Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 48 (2012) 533 – 542

Procedia Engineering

www.elsevier.com/locate/procedia

MMaMS 2012

Optimizing industry robot for maximum speed with high accuracy

Peter Papcun[a]*, Ján Jadlovský[a]

[a]*Department of cybernetics and artificial intelligence, FEEI TU in Košice, Košice 042 00, Slovak Republic*

Abstract

This article describes design of algorithm for finding maximum possible speed. Maximum speed of robot's endpoint is 4.4 meters per second, but this speed is achieved rarely by the robot. Robot name is MELFA RV-2DB. Robot has integrated development environment with name RT-Toolbox2. Robot maximum speed is under 1 meter per second in motions, where the trajectory is important. Algorithm analyzes and designs speed optimizing motions on high accuracy motions. Circular motions and motions on straight line require high accuracy of trajectory. In these types of motion trajectory is important. I also describe results and analyses of proposed algorithm in this paper.

Keywords: Industry robot, Mitsubishi, Optimizing, Joint, Robot movement, Accuracy, MELFA RV-2DB

Nomenclature

X, Y, Z	main axes in Cartesian system
XY, YZ	planes in Cartesian system
XZ	plane in Cartesian system
m/s	velocity (speed) unit meters per second (others units: cm/s, mm/s)
s	time unit seconds (other units: ms)
R	radius
B	byte (other units: kB, MB)
Subscripts	
P	required coordinate or main point coordinate
N	directional vector (example: X_N is first directional vector coordinate of line)
C	center of circle

1. Introduction

This paper inscribes monitoring the movement accuracy and optimizing for maximum speed with high accuracy of industry robot. Robot name is MELFA RV-2DB. This article is divided to seven chapters. The first chapter is naturally introduction. The second chapter designs method of data collection from industry robot. This design is realized on simple example. The third part describes data collections from simulation mode and real system. Data will collect from eight types

* Corresponding author. Tel.:+421-55-602-4218;
E-mail address: peter.papcun@tuke.sk.

1877-7058 © 2012 Published by Elsevier Ltd.Selection and/or peer-review under responsibility of the Branch Office of Slovak Metallurgical Society at Faculty of Metallurgy and Faculty of Mechanical Engineering, Technical University of Košice
doi:10.1016/j.proeng.2012.09.550

of movement at minimal ten different speeds. Data will be from 100 experiments from every type of movement at any speed. These are 8 000 measurements in simulation mode and 8 000 measurements in real system (real system is industrial robot Mitsubishi). The fourth part analyzes movement accuracy of robot and describes results of analyzes. The fifth chapter analyzes robot speed. The sixth chapter proposes the continuation of analysis and optimizations in the next continuation of research. Conclusion rates every measurements and analysis. Figure of industrial robot:

Fig. 1. Industrial robot: Mitsubishi RV-2SDB.

2. Design of data collection

Robot has multitasking capabilities, through threads programming. These threads are programs transferred in robot controller, which can call by other program, when some robot parameters are set. The maximum number of threads is eight. In additional, robot controller has integrated several timers to use in programs.

We make one thread. This thread captures robot positions into file every working loop of this thread. These positions are in Cartesian coordinates (X, Y, Z). Algorithm design of base and simple main program for one measure in points:

1) open file for write data,
2) move robot to the starting position,
3) set the desired speed,
4) run thread,
5) reset timer,
6) write string to file: “Begin at [timer value]”,
7) perform the movement for analysis,
8) write string to file: “End at [timer value]”,
9) stop thread,
10) close file.

These algorithms (main program and thread) are programmed and tried in simulation mode on linear robot motion. Here is text file or output from these algorithms:

```
...
(+270.07,+0.00,+504.65,+180.00,-0.07,+180.00)(7,0)
Begin at +42.6667
(+270.07,+0.00,+504.65,+180.00,-0.07,+180.00)(7,0)
(+270.07,+4.39,+504.65,+180.00,-0.07,+180.00)(7,0)
...
(+270.07,+192.56,+504.65,+180.00,-0.07,+180.00)(7,0)
(+270.07,+198.67,+504.65,+180.00,-0.07,+180.00)(7,0)
End at +469.333
...
```

On this output you can see, that we can check and analyze robot speed, not only accuracy. Main program make one instruction before measurement and one instruction after measurement. This instruction is writing marks of begins and ends to file. Thread is writing robot’s endpoint coordinates to file. Through timer was measured maximum time of duration

mentioned instruction for marking begins and ends. This time is 21.333 s. Then was measured the length of timer tact. Timer tact was identified so, that program writes timer time to file every working loop. File output is follows: 0, 0, 0, 0, 7.111, 7.111, 7.111, 7.111, 7.111, 14.222, etc. So one timer tact is 7.111 ms, that is around 140.5 Hz. This is means, that maximum mistake is 7.111 ms of time counting and time shift is 21.333 ms, this is 28.444 ms altogether.

In the case of multiple measurements in one main program is to text “Begin at [time]” adds type of movement and number of the experiment. Exercise of line “Begin” for multiple measurements: “Begin [type] [number] [speed] [time] [distance]”. Types of movement explain the next chapter.

3. Data collection

Algorithm, designed in the second chapter, is now extended. Now is possible to run a program to make several measurements with extended algorithm. Flowchart can be seen on figure 2:

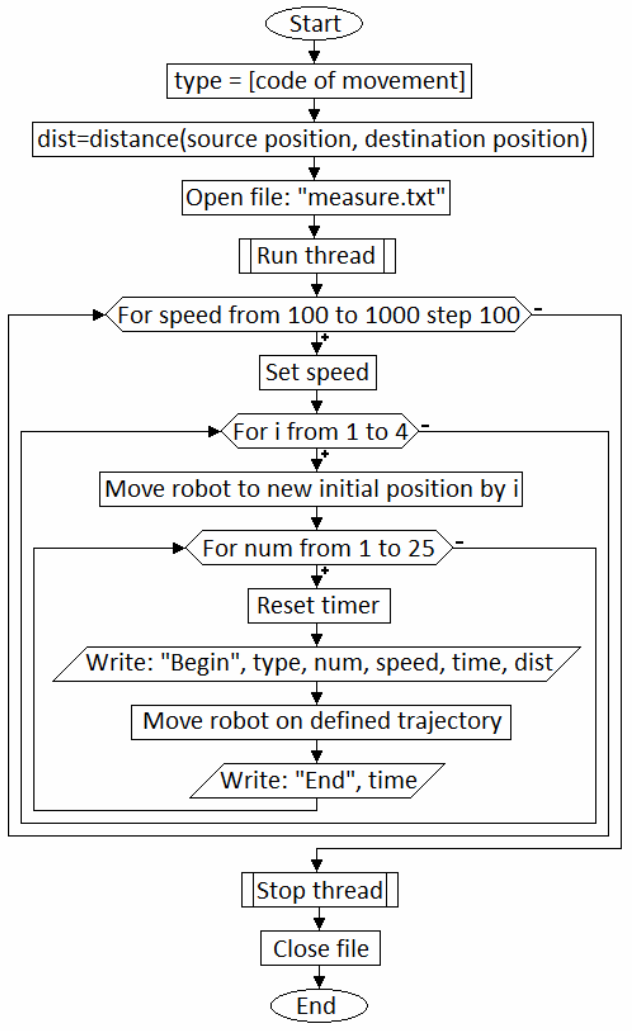

Fig. 2. Flowchart for data collection.

This flowchart is programmed in an integrated development environment RT-ToolBox 2 in programming language MELFA V. This integrated development environment has been supplied with industry robot Mitsubishi.

This algorithm performs 1 000 measurements per about 30 min. These trajectories were tested:

- linear movement in X-axis direction (coordinates Y and Z do not change during the motion) – movement sign: SX
- linear movement in Y-axis direction (coordinates X and Z do not change during the motion) – movement sign: SY
- linear movement in Z-axis direction (coordinates X and Y do not change during the motion) – movement sign: SZ

- linear movement in which all coordinates (X, Y, Z) change – movement sign: SA
- circular movement in plane which is parallel with plane XY – movement sign: CZ
- circular movement in plane which is parallel with plane XZ – movement sign: CY
- circular movement in plane which is parallel with plane YZ – movement sign: CX
- circular movement in the whole space (all coordinates is changing during motion) – movement sign: CA

One measure generates more then 10 MB of data. All measured data have more then 80 MB from simulation mode. Then this program run in robot controller and measured data had more then 100 MB. Measurements were carried out for several days. These data were necessary to handle. Due to their size was appropriate to design a program for data analysis.

4. Design and realization application for analyze robot accuracy

Data is quite a lot, it was well designed tool for the analysis. Flowchart of the analyzer accuracy can be seen on figure 3.

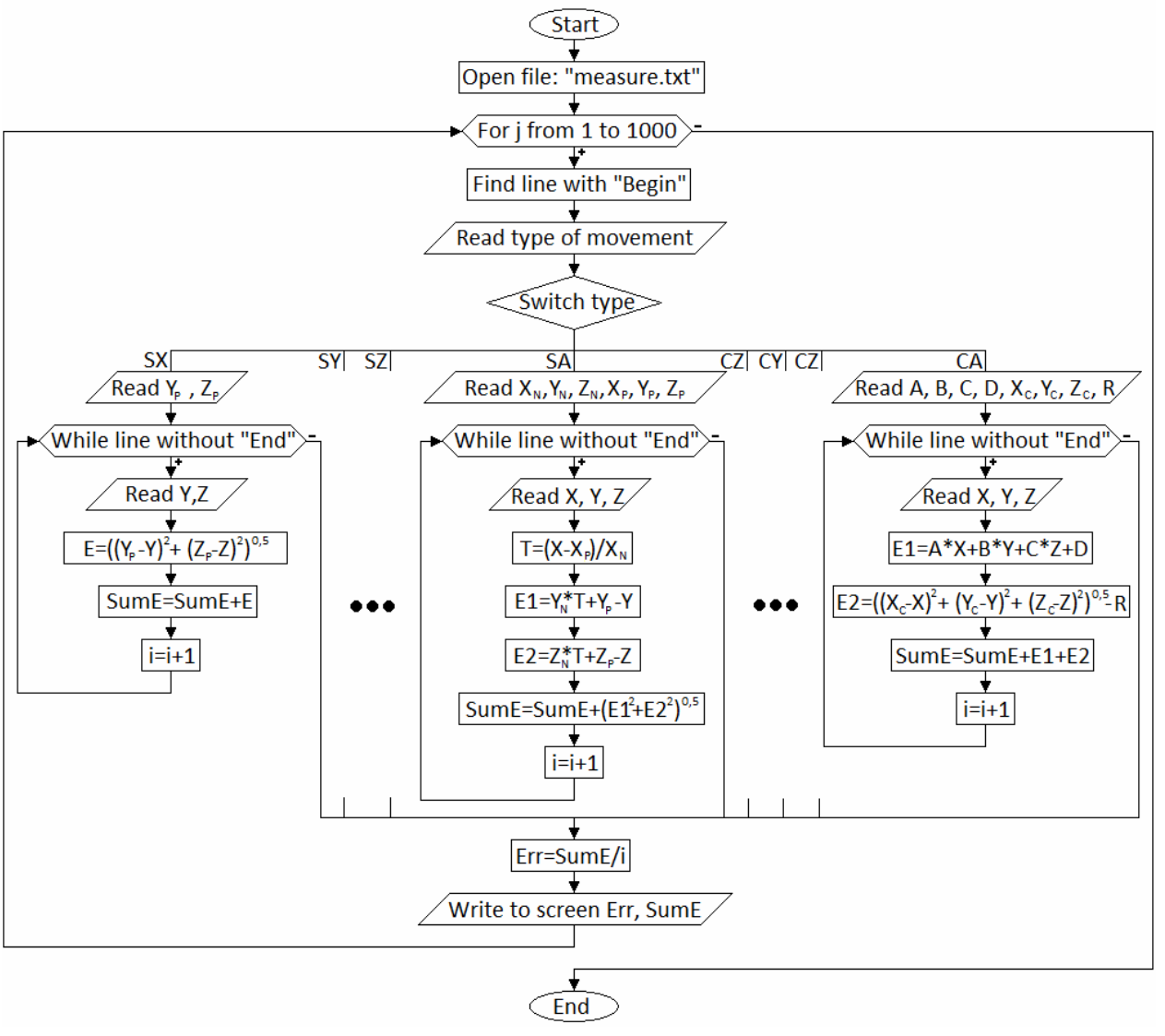

Fig. 3. Flowchart for accuracy analyzing

Program found line, which begins with word “Begin”. Program read type of movement from this line. Movement signs: SX, SY, SZ, SA, CZ, CY, CX, CA. These signs explain previous chapter. Then begin analysis of data, according to the type of movement. Program read necessary information for analysis before analysis start from keyboard or other file:

- For analysis movement SX are necessary the coordinates constants, during the movement would not change the coordinates Y a Z (Y_P, Z_P).
- For analysis movement SY are necessary the coordinates constants, during the movement would not change the coordinates X a Z (X_P, Z_P).
- For analysis movement SZ are necessary the coordinates constants, during the movement would not change the coordinates X a Y (X_P, Y_P).
- For analysis movement SA are necessary the line vector coordinates (X_N, Y_N, Z_N) and coordinates any point on this line (X_P, Y_P, Z_P).
- For analysis movements CX, CY a CZ are necessary coordinates of circle center (X_C, Y_C, Z_C) and radius R.
- For analysis movement CA are necessary parameters A, B, C, D, which representing the plane equation parameters ($0 = Ax + By + Cz + D$) where is the circle, and then program needs coordinates of circle center (X_C, Y_C, Z_C) and radius R.

Some parts of flowchart are omitted, because they are alike then shown parts. The analysis of movements SX, SY, and SZ are obvious, because point coordinates can change only one coordinate and the other coordinates have to be constant. This is reason, why program checks only these parameters, which have to be constant. With this checking program check that robot does not deviate from trajectory. Analyzer checks movement SA through parametric equations of line:

$$X=X_P+X_N*T$$
$$Y=Y_P+Y_N*T$$
$$Z=Z_P+Z_N*T$$
T is real number

Through these equations, analyzer determines how many cents of millimeters robot deviate from trajectory. Coordinate X is not changing during CX movement and other coordinates are changing during movement according to rule. This rule is: distance between analyzed points and circle centre must be still same and this distance must be circle radius (R). It is similar with the movements CY and CZ, but there are other constant coordinates. The movement CA is also similar, there is not checking constant coordinate, because all coordinates are changing in this type of movement, here are checking only two things: distance between points and circle center, and points membership (if points are points of plane).

Variables E, E1, E2 are errors in millimeters. Variable SumE is sum of these errors and variable Err is average error.

The analysis ends when program comes to file line that begins with “End”. Program writes values Err and SumE on screen. Then program returns to step where was looking the word “Begin” and performed a new analysis. When screen is full with values Err and SumE, then program waits for pressing the Enter key.

The analyzer was programmed in programming language C. The measured data have a precision of one cent of millimeter. Therefore, the variables E, E1 and E2 are round to two decimal places (cent of millimeter). All files with data were analyzed with programmed analyzer. The errors were zero in any case, even SumE variables were zero. Then we change formulas for error calculating in analysis SX, SY and SZ from flowchart:

from

$$E=((Y_P-Y)^2+(Z_P-Z)^2)^{0,5};\quad E=((X_P-X)^2+(Z_P-Z)^2)^{0,5};\quad E=((X_P-X)^2+(Y_P-Y)^2)^{0,5}$$

to

$$E=Y_P-Y+Z_P-Z;\quad E=X_P-X+Z_P-Z;\quad E=X_P-X+Y_P-Y$$

Also we create new variable SSumE, this variable sum all variables sumE. Even after these changes came all errors to zero, also the variable SSumE is zero in any examination of file. So the robot has a full accuracy (100%) in moving with accuracy one cent of millimeter. When we know that the robot is completely accurate, than we have to checked speed, if it still moves at the desired speed. This problem devotes the next chapter.

5. Design and realization application for analyze robot speed with high accuracy

Since the robot is completely accurate, the information about the positions of the robot are not necessary. So we can program a simple filter that removes any line beginning with bracket "(". Alternatively, we have not to lose information, filter creates a new file to copy all lines starting with letter "B" or "E". Now the file has less than 80 kB, before filtering file has more than 10 MB. But information is still a lot, so in this case is a necessary programming a tool for the analysis of this data, too. Flowchart of the speed analyzer can be seen on figure 4.

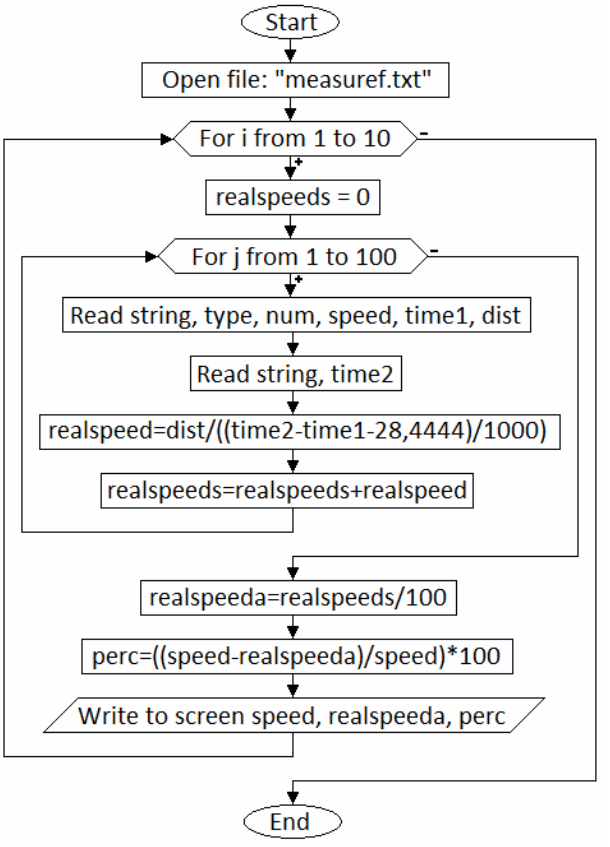

Fig. 4. Flowchart for speed analyzing

File measuref.txt is file after filtering. Value of 28.444 is time that is described in the second chapter. This value can be seen in the flowchart. Explanation of variables:

i, j	– auxiliary variables
time1	– in this time the measured movement began
time2	– in this time the measured movement ended
type	– type of movement
num	– sequence of measurements (variable in not needed in algorithm)
speed	– set speed of movement
dist	– distance that must be overcome by robotic arm
string	– auxiliary variable for string reading
realspeed	– calculated speed (real speed)
realspeeds	– sum of calculated speed
realspeeda	– average calculated speed
perc	– percentage error between set speed and real speed (calculated speed)

The algorithm works quite simply. We know that in file is 1 000 measurements, 100 measurements at 10 different speeds. The program reads type, number and speed of movement. Then program reads time when measured movement began and ended. At last program reads distance which has to be overcome by robot. Then program with simple formula calculates the speed of movement. Program calculates average speed from calculated speeds (real speeds) which have set the same speeds in robot controller.

This analyzer was programmed in programming language C. Measurement results can be seen in the following tables:

Table 1. Average speed of straight movement

Speed	Max. Speed	Min. Speed	SX Speed	Error	SY Speed	Error	SZ Speed	Error	SA Speed	Error
[mm/s]	[mm/s]	[mm/s]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]
100	100.24	99.76	100.03	0.03	99.99	0.01	100.11	0.11	100.13	0.13
200	200.95	199.06	199.98	0.01	199.86	0.07	199.89	0.06	199.81	0.10
300	302.15	297.88	299.91	0.03	299.76	0.08	299.73	0.09	299.68	0.11
400	403.83	396.24	399.51	0.12	399.45	0.14	399.28	0.18	399.41	0.15
500	506.00	494.14	498.67	0.27	497.72	0.46	498.91	0.22	500.98	0.20
600	608.66	591.59	598.34	0.28	582.45	2.92	580.45	3.26	598.94	0.18
700	711.81	688.57	641	8.43	628.65	10.19	642.05	8.28	692.42	1.08
800	815.46	785.11	708.94	11.38	633.21	20.85	705.45	11.82	788.63	1.42
900	919.62	881.20	749.85	16.68	636.45	29.28	792.46	11.95	881.56	2.05
1000	1024.28	976.85	771.56	22.84	639.54	36.05	832.18	16.78	882.45	11.76

Table 2. Average speed of circle movement

Speed	Max. Speed	Min. Speed	CZ Speed	Error	CY Speed	Error	CX Speed	Error	CA Speed	Error
[mm/s]	[mm/s]	[mm/s]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]
100	100.15	99.85	99.73	0.27	99.82	0.18	99.84	0.16	99.92	0.08
200	200.60	199.40	196.47	1.77	195.26	2.37	195.28	2.36	197.31	1.35
300	301.36	298.65	281.97	6.01	282.42	5.86	283.16	5.61	287.53	4.16
400	402.43	397.60	353.39	11.65	355.65	11.09	358.15	10.46	369.23	7.69
500	503.80	496.26	431.62	13.68	417.23	16.55	418.35	16.33	441.54	11.69
600	605.47	594.62	475.2	20.80	460.71	23.22	461.91	23.02	484.14	19.31
700	707.46	692.69	511.95	26.86	495.32	29.24	505.45	27.79	512.19	26.83
800	809.76	790.47	545.52	31.81	516.84	35.40	533.57	33.30	545.63	31.80
900	912.37	887.96	583.05	35.22	522.78	41.91	547.32	39.19	581.26	35.42
1000	1015.30	985.16	606.47	39.35	529.5	47.05	554.21	44.58	599.51	40.05

Columns Max. Speed and Min. Speed express speed with error of one timer tact. These speeds is calculated with average trajectory distance, linear motion has average distance 300 mm and circular motion has average distance 471.24 mm. Of course speed Max. Speed is calculated from measured time which is lower by one timer tact and speed Min. Speed is calculated from measured time which is higher by one timer tact. I remind to you, that the tact of timer is 7.111 ms.

Tables with results from the analysis in simulation mode are not written, because they are very similar as the results from the real system. The maximum difference between the errors is only 0.5%.

Through this analysis, we can read when the desired speeds are equal with the real speeds. I take new term **equal speeds** for these speeds. In a linear movement is the highest equal speed somewhere around 600 mm/s and in a circular movement is the highest equal speed somewhere around 300 mm/s. Of course, the maximum speed in both cases is higher. But the

mentioned speeds we can be almost certain that the endpoint of the robot will have just such speed. In other analysis is analyzed the maximum speed by type of movement. The best movement is SA where the highest equal speed is 900 mm/s. Entire analysis can be seen in the first and the second table.

Analysis may not only do for type of movement, but also for a particular movement. Particular movement is measured 25-times. Times are measured as in previous cases. Program calculates arithmetic average of real speed where is required speed same:

Table 3. Average speed of four types of movement

Speed	CZ - 1		CZ – 2		SA – 1		SA - 2	
	Speed	Error	Speed	Error	Speed	Error	Speed	Error
[mm/s]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]	[mm/s]	[%]
100	99.65	0.35	99.75	0.25	100.14	0.14	99.98	0.02
200	199.25	0.38	199.37	0.31	200.5	0.25	199.78	0.11
300	298.59	0.47	298.63	0.46	301.69	0.56	299.49	0.17
400	350.54	12.37	398.45	0.39	401.55	0.39	399.74	0.06
500	378.45	24.31	485.64	2.87	500.56	0.11	497.2	0.56
600	386.95	35.51	568.42	5.26	600.27	0.05	597.78	0.37
700	387.65	44.62	649.23	7.25	690.35	1.38	691.82	1.17
800	388.12	51.49	702.54	12.18	781.97	2.25	787.95	1.51
900	388.12	56.88	725.12	19.43	851.48	5.39	879.46	2.28
1000	388.06	61.19	745.65	25.44	866.57	13.34	915.12	8.49

Maximum speed of movement CZ – 1 can be read from this table. This speed is 388 mm/s, but robot has this speed only if you set software speed higher than 700 mm/s. In the same motion, the equal speeds are in the range from 0 to 300 mm/s. Movement CZ – 2 has maximum speed greater than 745 mm/s, additional measures measure the maximum speed to 755 mm/s. This motion has equal speeds in the range from 0 to 400 mm/s. The maximum speed of motion SA – 1 is 867 mm/s and the maximum speed of motion SA – 2 is 980 mm/s. The equal speeds are same in both cases (SA – 1 and SA – 2). Equal speeds are in range from 0 to 800 mm/s.

6. Proposes the continuation of analysis and optimizations

Since in this paper we have not enough space, we outline the continuation of our analysis and optimization. Another procedure would be a comparison of maximum speed above-mentioned movements with programmed movements with high accuracy by loop. This meant, that is not use commands to set the speed (SPD) and for the precise movement of the trajectories (MVS and MVC), that has been use so far. We will use the command “mov P”, which is command for the most effective move to a given point, this command does not take a care for accuracy. Therefore, we will program this command to be as accurate as possible and compare results with results in this article. We will try to compare measure from this paper with command „mov J“, too. This command directly controls the rotation of the robot’s joints.

Next step would be to program an application. Application would do own analysis and suggest the optimized speed for the given movement. Program will be designed for several types of analysis:

- Slow analysis – similar as in the article, only the analysis would be on-line
- Accelerated analysis – analyze only some speeds that would be enough for a full analysis, which would include a maximum speed and maximum equal speed
- Fast analysis – attempt to maximal speed up analysis

This application should monitored and drew at all time the actual position of robot's endpoint. Design of this application can be seen on figure 5. Currently, work is just beginning on this application.

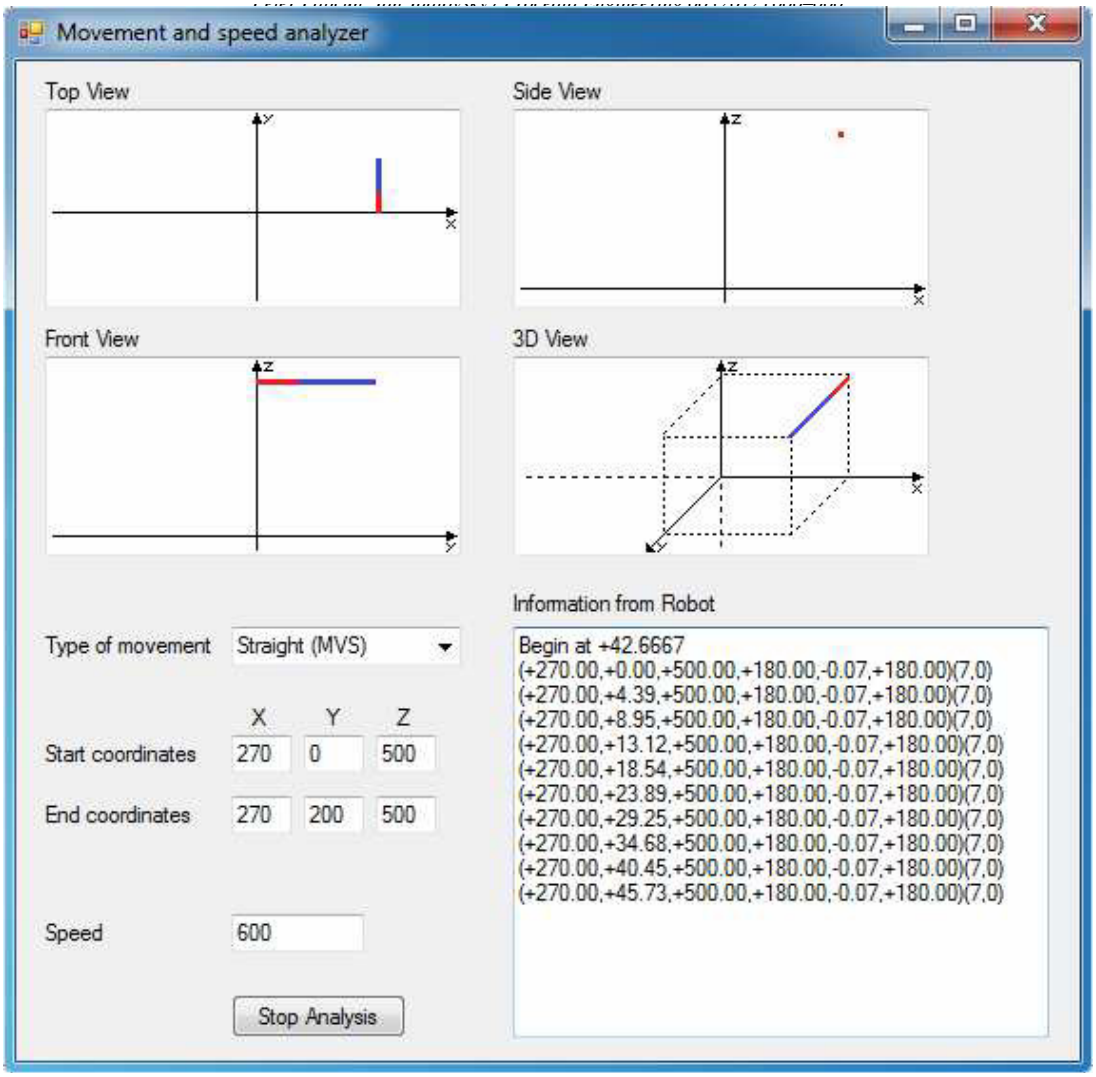

Fig. 5. Design of application for analyzing

7. Conclusion

In this paper, we dealt with the analysis of robot accuracy and robot speed analysis with high accuracy. After analyzing the robot speeds, we test these speeds on industrial robot, and we reached the optimization of the robot with this test. If the robot controller ordered move on selected trajectories, the robot controller will be controlled move so that move will has high accurate at the expense of the set speed. So the robot will go rather slower than it should break accuracy of trajectory.

The analysis found that the robot has one cent of millimeter accuracy. Maybe robot has higher accuracy, but we could not measure better accuracy, because outputs in file have only two decimal places. Robot had 100% accuracy in 16 000 measurements. Analyzer checked more than 1 500 000 values, every of these values do not be outside of their range, outside of selected trajectory.

Analysis of speed was not so clear. We were limited in analysis of accuracy, because values have precision only one cent of millimeter. We were limited in analysis of speed, too. Because one tact of timer takes longer then 7 ms. Analysis was set to take aspect of mentioned fact and to be known what time takes movement in mistake of one timer tact. The First two tables have columns min and max, values in these columns are values with mistake of one timer tact. We found in this analysis that the robot will move rather slowly, as it has set speed in program, because it does not want infringed its accuracy. When robot is moving in a linear movement, the maximum speed is around 1 000 mm/s, but a maximum design speed is 4 400 mm/s. When robot is moving in circular movement, the maximum speed is decreases to around 600 mm/s. Maximum equal speed is 600 mm/s in linear motion and 300 mm/s in circular motion. But with each new trajectories are

valid something else, end of the fifth chapter analyzing four specific trajectories. From this analysis it is clear that each trajectory can be analyzed separately.

The sixth chapter demonstrates how we would like to continue in this research. Chapter describes two points: compare these movements with other alternative programming of robot controller and program application for on-line analysis.

Fig. 6. Industrial robot in our laboratory

Acknowledgements

This work is the result of the project implementation:

Development of the Center of Information and Communication Technologies for Knowledge Systems (ITMS project code: 26220120030) supported by the Research & Development Operational Program funded by the ERDF.

This work has been supported by the Scientific Grant Agency of Slovak Republic under project Vega No.1/0286/11 Dynamic Hybrid Architectures of the Multiagent Network Control Systems.

References

[1] Product leaflets, RV-2SDB, Mitsubishi Electric, Ratingen, Germany, 2010.
[2] Instruction manual, CRnQ/CRnD Controller, Mitsunishi Electric, Ratingen, Germany, 2010.
[3] Papcun, P., 2011. Control of robot integrated in flexible production line. Diploma thesis, Košice, Slovakia, 2011.
[4] Papcun, P., Čopík, M., 2012. Remote control of Mitsubishi industrial robot - 2012. - In: SCYR 2012: proceedings from conference: 12th Scientific Conference of Young Researchers: May 15th, 2012, Herľany, Slovakia. Košice: TU, 2012 S. 212 - 215. ISBN 978-80-553-0943-9.
[5] Papcun, P., Čopík, M., Ilkovič, J., 2012. Riadenie robota integrovaného v pružnom výrobnom systéme - 2012. In: ElectroScope. Vol. 2012, no. 2 (2012), p. 1-9. - ISSN 1802-4564.