Science & Technology Development Journal – Engineering and Technology, 3(3):461-471

Open Access Full Text Article

Research Article

A study on application of real-time control method for controlling position of robots in the given time

Tri Cong Phung[1,2,*]

Use your smartphone to scan this QR code and download this article

ABSTRACT

In the field of robotics, the accurate control of the position and the velocity for making the robots operate in a desired time is an important requirement in the industry. The modern works suggest that the robots can work in the complicated systems that need the external information collected from the working environment of the robots. In this case, it is necessary to process the signals from the external sensors before sending the useful information to the controllers of the robots. Thus, the communication time between the host computers and the controllers of the robots plays an important role in the working of the system. In addition, the ability of processing signals in the real-time is also very important. There are many applications that require the strict time in the total system. The problem of the interruption often happens when the users use the standard Windows operating system. Thus it is very difficult to control the robots or the equipment so that they work in time. To overcome this problem, a real-time operating system should be used to control the robots. The open-source real-time operating systems not only have more advantages than the normal operating systems in both economy and flexibility but also meet the needs. This paper concentrates on building the algorithms for controlling the robot trajectory in a desired time using a modern real-time operating system called Linux-Xenomai. Firstly, the paper analyzes several advantages of the real-time operating system Linux-Xenomai comparing the general operating systems and other real-time operating systems. Secondly, a real-time controller of a 5 degree-of-freedom (DOF) robot is built based on the real-time operating system Linux-Xenomai. After that, the paper proposes the algorithms to test the ability of working in time of the robot. Finally, the real experiments are done to verify the proposed algorithms.

Key words: Industrial robots, inverse kinematics, operating system, position control, real-time control

[1]*Department of Mechatronics, Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, Ward 14, District 10, Ho Chi Minh City, Vietnam*

[2]*Vietnam National University Ho Chi Minh City, Linh Trung Ward, Thu Duc District, Ho Chi Minh City, Vietnam*

Correspondence

Tri Cong Phung, Department of Mechatronics, Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, Ward 14, District 10, Ho Chi Minh City, Vietnam

Vietnam National University Ho Chi Minh City, Linh Trung Ward, Thu Duc District, Ho Chi Minh City, Vietnam

Email: ptcong@hcmut.edu.vn

History

- Received: 27-8-2020
- Accepted: 26-11-2020
- Published: 03-12-2020

DOI : 10.32508/stdjet.v3i3.759

Check for updates

VNU-HCM Press

INTRODUCTION

Industrial robots nowadays are used popularly in the manufacturing and the industry. They normally are controlled by the distributed controllers to get the accuracy both in the position and the time. This structure is suitable for the simple works which do not require more information from the external sensors. However, the modern works suggest that the robots can work in the complicated systems that need the external information collected from the working environment of the robots. In this case, it is necessary to process the signals from the external sensors before sending the useful information to the controllers of the robots. Thus, the communication time between the host computers and the controllers of the robots plays an important role in the working of the system. In addition, the ability of processing signals in the real-time is also very important. There are many applications that require the strict time in the total system.

One problem is that the user control the robot so that its end-effector contacts with an object at a constant force. For example, it happens with the windows cleaning robots. The host computer needs to collect the signals from the force sensor attached to the wrist of the robot and sends the contact force information to the controller of the robot immediately. If there is the late time, the real contact force maybe larger than the desired force. Sometimes a broken system mays happen. Another example is that the user control the robot to grasp an object on a moving conveyor with the desired orientation. A camera is used to take the picture of the object. After that, the host computer processes this picture to get the information of the position and the orientation of the object. It is need that this information is sent to the controller in real time so that the robot can grasp the object at the desired position. If there are many objects or the speed of the conveyor is large, it is more difficult to finish the task and the communication time plays an important role in the system.

Cite this article : Phung T C. **A study on application of real-time control method for controlling position of robots in the given time**. *Sci. Tech. Dev. J. – Engineering and Technology;* 3(3):461-471.

The traditional structure of the controllers of industrial robots does not satisfy the requirement of collecting the information from the external sensors. To meet this requirement, the controllers of the robots should be bigger and the cost of the robots also increases. Another reason is that it is not easy to enter the information of the sensors to the controllers of robots because of restricting from the users to interfering into the controllers of the robots. Andrew A. Goldenberg et al. suggested a new structure that the users can use an additional host computer to collect the information and send the signal to the controller of the PUMA robot[1]. J. Gomez Ortega et al. proposed a new structure of the controller for the industrial robot Staubli RX60 so that the robot can get the information from the external sensors[2].

For the host computer, the users can install one of the popular operating systems. One problem is that the interruption often happens when the users use the standard Windows operating system. Thus it is very difficult to control the robots or the equipment so that they work in time. To overcome this problem, a real-time operating system should be used to control the robots. Building a real-time operating system based on the standard Windows operating system is very expensive. Many groups of researchers are concerned about building a real-time operating system based on a standard Linux open-source operating system. This solution can help the users to achieve the full potentials of the open-source software and solve the problem of the interruption which normally occurs in using the controlling softwares based on the standard Windows operating system.

Several researchers applied the real-time operating systems to control the robots but they used the old real-time operating systems that hadn't been supported such as RTLinux or RTAI Linux[1,3–6]. Robert S. G. et al. used a commercial real-time operating system, the Java real-time, to control the robot with very high cost for using[7]. Choi B. W. et al. used the real-time Linux-Xenomai to get the signals and controll a mobile robot but they just mentioned about the accuracy in the position, not discussing about the accuracy in the time[8]. To overcome the disadvantages of the previous researches, this paper concentrates on using a modern and open-source real-time operating system, the real-time Linux-Xenomai, to control the position of the robot in the desired time.

This paper proposes a new controller using the real-time operating system Linux-Xenomai based on a standard Linux operating system to control the robot. Section ***Overview about the advantages of the real-time operating system Linux-Xenomai*** shows the advantages of the real-time operating system Linux-Xenomai comparing the commercial controlling robot softwares. In addition, this section also shows the advantages of the real-time operating system Linux-Xenomai comparing with the other real-time operating systems. After that, a real-time controller of a 5-DOF robot based on the real-time operating system Linux-Xenomai is built and shown in section ***Method of building a real-time controller for the 5-DOF robot***. In section ***Experimental results and discussions***, the experiments are done to control the robot to track the desired trajectory in the desired time. The comparison about the control time between the standard Linux operating system and the real-time Linux-Xenomai operating system is also mentioned.

OVERVIEW ABOUT THE ADVANTAGES OF THE REAL-TIME OPERATING SYSTEM LINUX-XENOMAI

In this section, the paper compares the real-time operating system Linux-Xenomai with the commercial controlling robot softwares. This section also analyzes the characteristics of the real-time operating system Linux-Xenomai and other real-time operating systems. From that the paper shows the advantages of using the real-time operating system Linux-Xenomai in controlling robots.

As defined by Donald Gillies "A real-time operating system is one in which the correctness of the system depends not only on the logical results but also on the time at which the results are produced. If the timing constraints are not met, system failure is said to have occurred." Therefore, the real-time operating systems often provide the priority of tasks. The higher priority tasks will be done before the others. It is said that the most basic differences of the real-time operating systems with the non-real-time operating systems are the strict requirement of the time, the access of resources, and the importance of scheduling.

Advantages of Linux-Xenomai comparing commercial controlling robot softwares

In this part, the paper analyzes the features, the advantages and the disadvantages of the commercial controlling robot softwares. The chosen software is Robot Studio, a famous production of ABB Company for controlling ABB robots. Several conclusions are listed as follows. The first advantage of this software is that it has a friendly user interface. Users can interface with the software and feel easy to program with the simple programming steps. Customers can also get

the most out of the robot features they have in a short time getting used to the program. The second advantage comes from the high compatibility between the software and the real system. After being simulated, the program can be exported directly to the company's robots without intermediary steps.

However, this software also exists several disadvantages. The first one is high cost for use, nearly 7000$ per year. This price is so high to the small and medium companies. Most of Vietnamese enterprises do not have the high budget, so they mainly use the old production tools and machines imported from the developed countries. Therefore, when using the new updated software, the compatibility is not as high as expected. In addition, integrating with only one specific line of the robots is a disadvantage when they do not have many options in the maintenance installation. The second problem is that this software is only used for robots of ABB Company. Other types of industrial robots use their own softwares. Thus, users need time to train and learn how to use again. The third problem is the secret of the software. It is difficult for engineers to exploit and create the new functions of the robots. The interfering into the controllers of the robots is also limited.

Based on the monitoring criteria, a comparison about the advantages and the disadvantages between the ABB's controlling robot software and the real-time Linux-Xenomai is shown in the Table 1. From this, it is said that the real-time Linux-Xenomai has the ability to control the exact position as the Robot Studio software, while it has the outstanding advantages of the ability to ensure time accuracy control, the low cost, the used language, and the ability to interfere with the hardware of the robot.

Advantages of real-time operating system Linux-Xenomai comparing other real-time operating systems

Actually there is not much papers that mentioned about the comparison between real-time operating Linux-Xenomai and other commercial real-time operating systems. However, there are several researches about the advantages of the Linux-RTAI compared with other real-time operating systems. Thus, we can estimate the characteristics of operating system Linux-Xenomai comparing other real-time operating system by comparing them with operating system Linux-RTAI.

In the article "A Real Time Operating System (RTOS) Comparison"[9], the author had done many experiments to compare the performance of the Linux-RTAI with the commercial real-time operating systems such as Win CE, QNX Neutrino, VxWorks, and μC / OS. According to the experimental results, the paper concluded that the Linux-RTAI has small response time, small latency, and small latency jitter comparing the most of other real-time operating systems. Only μC / OS-II has better ability than Linux-RTAI but with very expensive cost.

Xenomai is a project that started from August 2001. In 2003, it teamed up with the RTAI project to create an industry-free integrated real-time software architecture for the GNU / Linux operating system called RTAI / fusion. However, in 2005, the Xenomai project was separated from the RTAI project to develop its own RTAI / fusion project and develop independently to this day. This is the reason why Linux-RTAI and Linux-Xenomai have several same characteristics. Koh et al.[10] showed that in both RTAI and Xenomai, the response time become faster when the task period was shorter. The paper also concluded that the real-time mechanism showed no difference in performance with changes in the load task. Barbalace et al.[11] mentioned that both RTAI and Xenomai are worth of consideration. The paper also concluded that Xenomai proved to be slightly less performing than RTAI, mainly because of its layered approach, which introduces some overhead in interrupt management. However, Xenomai is better structured and is available for a larger number of platforms.

From that, a comparison between Linux-Xenomai, Linux-RTAI and the other commercial real-time operating systems based on several important considered criteria have been done and shown in the Table 2. Several outstanding advantages of the real-time operating system Linux-Xenomai can be concludes as follows. This operating system has the free resource repository. By allowing the threads to switch positions within the domain, Xenomai allows a fast and stable response time to prevent the crashes. Moreover, Xenomai provides a set of simulation classes and is very useful when connecting to the large systems. Finally, the good communication ability in the communication environment with UDP protocol is also an advantage of Xenomai.

METHOD OF BUILDING A REAL-TIME CONTROLLER FOR THE 5-DOF ROBOT

Overview of the 5-DOF robot

The robot used in this paper is a 5-DOF robotic arm of five revolute joints, shown in the Figure 1 [12]. The first

Table 1: Comparison between real-time Linux-Xenomai and Robot Studio software

Criteria	Robot Studio	Linux-Xenomai
1. Schedule latency	Large ($\approx$ 424 μs)	Small ($\approx$ 11.4 μs)
2. Jitter	Large ($\approx$ 350 μs)	Small ($\approx$ 7.01 μs)
3. Response time	Large ($\approx$ 100 μs)	Small ($\approx$ 5 μs)
4. Accuracy of control time	Accuracy is not guaranteed by interrupt error in the Windows operating system	Ensuring the accuracy of control time
5. Accuracy in control position	Ensuring the accuracy on position control	Ensuring the accuracy on position control
6. Cost	High,7000$/year	Free
7. Programming language	Its own programming language, it takes time to learn new	Common C programming language
8. The friendliness with users	High compatibility and user friendly interface	Simple user interface
9. The ability to interfere with the robot's hardware	Inability to interfere with the robot's hardware	Able to interfere with the robot's hardware

Table 2: Comparison of Linux-Xenomai, Linux-RTAI and other real-time operating systems

Criteria	Linux-Xenomai	Linux-RTAI	Commercial RTOS (VxWorks, QNX Neutrino, Win CE)
1. Schedule latency	Small ($\approx$11.4 μs)	Small (11.4 μs)	Large (49.2 μs)
2. Jitter	Small ($\approx$7.01 μs)	Small (7.01 μs)	Large (43.73 μs)
3. Response time	Fast ($\approx$5 μs)	Fast (5 μs)	Slow (14.62 μs)
4. Price	Free	Free	High cost
5. Ability of Support	Widely	No longer support	Limited

three joints are used to move the tool point to its desired position, while the last two joints adjust the orientation of the end-effector. Each joint is controlled by a DC servo motor. These motors are the products of HARMONIC DRIVE. They are linked with the encoders, which have the resolution of 200 pulses/rev, and fixed with the gear box.

The use of the forward kinematics is to find the position and the orientation of all links and the end-effector of the robot [12]. DH method was used to develop the kinematic model of the robotic arm because it is very versatile. This method is suitable for any model with any number of joints and links. From the relationships of the forward kinematics, we can solve the inverse kinematics for the robotic arm to control the position and orientation of the end-effector to follow a desired orbit.

The electrical diagram of the robot's controller is shown in the Figure 2. The controller includes the computer for the calculation part and the microcontrollers for the control part. The computer communicates with the microcontrollers via RS232 protocol. In this controller, the computer is used to carry the calculation of robot's kinematic problem as well as plan the paths for the robot to follow a desired trajectory. The output of the computer part is a string of charac-

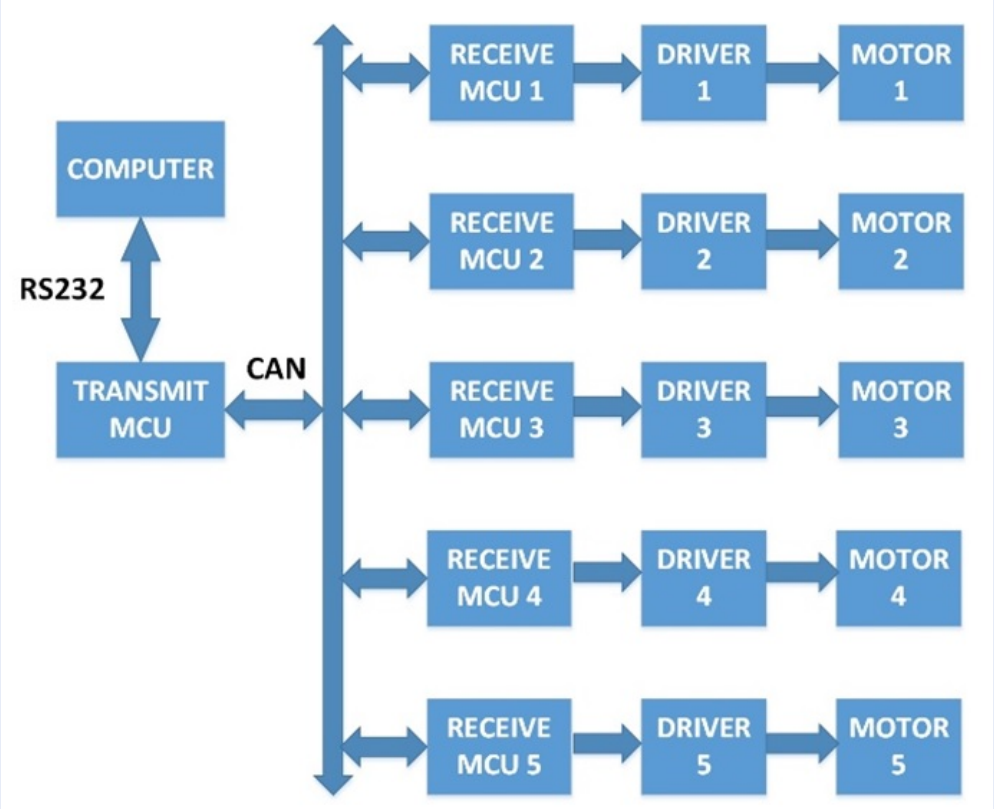

Figure 2: Electrical diagram of the robot controller

Figure 1: The 5-DOF robotic arm

ters that containing the desired pulses for each motor at the joints of the robot. For the control part, the distributed architecture is used to control the joints of the robot to reduce the number of tasks that a microcontroller should operate. In this research, the master-slave architecture is used which one microcontroller plays a master role and five microcontrollers play the slave role. The master microcontroller receives the data from the computer via RS232 protocol. This microcontroller will split the received string into five smaller strings that contain the numbers of pulses that each motor needs to rotate. These strings are sent to the slave microcontrollers via CAN communication between the master microcontroller and the slave microcontrollers. Each slave microcontroller will control the motor to get the desired pulses via a servo drive.

Real-time controller

In previous research [12], we built a graphical user interface (GUI) running on the Windows operating system to control the robot. However, using this GUI, we cannot sure that the robot will move and track the desired trajectory on time. In this research, we build a real-time controller based on the real-time operating system Linux-Xenomai. We apply the real-time controller for controlling the robot to track a desired trajectory on the desired time.

Tracking algorithm

In this section, we build an algorithm to control the robot to track a desired square orbit of 50mm x 50mm.

This trajectory includes four line paths with the length of 50mm. To control the robot to follows these line paths, we divide these lines into several intermediate points. At each point on the trajectory, we solve the inverse kinematic problem to get the desired angles and they are transformed to the desired pulses for each joint of the robot. The number of desired pulses for each joint is described by six data bytes. The robot has five motors, thus a frame of 31 data bytes is needed to transfer data from the host computer to the master microcontroller.

The algorithm of controlling the robot to track a desired square orbit of 50mm x 50mm is shown in the Figure 3. Firstly, the robot is controlled to move to the HOME position. After that, the robot is controlled to move to the FIRST position of the square trajectory. The square trajectory includes four linear trajectories of 50m length. Each line is divided into 50 segments with 51 intermediate points. Thus, the desired trajectory is divided into 200 intermediate points. At each intermediate point on the trajectory, the inverse kinematic problem is solved to get the desired pulses for each joint of the robot. Each 0.75 second, the host computer send the new update value of the desired pulses to the master microcontroller. The master microcontroller processes the data and sends the corresponding pulses to each slave microcontroller. The slave microcontroller will control the motor to the desired position. This work repeats until the robot reach the final point and finish the desired trajectory.

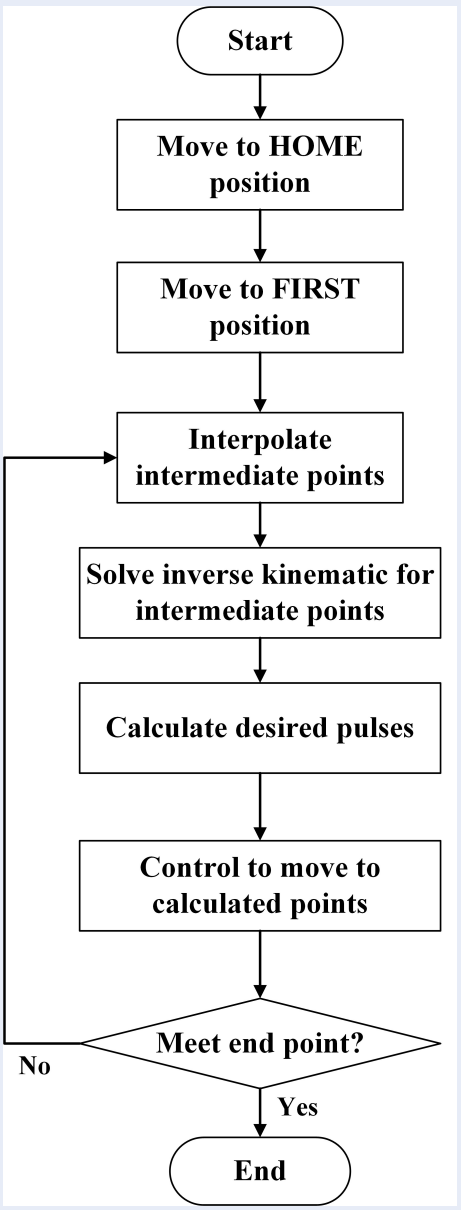

Figure 3: Tracking algorithm

Real-time program

In this section, we show how to build the real-time program to control the robot to follows a desired trajectory on time.

Actually, there are several difficulties to make the robot work on time. The first problem is that how we sure that the host computer send the signal to the master microcontroller at exact each 0.75 second. The latency in the normal operating system like Windows make it has the error in the processing functions. Because this reason, maybe the host computer need more 0.75 second to process the data and send the data to the master microcontroller. Thus, there are much errors of time during tracking the trajectory.

The second problem is that the data frame has 31 bytes and the host computer send this data to the master microcontroller by RS232 protocol. Using the normal driver of the standard operating system, there always exists the late time during transferring the data. In conclusion, it is impossible to make the system work on time with the standard operating system.

These above two problems can be solved by using the real-time operating system Xenomai [13]. Using the function rt_task_set_periodic, the program can sure that the host computer will send the data to the master microcontroller exactly at each 0.75 second. Besides, RT driver of RS232 protocol is used to sure about the real-time transfer in communication between the host computer and the master microcontroller [14]. The proposed program will be checked by the experimental model shown in the next section.

EXPERIMENTAL RESULTS AND DISCUSSIONS

Experimental setup

In this part, the experiments are made for the 5-DOF robot to track the desired trajectories. Two selected type of trajectories are square trajectory and circular trajectory.

To estimate the results, the experiments are made for both the normal controller and the real-time controller. To measure the time for each cycle of the trajectory, we use the proximity sensors and a real-time clock to determine the time for one period. Two proximity sensors are used to measure the time and they are put in the opposite positions. Figure 4 shows the hardware controller of the robot. The experimental setup to check the time to finish one period of trajectory is shown in the Figure 5.

Figure 4: Controller of robot

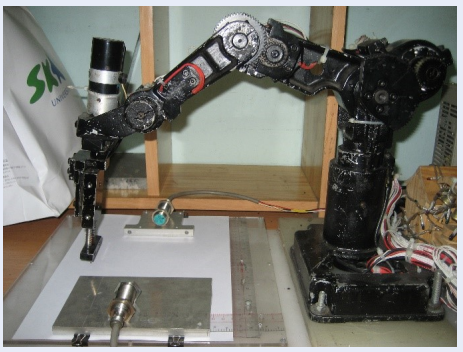

Figure 5: Experimental setup

Experimental results

The experimetal results of tracking the square trajectory

In the first experiment, we intend to control the robot to track a square trajectory of 50mm x 50mm in the desired time of 145 seconds. This work is made for 11 times for both the standard Linux operating system and the real-time Linux-Xenomai operating system. During the time of tracking the trajectory, we add several loads to the host computer so that the computer need to process more tasks similarly when the host computer read the signal from the external sensors. The results of the time for completing each round of the square trajectory are shown in the Table 3.

Figure 6 shows the result of drawing a square trajectory using the real-time Linux-Xenomai. The position error of tracking this trajectory is about 3%. Our group also made experiments of drawing the square trajectory in 11 times using both the standard Linux and the real-time Linux-Xenomai. The position errors in these two cases are nearly the same.

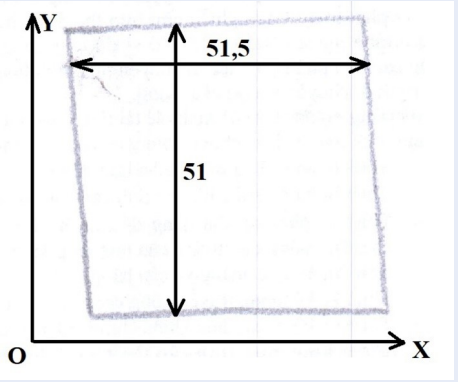

Figure 6: Result in drawing a square path of 50mm x 50mm using Linux-Xenomai.

From the results in the Table 3, we can estimate the time error for tracking a square trajectory. In case of using the real-time Linux-Xenomai, the average errors of sensor 1 and sensor 2 are 0.36 seconds and 0.91 seconds respectively. These errors are so small comparing the case of using the standard Linux, that are 8 second for sensor 1 and 10.36 seconds for sensor 2. When the loads are increased, the errors of time also increase shown in the case of number 10 and 11 in the Table 3. Thus, we can conclude that using real-time Linux-Xenomai will give more accurate results of desired time than using the standard Linux.

The experimental results of tracking the circular trajector

In the second experiment, we intend to control the robot to track a circular trajectory having diameter of

Table 3: Experimental results of controlling robot to track the square path using standard Linux and Linux-Xenomai

No.	Time in 1 round (seconds)			
	Linux-Xenomai		Standard Linux	
	Sensor 1	Sensor 2	Sensor 1	Sensor 2
1	144	146	145	150
2	145	145	145	145
3	144	145	146	149
4	144	144	150	153
5	145	143	155	153
6	145	145	157	155
7	145	147	150	152
8	145	145	150	149
9	144	144	149	152
10	145	144	167	175
11	145	147	169	176

40mm in the desired time of 61 seconds. This work is made for 11 times for both the standard Linux operating system and real-time Linux-Xenomai operating system. During the time of tracking the trajectory, we also add several loads to the host computer so that the computer need to process more tasks similarly when the host computer read the signal from the external sensors. The results of the time for completing each round of the circular trajectory are shown in the Table 4.

Figure 7 shows the result of drawing a circular trajectory using the real-time Linux-Xenomai. The position error of tracking this trajectory is about 3.75%. Our group also made experiments of drawing the circular trajectory in 11 times using both the standard Linux and the real-time Linux-Xenomai. The position errors in these two cases are nearly the same.

From the results in the Table 4, we can estimate the time error for tracking a circular trajectory. In case of using the real-time Linux-Xenomai, the average errors of sensor 1 and sensor 2 are 0.27 seconds and 0.91 seconds respectively. These errors are so small comparing the case of using standard Linux, that are 3.55 second for sensor 1 and 4.09 seconds for sensor 2. The reason that the error of time in this case is smaller than the case of tracking the square trajectory is that the desired time of tracking the circular trajectory is smaller than in case of tracking the square trajectory.

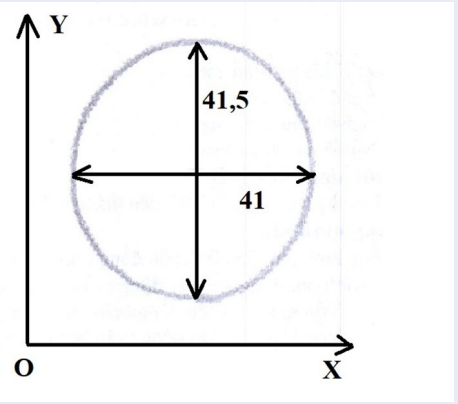

Figure 7: Results in drawing a circular path (diameter of 40mm) using Xenomai.

Discussions

The results of tracking the square trajectory and the circular trajectory show the errors of 1.5/50 mm for the square trajectory and 1.5/40 mm for the circular trajectory. The reasons of the errors can be explained as follows. The first reason comes from the mechanical structure of the robot. Although the robot has harmonic drives for each motor but at joint 2 and joint 3, there are a reduce gears. Thus, there is backlash in the gears and increase the motion errors of the robot. The second error comes from the electronic control hardwares. The error in each joint, using DC motor

Table 4: Experimental results of controlling robot to track the circular path using standard Linux and Linux-Xenomai

No.	Times in 1 round (seconds)			
	Linux-Xenomai		Normal Linux	
	Sensor 1	Sensor 2	Sensor 1	Sensor 2
1	61	59	60	63
2	61	61	62	63
3	61	62	63	63
4	61	62	64	65
5	62	62	64	67
6	61	60	69	69
7	61	61	65	68
8	61	62	66	64
9	62	62	65	66
10	61	62	65	64
11	62	60	65	64

control, add to the total error of the robot. The third reason comes from the control structure. Using master – slave architecture to control the robot and donot sure about the synchronous control of the joints of the robot.

However, the paper can conclude that using real-time Linux-Xenomai will give more accurate results of the desired time than using the standard Linux.

CONCLUSION

This paper proposes a real-time controlling method for the 5-DOF robot using the real-time operating system Linux-Xenomai. The controller of the robot has been built including both the hardware part and the software part. The paper also mentions the working of this controller by controlling the robot to track the desired trajectory in the desired time. Experimental models have been made to compare the working of the normal controller and the real-time controller. The experimental results show that the real-time controller can make robot work accurately both in the position and in time. In case of the normal controller, the system cannot complete the work on time. In the future, we will try to apply the real-time controller in several other tasks that require the accuracy of time.

ACKNOWLEDGEMENTS

This research is funded by the Department of Science and Technology of Ho Chi Minh City, Vietnam, under grant number 318/2012/HD-SKHCN. We acknowledge the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for this study.

LIST OF ABBREVIATIONS

DOF: Degree of Freedom
CAN: Controller Area Network
RTOS: Real Time Operating System
GUI: Graphical User Interface
MCU: Microcontroller Unit

CONFLICTS OF INTEREST

The author declares that he has no conflicts of interest.

AUTHOR' CONTRIBUTION

Tri Cong Phung has built the realtime controller and made experiements. He aslo has written and edited the manuscript.

REFERENCES

1. Goldenberg AA, Chan L. An approach to Real-Time Control of Robots in Task Space: Application to Control of PUMA 560 without VAL-II. IEEE Transactions on Industrial Electronics. 1988;35(2):231–238. Available from: https://doi.org/10.1109/41.192654.
2. Ortega JG, Garcia JG, Nieto LM, Garcia AS. Open Software Architecture for Advanced Control of Robotic manipulators in Non-structured Environments. Workshop at the IEEE 2010 International Conference on Robotics and Automation, Alaska, USA. 2010;.
3. Han SH, Seo WH, Yoon KS, Lee MH. Real-time control of an industrial robot using image-based visual servoing. The IEEE/RSJ International Conference on Intelligent Robots and Systems. 1999;3:1762–1767.

4. Valera A, Mata V, Valles M, Valero F, Rosillo N, Benimeli F. Solving the inverse dynamic control for low cost real-time industrial robot control applications. Robotica. 2003;21(3):261–269. Available from: https://doi.org/10.1017/S0263574702004769.
5. Zhou Y, Silva CW. Real-time control experiments using an industrial robot retrofitted with an open-structure controller. Proceedings of IEEE International Conference on System, Man, and Cybernetics. 1993;4:553–559.
6. Maccheli A, Melchiorri C. A real-time control system for industrial robots and control applications based on real-time Linux,. The 15th International Federation of Automatic Control World Congress. 2002;35(1):55–60. Available from: https://doi.org/10.3182/20020721-6-ES-1901.00821.
7. Robertz SG, Henriksson R, Nilsson K, Blomdell A, Tarasov I. Using real-time Java for industrial robot control. JTRES Proceedings of the 5th International Workshop on Java Technologies for Real-time and Embedded Systems. 2007;p. 104–110. Available from: https://doi.org/10.1145/1288940.1288955.
8. Choi BW, Shin DG, Park JH, Yi SY, Gerald S. Real-time control architecture using Xenomai for intelligent service robots in USN environments. Intelligent Service Robotics. 2009;2:139–151. Available from: https://doi.org/10.1007/s11370-009-0040-0.
9. Aroca RV, Caurin G. A Real Time Operating System (RTOS) Comparison. The 6th Workshop on Operating Systems. 2009;p. 2441–2452.
10. Koh JH, Choi BW. Real-time Performance of Real-time Mechanisms for RTAI and Xenomai in Various Running Conditions. International Journal of Control and Automation. 2013;6(1):235–245.
11. Barbalace A, Luchetta A, Manduchi G, Moro M, Soppelsa A, Taliercio C. Performance Comparison of VxWorks, Linux, RTAI, and Xenomai in a Hard Real-Time Application. IEEE Transactions on Nuclear Science. 2008;55(1):435–439. Available from: https://doi.org/10.1109/TNS.2007.905231.
12. Phung TC, Tran NH, Nguyen DA. Research on Building an Algorithm for a 5DOF Robotic Arm to Assemble Object on a Moving Conveyor Belt", National Conference on Machines and Mechanisms, Ho Chi Minh City, Vietnam. 2015;p. 31 –40.
13. Phung TC, Nguyen DA. Research and Improving the Accuracy of Time in Controlling Motors using Xenomai Realtime Software. National Conference on Mechatronics, Dong Nai province, Vietnam. 2014;p. 324 –329.
14. Kiszka J. The Real-Time Driver Model and First Applications. The 7th Real-Time Linux Workshop, Lille, France. 2005;.

Tạp chí Phát triển Khoa học và Công nghệ – Engineering and Technology, 3(3):xxx-xxx

Open Access Full Text Article

Bài Nghiên cứu

Nghiên cứu về ứng dụng phương pháp điều khiển thời gian thực nhằm điều khiển quỹ đạo robot trong khoảng thời gian xác định

Phùng Trí Công[1,2,*]

Use your smartphone to scan this QR code and download this article

TÓM TẮT

Trong lãnh vực robot, việc điều khiển chính xác vị trí và vận tốc để làm cho robot vận hành trong một khoảng thời gian mong muốn là một yêu cầu quan trọng trong công nghiệp. Những công việc hiện đại yêu cầu robot có thể làm việc trong các dây chuyền phức tạp mà cần phải thu thập thông tin từ môi trường làm việc xung quanh. Trong trường hợp này, tín hiệu từ cảm biến cần phải được xử lý trước khi gửi đến bộ điều khiển robot những thông tin cần thiết. Vì vậy thời gian giao tiếp giữa máy tính chủ xử lý tín hiệu cảm biến và bộ điều khiển robot đóng vai trò quan trọng trong hoạt động của hệ thống. Hơn nữa, khả năng xử lý tín hiệu trong khoảng thời gian thực cũng rất quan trọng. Có rất nhiều ứng dụng đòi hỏi sự tuân thủ nghiêm ngặt về thời gian hoạt động của toàn bộ hệ thống. Vấn đề ngắt thường xảy ra đối với người sử dụng hệ điều hành Windows thông thường. Vì thế khó dung Windows để điều khiển robot và các thiết bị để chúng làm việc đúng giờ. Để giải quyết vấn đề này, một bộ điều khiển thời gian thực nên được sử dụng để điều khiển hoạt động của robot. Các hệ điều hành thời gian thực mã nguồn mở không những có lợi thế hơn các hệ điều hành thông thường về tính kinh tế và độ linh hoạt mà còn đáp ứng được yêu cầu đặt ra ở trên. Bài báo này tập trung vào việc xây dựng các giải thuật điều khiển quỹ đạo robot trong một khoảng thời gian xác định sử dụng một hệ điều hành thời gian thực mã nguồn mở gọi là Linux-Xenomai. Đầu tiên bài báo sẽ tiến hành phân tích các ưu điểm của hệ điều hành thời gian thực Linux-Xenomai so với các hệ điều hành thông thường và các hệ điều hành thời gian thực khác. Sau đó một bộ điều khiển thời gian thực của một robot 5 bậc tự do được xây dựng dựa trên hệ điều hành thời gian thực Linux-Xenomai. Kế đến bài báo trình bày các giải thuật để kiểm tra khả năng làm việc đảm bảo độ chính xác về thời gian của robot. Cuối cùng các thí nghiệm được thực hiện nhằm kiểm chứng giải thuật đề ra.

Từ khoá: robot công nghiệp, động học ngược, hệ điều hành, điều khiển vị trí, điều khiển thời gian thực

[1]*Bộ môn Cơ điện tử, Khoa Cơ khí, Trường Đại học Bách khoa (HCMUT), 268 Lý Thường Kiệt, Phường 14, Quận 10, TP.HCM, Việt Nam*

[2]*Đại học Quốc gia Thành phố Hồ Chí Minh, Phường Linh Trung, Quận Thủ Đức, Việt Nam*

Liên hệ

Phùng Trí Công, Bộ môn Cơ điện tử, Khoa Cơ khí, Trường Đại học Bách khoa (HCMUT), 268 Lý Thường Kiệt, Phường 14, Quận 10, TP.HCM, Việt Nam

Đại học Quốc gia Thành phố Hồ Chí Minh, Phường Linh Trung, Quận Thủ Đức, Việt Nam

Email: ptcong@hcmut.edu.vn

Lịch sử

- Ngày nhận: 27-8-2020
- Ngày chấp nhận: 26-11-2020
- Ngày đăng: 03-12-2020

DOI :10.32508/stdjet.v3i3.759

Check for updates

VNU-HCM Press

Trích dẫn bài báo này: Công P T. **Nghiên cứu về ứng dụng phương pháp điều khiển thời gian thực nhằm điều khiển quỹ đạo robot trong khoảng thời gian xác định**. *Sci. Tech. Dev. J. - Eng. Tech.;* 3(3):461-471