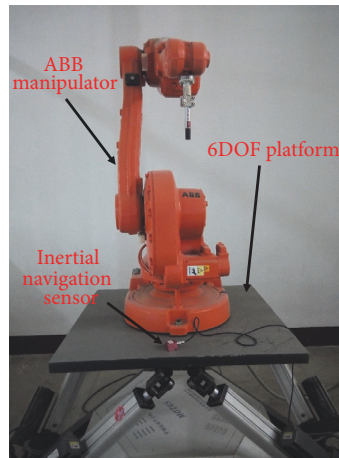

FIGURE 2: View of the proposed virtual submerged floating operational system (VSFOS) in the laboratory.

Shenyang Institute of Automation (SIA) can work as a moving underwater platform for monitoring marine environment under ice [38].

Although these applications of SFOS have already been so extensive, there are still a lot of limits when it comes to the experimental research and system test of SFOS. These limitations are as follows: First of all, based on underwater operational experiments of SFOS, it may cost a lot of time and money as well as many labor force. Secondly, suitable environment conditions such as weather condition will be required for the success of the underwater operations. Thirdly, harsh and dangerous undersea environments with high uncertainty may render the experiments very delicate to be carried out [39–43].

In order to help researchers performing all kinds of underwater operational experiments, a virtual submerged floating operational system (VSFOS) was built up at the school of Naval Architecture and Ocean Engineering in Wuhan, China. As illustrated in Figure 2, the VSFOS is mainly composed of four parts, the parallel kinematic platform, the ABB serial manipulator, the inertial navigation sensor, and the real-time industrial computer. The ABB manipulator is installed on the upper surface of the parallel platform executing an operational intervention tool. At the same time,

the 6-DOF platform completes its translational and rotational motions in the 3D space to simulate the movement of the underwater vehicle and environmental disturbances. The inertial navigation sensor is mounted on the moving plate of the parallel platform to monitor its attitude change. The main control part of the VSFOS is a real-time industrial computer, responsible for providing instructions to the ABB serial manipulator. In fact, the underwater vehicle and carrying manipulator are disturbed with complex underwater environment. The manipulator and the platform are joined rigidly in underwater operation system and VSFOS. And the ocean disturbance on the manipulator robot is difficult to simulate actively. Hence, for the control scheme of the manipulator, the disturbances from underwater environment and platform can be equivalent to the slow disturbance from the platform. In this sense, the proposed VSFOS is used to replace the underwater vehicle and support the manipulator. Yet, for the manipulator, the parallel platform is used to generate all external disturbances to the manipulator. An inertial navigation sensor is mounted on the lower surface of the 6-DOF platform to monitor its attitude change. The main control part of our VSFOS is a real-time industrial computer, responsible for providing instructions to the ABB serial manipulator.

Equipment	Composition	Industrial applications	Role in VSFOS
ABB manipulator	Manipulator/Controller	welding/assembly handling	Operational tool
6-DOF parallel platform	electric cylinders/AC servo motors/steel platform	virtual reality for simulated driving cars, aircraft	operational: Simulates the underwater vehicles
Inertial navigation sensor	gyroscope/accelerometer/accelerometer/thermometer	automobile/aircraft/ocean	measures the movement of the 6DOF platform

In terms of control point of view, two critical questions may arise. The first one deals with the communication problem with the ABB manipulator, where a lot of efforts have been made to break the communication blockade of this industrial manipulator to be able to have access to the low level control. Indeed, commercial industrial robotic manufacturers typically provide a joystick or a static off-line programming interface for the customers, where the control type is mainly appropriate for simple and respective tasks, but not for the VSFOS since a sophisticated real-time control of the manipulator is necessary [44]. To overcome this problem, we developed control software with an open monitoring interface [45], thanks to which, the communication bridge between the manipulator and the industrial computer was successfully built up.

The second critical question is how to simulate the underwater operational tasks making use of the hardware equipment of the VSFOS. Controlling the 6-DOF parallel platform and the ABB manipulator separately is a relatively easy task. However, when it comes to controlling the manipulator to do some accurate operations while its base is following the movement of the parallel platform, the task will be much more challenging. To achieve this function, two requirements ought to be met. Firstly, the attitude change of the parallel platform should be measured in real-time; secondly, the manipulator ought to compensate the movement of the platform in real-time. Indeed, the inertial navigation sensor can work under the condition of high speed, and control software was developed to perform the real-time control of the manipulator. In this way, the two requirements are fulfilled. As a result, the VSFOS can be used to conduct the simulated underwater experiments in a laboratory environment.

To sum up, this paper mainly introduces a new virtual submerged floating operational system (VSFOS) designed to make it easier and safer to conduct underwater operational simulations. The organization of this paper is as follows. In Section 2, a detailed description of our VSFOS, including hardware and software, is presented. The solution to the two critical questions regarding the control of the VSFOS system is detailed in Sections 3 and 4. In Section 5, simulation and real-time experiments are presented and discussed. The last part presents the concluding remarks and future works.

2. System Architecture

In this section, the hardware framework of VSFOS is presented firstly with a description of each component, and then the connections of all the hardware equipment are presented in the software description part.

2.1. Description of the Hardware. VSFOS consists of five main equipment pieces, the six-degree-of-freedom (6-DOF) parallel robotic platform, the ABB serial robotic manipulator, the manipulator controller, the inertial navigation sensor, and the real-time industrial computer. The overall architecture of the system is illustrated in Figure 3.

The 6-DOF parallel platform is fixed on the ground with three bolts. A laptop is connected to the platform through an ethernet cable to send the control commands to drive the platform. The ABB manipulator is mechanically attached to the moving base of the parallel platform by three positioning screws. Consequently, the base of the serial manipulator follows the movement of the 6-DOF parallel platform. The inertial navigation sensor is fixed on the upper surface of the platform to measure in real-time the orientation of the platform. The information collected by this sensor is transmitted to the real-time industrial computer through a USB data cable. The industrial computer and the manipulator controller are connected through another ethernet cable, thanks to which, the control instructions can be sent to the manipulator controller and the real-time status can be fed back to the computer.

The features description of the ABB manipulator, the 6-DOF platform, and the inertial navigation sensor is summarized in Table 1.

The ABB manipulator IRB 1600 is a 6-axis industrial manipulator produced by ABB Company. This manipulator is equipped with a strong power system, and its end-effector can carry a payload do up to 10 Kg; moreover, it is also suitable for high precision operations, since its positioning accuracy is 0.35 mm and its repeatability is 0.05 mm. Owing to these characteristics, the IRB 1600 is widely used in many industrial tasks, like arc welding, packaging, and material handling. Taking advantage of its strong power and high precision, we make it an operational tool in the proposed VSFOS.

The 6-DOF parallel platform created by MOTUS Company is a six-degree-of-freedom parallel kinematic machine (PKM). Its six legs are composed of electric cylinders, equipped with six AC servo motors inside the legs to control the piston rod of the electric cylinders. This platform can be typically used to provide a simulation environment for automobile driving, aircraft piloting, and some other entertainment devices. Therefore, we combine its translational and rotational motions to simulate the movement of underwater vehicles in VSFOS, as illustrated in Figure 16.
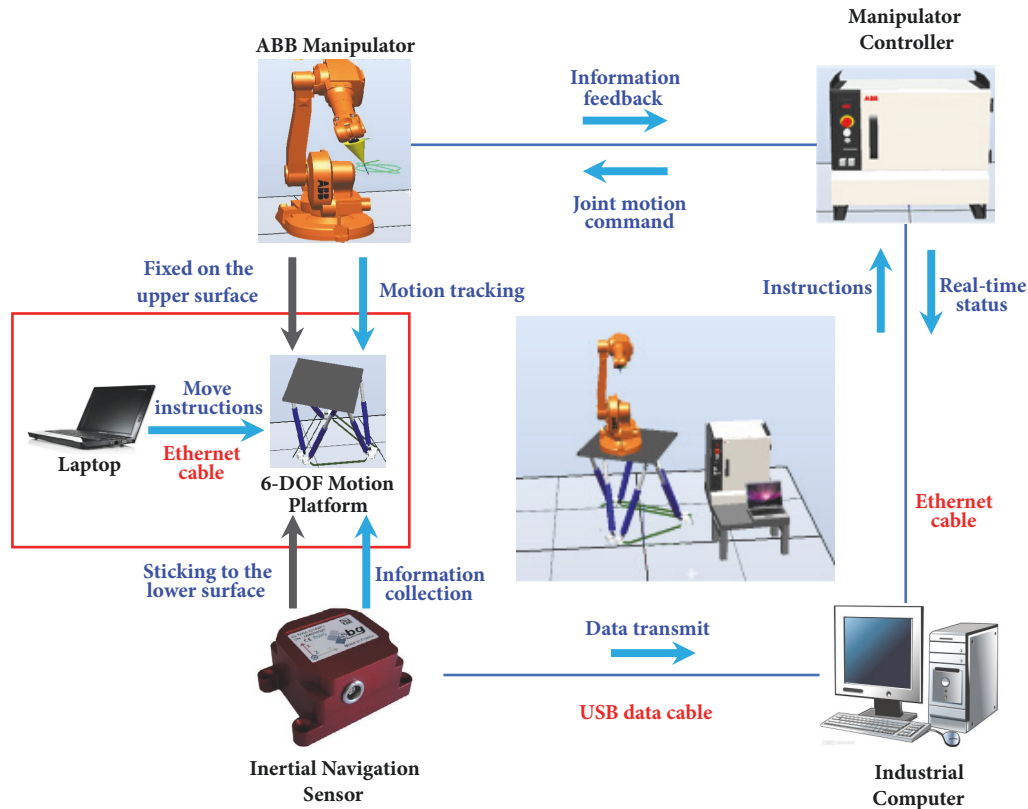
FIGURE 3: Illustration of the overall composition of the VSFOS. Connections between all the components are represented by arrows, and the overall layout of the system is in the middle of the picture.

The INS (inertial navigation sensor) IG-500A is used to accurately measure the attitude angles of the VSFOS. Based on the MEMS (Microelectromechanical Systems) technology, the INS IG-500A can calculate the precise 3D direction data by 3 gyroscopes, 3 accelerometers, 3 magnetometers, and a thermometer. It is commonly used in automobile, aerospace, and other vessels. In our VSFOS, the INS IG-500A is used to measure the attitude angles of the 6-DOF parallel platform.

2.2. Software Description. In order to control the whole system and monitor it in real-time, control software programmed in C SHARP language is developed. Taking advantage of this software, different functions have been developed for the following purposes:

(i) receiving the information collected by the inertial navigation sensor through serial port communication;

(ii) planning the trajectory for the serial manipulator by combining the received data, the manipulator kinematics, and the manipulator motion constraints;

(iii) transmitting motion instructions to the IRC5 controller and get the real-time status of the manipulator back through network communication.

In summary, three main functions of the software which ensure well operating conditions are the serial port communication, the manipulator control, and the network communication, as illustrated in Figure 4.

The sensor sticking to the upper moving base of the 6-DOF parallel platform can measure the attitude angles of the platform. The collected data will then be sent to the industrial computer through the serial port, after being filtered and analyzed. The attitude measurement information will be parsed out from the processed data. As illustrated in Figure 4, the letter S refers to the processed data, and the letter T denotes the calculated homogeneous transformation matrix. Once the 6-DOF parallel platform starts moving, the manipulator ought to follow the motion. In order to make each joint of the manipulator moving to the target position smoothly, we need to plan carefully the reference joint trajectory for the mobile manipulator. Finally, the joint angles instructions newly generated will be sent to the controller through the network communication; meanwhile, the real-time status of the manipulator will be fed back to the industrial computer.

2.2.1. Serial Port Communication. The data collected by the inertial navigation sensor are transmitted to the industrial computer through the serial port communication. The received data should be packaged in a certain format
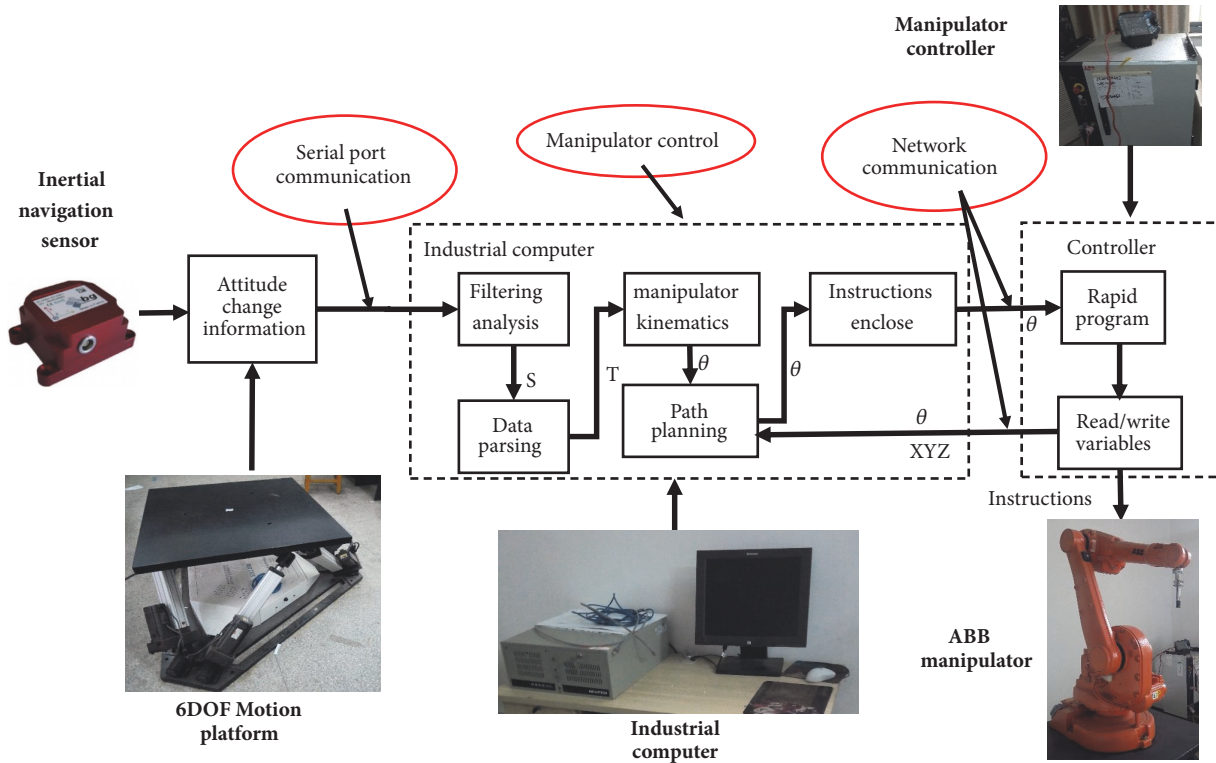
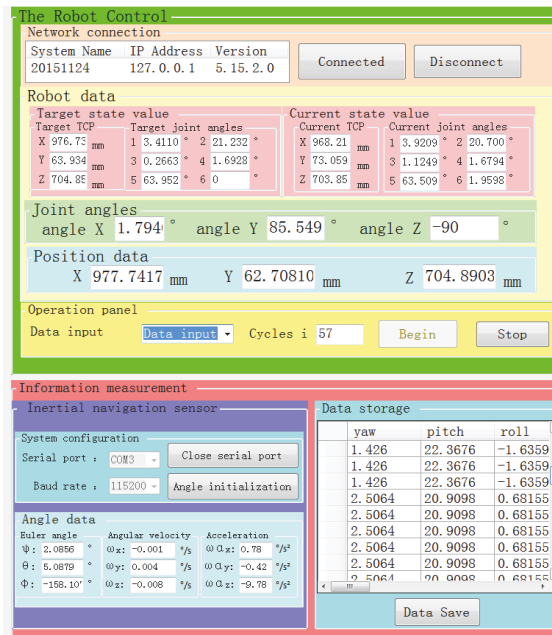FIGURE 4: View of the proposed software architecture of the VSFOS.

FIGURE 5: View of the control software interface.

according to the communication protocol; thus, we can parse and analyze it according to this protocol. The communication protocol is described in Table 2.

As shown in Figure 5, the serial port communication unit is displayed at the bottom of the control software interface. To receive the collected data, we need to choose the proper port and bound rate. Then by clicking the "Open serial port" button, the software will parse the data and display it in the corresponding frame. For example, the "Angle data" frame tells the Euler angle of the platform relative to the base coordinate system as well as its linear velocity and angular acceleration; the "Angle initialization" button is set to define
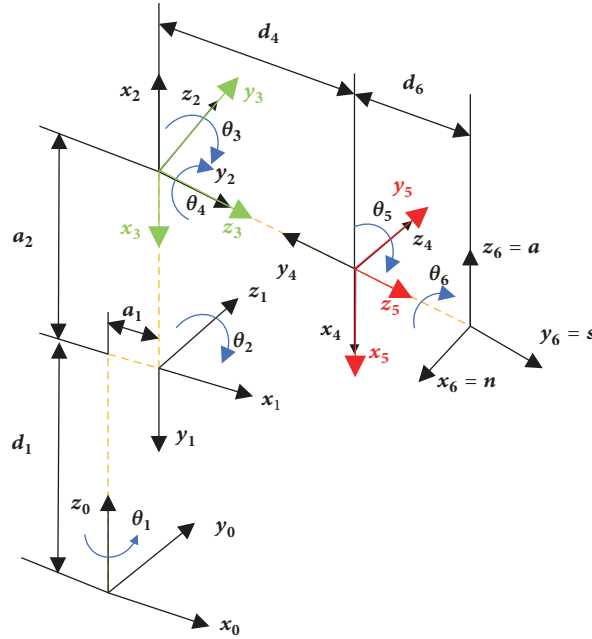
FIGURE 6: Joint diagram of IRB 1600. x_i, y_i, z_i represent the coordinate axes. a_i, θ_i, d_i α_i refer to the D-H parameters.

TABLE 2: Serial port communication protocol.

Frame domain	Size(bit)	Description
Packet header	1	0XA5
Data ID	1	01~06
Data length	1	n
Data domain	0~8	Data
CRC check	16	Cyclic Redundancy Check
Packet Tail	1	0X5A

the initial state of the sensor. The bottom right side of the interface is an Excel control, which can be used to display all the current data, as well as save and export it to an Excel file.

2.2.2. Manipulator Control. The manipulator control scheme gives joint angles instructions to joint motors to make the end-effector reach the right position and orientation. Therefore, it is necessary to set a target value for each joint and monitor the real-time position and orientation of the end-effector. As shown in Figure 5, in the manipulator control unit, the "Operational panel" frame is used to choose the planning trajectory for the manipulator, and the "Position data" and "Joint angles" frames are designed to display the position and orientation information of the TCP (tool center point). In addition, there is a "Manipulator data" frame, in which the target and current position and joint angles of the manipulator are displayed.

2.2.3. Network Communication. In order to perform the real-time control of the manipulator, we choose to communicate with the manipulator controller through the network

communication. As illustrated in Figure 5, the network communication part is located at the top of the interface. Once starting the software, by clicking the "Connect" button, it will search the on-line manipulator controller automatically. There are two pieces of information to identify the correct controller, which are the "system name" and the "IP address". Once the connection with the manipulator controller is set, we get access to exchange data with the manipulator controller. In other words, we can write variables to the controller to update the position and orientation of the end-effector. Meanwhile, the system variables are available to observe the real-time status of the manipulator, which include joint angles and the position information of the TCP.

3. Manipulator Kinematics

The ABB serial manipulator IRB 1600 is a robotic arm with six rotational joints, equipped with an operation tool installed at the end joint. In order to get the position and orientation of this TCP (tool center point) with respect to the base, it is necessary to model the manipulator mathematically. The universal way to model a manipulator is based on the Denavit-Hartenberg (D-H) method [46]. The D-H method introduces the four steps of manipulator model, which are manipulator modeling, forward kinematics, inverse kinematics, and path planning.

3.1. Manipulator Modeling. The first step in the Denavit-Hartenberg (D-H) method is to build the D-H model of the manipulator; this model contains two parts, namely, the joint diagram of the manipulator and the D-H parameters table. The joint diagram can be drawn according to the joint features and mechanical structure parameters, as illustrated in Figure 6.

TABLE 3: Denavit-Hartenberg parameters table of IRB 1600 robot manipulator.

i	$a_i(mm)$	$\alpha_i(°)$	$d_i(mm)$	$\theta_i(°)$
1	150	-90	486.5	θ_1
2	700	0	0	θ_2-90
3	0	90	0	θ_3+180
4	0	-90	600	θ_4
5	0	90	0	θ_5
6	0	90	65	θ_6-90

TABLE 4: D-H parameters explanation.

Parameters	Meaning	Detailed explanation
a_i	Link length	the distance from joint axis (i-1) to joint axis i
α_i	Link twist	the angle from joint axis (i-1) to joint axis i
d_i	Link offset	the distance from joint X_{i-1} to X_i
θ_i	Joint angle	the angle between joint X_{i-1} to X_i

Based on the joint diagram and the products brochure of the ABB manipulator IRB 1600, the key parameters can be listed in Table 3. The key parameters are reconfirmed by manoeuvring the manipulator and observing the data feedback on the host computer.

Four important D-H parameters used in D-H method, namely, $a_i, \alpha_i, d_i, \theta_i$, are explained in Table 4.

3.2. Manipulator Forward Kinematics. To describe the translation and rotation of the frame $i-1$ with respect to the frame i, a 4×4 homogeneous transformation matrix $^{i-1}_i A$ is used with the following structure.

$$^{i-1}_i A = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & \cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

As explained above, the IRB 1600 manipulator has six rotational joints. Then, to get the transformation matrix from the base frame to the TCP, we are supposed to multiply all the transformation matrices together from i=1 to i = N, where N is the number of joints of the manipulator. The mathematical expression of the transformation matrix from the base frame to the TCP is then given by the following.

$$^0_6 T = ^0_1 A \times ^1_2 A \times ^2_3 A \times ^3_4 A \times ^4_5 A \times ^5_6 A \quad (2)$$

All of the D-H parameters used in the transformation matrix are constant except for the joint angles θ_i. Therefore, once the values of the joint angles are obtained, it is simple and straightforward to get the transformation matrix $^0_6 T$, in which the position and orientation of the end-effector with respect to the base are calculated.

3.2.1. Manipulator Inverse Kinematics. In the general sense, the inverse kinematics of manipulator means calculating the joint angles when the position and orientation of the end-effector is given. This problem is much more complex than the forward kinematics due to the fact that there often exists more than one set of joint angles that satisfy the given position and orientation. For the case of the IRB 1600 manipulator, on the premise of knowing the transformation matrix $^0_6 T$ between the base center and the TCP of the manipulator, the process of solving the inverse kinematics equation to get the joint angle of joint 1 is as follows.

$$^0_6 T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The elements $n_{x,y,z}, o_{x,y,z}, a_{x,y,z}$ refer to the attitude of the operational tool, and the element $p_{x,y,z}$ refers to its position with respect to the base. Similarly, another expression of the transformation matrix $^0_6 T$ is as follows.

$$T = A1 \times A2 \times A3 \times A4 \times A5 \times A6 \quad (4)$$

Multiplying both sides of (4) by the inverse of the matrix A1, we get the following.

$$A1^{-1} \times T = A2 \times A3 \times A4 \times A5 \times A6 \quad (5)$$

Calculating both sides of (5), the matrix on the left side of the equation equals the one on the right. Thus, we get the following.

$$\theta_1 = \pm \arctan\frac{p_y - 650o_y}{p_x - 650o_x} \quad (6)$$

From (6), it can be seen that the joint angle θ_1 has two solutions.

$$\theta_1 = \arctan\frac{p_y - 650o_y}{p_x - 650o_x}$$

$$\theta_1 = \arctan\frac{p_y - 650o_y}{p_x - 650o_x} + \pi \quad (7)$$
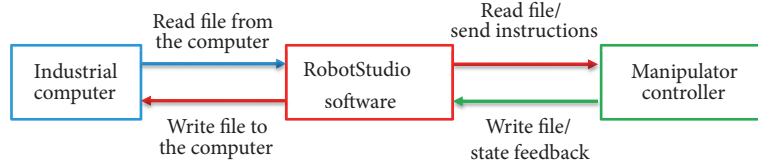
FIGURE 7: Communication architecture for read/write files.

In the same way, the rest of joint angles from θ_2 to θ_6 can be computed. Hence, there are multiple sets of solution for the manipulator robot. And there is a great deal of difference between the two solutions for each joint angle. In the numerical control process, the solution which has smaller difference with the joint angle in the last control period is preferred. Hence, the optimization principle can be ensured.

3.3. Path Planning. Path planning, also referred to as trajectory generation, is the act of describing the desired motion of a manipulator in multidimensional space over a period of time. To decrease the vibration of the manipulator at the initial and final points, it is very common to use cubic polynomials interpolation methods to generate point-to-point trajectories of the joint angles. Taking the current joint angles as the initial value and the joint angles calculated by solving the inverse kinematics equations as the target value, the planned joint angles expression with respect to time can be expressed by the following.

$$\theta_i(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \quad i \in (1,2,3,4,5,6) \quad (8)$$

Four constraints about the initial and final conditions of the joint angles are used to compute the coefficients of (8). These constraints are expressed by the following formula.

$$
\begin{aligned}
\theta_i(0) &= \theta_{0i}, \\
\theta_i\left(t_f\right) &= \theta_{fi}, \\
\dot{\theta}_i(0) &= 0, \\
\dot{\theta}_i\left(t_f\right) &= 0
\end{aligned}
\quad (9)
$$

By solving (8) and (9), we can evaluate the designed joint angles at any sample time.

4. Communication with the Robotic Manipulator

For the ABB manipulator, a simulation and off-line programming software, RobotStudio, are provided by the manufacturer. Programming with RAPID language and downloading it to the manipulator controller are allowed by using the RobotStudio software. Indeed, this might be the simplest method to achieve the communication with the manipulator compared with other manual control methods. However, it is not suitable for the VSFOS, due to the high real-time control demands of the VSFOS. Therefore, getting rid of the traditional control method for the manipulator and breaking

the communication blockade of this industrial manipulator are demanded. During our research and development of VSFOS, three methods have been tested to communicate with the manipulator controller, which are "read/write files", "socket communication", and "develop software using PC SDK"; detailed descriptions of these methods are presented in the following paragraph.

4.1. Read/Write Files. Read/write files make use of files to transmit information between the industrial computer and the manipulator controller. At the computer side, a ".doc" file is edited with control instructions inside. Meanwhile, at the controller side, a RAPID program is edited to receive and parse control instructions. Similarly, the current status of the manipulator can be written in another file and then fed back to the industrial computer. To realize the "sending" function, we take advantage of RobotStudio software, thanks to which, the industrial computer can get the "write access" of the manipulator controller. Detailed description of this communication architecture is illustrated in Figure 7.

The control variables of the manipulator, such as the "jointtarget" variables (joint angle) and the "robtarget" variables (position of TCP), are written in those files and then transmitted from the computer to the controller. Therefore, with RobotStudio software, conducting the transmission job, the values of these variables can be changed by reading/writing these files. Indeed, control instructions can be sent to the controller and the status information of the manipulator can be fed back. The principle of this method is very simple; however, when it comes to the problem of real-time constraints, this method cannot meet the needed requirements.

4.2. Socket Communication. This method ensures the communication function based on the client/server mode. The RobotStudio software installed on the industrial computer works as a client, and the manipulator controller connects to the industrial computer with an ethernet cable acting as a server. When the program starts, the client will first request for the "write access" from the server and then send motion instructions to the server periodically after obtaining the "write access". On the server's side, the controller will get these instructions to control the manipulator; meanwhile, real-time status of the manipulator will be fed back to the client; the specific process of the socket communication is illustrated in Figure 8.

It is necessary to edit programs on both RobotStudio software and the controller if this method is applied to realize the real-time control of the manipulator. Definitely, this method
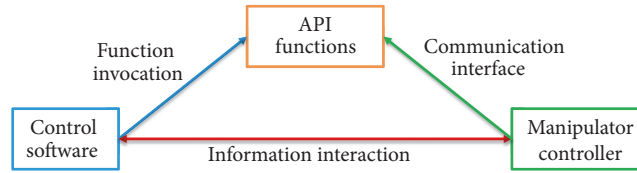
FIGURE 8: View of the socket communication.

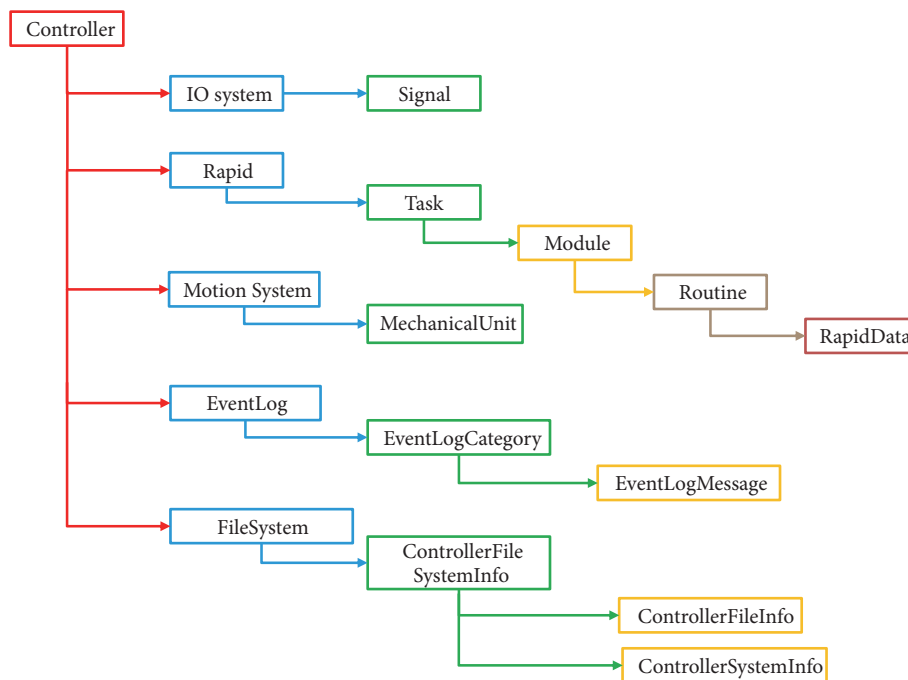FIGURE 9: Proposed PC SDK communication method.

FIGURE 10: View of the different classes of controller API.

allows us to control the manipulator in real-time; however, we cannot get rid of this RobotStudio software which does not provide an external device input interface. In other words, reading the data sent from the inertial navigation sensor with this software is not available; therefore, this method is also not suitable for our VSFOS.

4.3. Developing Software Using PC SDK. PC SDK (Software Development Kit) allows end-users to add their own customized operator interfaces for the IRC5 controller. Hence, users can control the manipulator over a network from a tailored application on a PC. Furthermore, this application can be developed with any programming language in any development environment. In our case, we developed a control software application written in C Sharp language in Visual Studio compiling environment. This software makes use of the API functions provided by the ABB Company to get

access to the manipulator controller very easily; the proposed solution is illustrated in Figure 9.

The classes used to access to the manipulator controller functionality together make up the controller API, in which there are many functions that can be used to obtain or change the value of the control variables of the manipulator. A part of the controller API object model is shown in Figure 10.

Instead of requesting the "write access" in advance, the control software can send instructions to the manipulator directly by invoking the API functions, as illustrated in Figure 10. Having connection with the controller, going through the "Rapid-Task-Module-Routine-RapidData" route, we can get access to the "RapidData". This last one model stores the information of the control variables of the manipulator. Hence, we can read the values of the variables to get the real-time information about the manipulator and send instructions to control the manipulator by writing new values for

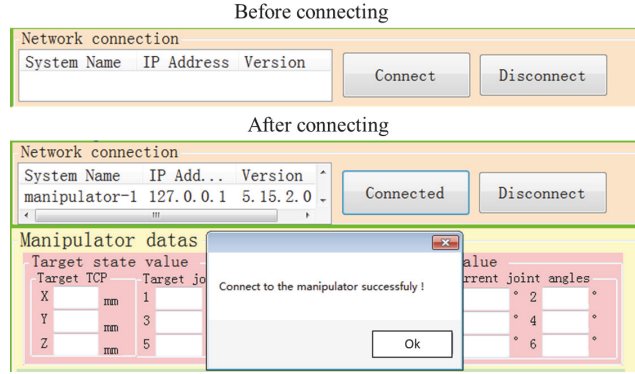Before connecting

After connecting

FIGURE 11: Connection to the manipulator controller.

the variables. The software interface is very simple and easy to operate with many integrated functions, making use of which we can truly realize the real-time control of the manipulator. Not restricted by the given RobotStudio software, as well as the programming language, this method is confirmed to be the best one to ensure the communication with the manipulator compared to the above two methods.

5. Simulation and Experimental Results

For the sake of validation of the performance of the proposed solution, two experiments have been carried out, one of which is a simulation test to demonstrate the network communication and function of the control software, while the second one is performed in the laboratory with the real VSFOS working in 3D space.

5.1. Simulation Scenario: Communication and Function Tests. In this simulation test, the control software and the Robot-Studio software are running on the same industrial computer. The RobotStudio software is used to build the manipulator model and create a virtual controller for IRB 1600. The default IP address of the virtual controller is "127.0.0.1", and for distinction, we change its system ID to "manipulator-1". The control software can be connected to the manipulator by searching the on-line controllers. Once the virtual controller is found, the network connection will be established after confirming the IP address and the system ID.

As shown in Figure 11, when the communication bridge between the controller and the control software is built up, the button name will change from "Connect" to "Connected". At the same time, a prompt box will pop up to inform the user of the connection status. Since the communication is unblocked, motion instructions can be sent to the controller. A particular geometrical path for the end-effector of the manipulator beforehand and the path function is expressed as follows.

$$
x = 800 + 200 \sin\left(2t + \frac{\pi}{2}\right)
$$
$$
y = 200 \cos\left(2t + \frac{\pi}{2}\right) \tag{10}
$$
$$
(0 \leq t < 100)
$$

$$
x = 1000 - 100\,(t - 100)
$$
$$
y = -\frac{10t}{\sqrt{3}}
$$
$$
(100 \leq t < 130)
$$

$$
x = 700
$$
$$
y = -100\sqrt{3} + 10\sqrt{3}t
$$
$$
(130 \leq t < 150)
$$

$$
x = 700 + 10t \tag{11}
$$
$$
y = 100\sqrt{3} - 10\sqrt{3}t
$$
$$
(150 \leq t < 180)
$$

$$
x = 1000 - 7.5t
$$
$$
y = -\frac{7.5t}{\sqrt{3}}
$$
$$
(180 \leq t < 200)
$$

$$
x = 800 + 100 \sin\left(2t + \frac{5\pi}{6}\right)
$$
$$
y = 100 \cos\left(2t + \frac{5\pi}{6}\right) \tag{12}
$$
$$
(200 \leq t < 300)
$$

The corresponding trajectory is illustrated in Figure 12. In order to get a better observation of the simulation results, we tracked the actual trajectory of the TCP. This trajectory is able to be displayed on RobotStudio software in real-time, as illustrated in Figure 13.

While conducting the simulation test, it can be seen from the RobotStudio software that the manipulator TCP moved along the designed path very accurately and quickly. In addition, the actual end trajectory is quite similar to the designed one according to Figures 12 and 13. In order to demonstrate the network communication function of the control software, further analysis of the control result is necessary. Since the motion instructions are distributed to
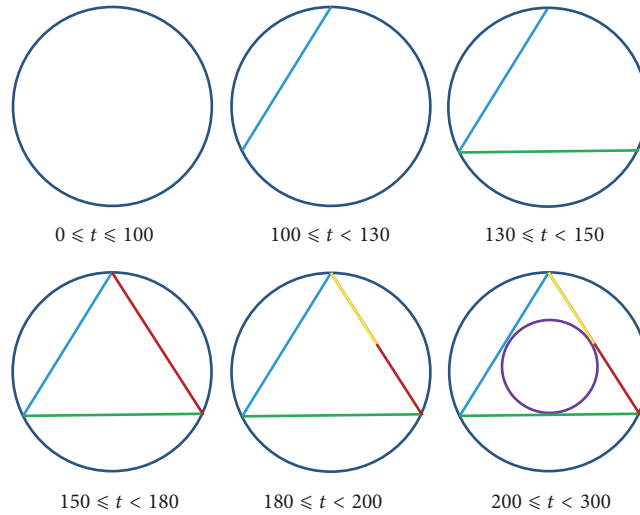
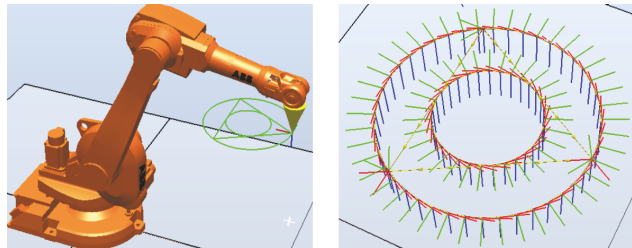FIGURE 12: The designed trajectory of the end-effector.
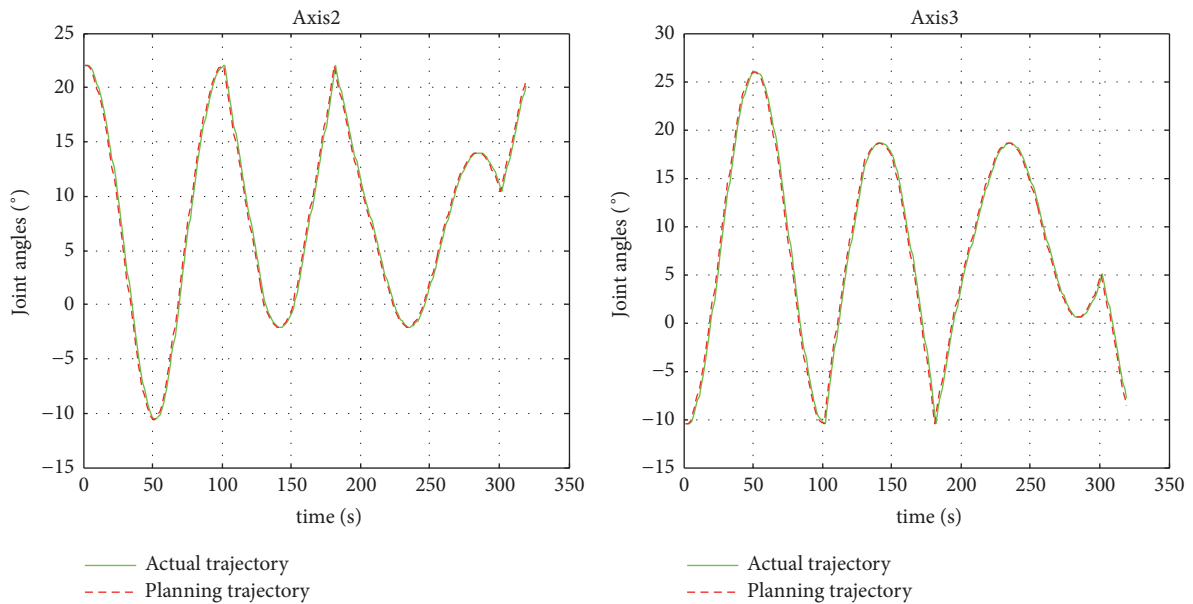

FIGURE 13: Trajectory of the TCP of IRB 1600.

FIGURE 14: The planned joint angles and the actual joint angles of axis 2 and axis 3.

the joint motors with a series of joint angles, it is intuitive to compare the planned joint angles to the actual joint angles. Therefore, we collect the planned joint angles data and the actual joint angles data which are recorded in the simulation test. The joint 2 and joint 3 with quick movement in the test are taken as an example. A time plot of these joint angles is presented in Figure 14, and the time plot for the tracking error of the joint angles is displayed in Figure 15.

From Figures 14 and 15, it can be seen that the joint angles curves of the planned trajectory and the actual one are almost
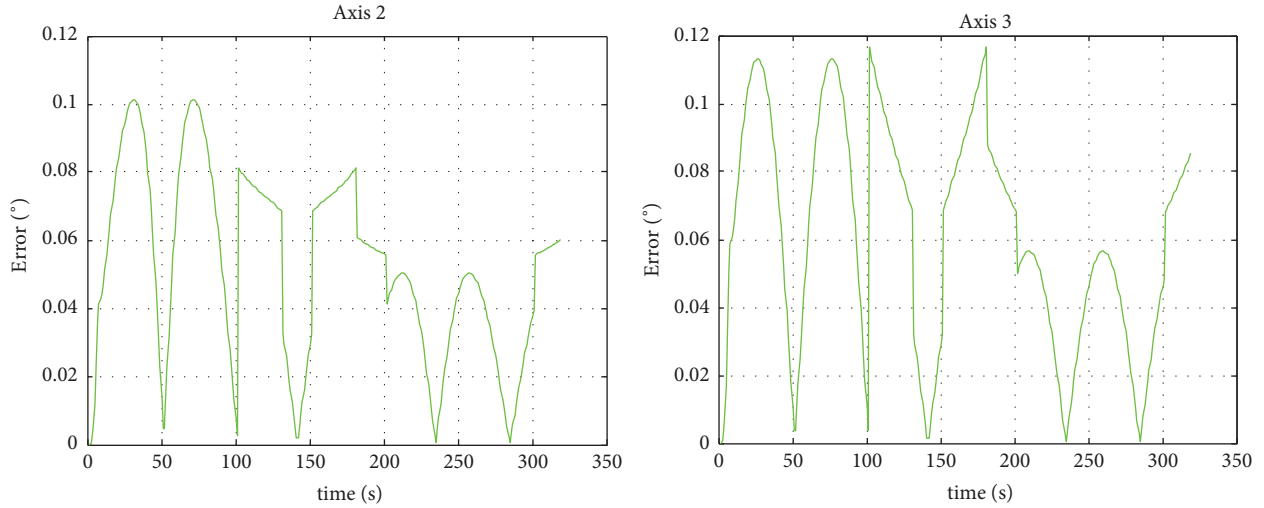
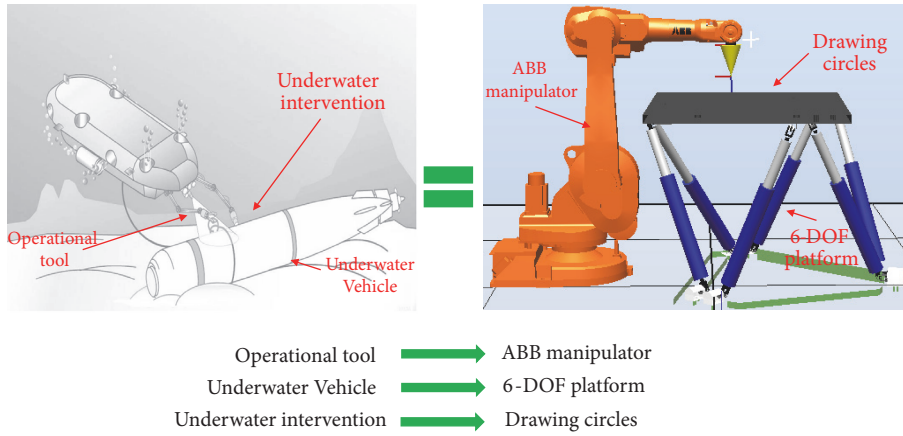FIGURE 15: The tracking error of joint angles of IRB 1600.

Operational tool	→	ABB manipulator
Underwater Vehicle	→	6-DOF platform
Underwater intervention	→	Drawing circles

FIGURE 16: Simulation of the underwater salvage.

overlapping and the tracking errors of the joint angles of axis 2 and axis 3 are less than 0.12˚. Therefore, we can conclude that the manipulator moves in strict accordance with the planned joint angles; this also means that the generated instructions have been correctly transmitted to the controller, which can prove that the network communication is well working experiment.

5.2. Experiment Scenario: 3D Space Operation Test. Since the research of underwater docking is becoming a timely research topic recently, then before we put the manipulator on the 6-DOF parallel platform, we conduct a 3D space operation test in our laboratory with the current system. This 3D space operation test is carried out to test the performance of the hardware equipment of VSFOS while performing the simulation operations of underwater docking. In this test, a marking pen is installed at the end-effector of the serial manipulator as an operation tool, and the 6-DOF parallel platform rotates along its central axis to simulate the underwater vehicle which is disturbed by the current and waiting for docking. In the

end, the marking pen will draw circles in the corresponding plane to which the 6-DOF parallel platform is moving, as illustrated in Figure 16. The operational tool is installed on the mobile vehicle platform. And the operating objective is a fixed underwater vehicle. Since the INS cannot accurately measure the linear movements in 3 axes, the underwater salvage scenario is equivalently translated into the experiment scenario as shown in the right subgraph of Figure 16. The operational tool, underwater vehicle, and underwater interfacing are simulated with ABB manipulator, 6-DOF platform, and drawing circles, respectively. The relative movement situations of simulation scenario are similar to the actual underwater scenario.

When the 6-DOF parallel platform starts moving, the inertial navigation sensor measures the attitude change of the platform, namely, the roll and the pitch angles of the plane that the 6-DOF parallel platform changed to. Since the inertial navigation sensor cannot measure the position change information, we make the central point of the 6-DOF parallel platform remain unchanged. As a result, we get the new position and orientation information about the
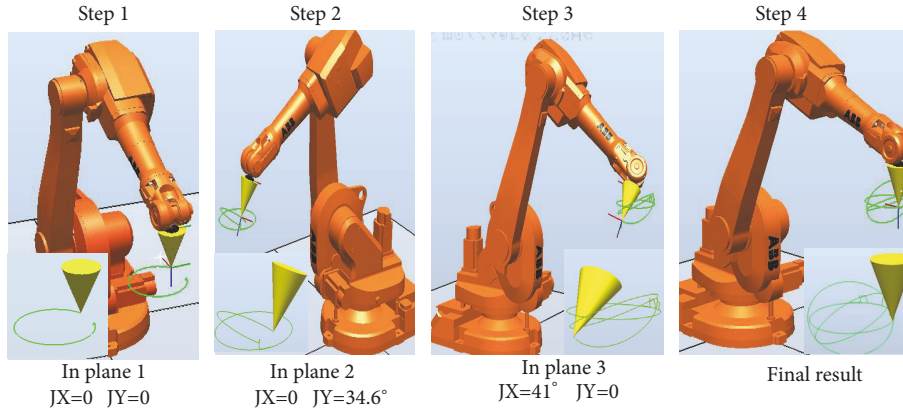
FIGURE 17: Expected result of the 3D space operation test. JX, JY mean the rotation angles along X axis and Y axis.
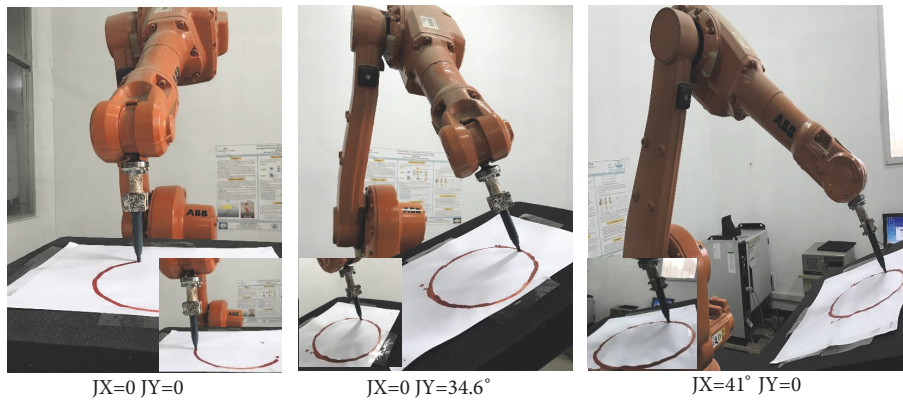

FIGURE 18: Experimental result of the 3D space operation test.

target points of the operation tool of the manipulator. At the beginning, when the 6-DOF parallel platform stays in the horizontal plane, the manipulator is designed to follow the planned circular trajectory. The coordinates of its central point are denoted by O (a, b, c), and the radius of the planned circle is R. If we consider a point M on this circle, the position coordinates of this point M (X, Y, Z) can be expressed as follows.

$$X = a + R \sin(\theta)$$
$$Y = b + R \cos(\theta)$$
$$Z = c \tag{13}$$
$$(0 \le \theta \le 2\pi)$$

When the platform rotates JX degrees along its X axis, JY degrees along its Y axis, and JZ degrees along its Z axis, the upper surface of the platform changes from the horizontal plane to plane 1, whose yaw, pitch, and roll angles are JX degrees, JY degrees, and JZ degrees. Similarly, point M will change to point M′; the position coordinates of point M′(X′, Y′, Z′) are as follows.

$$X' = R \cos(\theta) \cos(JY) + a$$
$$Y' = R \sin(\theta) \sin(JX) \sin(JY) + R \cos(\theta) \cos(JX)$$
$$\quad + b$$

$$Z' = R \sin(\theta) \cos(JX) \sin(JY) - R \cos(\theta) \sin(JX)$$
$$\quad + c$$
$$(0 \le \theta \le 2\pi) \tag{14}$$

From (13) and (14), it is obvious that as long as we get the position coordinates of the central point and the attitude change information, we can calculate the corresponding designed trajectory in plane 1. In this test, considering the height of the 6-DOF parallel platform, we set the position coordinates of the central point of the planned circle as (900mm, 0, 700mm), and the radius is set to 100mm; thus, the trajectory equations of the TCP in plane 1 are as follows.

$$X = 100 \cos(\theta) \cos(JY) + 900$$
$$Y = 100 (\sin(\theta) \sin(JX) \sin(JY) + \cos(\theta) \cos(JX))$$
$$Z = 100 (\sin(\theta) \cos(JX) \sin(JY) - \cos(\theta) \sin(JX)) \tag{15}$$
$$\quad + 700$$
$$(0 \le \theta \le 2\pi)$$

By collecting the planned and actual position information of the TCP, the results are displayed in Figures 17 and 18.

Step 1 In horizontal plane

Step 2 JX=0 JY=34.6°

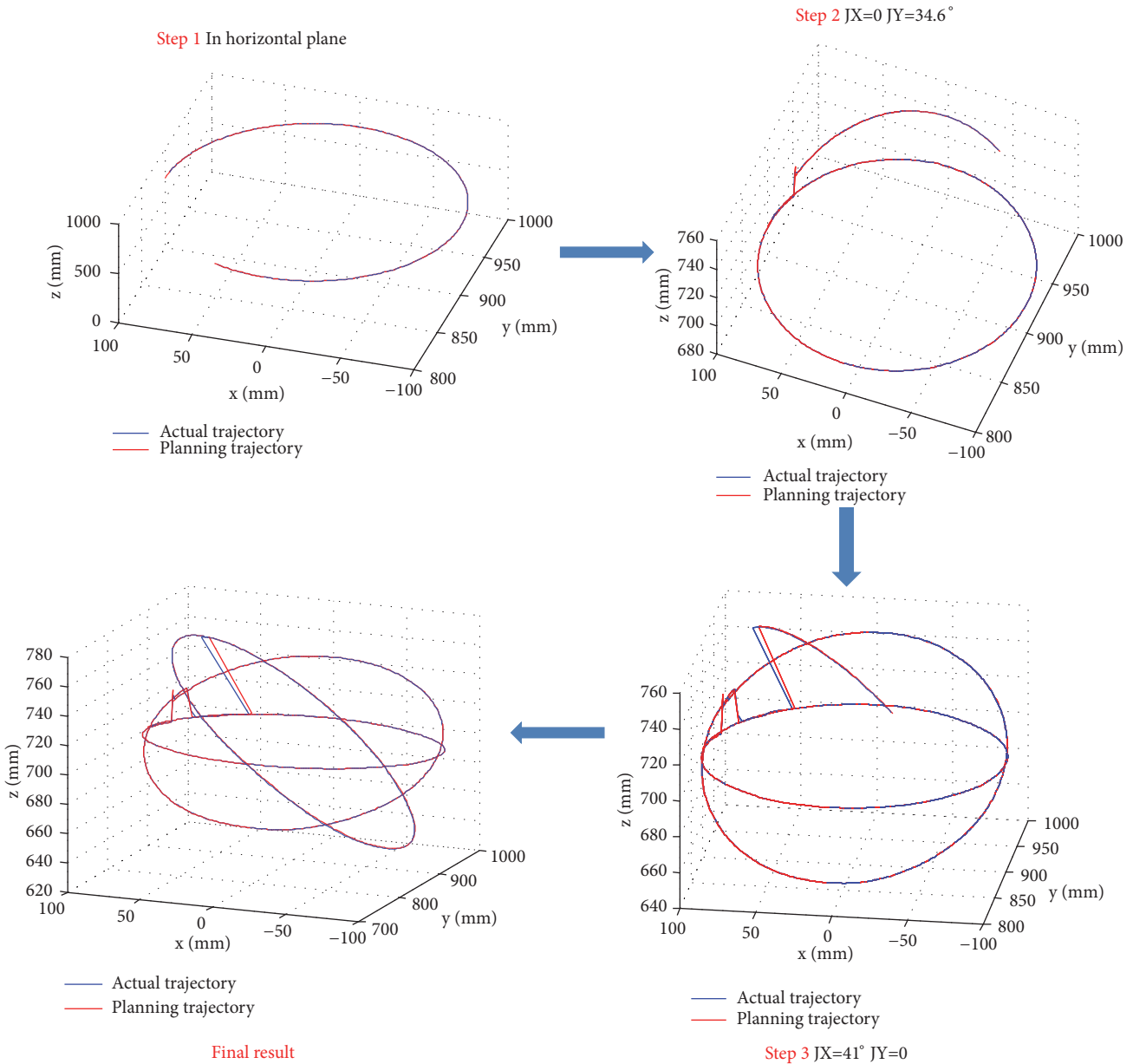Step 3 JX=41° JY=0

Final result

FIGURE 19: View of the planned and the actual trajectories of the 3D space operation test.

In the first stage of this experiment, the platform remains still; thus, the first circle is drawn in the horizontal plane, i.e., plane 1. Then, the platform rotates 0 degrees along the X axis and 34.6 degrees along the Y axis. As a result, the manipulator adjusts the attitude of the marking pen and draws a circle in plane 2. In stage 3, the platform is rotated 41 degrees along the X axis and 0 degrees along the Y axis. And the last circle is drawn in plane 3, as illustrated in Figure 19.

According to test results in Figure 19, the TCP of the manipulator moves in a well accordance with the planned trajectory when drawing the circles in 3D space, except for the moments when the platform changes its attitude suddenly. At those transient instants, the tracking process is a little delayed.

To take a further analysis of the reasons, we can make use of the collected joint angles information.

Since the variation range of joint 5 is the biggest one and joint 1 is the smallest, then, we chose to take the angles of joint 5 and joint 1 as an example; a time plot of these joint angles is shown in Figure 20, and their tracking errors are illustrated in Figures 21 and 22.

As presented in Figures 19 and 20, the joint angles of the actual trajectory are almost overlapped with the planned trajectory. And the joint angles can be fast and stable track to complete the entire trajectory. As it can be observed in Figures 21 and 22, the tracking error in three interim instants is bigger than the others.
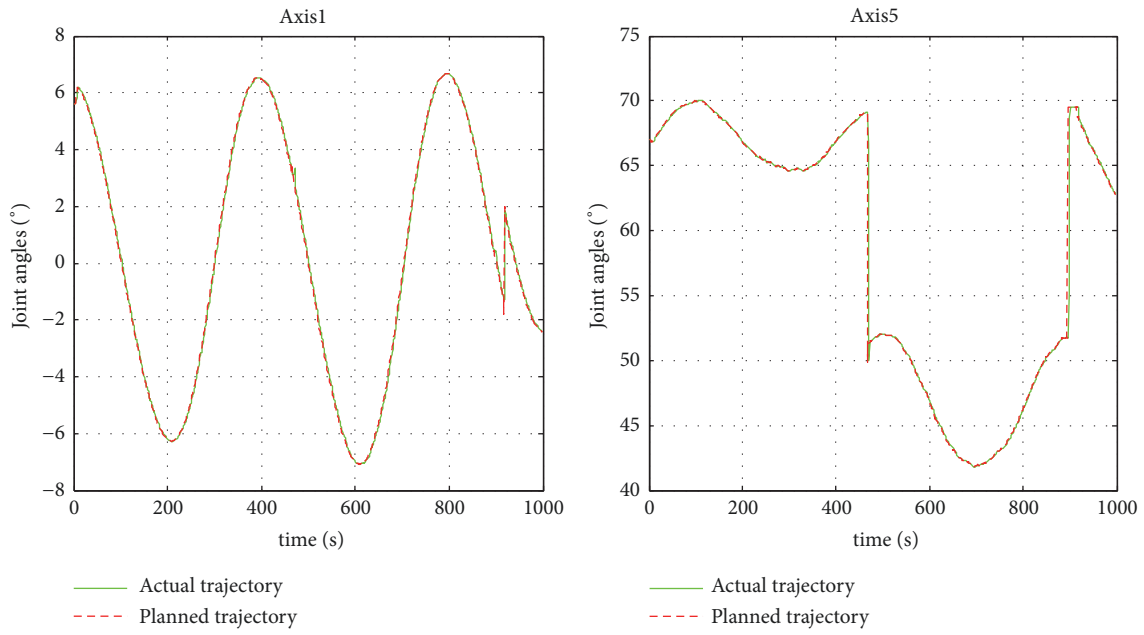
FIGURE 20: Joint angles of the planned trajectory and the actual trajectory.
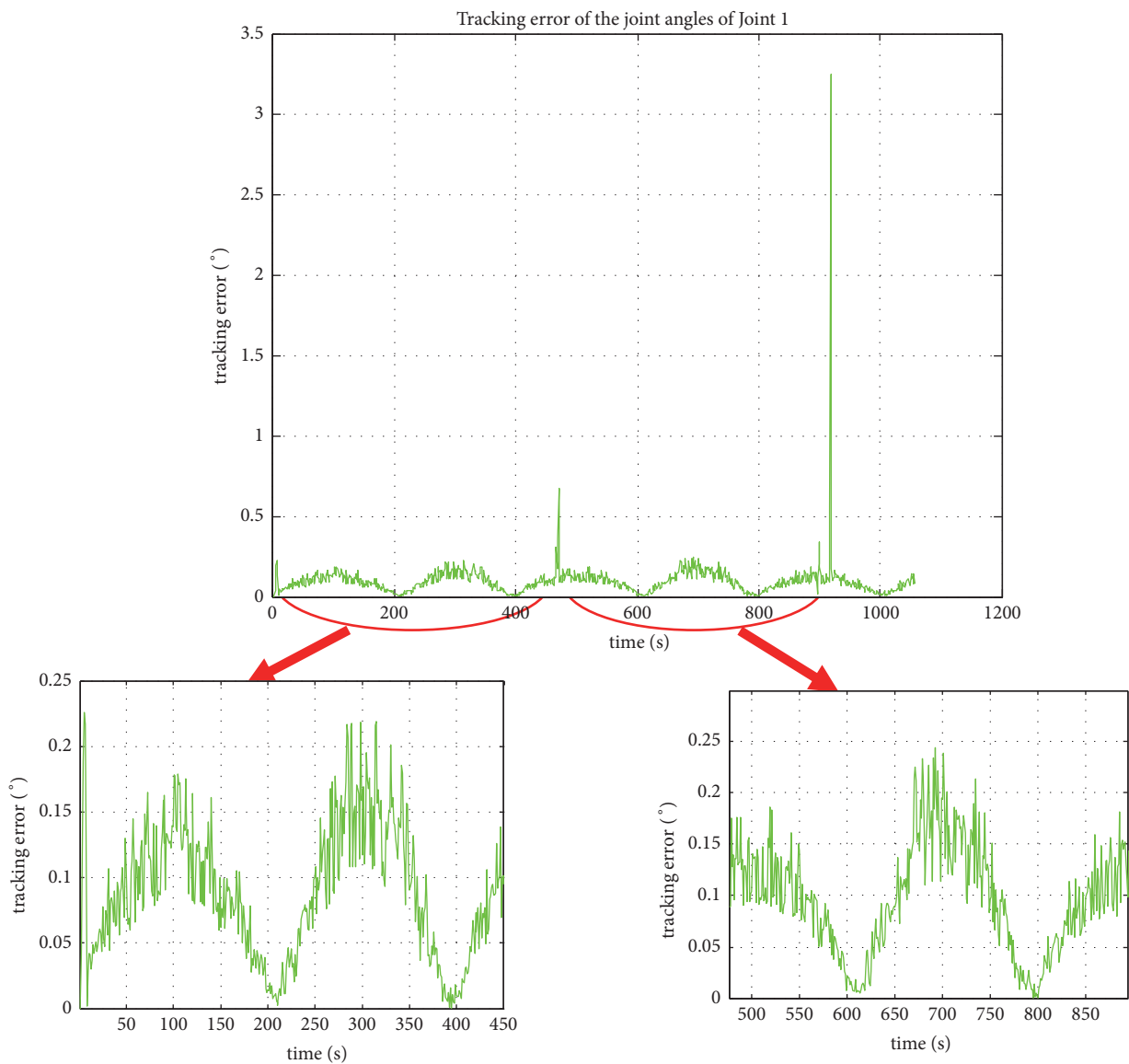

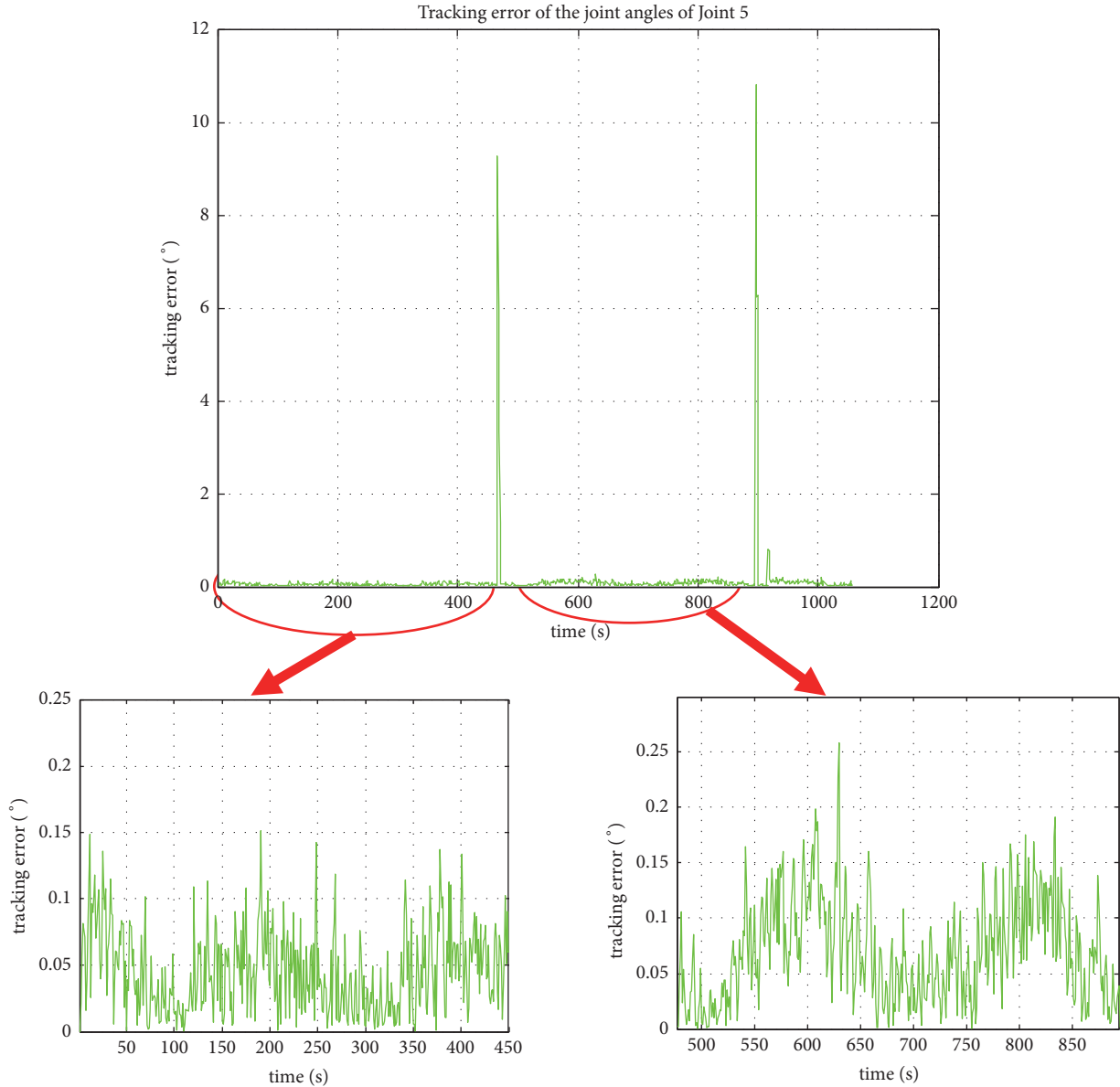FIGURE 21: The tracking error of the joint angles of joint 1.

FIGURE 22: The tracking error of the joint angles of joint 5.

It is worth noting, from these two figures, that at the instant we change the attitude of the platform, the joint angles of the planned trajectory change quickly, while the actual ones are slightly delayed. Concerning the control cycle of the control software and the velocity of the TCP, the attitude changing speed of the platform is much faster than the moving speed of the manipulator. At the moment when the platform changes its attitude, the inertial navigation sensor detects that change information; as a result, a new trajectory in a new plane will be generated. Meanwhile, the manipulator is still following the orders from the previous control cycle and moving in the previous plane; therefore, the joint angles of the planned trajectory are very different from those of the actual trajectory at this instant. However, except for these three moments, an average value of the tracking error of the joint angles is mainly less than 0.7°,

which is acceptable when taking into account the measuring errors. The obtained experimental results show clearly that the proposed equipment hardware of VSFOS is working well in 3D space manipulation tasks.

6. Conclusion and Future Work

This paper mainly introduces a new virtual submerged floating operational system (VSFOS), which makes it faster and much sager to conduct underwater operational simulations. The hardware framework and software structure of VSFOS are introduced in detail. And two critical questions regarding the manipulator kinematics and the communication with the manipulator are highlighted. In the end, two experiments have been conducted to demonstrate the effectiveness of the proposed VSFOS.

The developed control software and the RobotStudio modeling software are combined together to test the developed system hardware and software framework. The test results have shown that the end-effector of the manipulator can follow the desired trajectory with developed system framework. An underwater salvage simulation test is then conducted. The end-effector of the manipulator can move according to specified trajectory on the mobile operational interface.

In the future, some follow-up works demand further research, namely, improving the operational precision and reducing the response time of VSFOS. Some operational tools will be designed and installed on the end-effector of the manipulator to perform difficult subject.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request with guarantee for self-use and not sharing with other parties.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

References

[1] D. J. Fornari, S. E. Humphris, and M. R. Perfit, "Deep submergence science takes a new approach," *Eos, Transactions, American Geophysical Union*, vol. 78, no. 38, pp. 402–408, 1997.

[2] F. Zhang, G. Marani, R. N. Smith, and H. T. Choi, "Future trends in marine robotics," *IEEE Robotics and Automation Magazine*, vol. 22, no. 1, pp. 14–122, 2015.

[3] X. Xiang, C. Yu, L. Lapierre, J. Zhang, and Q. Zhang, "Survey on fuzzy-logic-based guidance and control of marine surface vehicles and underwater vehicles," *International Journal of Fuzzy Systems*, vol. 20, no. 2, pp. 572–586, 2018.

[4] Z. Ju, G. Ouyang, M. Wilamowska-Korsak, and H. Liu, "Surface EMG based hand manipulation identification via nonlinear feature extraction and classification," *IEEE Sensors Journal*, vol. 13, no. 9, pp. 3302–3311, 2013.

[5] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.

[6] Z. Ju and H. Liu, "A unified fuzzy framework for human-hand motion recognition," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 5, pp. 901–913, 2011.

[7] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.

[8] Q. Zhang, L. Lapierre, and X. Xiang, "Distributed control of coordinated path tracking for networked nonholonomic mobile vehicles," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 472–484, 2013.

[9] X. Xiang, C. Liu, H. Su, and Q. Zhang, "On decentralized adaptive full-order sliding mode control of multiple UAVs," *ISA Transactions*, vol. 71, pp. 196–205, 2017.

[10] Z. Peng, J. Wang, and D. Wang, "Distributed Maneuvering of Autonomous Surface Vehicles Based on Neurodynamic Optimization and Fuzzy Approximation," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 3, pp. 1083–1090, 2018.

[11] N. Wang, S.-F. Su, J. Yin, Z. Zheng, and M. J. Er, "Global Asymptotic Model-Free Trajectory-Independent Tracking Control of an Uncertain Marine Vehicle: An Adaptive Universe-Based Fuzzy Control Approach," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 3, pp. 1613–1625, 2018.

[12] Z. Zheng, Y. Huang, L. Xie, and B. Zhu, "Adaptive Trajectory Tracking Control of a Fully Actuated Surface Vessel With Asymmetrically Constrained Input and Output," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1851–1859, 2018.

[13] J. Gao, A. A. Proctor, Y. Shi, and C. Bradley, "Hierarchical Model Predictive Image-Based Visual Servoing of Underwater Vehicles with Adaptive Neural Network Dynamic Control," *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2323–2334, 2016.

[14] Z. Chu, X. Xiang, D. Zhu, C. Luo, and D. Xie, "Adaptive fuzzy sliding mode diving control for autonomous underwater vehicle with input constraint," *International Journal of Fuzzy Systems*, vol. 20, no. 5, pp. 1460–1469, 2018.

[15] L. Qiao and W. Zhang, "Adaptive second-order fast nonsingular terminal sliding mode tracking control for fully actuated autonomous underwater vehicles," *IET Control Theory & Applications*, vol. 11, no. 8, pp. 1293–1306, 2017.

[16] C. Yu, X. Xiang, L. Lapierre, and Q. Zhang, "Robust Magnetic Tracking of Subsea Cable by AUV in the Presence of Sensor Noise and Ocean Currents," *IEEE Journal of Oceanic Engineering*, vol. 43, no. 2, pp. 311–322, 2018.

[17] R. Cui, X. Zhang, and D. Cui, "Adaptive sliding-mode attitude control for autonomous underwater vehicles with input nonlinearities," *Ocean Engineering*, vol. 123, pp. 45–54, 2016.

[18] X. Xiang, C. Yu, and Q. Zhang, "On intelligent risk analysis and critical decision of underwater robotic vehicle," *Ocean Engineering*, vol. 140, pp. 453–465, 2017.

[19] R. Yang, B. Clement, A. Mansour, M. Li, and N. Wu, "Modeling of a Complex-Shaped Underwater Vehicle for Robust Control Scheme," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 3-4, pp. 491–506, 2015.

[20] R. Cui, L. Chen, C. Yang, and M. Chen, "Extended State Observer-Based Integral Sliding Mode Control for an Underwater Robot With Unknown Disturbances and Uncertain Nonlinearities," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 8, pp. 6785–6795, 2017.

[21] Z. Zeng, K. Sammut, L. Lian, F. He, A. Lammas, and Y. Tang, "A comparison of optimization techniques for AUV path planning in environments with ocean currents," *Robotics and Autonomous Systems*, vol. 82, pp. 61–72, 2016.

[22] C. Yu, X. Xiang, Q. Zhang, and G. Xu, "Adaptive fuzzy trajectory tracking control of an under-actuated autonomous underwater vehicle subject to actuator saturation," *International Journal of Fuzzy Systems*, vol. 20, no. 1, pp. 269–279, 2018.

[23] J. T. Pado, "Remotely operated underwater vehicle," *uS Patent 4,721,055 (Jan. 26 1988)*.

[24] X. Xiang, C. Yu, J. Zheng, and G. Xu, "Motion forecast of intelligent underwater sampling apparatus — Part I: Design and algorithm," *Indian Journal of Geo-Marine Sciences*, vol. 44, no. 12, pp. 1962–1970, 2015.

[25] Z. Chu, D. Zhu, and S. X. Yang, "Observer-based adaptive neural network trajectory tracking control for remotely operated vehicle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1633–1645, 2017.

[26] J. Yuh, "Design and control of autonomous underwater robots: A survey," *Autonomous Robots*, vol. 8, no. 1, pp. 7–24, 2000.

[27] D. Wu, T. Chen, and A. Li, "A high precision approach to calibrate a structured light vision sensor in a robot-based three-dimensional measurement system," *Sensors*, vol. 16, no. 9, article no. 1388, 2016.

[28] R. B. Wynn, V. A. I. Huvenne, T. P. Le Bas et al., "Autonomous Underwater Vehicles (AUVs): their past, present and future contributions to the advancement of marine geoscience," *Marine Geology*, vol. 352, pp. 451–468, 2014.

[29] A. D. Bowen, D. R. Yoerger, C. C. German et al., "Design of Nereid-UI: a remotely operated underwater vehicle for oceanographic access under ice," in *Proceedings of the OCEANS 2016 MTS/IEEE St. John's, OCEANS '14*, Canada, January 2015.

[30] X. Xiang, L. Lapierre, and B. Jouvencel, "Smooth transition of AUV motion control: from fully-actuated to under-actuated configuration," *Robotics and Autonomous Systems*, vol. 67, pp. 14–22, 2015.

[31] J. Ghommam and M. Saad, "Backstepping-based cooperative and adaptive tracking control design for a group of underactuated AUVs in horizontal plan," *International Journal of Control*, vol. 87, no. 5, pp. 1076–1093, 2014.

[32] O. Khatib, X. Yeh, G. Brantner et al., "Ocean one: A robotic avatar for oceanic discovery," *IEEE Robotics and Automation Magazine*, vol. 23, no. 4, pp. 20–29, 2016.

[33] F. A. Azis, M. S. M. Aras, M. Z. A. Rashid, M. N. Othman, and S. S. Abdullah, "Problem identification for Underwater Remotely Operated Vehicle (ROV): A case study," in *Proceedings of the 2nd International Symposium on Robotics and Intelligent Sensors 2012, IRIS 2012*, pp. 554–560, Malaysia, September 2012.

[34] B. Bingham, N. Kraus, B. Howe et al., "Passive and active acoustics using an autonomous wave glider," *Journal of Field Robotics*, vol. 29, no. 6, pp. 911–923, 2012.

[35] P. J. Sanz, A. Peñalver, J. Sales et al., "GRASPER: A multisensory based manipulation system for underwater operations," in *Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*, pp. 4036–4041, UK, October 2013.

[36] S. Heshmati-Alamdari, A. Eqtami, G. C. Karras, D. V. Dimarogonas, and K. J. Kyriakopoulos, "A self-triggered visual servoing model predictive control scheme for under-actuated underwater robotic vehicles," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA '14*, pp. 3826–3831, Hong Kong, June 2014.

[37] X. Xiang, Z. Niu, L. Lapierre, and M. Zuo, "Hybrid underwater robotic vehicles: The state-of-the-art and future trends," *HKIE Transactions*, vol. 22, no. 2, pp. 103–116, 2015.

[38] J. Zeng, S. Li, Y. Tang, and Y. Wang, "The application of Polar-ARV in the fourth Chinese National Arctic Expedition," in *Proceedings of the MTS/IEEE Kona Conference, OCEANS'11*, USA, September 2011.

[39] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.

[40] F. Sanfilippo, L. I. Hatledal, H. Zhang, M. Fago, and K. Y. Pettersen, "JOpenShowVar: An open-source cross-platform communication interface to Kuka robots," in *Proceedings of the 2014 IEEE International Conference on Information and Automation, ICIA 2014*, pp. 1154–1159, China, July 2014.

[41] D. Maalouf, A. Chemori, and V. Creuze, "L1 Adaptive depth and pitch control of an underwater vehicle with real-time experiments," *Ocean Engineering*, vol. 98, pp. 66–77, 2015.

[42] Z. Chu, D. Zhu, and G. Eu Jan, "Observer-based adaptive neural network control for a class of remotely operated vehicles," *Ocean Engineering*, vol. 127, pp. 82–89, 2016.

[43] X. Xiang, C. Yu, and Q. Zhang, "Robust fuzzy 3D path following for autonomous underwater vehicle subject to uncertainties," *Computers & Operations Research*, vol. 84, pp. 165–177, 2017.

[44] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.

[45] Z. Jiang, X. Xiang, and H. Liu, "An open-source control software to the virtual submerged floating operational system (VSFOS)," in *Proceedings of the OCEANS 2016 MTS/IEEE Monterey, OCEANS '16*, USA, September 2016.

[46] C. R. Rocha, C. P. Tonetto, and A. Dias, "A comparison between the DenavitHartenberg and the screw-based methods used in kinematic modeling of robot manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 4, pp. 723–728, 2011.