ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 2, August 2014

Development of Flexible Control Software for Any Articulated Manipulators

Taeyong Choi, Hyunmin Do, Chanhun Park, Dongil Park, Jinho Kyung and Doohyeong Kim
Department of Robotics and Mechatronics, Korea Institute of Machinery & Materials,
156, Gajeongbuk-Ro, Yuseong-Gu, Daejeon, 305-343, Korea

***Abstract*—Developing control software for a developed robot is not only difficult, but also considerably time consuming. Moreover, many modifications to the control software are required during robot development. There are several general-purpose robot control software, however applying them to certain robots that need real-time control performance is almost impossible. In this paper, we introduce a real-time controllable framework for a general-purpose controller that can be applied to any number of articulated robots with minimal effort.**

***Index Terms*—robot operating system, robot software.**

I. INTRODUCTION

The control software for articulated manipulators may need to be reconfigured in the event of certain unforeseen situations. This is especially true during the development of robots, when frequent hardware changes are needed to address development issues. For example, developers may need to modify the original control software to allow debugging in the case of a temporarily reconfigured robot as soon as possible. Further, the development of a robot usually occurs in a systematic fashion; from a prototype to the final version, multiple versions might be developed, such as version 1, version 2, and so on. In such cases, modifying the control software used for version 1 of the robot such that it can be used for version 2, is a time consuming process. Such situations are frequently encountered during the development of robots. To address the aforementioned problems, flexible control platform software is necessary; here, the word "platform" is used to indicate that the control engine, hardware manager, simulator, and graphical user interface (GUI) are included in the control software. A considerable amount of effort is required for the implementation of robot control. In [1], a general software platform for robot control, OpROS is described. OpROS contains a module-based controller configuration function, but lacks a real-time robot control function, which is a basic aspect in case of an industrial robot. In [2], an open source robot operating system, ROS, is described. ROS has a similar tool-set as OpROS, but it is based on the Linux operating system. We aim to develop a robot operating system based on Microsoft's Windows OS so that their Visual Studio tool can be used to develop our system in a convenient manner. Microsoft's Robotic Developer Studio (MRDS) [3] is one of the general software frameworks for robot control. Though, MRDS is convenient to use, it lacks the real-time robot control functions, and lacks efficient kinematics and dynamics to control the manipulator in it. Simlab's Roboticslab [4] is very similar to our proposed software. Roboticslab contains specialized manipulator control functions and real-time functions on the windows. However, its real-time function is incomplete, and is restricted by many constraints. In general, the implementation of flexible control software is difficult owing to its hardware dependent properties, and thus, it cannot be implemented in a generalized manner. Furthermore, not only does each robot have different servo-drives, different sizes, different gear ratios, different motors, etc., but the communication protocol they use may differ; Some robots use CAN, some use PROFINET, and others might use some other communication protocol as there are many options for robot communication protocols. As explained above, it is not possible to completely automate control software to account for changes in the robot hardware configuration. We say that the control module including kinematics and dynamics for two type of robot, real-time service module, Ether CAT communication module and hardware configuration module are given to user with minimal effort. We selected Ether CAT [5-7] as the main communication protocol in our proposed system because of its high speed and easy configuration.

II. DEVELOPMENT OF CONTROL SOFTWARE FRAMEWORK

A. *Data exchange between GUI and control module*

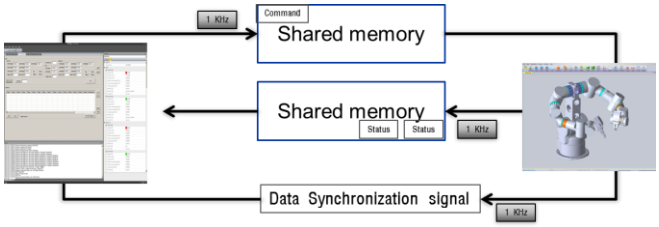

Fig. 1 The control software framework consists of GUI and control module with 'shared memory'.

Our proposed control software framework is divided into a graphical user interface (GUI) module and a control module as shown in Fig. 1. The GUI module is not a real-time program. It is a normal program that runs on the Windows environment. On the other hand, control module is a real-time program that runs on a real-time layer provided by the

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 2, August 2014

commercial real-time kernel RTX [8]. The exchange of data between the GUI module and the control module is through shared memory. It is important that the control algorithm including kinematics and dynamics that is provided by default should be located at the control module. Monitoring data such as joint position, velocity and torque are transmitted to the GUI module from the control module. Command values such as joint command, velocity command, torque command and task space command are transmitted to control module from GUI module. The monitoring data shown on the GUI is not the real-time data, but past data, because the data is transmitted through the shared memory, and thus, does not reflect the real-time process.

B. Reconfiguration automation

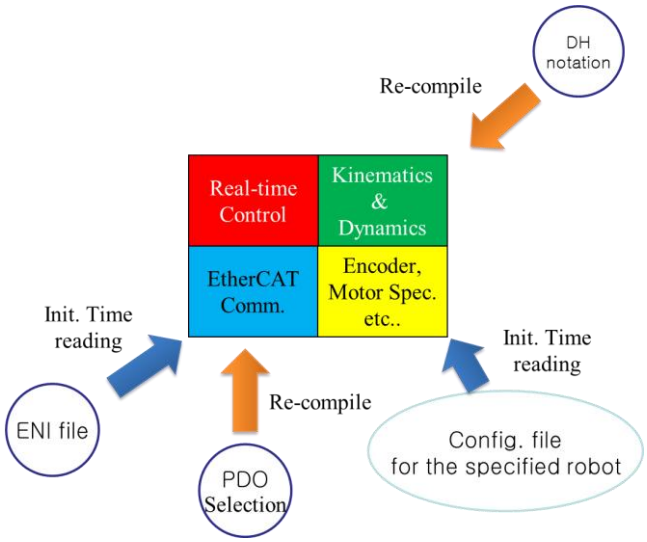

Fig. 2 User must set four categories of configurations such as ENI file, PDO data structure, hardware configuration, and DH parameters to operate the developed control software framework.

We aim to minimize the user effort in modifying the original control software such that it can be used for the new robot system. Nonetheless, the user will have to set some properties related to the hardware configuration. Our proposed control software consists of four modules namely, real-time module, kinematics/dynamics module, Ether CAT communication module and hardware configuration module. From among the aforementioned modules, the real-time module does not need the user to set any properties, because the structure is predefined in case of the hard real-time timers on the multi-core processors. The details are explained in the next section. In case of reconfiguring the Ether CAT communication module, the user should extract the ENI file of the target robot. An ENI file is a standard file that contains the slave configuration information that is conveyed to the master on the Ether CAT communication. The ENI file is auto-generated with the Ether CAT manager tool. In addition, the user must define the data configuration in the data train of the Ether CAT communication. The user is expected to manually program the method to obtain the input PDO data based on the servo drives. This process can be automated with no considering real-time performance. However, we decided to have a step for recompilation because it is more suitable for the purpose of our software. Hardware configuration details such as gear ratio, torque limit, control frequency, and motor constant should be provided via a configuration file. This configuration file is loaded during the initial execution. The most difficult one among the four user configuration settings that the user has to define for our software is the Denavit-Hartenberg (DH) parameters set. However, we expect users with a little knowledge about kinematics can configure DH parameter easily, using our proposed software. The defined DH notation is used to implement the kinematics and dynamics module. The kinematics and dynamics module needs to be recompiled after setting DH parameters.

C. Real-time process in multi-core CPU

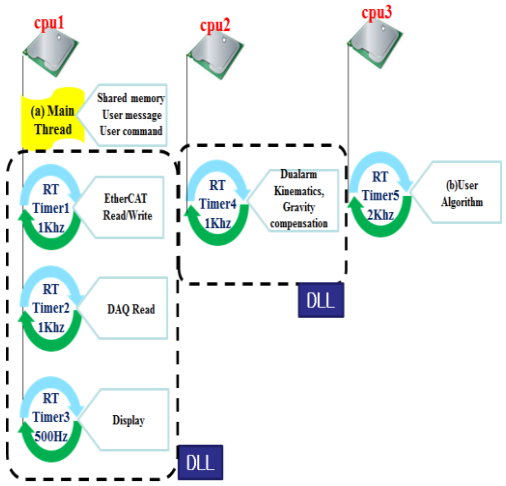

Fig. 3 Timer and module arrangement on the multi-core CPU.

Our software requires three CPUs for operation. Communication between the GUI and control modules, functions related to Ether CAT and small functions related to system are executed on the CPU1 as shown in Fig. 3. Kinematics and dynamics calculations use another CPU (CPU2 in Fig. 3) because of the high-level computations involved and this module is compiled to a dynamic link library (DLL) or real-time DLL (RTDLL). The user works on CPU3 shown in Fig. 3. Feedback information such as joint position, velocity, torque, and servo-drive status are provided to user. The user can command the robot by specifying the joint position, velocity, and torque. Further, the user can send the task space position command using the kinematics module.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 2, August 2014

D. *Control module details*

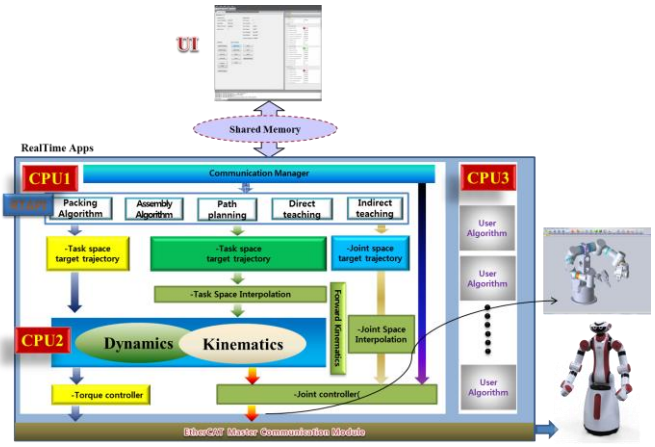

Fig. 4 The proposed control software framework overview.

The detailed configuration of our proposed control software framework is shown in Fig. 4. It is noteworthy that the user can run the target robot on the simulation environment or on the practical robot, using our proposed software. Our developed control software has many other useful functions. For example, functions related to cellular phone packaging, cable connecting and box packaging have been implemented for use in the practical production line.

III. SIMULATION STUDY AND EXPERIMENT

In this section, the performance of our proposed control software platform is evaluated. The real-time performance was checked on the simulation. Further, flexibility of the software was proven using many robots.

A. *Calculation performance*

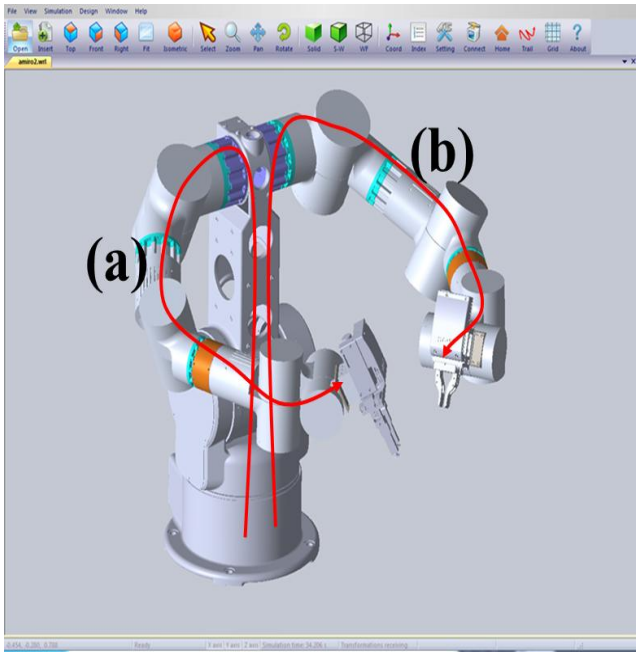

Fig. 5 Kinematic solver configuration for the dual-arm robot.

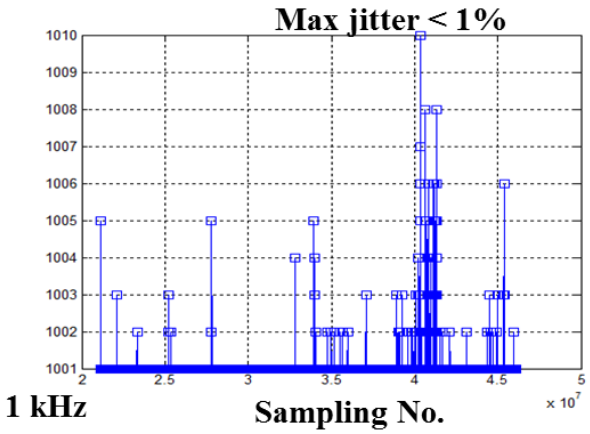

Fig. 6 Measured control time at each sampling.

A dual-arm robot with 16 degrees of freedom (DOF) is considered as the target robot. There are 7 DOF for the right arm, 7 DOF for the left arm, and 2 DOF for the waist. Two kinematic computations are implemented. In reality, the kinematic computations are automatically generated by our control software, if the user sets the DH parameters correctly. One kinematic computation is for "waist + right arm" labeled (a) in Fig. 5 and the other kinematic computation is for "waist + left arm" labeled (b) in Fig. 5. Each kinematic computation solves 9 DOF serial manipulators' kinematics. In addition, the gravity compensation algorithm is implemented. In the practical control scheme, the gravity compensation and an additional simple linear position controller are sufficient to show tough tracking performance. Fig. 6 shows the resultant graph for the measured control time at each sampling. The control frequency was set as 1 kHz, i.e., a sampling time of 1㎳. The time consumption during each control frequency was measured. Unit of the measured time is ㎲. It is noteworthy that it showed jitters less than 1%. Moreover, the kinematics solver and gravity compensation solver spend less than 200㎲ during the test motion.

B. *Practical application cases*

Our developed control software was applied to four real robots. Three of them, specifically (a), (b), and (d) shown in Fig. 7 have a similar structure. In addition, (b) and (d) have same DOF, though their link length, weight, gear ratio, and servo-drive properties are different. In reality, (b) is an advanced version of (d). The detailed information of (d) is given in [9]. Robot (a) is unique because its waist has 3 DOF. Robot (c) has both arms, but has only its right arm actuated. In addition, robot (c) has 6 DOF in the kinematic joint and 12 servo-drives to actuate each joint with the antagonistic configuration. The technicalities of robot (c) shown in Fig. 7 are explained in [10]. It costs one or two days to apply the developed control software platform to a certain robot only if the robot is connected through EtherCAT protocol.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 2, August 2014

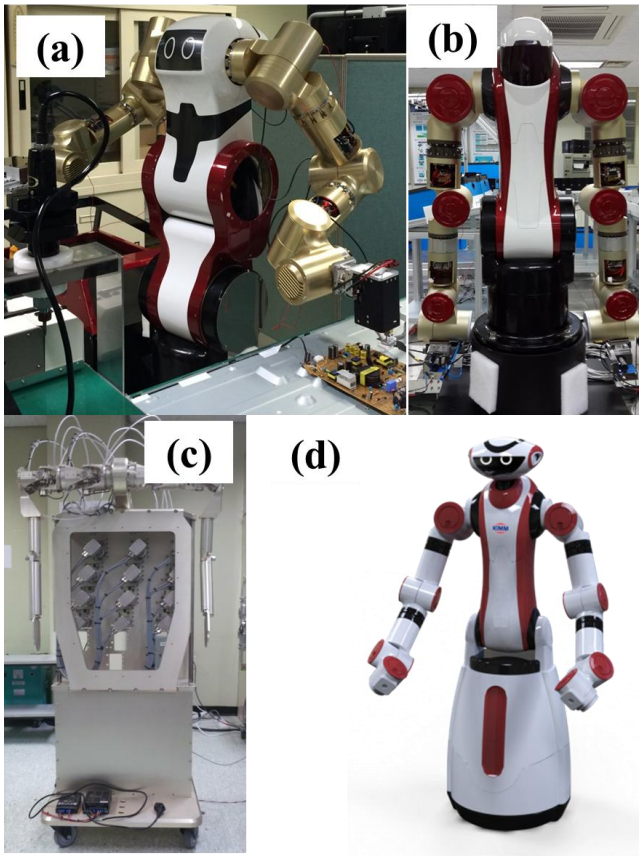

Fig. 7 Practical examples that the developed control software framework were applied to.

IV. CONCLUSION

A general control software framework for any articulated robot was developed. It is flexible and useful, assuming that robot is connected through the EtherCAT protocol. In our developed system, configuration of only four categories such as ENI file, PDO data structure, hardware configuration, and DH parameters is sufficient to operate any articulated robot. The developed control software was applied to many practical robots and it yielded a good real-time control performance.

ACKNOWLEDGMENT

This research was supported by the Ministry of Trade, Industry & Energy and the Korea Evaluation Institute of Industrial Technology (KEIT) with the program number of "10038660"

REFERENCES

[1] B. Song, S. Jung, C. Jang, and S. Kim, "An introduction to robot component model for opros (open platform for robotic services)," in Workshop Proc. of SIMPAR, 2008, pp. 592-603.

[2] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, et al., "ROS: an open-source Robot Operating System," in ICRA workshop on open source software, 2009.

[3] Microsoft. Microsoft Robotics Developer Studio. Available at : http://www.microsoft.com/robotics/, 2014

[4] T. Choi, H.-M. Do, J.-H. Kyung, D. Park and C. Park, "Control of 6DOF Articulated Robot with the Direct-teaching Function using EtherCAT," in Proc. of IEEE Int. Symp. on Robot and Human Interactive Communication, Gyeongju, Korea, Aug., pp. 26-29, 2013

[5] D. Jansen and H. Buttner, "Real-time Ethernet: the EtherCAT solution," Computing and Control Engineering, vol. 15, p. 16, 2004.

[6] G. Cen, I. C. Bertolotti, S. Scanzio, A. Valenzano and C. Zunino, "Evaluation of EtherCAT distributed clock performance," IEEE Trans. on Ind. Inform., vol. 8, no. 1, pp. 20-29, 2012.

[7] EtherCAT_Technology_Group. Available at : http://www.ethercat.org/

[8] IntervalZero. RTX Real-Time Platform. Available at : http://www.intervalzero.com/products/rtx-at-a-glance/ , 2014

[9] T. Choi, H.-M. Do, C. Park, D. Park and J.-H. Kyung, "Development of small-sized industrial dual-arm robot with convenient interface," Int. Journal of Engineering and Innovation Technology, vol. 3, issue 2, Aug. 2013

[10] T. Choi, H.-M. Do, K. Chung and D. Kim, "Development of Shoulder Complex Structure," in Proc. of Int. Conf. on Ubiquitous and Ambient Intelligence, Jeju, Korea, Oct., 2013.

AUTHOR'S PROFILE

Taeyong Choi received the B.S. degree in Electronic and Electrical Engineering from Pohang University of Science and Technology (POSTECH), Pohang, Korea, in 2003, and the Ph.D. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2010. He was a senior engineer at the Samsung Electronics, Korea, in 2010. He has been working for Korea Institute of Machinery and Materials (KIMM), Korea, since 2011. His current research interests include safe robot, robotics and machine learning.

Hyunmin Do received the B.S. and M.S., and Ph.D. degrees in the School of Electrical Engineering from Seoul National University, Seoul, Korea, in 1997, 1999, and 2004, respectively. From 2004 to 2006, he was with the Hyundai Mobis, Korea and from 2006 to 2007, he was with the Hyundai Motor Company, Korea, where he was a Senior Research Engineer. From 2007 to 2010, he was the National Institute of Advanced Industrial Science and Technology, Japan, where he was a Postdoctoral Research Fellow. In 2010; he joined the Korea Institute of Machinery and Materials, where he is a Senior Researcher of Robotics and Mechatronics Research Center. His research interests are neuro computing and control, adaptive and learning control, ubiquitous robotics and robot middleware, mobile robot navigation, and manipulator control of a dual-arm robot and a parallel kinematic machine.

Chanhun Park received BS degree in Mechanical Engineering from Yeungnam University, Korea, in 1994 and MS degree in Mechanical Engineering from POSTECH, Korea, in 1996 and PhD degree from KAIST, in 2010. He has been working for Korea Institute of Machinery and Materials (KIMM), Korea, since 1996, where he is currently a senior researcher. His research fields include human-robot cooperation and design and control of dual arm robot manipulator for industrial applications.

Dongil Park received a BS degree in Mechanical Engineering from Korea Advanced Institute of Science and Technology (KAIST) in 2000, a MS degree in Mechanical Engineering from KAIST in 2002 and a PhD degree from KAIST in 2006. He has been researching the robotics in Korea Institute of Machinery and Materials (KIMM) since 2006. His research fields are the design, control and application of robot manipulators and mobile robots.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 2, August 2014

Jinho Kyung received his B.S. and M.S. degree in Mechanical Engineering from Korea Aerospace University and KAIST in 1985 and 1988, respectively. He was awarded his Ph.D. degree from KAIST in 2003. Dr. Kyung is currently a Principal Researcher at Dept. of Robotics & Intelligent Machinery at Korea Institute of Machinery and Materials, and he has served as a head of Robot team. Dr. Kyung's research interests include the design and control of robotic systems, and manipulation technology for human-robot cooperation.

Doohyeong Kim received the B.S. degree in Mechanical Design Engineering from Seoul National University, Seoul, Korea, in 1982, and the Ph.D. degree in Mechanical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2003. He has been working for Korea Institute of Machinery and Materials (KIMM), Korea, since 1982. His current research interests include design and control of robot.