Robot Control Unit for Educational Purposes

Gerhard Rath, Gerold Probst and Werner Kollment
Department Product Development
University of Leoben
Leoben, Austria

***Abstract*—A standard serial 6-DOF industrial robot was equipped with an electronic control unit designed especially for educational purposes. A fast Ethernet LAN with UDP communication allows a real-time control from a PC, which runs a widely used scientific software. Even inverse and direct kinematics are computed on the PC and hence are available for education. The configuration allows interactive reaction to sensor signals or image processing applications. In a student project a motion sensing device for video games was implemented for interactive control of the robot with the human body.**

Keywords-robotics; education; control

I. Introduction

A. Robotics in Education

Since robots play an important role in industry for several decades, education in robotics is important in engineering curricula. Teaching on real robots is critical due to high dynamics and power of these devices. Safety measures required during the work with the robot decrease the chance for creative experiments for the students. Mistakes cannot be tolerated. Furthermore control strategies are hidden inside the controller and kept as secrets by the manufacturers. Programming is inflexible and restricted to turn-key controllers. Only teach-in procedures provide some interesting flexibility. From interviews with our students and those of other institutions we hear that, after enthusiasm at the begin of the class, the work with the robot is experienced increasingly boring because of these restrictions. Activities to make learning about industrial robots more effective and interesting need the connection of electronic controllers with scientific tools like Matlab™ [1][2].

As an alternative, the idea of using non-industrial robots with lower price and lower risks for teaching came up much earlier, inspired by the Braitenberg vehicle [3]. Institutions started to develop their own devices to fascinate their students for robotics [4] and to minimise costs [5]. The introduction of competitions attracted students to acquire the skills to construct robots, which are mechanical, electrical and electronics engineering, and theoretical basics like dynamics, kinematics, control theory, and computer languages [6][7]. Competition may take place either with a number of robots of identical hardware [8][9] or free design of hard- and software [10]. Standard learning platforms arose that lowered the costs and offered higher versatility to address the creativity of students [11]. Even children were addressed in the pioneer work of Seymoor Papert [12], resulting in a product called *Lego mindstorms.*

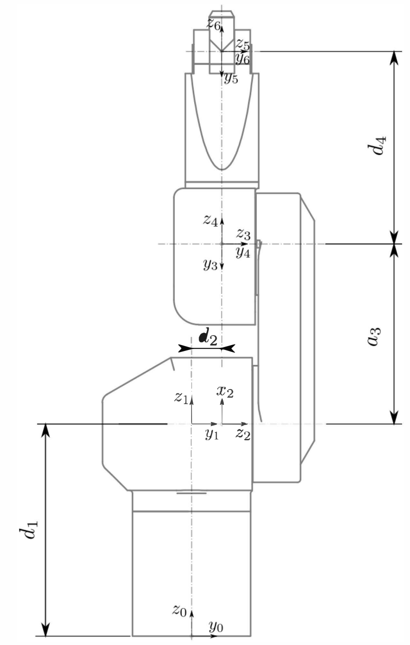

Fig. 1. Front view of the robot with Denavit-Hartenberg parameters

This trend was reinforced with the progress in software development. It is standard today that educational robots may be programmed with different software platforms [5]. Dedicated tools like Microsoft's *Robotics Developer Studio (MRDS)* or Python-based *Myro* are examples.

The high attractiveness of robots is not only used to transport dedicated skills, but also to increase the number of students for technical studies [13]. For example, studying the behaviour of crickets with the help of *e-puck* robots is much more interesting than using pure computer simulation [14].

Recent developments of inexpensive hardware platforms, e.g. *Arduino*, *Raspberry Pi* and *BeagleBone*, together with open source platforms like real-time *Linux* [15] and *LinuxCNC*, enable students who are not IT experts to create their own machines, for example 3D printers. They accomplish this even outside of their classes with the help of community-driven support.

B. Intention of the Actual Project

In this situation we started our project to make standard industrial robots more attractive for our students who are mainly mechanical engineers. The intention was to allow more flexibility for creative experiments

ISBN 978-80-261-0386-8, © University of West Bohemia, 2015

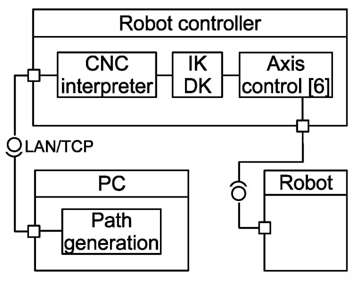

Fig. 2. Structure of a standard industrial robot control system

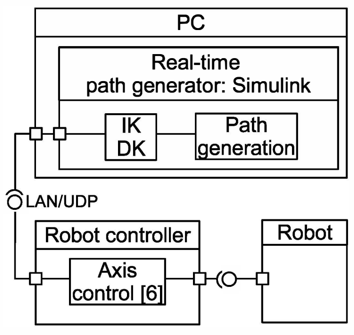

Fig. 3. Structure of the actual educational control system

without reducing safety and reliability. Our 6DOF manipulator (Fig. 1) was mainly used for auxiliary tasks in our laboratory, for example for positioning of targets for image processing, or moving specimens in test facilities for measurements. The intention was to equip the robot with a new controller providing a simple and fast interface to standard scientific software. This should open the system for student education and experiments.

C. Structure of a Standard Robot Controller

Many different approaches exist to control a serial robot [16]. For standard industrial robots usually all axes are controlled individually without regarding dynamic or kinematic interaction [17]. The structure of an industrial control system is according to Fig. 2. The robot controller is the part working in real-time and consists of six motion controllers, modules for inverse and direct kinematics, and a CNC code (G code) interpreter. The G code is a sequence of commands that describes the path of the tool. The code is generated offline on a PC maybe from CAD data. With this configuration, the robot can perform working tasks in production industry.

II. Description of the Educational Robot Control System

A. Structure of the Controller

The system in Fig. 2 interpretes path data and encloses the inverse and direct kinematics (IK and DK), hence it is not possible to include sensor data in a real-time manner into the control strategy. In the alternative design in Fig. 3 the PC does the path generation and calculation of the kinematics in real-time. Only the set values for the six axes angles are transmitted, while the actual values are received from the robot controller. Path data can be calculated in quasi real-time and may include sensor data. Even the kinematics is open for teaching.

B. Inverse and Direct Kinematics

IK and DK are central tasks for robot control. In a classical system (Fig. 2), this computation was critical due to the high numerical effort. In the past it was a challenge for microprocessors to calculate this in real-time. With modern hardware this job can be easily accomplished. One important idea of our project was to do the IK and DK on a remote hardware, since it does not essentially increase the CPU load, and the communication over LAN is no longer a bottleneck for a real-time system. This configuration makes kinematics available for education of upper grade students. For example, they might increase the accuracy for fast motions using the Jacobian matrix.

To control the robot, the DK is required to find the initial pose of the end-effector to start the motion smoothly from a given point. The IK then is used to find the joint angles for a desired motion path.

Denavit-Hartenberg convention [16] is used for kinematics (Fig. 1). The IK delivers eight solutions for each position and orientation of the end-effector. From this set, the angles with the least mean square differences to the previous solution are chosen for a new pose. The implementation was done as a Matlab™ script block in a Simulink™ model (Fig. 3). These objects now can easily be copied and pasted into a new application.

C. Communication

A fast communication is crucial in real-time systems. UDP is a connectionless, unreliable protocol of the TCP/IP suite and covers levels four and five of the OSI communication model (transport and session layers). Most industrial fieldbuses are based on UDP. Data integrity of a single datagram is given with a CRC sum. Adding a simple protocol establishes the reliability of the communication. To obtain a real-time behaviour, the LAN has to be a collision-free domain, which is obtained with switches and a proper protocol, such as Master/Slave.

In the actual project the PC (master) transmits the set values for the position of the six axes, and the controller responds the actual values. A cycle time of 10 ms is achieved. For safety reasons a timeout of 100 ms stops the robot, when no data were received.

D. Basic Control Program for Student Work

The basic example program in Fig. 4 shows how interactive robot control can be established for educational purposes in a simple manner. The tool centre point is moved with a 3D mouse as HMI device. Speed signals are integrated and fed into the IK block that calculates the angles for the six drives. The orientation of the end-effector (EE) is represented by two vectors f_1 and f_2 defining a plane in the EE coordinate system and is kept constant in this example. Before starting the first motion with the program, the integrator is initialised with the actual robot positions. For this

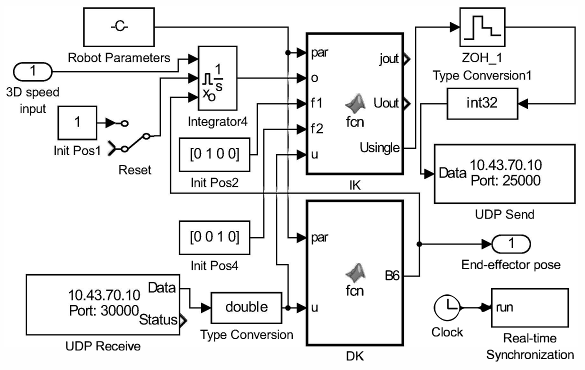

Fig. 4. Example robot control program

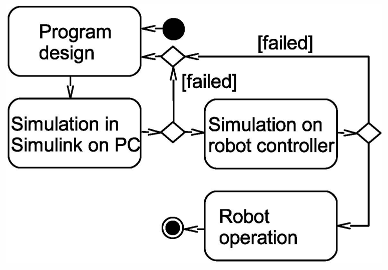

Fig. 5. Workflow of simulation tests

purpose the program needs the DK to calculate the EE pose from the actual robot position. IK and DK blocks contain Matlab™ scripts and are also available to be modified by the students.

III. Safety Considerations

Protection against human injury: The standard EN ISO 10218-1:2012-01 defines safety requirements to protect people from beeing hurt by the robot. In the actual project, a dedicated safety PLC and a light barrier are the main components to establish obligate safety.

Protection against software mistakes: Students' software is expected to be error prone. To protect the robot against damage caused by program mistakes, the development process was divided into four steps according to Fig. 5. At first the program is developed. In the second step, it is tested in a virtual reality environment with a virtual robot model. It is remarkable that the control program written in Simulink™ is almost identical with the final program. The simulation results can be observed on the screen of the PC. In the third step, the test includes the robot controller, which provides an additional simulation mode. Now also communication and real-time behaviour can be checked. The simulation output appears on the screen of the robot controller. The last step, after passing two simulation tests, is to activate the robot drives. Now the majority of software mistakes should be removed.

IV. Real-Time Performance

Real-time capability of the communication was crucial for the project, since important control tasks are carried out on a separate hardware, a PC in our case. Consequently, the performance of the UDP transmission over LAN was measured. To do this, a task on

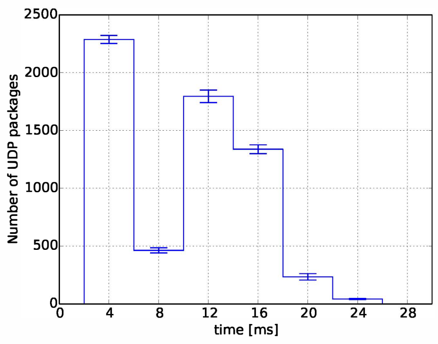

Fig. 6. Mean package interarrival-times with standard deviations

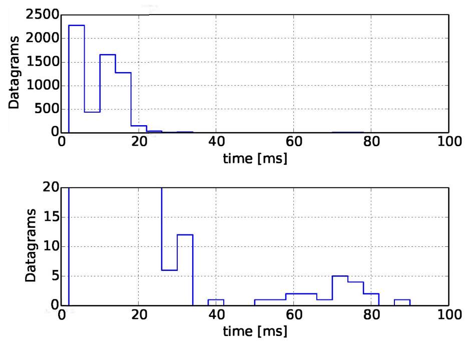

Fig. 7. Mean package interarrival-times witch human interactions I

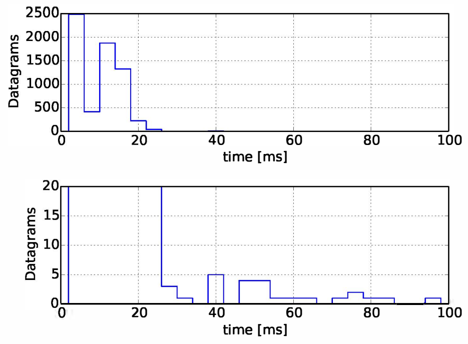

Fig. 8. Mean package interarrival-times witch human interactions II

Fig. 9. Student conducting the robot with Kinect™ sensor.

the real-time motor control system was programmed to take histograms of the interarrival times of the UDP datagrams. A crossover cable connected the robot controller directly with the PC with the path controller programmed in Simulink™. The nominal duration of one cycle was 10 ms. The duration of the measurement was 100 seconds. Fig. 6 shows that in the free running system no package has a longer delay than 30 ms. Standard deviations, also displayed in Fig 6, are very small, which means, the system behaves in a nearly deterministic manner. Counting the datagrams proved that no telegram got lost. These tests indicate real-time property of the controller, even if the operating system Windows™ is far away from being a real-time system. Also the program on Simulink™ was not optimised for computing speed in any way. Even when the real-time property cannot be guaranteed, it is sufficient for making experiments with the robot. The control task worked several hours without causing a timeout condition of more than 100 ms, which stops the robot motion.

Of course, arbitrary human interactions on the PC are not allowed during operation. Fig. 7 was taken over a minute with ten activations of a *manual switch* from the Simulink™ library. For a second experiment, ten value changes of a *constant* block type were done within one minute (Fig. 8). In both cases, there are few datagrams with interarrival times close to 100 ms, but no timeout condition ocurred. Other actions, like opening or closing a window, extend the time beyond the timeout value of 100 ms and will stop the motors.

This means, that it is possible to run a control program even while changing or switching parameters or values by human operators during run-time.

V. Conclusion

An industrial 6DOF robot was equipped with a new electronic control unit with an interface specifically designed for educational purposes. The set values and the actual values of the six motor drives are communicated via UDP protocol directly from and to Matlab™/Simulink™ programs. Safety-related programs stay on the electronic control, while computation of the direct and inverse kinematics are done on the PC. The set-up enables experiments in a quasi-real-time mode in a 10 ms cycle, even though software on the PC is not running under control of a dedicated real-time system. The system is open now to experiment with new algorithms for path planning or image processing, or for simple integration of sensors into the control loop. Working with the robot became more attractive for students.

Acknowledgment

The authors would like to thank Paul O'Leary, the head of our department, for enabling this project. Anton Gfrerrer helped us solving the kinematics. Roland Riepl from the Johannes Kepler University in Linz gave us valuable tips for successfully designing the robot controller. Last not least we thank our supplier of the electronic motor and logical controllers, Bernecker & Rainer, for their great support when facing problems.

References

[1] P. I. Corke, *Robotics, Vision & Control: Fundamental Algorithms in Matlab*, vol. Springer tracts in advanced robotics. Springer, 2011.

[2] T. Pawletta, B. Freymann, C. Deatcu, and A. Schmidt, "Robotic control and visualization toolbox for matlab," in *Proc. of MATHMOD 2015 - 8th Vienna Int. Conf. on Mathematical Modelling* (F. Breitenecker, A. Kugi, and I. Troch, eds.), (Vienna), pp. 371–372, 2015.

[3] V. Braitenberg, *Vehicles: Experiments in Synthetic Psychology*. Cambridge, MA, USA: MIT Press, 1984.

[4] B. Maxwell and L. Lisa Meeden, "Integrating robotics research with undergraduate education," in *IEEE Intelligent Systems Magazine*, pp. 22–27, 2000.

[5] C. Cardeira and J. da Costa, "A low cost mobile robot for engineering education," in *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE*, 2005.

[6] L. Almeida, J. Azevedo, C. Cardeira, P. Costa, P. Fonseca, P. Lima, F. Ribeiro, and V. Santos, "Mobile robot competitions: Fostering advances in research, development and education in robotics," in *Proceedings of CONTROLO2000, the 4th Portuguese Conference on Automatic Control, Guimares*, pp. 4–6, 2000.

[7] I. Verner and D. Ahlgren, "Robot contest as a laboratory for experiential engineering education," *Journal on Educational Resources in Computing*, vol. 4, June 2004.

[8] T. Collins and T. Balch, "Teaming up: Georgia tech's multi-robot competition teams," in *Proceedings of the National Conference on Artificial Intelligence*, (Providence, RI, U.S.A), pp. 785–786, 1997.

[9] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *In Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, pp. 59–65, 2009.

[10] H. Kitano and M. Asada, "Robocup humanoid challenge: Thats one small step for a robot, one giant leap for mankind," in *Proceedings of International Conference on Intelligent Robots and Systems*, (Tokyo), pp. 419–424, 1998.

[11] T. Balch, J. Summet, D. Blank, D. Kumar, M. Guzdial, K. O'Hara, D. Walker, M. Sweat, C. Gupta, S. Tansley, J. Jackson, M. Gupta, M. Muhammad, S. Prashad, N. Eilbert, and A. Gavin, "Designing personal robots for education: Hardware, software, and curriculum," *Pervasive Computing, IEEE*, vol. 7, pp. 5–9, April 2008.

[12] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*. NY, USA: BasicBooks, second ed., 1993.

[13] NASA, "The robotics alliance project." Website, 2014. Available online at http://robotics.nasa.gov/goals.php; visited on June 10th 2014.

[14] I. Fertschai, J. Stradner, and H. Römer, "Neuroethology of female preference in the synchronously singing bushcricket mecopoda elongate (tettigoniidae: Orthoptera): why do followers call at all?," *The journal of experimental biology*, vol. 210, pp. 465–476, 2007.

[15] B. Wagner, P. Hohmann, U. Gerecke, and C. Brenneke, "Technical framework for robot platforms in education," in *Proceedings of International Conference on Engineering Education and Research*, (Ostrava, CZ), pp. 699–703, 2004.

[16] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. Advanced Textbooks in Control and Signal Processing, London, UK: Springer, second ed., 2000.

[17] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, ch. 6. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2005.