Mladen Crneković
Davor Zorc

ISSN 1333-1124

KINEMATIC CONTROLLER FOR A MITSUBISHI RM501 ROBOT

UDC 621.865.8

Summary

In this paper we describe the revitalization and upgrading of a Mitsubishi RM501 robot, using a "kinematic controller". With this upgrade, problems of the old fashioned Centronics parallel communication to the robot are solved and the command set of the robot and its abilities are improved. The basic function of the kinematic controller is to enable the user to set the robot position directly in external coordinates and to receive information about the position and status of the robot. For this purpose, a complete kinematic model of the robot is put into the kinematic controller. This feature also enables the robot not to collide with the basis. Besides the basic commands of the robot, a set of useful commands are added to improve the robot language syntax. The kinematic controller was designed using the Atmel 8-bit microcontroller that is used in education. All these changes result in a great improvement in the educational process with the robot and enable the user to program the robot in a higher level.

Key words: *robot control, kinematic controller, engineering education*

1. Introduction

The rate at which new technologies come into our lives and also into our educational process results in the educational equipment becoming obsolete quickly and in its piling up in laboratories. From the aspect of functionality, this "old" equipment in our laboratories is still usable and meets all the necessary criteria. In educational sense, it is sometimes better than new equipment because the manufacturer has paid more attention to its function and method than its outward “cosmetic” appearance.

On the other hand, there is educational equipment that could use modernization because it would significantly improve the educational process. If we look carefully, very often the “repair” is not necessary for the whole piece of equipment but only for its informatics system. It is a good idea to include this “repair” into the educational process, not only as an idea but in full realization.

With the modernization of old equipment, the part including knowledge and ideas is much more dominant than the financial part, therefore it is cost-effective. The price of the parts necessary to improve equipment is sometimes negligible when compared to the price of new equipment.

One of these pieces of equipment is a Mitsubishi RM501 robot which students use at the beginning of their education in robotics, and later they move on to newer robots. Every academic year, around a hundred students go through exercises with robots. The exercises consist of three parts that progress from simple to more complex problems in robot programming.

2. Motivation

The purchase of a Mitsubishi RM501 robot in the second part of the 80s significantly improved the education in robotics at the Faculty of Mechanical Engineering and Naval Architecture. The students could operate a robot at a time when the term robotics was fairly unknown. The robotic system (Figure 1) consisted of a robotic arm, a two-finger gripper, a teaching box and a drive unit [1]. No software came with the robot and the manual was quite unhelpful. It was clear that software support had to be written to enable easier interaction with the robot. The problem was even more complex due to the fact that all the work invested into robot programming was lost when the robot was shut down because the program and the learned points did not stay in the memory of the drive unit. There was a way to memorize the data (using the EPROM programmer), but that was not practical from the educational aspect. A TANDY 102 computer that came with the robot was already obsolete at that time because PC computers had arrived into our laboratories. Therefore, our own software had to be created to enable easier interaction with the robot. For a short period of time the program was run on a PC-XT computer and it worked well. Because of the purchase of a PC-AT computer, the program was transferred to the new platform. However, the robot could not operate because communication with the robot could not be established. That is because the robot and the PC computer were connected by a parallel Centronics connection, which was usual for printers and worked well at the time.

Working on hardware was the only thing that could be done. After having opened the drive unit and the connectors, we established that the signal ACK of the Centronics connection on the robot side was missing. The XT computer could manage that, but the AT could not. Some changes had to be done, so instead of the command to write on the parallel port Write (LPT, …), we wrote an entire procedure writing on the parallel port of the PC, without the ACK signal. In that form, the program was functioning for a long time, even with Windows. However, with Windows XP, which do not allow a direct access to ports, the connection with the robot could not be established once again. Besides that, some notebook computers did not have a parallel port, so they could not be connected to the robot. At that point, a decision was made to ultimately solve the problem.

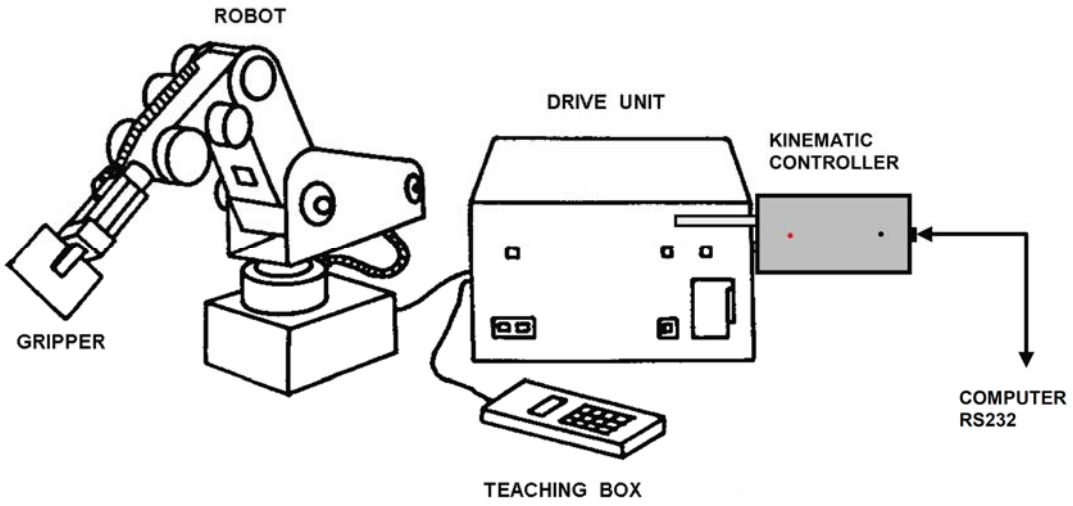

Fig. 1 Mitsubishi RM501 educational robotic system

On the other hand, the commands accepted by the drive unit of the robot, which refers to the robot movement, were very primitive and involved only setting the joint coordinates of the robot [2]. In these conditions, it was very hard to program even simple tasks (e.g. extraction from a matrix pallet). In the first version of our program, a kinematic model of the robot was already implemented, which enabled easier operating and programming of the

robot, but it had to be implemented into every new program language [3]. The result was that with every improvement in the robot functioning you had to start from the beginning, i.e. from the kinematic model of the robot. All that led to a lack of interest in the robot, even though it was still functional and required from the educational aspect.

3. Conceptual Solution

A conceptual solution to the problems stated above and an improvement in the functioning of the robot were determined and restricted by the poor documentation accompanying the robot. The manufacturer had not anticipated the user to make any changes to the drive unit. However, some groups were trying to find a solution to the robot improvement by replacing the existing controller with a completely new one based on a personal computer and some additional hardware [4] [5]. They report good results, but in this case, the level of necessary knowledge, solution time and cost represent a goal that is unattainable for us.

We have tried to solve the problem of communication with a serial connection by connecting to the serial connection of the robot, but it did not work. In discussions with news groups, we have found out that the serial connection did not work on other robots either, so we gave up on that. The final solution was: a converter from serial to parallel communication has to be designed. Additionally, a kinematic model of the robot with useful functions that can later be easily expanded has to be incorporated into the assembly. The device was called a "kinematic controller". That way, the only function of the robot drive unit was to control the robot movement. An Atmel AT89C5ID2 microcontroller was chosen for the kinematic controller [6] because its features met all the demands. The price of hardware is not more than 50 €, which is a trifling sum.

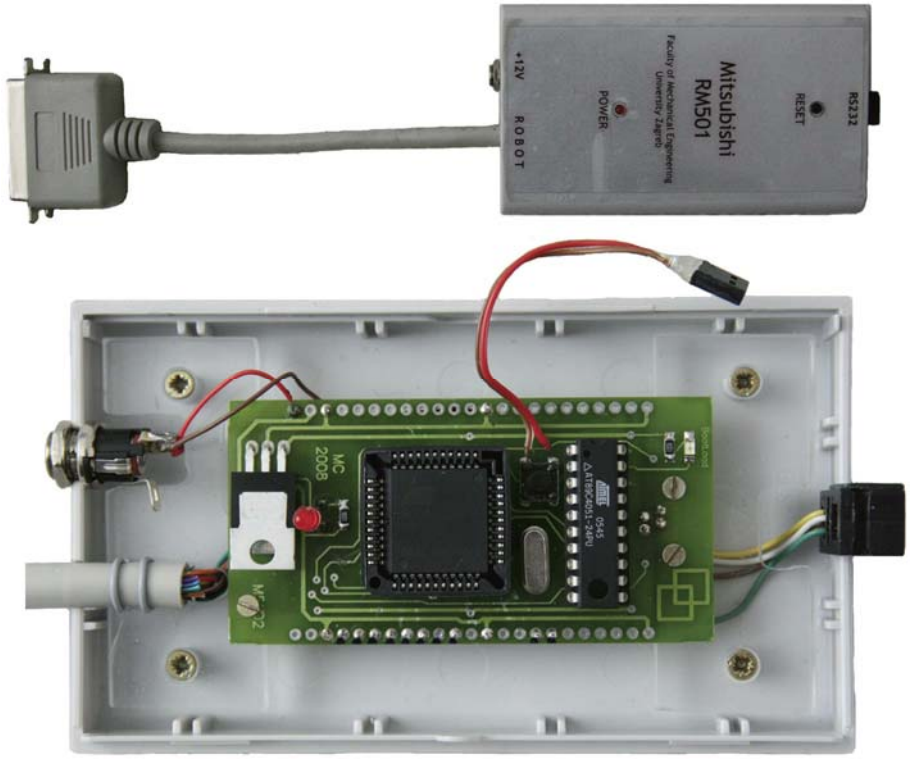

Fig. 2 Kinematic controller

4. Robot Kinematics

The Mitsubishi RM501 robot has 5 degrees of freedom (Figure 3) and the ability to grip with its two fingers. Its position can be described by the vector of joint coordinates $\mathbf{q} = [q_1\ q_2\ q_3\ q_4\ q_5]^T$ and the vector of external coordinates $\mathbf{r} = [\ x\ y\ z\ \varphi\ \psi]^T$. Figure 4 shows the robot's arm with its basic dimensions, from which the kinematics can be derived, i.e. the link between the **q** and **r** vectors.

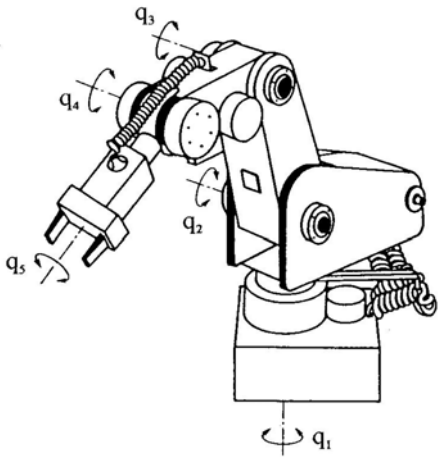

Fig. 3 Degrees of freedom of the Mitsubishi RM501 robot

Using the matrices of homogenous transformations and robot geometry, Fig. 4, we can solve equations of the direct and inverse kinematic problem [3]. The l_{TOOL} variable is added to the kinematic equation to determine the length of a tool or an object the robot is carrying (in the direction of the gripper), and can be changed during the robot task execution.

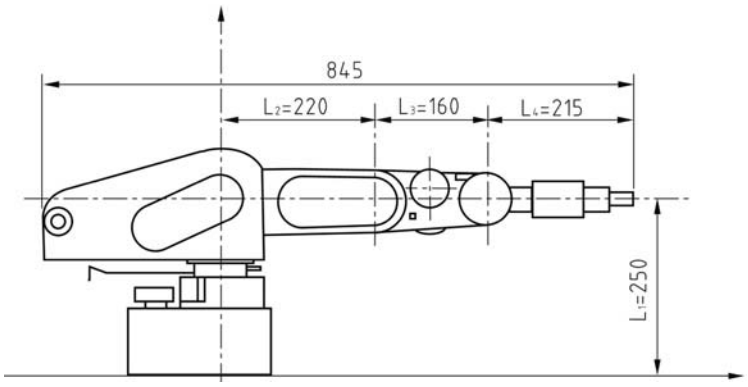

Fig. 4 Basic dimensions of the Mitsubishi RM501 robot

Feedback on every degree of freedom of the robot is obtained by means of optical incremental encoders. The resolution of q_1, q_2 and q_3 coordinates is 0.025°, and of q_4 and q_5 coordinates 0.075°. That is why the movement of every degree of freedom is set by an integer representing a unit of the coordinate resolution, e.g. parameter $\mathbf{a} = [\ a_1\ a_2\ a_3\ a_4\ a_5\ ]^T$. According to microswitch position of each joint coordinate, it is possible to establish a connection between coordinate **q** and parameter **a** [3].

Since microswitches of the joint coordinates are not mounted on the robot precisely, the robot position parameter should be corrected for a constant amount **c**. This constant is acquired by robot calibration and is measured as $\mathbf{c} = [\ 0\ \ 163\ \ 8\ \ 925\ \ 925\ ]^T$ [3]. Now, from prior equations for q_i, correct inverse relations can be obtained.

If the command "MI -6000, -2600, 1800, 1200, -1200, 0" is executed after initialization, parameters a_i have symmetrical limits which are suitable for programming.

The kinematic equations from [3] are the basis for making the program support. The program substitutes all the functions of the teaching box completely and enables the programming of the robot.

5. Controller Hardware

The kinematic controller is built around an Atmels 8 bit microcontroller AT89C51ID2 and works effectively on 1 MHz clock. It has 64 kB of program memory and 4 kB of EEPROM memory. The whole program takes 14.5 kB of program memory and is capable of calculating the robot kinematic equations with no significant delay. Besides the microcontroller, the kinematic controller contains voltage stabilization LM7805 protected from reverse polarity and a RS232 communication signal level converter ADM232 for the connection with PC. External power supply of 12V is required. Figure 5 shows connections between main parts.

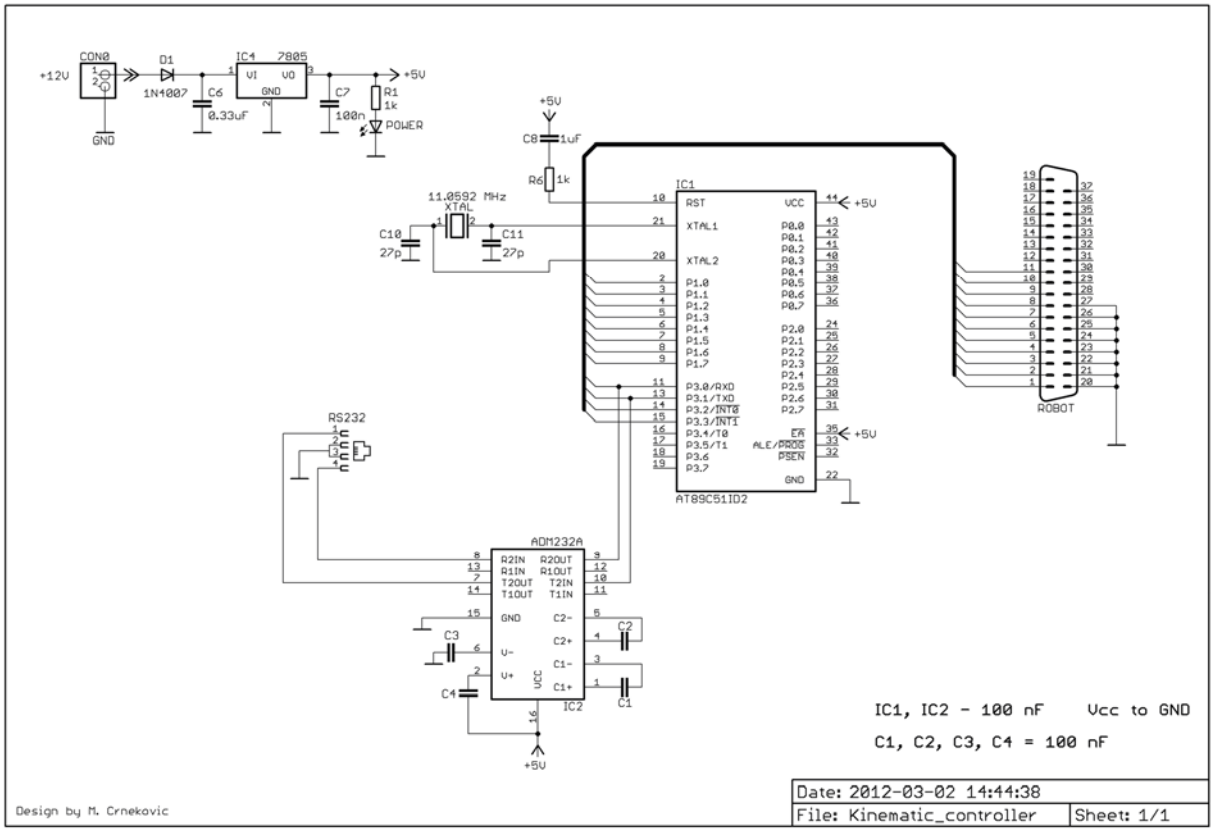

Fig. 5 Kinematic controller schematic diagram

The only hardware going towards the robot is the connection of the microcontroller to the robot Centronics connection [7]. The transmission of 8-bit data is synchronized with two additional signals, BUSY and STROBE. By reading them (Table 1), we can determine the state of connection between the robot and the kinematic controller.

Table 1 Reading the state of a robot

BUSY	STROBE	Robot state
0	0	Robot is connected and off
0	1	Robot is connected and on
1	0	Robot not connected
1	1	Robot not connected

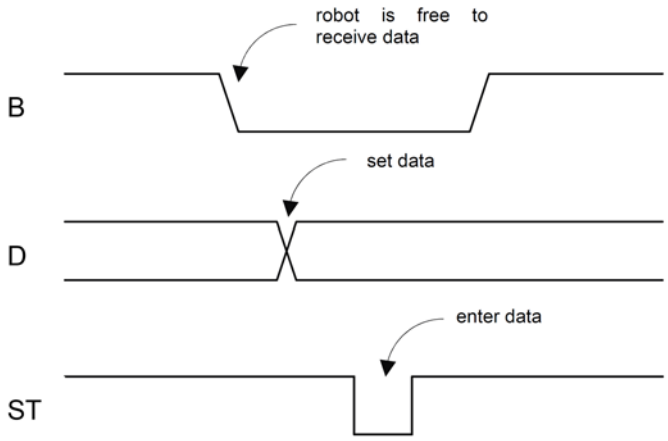

Fig. 6 Transmission of data and synchronization signals

The time graph of data transmission and synchronization signals are shown in Fig 6. The transmission is possible only if the STROBE signal is in high state, which means that the robot control unit is connected to the kinematic controller and switched on. The speed of transmission is determined by the robot with the feedback signal BUSY. When the BUSY is in high state, transmission is not possible. After transition of the BUSY signal to low state, 8-bit data can be set on data bus. The microcontroller then pushes the STROBE to low state, which is a signal to the robot control unit that data on the bus is valid and can be read. The duration of the STROBE signal is approximately 1 millisecond. After that, the whole protocol repeats for the next data transmission. It has been noticed that the robot has an internal receive buffer which gives the robot the ability to receive one command in advance.

6. New Instruction Set

The idea of kinematic controller has opened up many possibilities to modify the functions of the robot. The basic function of the kinematic controller was the ability to directly set external coordinates of the robot. Therefore, the whole kinematic model of the robot, with all physical limitations, is incorporated into the kinematic controller. This ability is very useful when a task-oriented robot language is used [8]. Also, it helps in robot path planning where no collision with obstacle occurs [9]. In order to reduce the possibility of errors in communication, the commands are set by ASCII strings. Each string has to end with 0x0D code (ASCII code for CR). Thus, commands can be set from a terminal program and in any program language. For every command, the kinematic controller answers a question or signals an error. Table 2 gives an overview of the commands implemented so far and brief explanations of them.

Table 2 Implemented commands of the kinematic controller

Command from PC	Kinematic controller answer	Command meaning
x127.3y-57.4z126.2f-52.6p0	Ex	Set the robot absolute position.
Dx10.5y52.7z20f0p10	Ex	Set the robot relative position.
SPEED=7	Ex	Set the robot speed.
OPEN	Ex	Open gripper.
CLOSE	Ex	Close gripper.
GPAR=7,7,1	Ex	Set grip parameters.
INIT	E0	Start the program for the robot initialization.
HOME	Ex	Go to HOME position.
LTOOL=25.7	Ex	Set the variable l_{TOOL}.
ZMIN=10.5	Ex	Set the variable z_{MIN}.

Command from PC	Kinematic controller answer	Command meaning
RESET	E0	Reset the robot error.
?R	#R=(127.3,-215.7,64.1,-32.9, 20.1)	Return vector **r**.
?Q	#Q=(27.7,-27.3,52.1,-40.9,0)	Return vector **q**.
?A	#A=(100,2731,-1800,547,73)	Return vector **a**.
?INIT	#INIT=YES	Return initialization status (Yes, No)..
?SPEED	#SPEED=5	Return speed status.
?GRIP	#GRIP=OPEN	Return gripper status (Open, Close).
?GPAR	#GPAR=i1,i2,i3	Return grip parameter status.
?LTOOL	#LTOOL=0	Return variable l_{TOOL}.
?ZMIN	#ZMIN=10.5	Return variable z_{MIN}.
?ROBOT	#ROBOT=ON	Return robot status (On, Off, Disconnected).
?SIM	#SIM=NO	Return simulation status (Yes, No).
SIM	E0	Communication simulation
NOSIM	E0	No communication simulation
Tx=ON or OFF or RESET	Ex	Timer x command (x=0 .. 5)
?Tx	#Tx=123, ON	Return value and status of the timer x .
ECHO	E0	Echo input
NOECHO	E0	No echo input
HELP	list of all commands	Return the list of all commands.
BASICMOD	OK	Enter a basic work mod

If we wish to send the robot to the point x=350.2, y=-27.5, z=113.4, φ=-90, ψ=0, the command will be: x350.2y-27.5z113.4F-90P0. If it is possible to move the robot to the set point, the kinematic controller will return the E0 message after the movement is made. If the robot, for whatever reason, cannot move to the set point, we will receive a message stating the reason for refusing to carry out the command, Table 3. For example, E6 means that the set point is out of reach of the robot.

Table 3 Meaning of error codes Ex

x	Meaning
0	No error
1	Parameter a1 out of range
2	Parameter a2 out of range
3	Parameter a3 out of range
4	Parameter a4 or a5 out of range
5	Input parameter out of range
6	Point is out of robot reach
7	Robot not initialize
8	Danger of collision with base
9	Error in command

Also, at any moment, it is possible to receive the information about the position of the robot from the kinematic controller in **r**, **q** and **a** coordinates.

The kinematic controller will also verify whether the robot is connected to the parallel port at all, and if so, whether it is on. The answer to the command ?ROBOT can be ON, OFF or DISCONNECTED. If the robot is not connected or is off, the simulation work mode of the kinematic controller is initiated.

One of the most important additional functions of the kinematic controller is the prevention of collision of the robot with the surface where the robot is mounted. That is because it is possible to define the lowest z coordinate of the robot, z_{MIN}, below which the robot will not go (the surface equals z=0). Even if the robot is holding an object (a tool), a collision with the surface will not occur as long as the length of the tool is set properly.

Timers enable the timing of each operation with accuracy of 0.1 seconds. They can be started, stopped, reset and scanned at any moment. Timer 0 is started when the kinematic controller is turned on and it times the overall time of the work.

Setting a relative position of the robot has to be avoided because it has been noticed that the error is accumulated since the numbers in the robot kinematic model have to be rounded.

In BASICMOD, the kinematic controller is used only as a converter from a serial to parallel communication. It brings the robot to the native work and is useful only when old programs are executed.

7. Conclusion

With low investment and good knowledge, it is possible to renew and improve old equipment. The addition of kinematic controller enabled a more-than-20-year old robot to be used once again in the educational process. The renewal itself can be an educational process if students are involved.

A kinematic controller has to do several basic tasks:

- transformation of RS232 communication into the parallel Centronics communication
- execution of direct and inverse kinematic model, while considering physical limitations of a robot
- control and prevention of robot collision with the surface
- measuring the time of robot movements and doing tasks
- additional features (gripper, tool, speed, simulation, …)

After having examined the renewed and improved system, we noted a possibility for additional functions, the most important of which is the ability of continuous movement of the robot in line, in circle and in arc. The step that would lead to these achievements is shorter than the one taken by now.

Also, the serial RS232 communication can be replaced with the USB communication (using an integrated circuit FT232RL). Thus, the power supply of kinematic controller would also be solved, so there would be one connector less on the box. The whole intervention on the robot would acquire a new dimension if the kinematic controller box was placed in the drive unit of the robot.

In further improvement, digital and analogue input and output signals could be added to the kinematic controller, along with a new programming language. In this way, the original drive unit of the robot would only be used for robot movement. Writing an appropriate software library in the form of DLL file would simplify the access to kinematic controller.

It is important to note that all these steps require only the knowledge that students get during their educational process anyway, and a robot and its kinematic controller are excellent objects on which that knowledge can be used. Therefore, we are planning to create a series of tasks for students that will go in that direction.

Hopefully, this work will encourage others to improve their old equipment (and not just robots) and to breathe life into it again. I will be happy to help them with my knowledge and experience.

REFERENCES

[1] Mitsubishi Electric, Move Master II, Instruction Manual, 1986

[2] Mitsubishi Electric, Move Master II, Programming Manual, 1986

[3] Crneković, M., A Contribution to Mitsubishi RM-501 Robot Programming, *Faculty of Mechanical Engineering Proceedings*, University of Zagreb, pp. 165-174, 1993

[4] Hitam, M.S., Design and Implementation of Fuzzy Control for Industrial Robot, *Chapter 15 of Industrial Robotics: Theory, Modeling and Control*, plV pro literatur Verlag Robert Mayer-Scholz, 2007

[5] Erlic, M., Jones, K., Lu, W.S, Hardware interface Configuration for Motion Control of the Puma-560 and the Mitsubishi RM-501 Robots, *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, May 9-10, 1991*

[6] Atmel data manual, AT89C51ID2.pdf

[7] Ogren, J., Help file for Centronics communication

[8] Crneković, M., Task Oriented Language for RM501 Robot, *Strojarstvo 37(1995) 1/2*, pp. 31-35.

[9] Crneković, M., Šitum, Ž., Collision Space Modeling for RM501 Robot, *9th International DAAAM Symposium*, Cluj-Napoca, Romania, pp. 137-138, 1998

Submitted: 20.1.2012

Accepted: 23.3.2012

Prof.dr.sc. Mladen Crneković
mladen.crnekovic@fsb.hr
Prof.dr.sc. Davor Zorc
davor.zorc@fsb.hr
University of Zagreb
Faculty of Mechanical Engineering and Naval Architecture
Ivana Lučića 5
10000 Zagreb, Croatia