Journal of Achievements in Materials and Manufacturing Engineering
VOLUME 47
ISSUE 1
July
2011

Automatic programming and generation of collision-free paths for the Mitsubishi Movemaster RV-M1 robot

K. Foit*, G.G. Kost, D. Reclik
Institute Of Engineering Processes Automation And Integrated Manufacturing Systems, Silesian University of Technology, ul. Konarskiego 18a, 44-100 Gliwice, Poland
* Corresponding author: E-mail address: krzysztof.foit@polsl.pl

Received 03.05.2011; published in revised form 01.07.2011

Manufacturing and processing

ABSTRACT

Purpose: of this paper: This paper discuss the possibility to develop and implementing the computer system, which could be able to generate a collision-free path and prepare the data for direct implementing in the robot's program.
Design/methodology/approach: The existing methods of planning of the collision-free paths are mainly limited to the 2D issue and implemented for the mobile robots. The existing methods for planning the trajectory in 3D are often complicated and time-consuming, so most of them are not introduced in reality, being only a theory. In the paper the 2½D method has been presented together with the method of smoothing the generated trajectory. Experiments have been carried out in the virtual environment as well as on the real robot.
Findings: The developed PLANER application has been adapted for cooperation with the Mitsubishi Movemaster RV-M1 robot. The current tests, together with the previous one carried out on the Fanuc RJ3iB robot, have shown the versatility of the method and the possibility to adapt it for cooperation with any robotic system.
Research limitations/implications: The further stage of research will be concentrated on the consolidation of trajectory generating and simulation phase with the program execution stage in such a way, that the determination of collision-free path could be realized in real time.
Practical implications: This approach clearly simplifies the stage of defining the relevant points of the trajectory in order to avoid collisions with the technological objects located in the robot's manipulator environment. Thereby it significantly reduces the time needed for implementation of the program to the production cycle.
Originality/value: The method of generating the collision-free trajectories, which is described in the paper, combines some of the existing tools with the new approach to achieve the optimal performance of the algorithm.
Keywords: Robotics; Collision-free paths; Automatic programming

Reference to this paper should be given in the following way:
K. Foit, G.G. Kost, D. Reclik, Automatic programming and generation of collision-free paths for the Mitsubishi Movemaster RV-M1 robot, Journal of Achievements in Materials and Manufacturing Engineering 47/1 (2011) 57-65.

1. Introduction

Industrial robotics is one of the most dynamically developing branches of robotics and robotic technology, and range of its applications is becoming broader and more common.

One of the most important aims of robotics, developed and targeted for industrial applications, is the scheduling of robots tasks. This large issue contains of many research problems, but one of the most important tasks is the collision-free motion planning of robotic manipulator, particularly in the context of the significant development of the off-line programming systems of industrial robots. These systems are increasingly used in the tasks, which are carried out in the field of technological preparation of production, allowing faster and more efficient programming of robots. This issue is important because existing off-line programming systems are not equipped with motion planning algorithms. The process of the off-line programming of robots, as well as its qualitative assessment, is not well described by the algorithms and its optimization is hardly possible, because it is rely exclusively on the subjective evaluation done by the robot programmer. For these reasons, is particularly important to develop collision-free trajectory planning methods for industrial robots. This task is multi-stage and requires solution of many problems, where the one of the most important is to simplify the computational complexity of this process, since the existing methods - taking into account the importance of this criterion - have purely theoretical meaning. Therefore, searching for high-speed, collision-free trajectory planning methods for industrial robots, taking into consideration the qualitative assessment of obtained solutions in terms of optimality, have the great practical importance [1]. This paper is dedicated to the mentioned issue.

2. Overall description of the developed method

In the group of application, which aid the programmer in his work of programming manufacturing machines, off-line programming of industrial robots is the least advanced, with a high degree of development of computer graphics [2-5]. The domain of issues, which concern the path planning, is particularly poor developed in these systems, including the designation of trajectory along which the manipulator should pass, avoiding collisions with other technological objects. There is no tool for effective management the programming work by automatically generating the high quality paths (understood as a set of consecutive positions between the initial and final position of the robot characteristic point - TCP), which makes the programming process harder and more time-consuming, because the programmer must check the possibility of a collision in each case. The introduction of the software tool which allows to automatically generating the trajectory, avoiding any other technological equipment would help to speed up the programming of the robot, as well as could improve the quality of the generated program.

The performed analysis of the available literature resulted in a fact, that many research centres have been attempted to develop effective methods for planning the mobile and industrial robot motion. According to [2-9], the task of the robots' trajectory planning (in short: the task of planning) is defined as the process of determining of the consecutive stages associated with defining of a parameterized function of time, which describes the manipulator's transitions from the initial position, through the intermediate to the final position specified in the configuration space of the robot

From the cited definition of the task of planning could be concluded that the problem of robot path planning is reduced to designate a mathematical equation defining the shape of the so-called template path. According to the [2-5] the whole process should be divided into the following sub-tasks that constitute the complete planning process:

- analysis of the space which consists of identifying barriers and isolation of collision-free areas, which may contain potential path,
- determining of the possible solutions on the basis of the assumed strategy for action (eg, implementation of the movement: straight to the point), specific positions (the initial and final) - so the potential collection of intermediate positions which allow to provide the movement in the collision-free manner,
- define of the optimization criterion for, which allows the selection of suboptimal solution, that best meets this criterion.

An additional function, described in the available literature [6,7,9,10] is an interpolation of the resulting motion path (or set of consecutive positions of the robot during the committing of the task) using a high order parametric curve.

2.1. The spatial methods for robots motion planning

Preserving the order of formation of robots' motion planning methods, it was decided to first undertake an analysis of the available methods for spatial traffic planning. According to the literature [2-5], there are two basic methods of spatial motion planning of manipulators. The first of these methods is based on discrete spatial representation of a robot's environment, called a raster spatial method. The second method involves the artificial introduction of potential interactions between the links of the manipulator's kinematic chain and technological environment, also called "the method of artificial potentials".

The spatial raster methods are based on a discrete raster representation of space around the robot. For this purpose whole space is divided into cubic or rectangular subspaces (raster). All of the subspaces are analyzed for the presence of technological objects. In case of dividing the scene using a dynamically allocated size of the raster (called OctTree), the octal division is used - hence the name: octants. This method assumes that after the first division (the division at the base), next phases of partitioning generate from any partly occupied octant the eight new areas. The idea of the division of partially occupied octants is shown in Fig. 1

The main disadvantage of raster methods in case of the analysis of three-dimensional space is very high computational complexity of determination of the safe path of movement. The methods of global optimization require the possibility of storing

in memory the entire image space representation of discrete scenes of the robot, which in turn causes a very significant limitations connected with the amount of free octants (raster) which may be subjected to a process optimization. This aspect combined with very high computational complexity makes the spatial raster method difficult for practical use in industrial robots, motion planning.

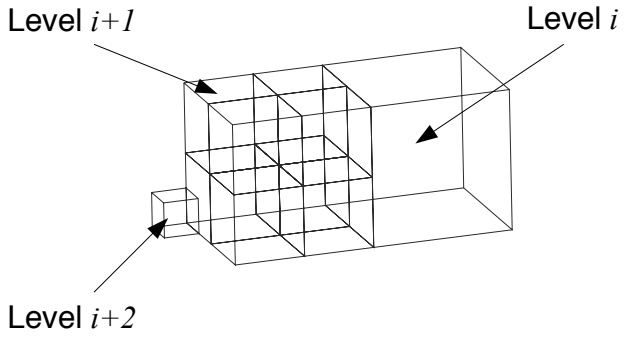


Fig. 1. The idea of partitioning of the partially occupied octants

The second method of spatial robot motion planning [2-5] is based on the influence of a repulsive force of potential fields associated with subsequent links of the kinematic chain of the robot. In order to allow the implementation of computation of the potential repulsive effect it is necessary to replace the complex shapes of individual links of the kinematic chain of the robot with the cubical solids. Any technological object located in the analyzed space, produces a repulsive potential described by the general Coulomb equation (1).

$$U_{rep}(q) = \begin{cases} \frac{1}{2} * \mu * \left(\frac{1}{\rho(q)} - \frac{1}{\alpha} \right)^2 \Leftrightarrow \rho(q) \le \alpha \\ 0 \Leftrightarrow \rho(q) > \alpha \end{cases} \quad (1)$$

where:

q - position of investigated point of the robot scene,

$\rho(q)$ - the distance from the object, which generates the potential field,

α - the factor of the influence range of the given object,

μ - the scale factor that matches the value of the influence for the size of analysed space.

During moving the robot between obstacles in case when any link of the kinematic chain will be too close to any other objects, the repulsion force is applied to the centre of the mass of the given link, resulting in automatic correction of the path. As a result, this leads to the preservation of a safe distance of each link from the other objects.

Despite the fact that arithmetic operations are not time-consuming, which virtually allows to carry out the planning in real time, the method of artificial potential has not found practical application in the programming of robot motion. The main reason is, according to [2-5], a significant difficulty to find the solution of complex equations, which define the state of the load during movement of the kinematic chain. Another major drawback of this method is the possibility of encountering a local minimum, which corresponds to the achievement of balance between a repulsive force of the scene objects and the attracting force of the final position. This lead up to the “endless loop” and the process of path planning will never be completed.

Analyzing the available literature it can be concluded that the existing methods for planning collision-free motion paths of industrial robots, have their origins in the 2D planning methods, so-called flat methods (mostly quad-tree method, as the basis for the octal method), which are computationally very efficient and allow the on-line robot motion planning. However, the use of flat methods to realize the spatial motion planning of industrial robots, reveals significant disadvantages resulting from the construction of these methods and their purpose - the necessity of "spatialization" by introducing extra dimensions of space (3D) and the positioning the robot in this space. The “spatialization” by simply introducing a third dimension, as it is done in the case of the octal method, significantly impairs the computational efficiency. Therefore, it is advisable to seek for solutions that will allow adapting the efficient, flat planning, computational methods to the spatial planning of collision-free motion of manipulators.

Implementation of such defined goal requires the development of the range of tasks relating to:

- to work out a base method of planning of the collision-free motion based on one of the methods of planning the safe movement of robots on the plane; the method should allow rapid determination of the collision-free path in the R^3 space,
- designation of a set of possible solutions of the robot trajectory, based on the resulting collision-free path, in the form of equation in the time domain, based on spatial curvilinear interpolation with the use of parametric equations of C^2 spatial curves,
- development of optimization procedures for obtained solutions,
- implementation of this method to the off-line programming system for industrial robots, which allows to perform its verification.

2.2. Simplifying the robot workspace analysis using the 2½D algorithm

Application of the robot motion planning methods at the level of spatial tasks has important disadvantages. The method based on the 2D view from the top, because of a simplified analysis of the scene of the robot, cannot guarantee the optimal movements paths. The main limitation of these methods is that there is impossible to carry the path over the low obstacles, which arises from the fact, that the trajectory is determined on the plane, so there is no connection between the height of the obstacle and the elaborated path.

This limitation is the main reason for which the possibility of designating a globally optimal solution is not possible In order to achieve the globally optimal path of motion is necessary to set the parametric description of the spatial image of the scene, which contains all completely empty octants. However, the attempt to implement such a defined field of possible solutions, did not

allow determining the optimal motion path. The main reason for this failure was high complexity of the defining process of the optimal spatial motion path. The computational complexity of the planning process stemming from a significantly greater number of the octants, which requires the occupancy checking in comparison to the base method [2-5], which allows to quickly find a solution, thanks to the preliminary motion planning algorithm, which - combined with the stochastic algorithm that implements the appropriate strategies - allowed to find the suboptimal solution (i.e. optimal for a view from the top).

To determine the optimal solution, while reducing the computational complexity of the task the 2½D interpolation mechanism has been used. In the described approach (using the 2½D method) the top view (flat, 2D scene) has been replaced by a cross-section plane, which is placed on the given height, counting from the scene ground as the base. The proposed algorithm 2½D is reduced to the implementation of the base method of finding the suboptimal movements paths together with the appropriate control algorithm for positioning the cross-section plane in the 3D space. This simplification is used for the generation of the subsequent planes in the spatial robot scene (Fig. 2). These sections, collected together, create the spatial form of the analyzed issue while considered separately allow to simplify the task of finding the safe paths to the flat (2D) problem.

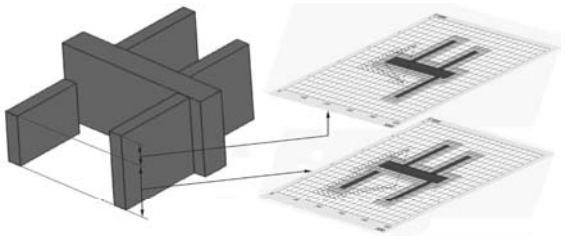


Fig. 2. The robot surroundings and the sections generated using 2½D algorithm

In order to consider the spatial robot's scene as a set of flat sections, certain conditions must be satisfied. A necessary condition is to assume that technological objects in the robot's environment have the form of rectangular blocks. Another assumption is that the base planes of objects line up with the scene's plane. This criterion also includes the impossibility of locating objects with empty space inside on the scene. If it is needed, to locate that objects in the robot's environment, it is necessary to replace them with cuboids [11].

An important aspect of path planning based on the 2½D algorithm is the appropriate matching successive intersections planes. Excessive number of sections increases the number of unnecessary calculations - each additional section generates the need for a flat scene analysis and the designation of sub-optimal path of motion. To ensure the possibility of generating the optimal - in terms of length - paths of motion, it is necessary to analyze all of the intersections, which differ in number of geometric traces, obtained the intersection of the technological objects [11].

Because each object has its own height, so the subsequent sections, going from the base plane towards the top, will have less or equal amount traces of cut objects in comparison to the previous section. The easiest way would be to sidestep the objects holding the robot's tool over them. Because it is not always possible to lift up the robot's wrist to the appropriate height, so the next condition, which had to be included in the developed algorithm, was the boundary condition which determines the maximum height to which the robot's wrist can be picked up [11]. It was assumed that the correct solution would be to generate the subsequent sections of the scene for the appropriate range of the height, determined by the range of the manipulator. The height scope is divided into small intervals, which are related to the heights of technological objects and include safety allowance, which is equal to the diameter of the sphere circumscribed on the object of manipulation (Fig. 3). The sections are done sequentially from the bottom, i.e. the first scope ranging from zero (the plane of the robot base) to the height of the lowest obstacle plus safety allowance. Another section range is measured from the height of the previous section to the one of the next, higher obstacle etc. In the same way the remaining parts of the scene are divided (up to a height limit).

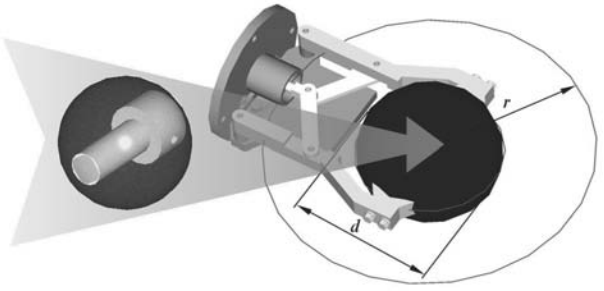


Fig. 3. Approximation of the object of manipulation by the sphere and spherical safety zone (r=d) around the gripper

The 2½D method simplifies the analysis of 3D space, in which all vertical edges (perpendicular to the ground) of the rectangular objects-obstacles are being replaced with corner points. This solution allows the determination of characteristic points, which constitute the basis for further analysis by determining the visibility graph vertices.

The process of searching for solution of the visibility graph is realized using the Floyd-Warshall algorithm. It allows determining the best transition between successive vertices of the visibility graph. It is very important that the projections of the start and end points of motion have well-defined positions in the visibility graph, since the result will require verifying the route between a particular pair of vertices, which must comply with projections of the robot position at the beginning and at the end of motion. It was assumed that it will be the first and the second vertex of the visibility graph.

As the result of assumptions about the implementation of movement between the empty cells with a common edge, the obtained path is described as a step function, which is not directly suitable for the robot control system. The Fig. 4 shows an example of such a path.

To smooth out the stepped function the NURBS curves has been used. They fulfil the requirement of the C^2 class smooth curves as well as the accurate interpolation of the first and the last checkpoint.

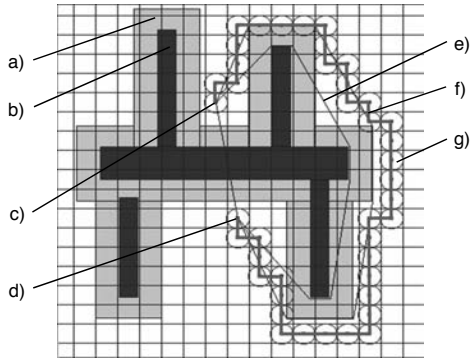


Fig. 4. An example of generated path: the safety zone (a), the obstacle (b), the start point of the motion (c), the end point of the motion (d), the visibility graph (e), the step function (f), the sphere that approximate the object of manipulation

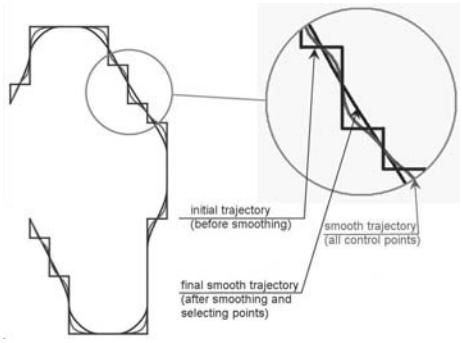


Fig. 5. Optimized and interpolated trajectory

The analysis have shown, that for the step nature of the motion path (broken line shown in Fig. 4), the implementation of a set of interpolation control points, which contains all of the intermediate positions of the robot, do not provide the optimal shape of the resulting trajectory (C^2 class) [12]. Attempts were made to increase the degree of the interpolation polynomial, however, due to the size of the set of control points (resulting from the properties of the function describing the path of movement), there were no way to get the monotonous trajectory - optimal in terms of the criterion of a minimum length of the movement. The solution to the problem was the variation of the weight factors of the control points. Performed tests allowed determining that the best shape of the final trajectory is attained in the case of minimum positive weight of the corner points for initially chosen step function. However, since the iterative process of choosing the weights of the control points require large amounts of computation, therefore other solution should be elaborated eliminate the stepped nature of the calculated safe path of movement.

It was decided to develop the algorithm of selective choice of control points, whose main task is to remove a set of intermediate points from the initially chosen path of movement. This set of points contains all of the corner points, whose removal does not cause a collision, but improve the quality of the resulting trajectory and accelerates the process of data analyzing. The result of the experiment is shown in Fig. 5.

The methods described in this chapter are used in the application called PLANER, intended for the planning of the collision-free path of the robot's manipulator.

3. The practical implementation of the elaborated method

The developed algorithm for collision-free trajectory planning has been implemented in the PLANER application [13]. Developed software consists of the forms for entering and processing the input data, as well as the graphics presentation of the calculations results. The main window is shown in Fig. 6.

In order to analyze the work space it is necessary to define the size of the scene and determine the amount, size and position of the technological environment objects. The application is equipped with the scene wizard - for this purpose it is necessary to run the ***Edycja Sceny*** (***Edit Scene***) function, which is available in the ***Scena*** (***Scene***) drop-down menu in the menu bar.

To increase the convenience of determining the coordinates of the beginning and end of movement there is the possibility of use the graphic method for entering the possible points. After pressing the corresponding key ***Wskaż*** (***Select***) the mouse cursor can be used to indicate the position (x, y) of the start point and end point of movement (Fig. 7).

In order to increase the readability of the indicated position (dark red colour) it has been decided to show also the second characteristic point of movement - depending on the choice done in the main screen, this is the start point (during the end point definition process) or end point (during the start point definition process). The z coordinate (the height) must be entered from the keyboard, because the process of defining points is done using view from above the work scene. If the position of any point is not already indicated, then no point is presented and the coordinate's box remains empty until the first selection is done. The graphical interface is built similarly as the scene graphics editor. The user interface window has the appropriate ratio to the specified dimensions of the scene - the default size of the scene is set to 10000 mm × 10000 mm.

The board has a grid with gradation equal to the diameter of the sphere circumscribed on the manipulation object. Leaving the grid on the screen was intended to facilitate the conversion of the coordinates.

In order to help identify the position of characteristic points on the scene, the board was complemented by technological environment objects along with their safety zone.

After running the scene wizard, the user can enter the parameters using tabular records. The number of rows in the table of objects can be arbitrarily increased or decreased, depending on the needs.

The change of the objects dimensions can be done by entering the coordinate values using keyboard or through a graphical indication of the position of the object. Editing or placing a new object by using the graphics editor is done by selecting two characteristic points - diagonally located corners of the object. The algorithm automatically assumes, that the lower values of the coordinates are the initial one and higher values describes the endpoint of the diagonal of the rectangle accept change as the coordinates of the lower initial coordinates of objects - the objects are defined by a vector (Fig. 8).

Fig. 6. The main window of the PLANER application: the size of subject of manipulation (1), the maximum height of the movement (2), the fields for entering the coordinates of the start and the end point (3), the button for graphical entering of the coordinates (4), the information about active scene objects (5), the number of points of the initial trajectory (6), the button for starting the computation process (7), the main menu bar (8)

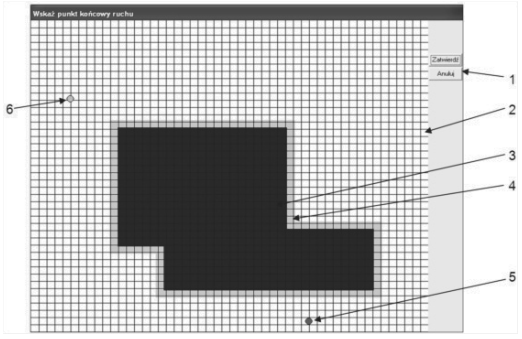


Fig. 7. The coordinate's editor: the decision keys (1), the scene grid (2), the scene object (3), the safety zone (4), the defined point (5), the other characteristic point (6)

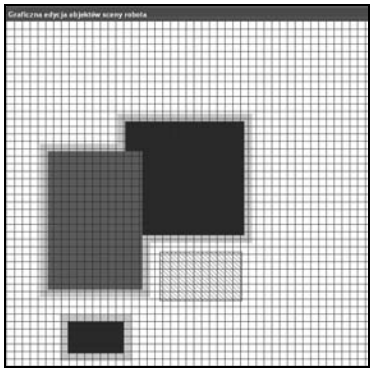


Fig. 8. The scene editor: the red filled rectangle indicates the object, which properties are edited; the red hashed rectangle represents the new position of the edited object or a brand new object

The results of computation are presented in the graphical form. The ***PLANER*** application has implemented the simple simulation module, which shows the resulted path in 3D environment. Moreover, the user can animate whole process of passing the object of manipulation (represented by the sphere) along the path and test whether in the given case the collision may occur. The example of the simple visualization is shown in Fig. 9.

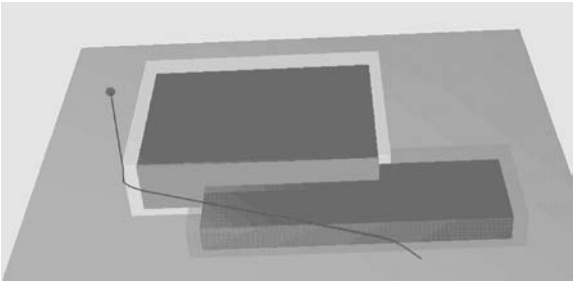
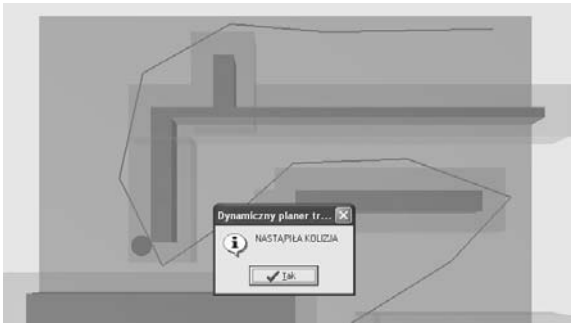


Fig. 9. The simple visualization of the resulted path: the red point represents the sphere related to the manipulated object

a)



b)

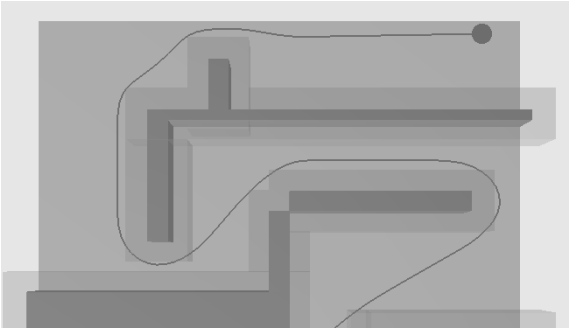


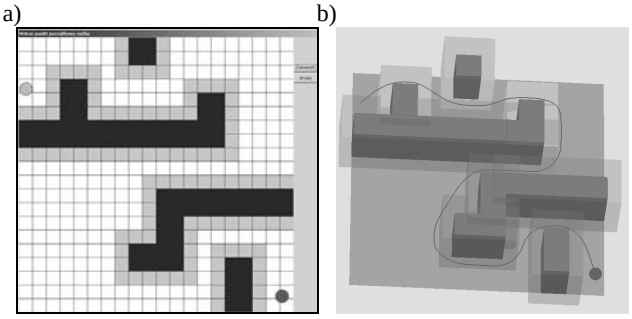
Fig. 10. The example of simulation using the path based on small number of points (a) and the proper number of points (b)

If the number of points is too small, then there could be possibility of a collision. The user will be warned about this fact, and will be able to set the correct (larger) value of points. The application has implemented the special algorithm, which provides an early detection of incorrect input data. The Fig. 10 shows an example of the trajectory, which has been generated two times: first for the small number of points (which was led to collision) and the second time for the proper number of points - simulation has confirmed the safety of the computed path.

4. The tests of the generated trajectories

Originally, the trajectories generated by the program ***PLANER*** were tested in the ***ROBOGUIDE*** application, then in reality, using the robot Fanuc RJ3iB ArcMate (Fig. 11) [13]. Later it has been decided to adapt the method of determining collision-free motion paths for use with the Mitsubishi Movemaster RV-M1robot. In the further part of the paper the example of generation of the collision-free trajectory for the Movemaster RV-M1 robot will be presented.

a) b)



c)

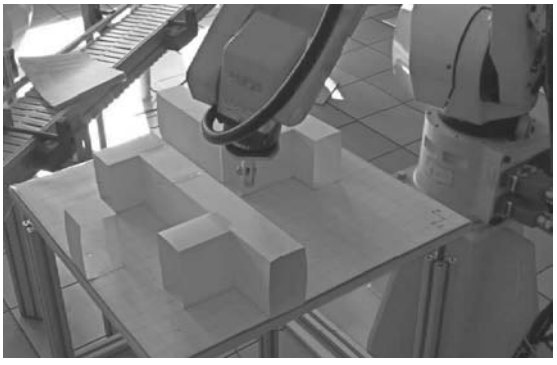


Fig. 11. The previous experiment: model of the scene (a) and the trajectory (b) generated in the PLANER application, which is realized using Fanuc RJ3iB robot

4.1. The short description of the used robot

The Mitsubishi Movemaster RV-M1 robot is fully equipped, small industrial machine. It has 1.2 kg payload and could be equipped with pneumatically or electrically driven tool (a gripper, a grind or drill etc.). The robot (the manipulator and the control unit is shown in Fig. 12.

The complete system consists of the control unit, teachbox and the manipulator. The robot could be controlled using teachbox, but the programming tasks must be done using personal computer, equipped with the Centronics or RS-232C port. The other option of programming is to use the pre-programmed EPROM memory, mounted inside the controller case. Because the control unit is equipped with the EPROM writer, the program could be also stored on the EPROM memory chip. Connecting the PC using the Centronics (printer) cable allows one-way communication from the PC to the robot. In this way, the program can be sent to the controller, but it is not possible to backup on your computer the program that is already stored in the robot's memory. Using the RS-232C cable (null modem type) allows the user to communicate in two directions. This mode not only allows downloading the program from the controller, but also enables the two-way interactive communication with the use of any terminal program. The RS-232C transmission is therefore often used to connect the robot to the simulation programs.

Fig. 12. The RV-M1 robot's manipulator mounted on the stand in the Institute's robotics laboratory. Under the table the controller is mounted

The robot has its own, a simple programming language called Movemaster Command, which in future versions of the robot was developed to the MELFA BASIC dialect. The program line is a string containing the two-letter instruction and subsequent parameters (Fig. 13). If before the instruction a decimal number is given, it means that the line should be stored in memory for later use. The program line which is not preceded by a number is executed immediately after confirming it by the *Enter* key. To transmit commands to the controller, the user could utilize the standard Application Programming Interface or write his own handler for the parallel or serial port. In the case of use the immediate execution mode, it should be kept in mind that the proper waiting times should be selected, because of interruption of the data transmission during the manipulator movement.

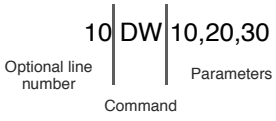


Fig. 13. The structure of program line in the Movemaster Command language

The programming capabilities of the robot are among others determined by the amount of the memory installed in the robot's

controller. In the case of the Mitsubushi Movemaster RV-M1 robot the memory capacity is 8KB for the positions storage (what is equivalent of 629 positions) and 16KB for the program (what is equivalent of 2048 program steps).

4.2. The tests in the ROBBO application

After generation of the optimal trajectory in the ***PLANER*** application it has been decided to do some tests in the virtual environment in order to check the correctness of the written program.

The ***ROBBO*** application [14-16] is the main program of the ***ROBBO*** package, which consists of

- the server application for remote programming of the robot,
- the web camera service application,
- the main application used for simulation and testing the programs.

The server and the web camera service applications reside on the computer, which is connected directly to the robot controller - in this case, the RS-232C port is used. The web camera is used for remote monitoring of the robot's workspace and its usage is optional.

The ***ROBBO*** simulation program could be used in connection with the server and web camera service by communicating with them through the TCP/IP protocol or can act as standalone application. In the case of current research, only simulation module is used on the computer using the direct connection with the robot. The main window of the ***ROBBO*** program (shown in Fig. 14) consists of two areas: the control panel (1) and the simulation window (2).

The control panel has five sections (Fig. 15). The first one is used for selecting the active robot in the case, when the virtual environment is shared between computers connected to the network and running the ***ROBBO*** application - up to five robots can be simulated simultaneously, where one of them can be a virtual clone, connected to the real robot. The second section of the panel is used for displaying the current coordinates. The third one consists of five sliders, where each one of them is bound to one axis of the robot - the user can use them (sliding with the mouse) to roughly set the position of the manipulator. The fourth section of the control panel is used for entering the robot's position by keyboard. Section five is the simple editor, where the program could be loaded and edited. There is also a possibility to automatically enter the position declaration to the program using sliders (section 3) and the button "V" located in the section 2 or manually entering the coordinates using text fields from section 4.

The simulation in the ***ROBBO*** application does not allow to draw the trajectory directly on the screen (i.e. the tool does not leave a trace), but it could be used for checking the correctness of the program. Running the program in the virtual environment allows checking the arrangement of the robot's arms, position of the tool and the general quality of the generated trajectory.

The ***PLANER*** application has been adjusted to generate a set of the trajectory points in accordance with the requirements of the Movemaster Command syntax. The resulted file, in text format, can be sent to the robot controller's memory regardless of the use of the ***ROBBO*** program. The maximum number of the generated points cannot be greater than 629 entries, because there is limited capacity of the memory, so it was necessary to impose additional conditions for validating the data entered in the program.

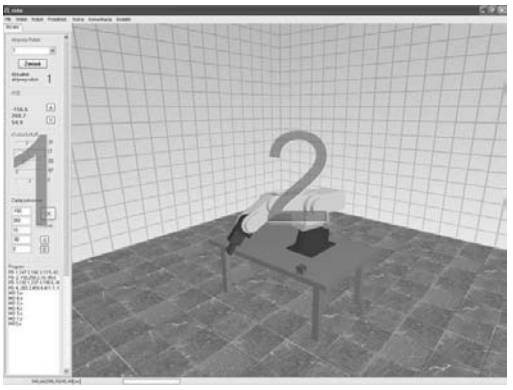


Fig. 14. The main window of the ROBBO program: the control panel (1) and the simulation window (2)

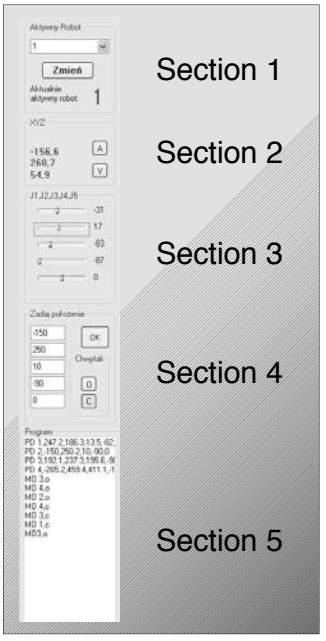


Fig. 15. Five sections of the control panel (detailed description in the paper text)

For testing purposes the special board has been prepared. The objects arranged on the board served as the scene elements that determine the shape of collision-free path. The data describing the size and location of the scene elements were introduced into the ***PLANER*** application. For the specified start and end points of the path the optimal trajectory has been generated. The resulting set of points has been entered from disk to the ***ROBBO*** application and then the necessary changes have been made using built-in editor, so the complete program has been created. The manipulator of the robot is equipped with a gripper, and then in order to better visualization of the movement along the created trajectory an object of manipulation has been used - it was a

whiteboard marker with the coloured cap, whose tip was moving along the specified path. After initial testing in a virtual environment it was decided to shorten the generated path due to the limitations of the manipulator workspace. Revised program has been sent to the robot's controller. For the experiment, the identical testing board has been used as for tests done with Fanuc robot (Fig. 11), but the path length has been shorten to meet technical restriction of the manipulator movement.

The board was placed on the robot's table (Fig. 16) and adjusted in the relation to the start point and the end point. Then the manipulator has been moved to the start point and the program was executed. The experiment carried out in the real environment confirmed the correctness of the algorithm and the possibility of its adaptation to cooperate with another robot.

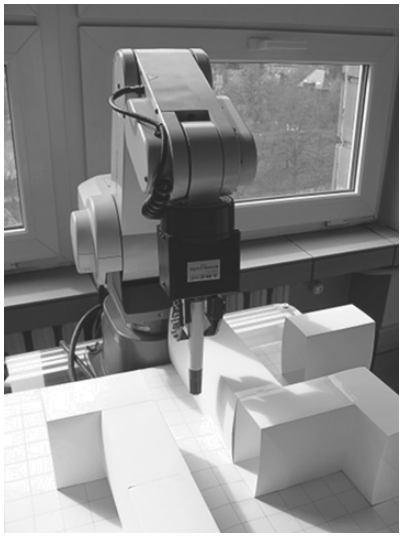


Fig. 16. The Mitsubishi Movemaster RV-M1 robot realizes the trajectory generated in the ***PLANER*** application

5. Conclusions

The automatic generation of optimal, collision-free motion paths of the robot's manipulator, strongly supports the process of robot programming, in most cases giving the desired results without the need of manually entering or correcting the data. Following the principle of limited confidence, an important step is to check the correctness of program by executing it in a virtual environment. The final step is testing the program on a real robot. In this manner, the operator can significantly shorten the time needed to create the program and start working cycle. This approach clearly simplifies the stage of defining the relevant points of the trajectory in order to avoid collisions with the technological objects located in the robot's manipulator environment. Thereby it significantly reduces the time needed for implementation of the program to the production cycle.

The further stage of research will be concentrated on the consolidation of trajectory generating and simulation phase with the program execution stage in such a way, that the determination of collision-free path could be realized in real time.

References

[1] J.C. Latombe, Robot motion planning, Kluwer Academic Publishers, London, 1993.

[2] G.G. Kost, Method of avoiding collisions of a robot with its environment based on the system of weights, Proceedings of the 7th International Conference "Automation/Robotics in theory and practice" ROBTEP'2004, Vysne Ruzbachy, 2004, 301-304.

[3] G.G. Kost, Strategy of determining non collision trajectories of robot operating in a complex process environment, Proceedings of the 7th International Conference "Automation/Robotics in theory and practice" ROBTEP'2004, Vysne Ruzbachy, 2004, 305-308.

[4] G.G. Kost, System of designing robot trajectory on the grounds of Markov's decision processes and Q-learning algorithm, Proceedings of the 7th International Conference "Automation/Robotics in theory and practice" ROBTEP'2004, Vysne Ruzbachy 2004, 309-312.

[5] G.G. Kost, The use of the Q-learning algorithm for programming the industrial robots, Proceedings of the 12th International Scientific Conference "Achievements in Mechanical and Materials Engineering" AMME'2003, Zakopane, 2003, 505-508 (in Polish).

[6] I. Dulęba, The algorithms of optimal planning of the robot's trajectory with use of the discretized phase plane, PhD thesis, Wrocław University of Technology, 1992 (in Polish).

[7] I. Dulęba, Methods and algorithms of motion planning of the mobile and manipulation robots, EXIT, Warsaw, 2001.

[8] M. Egerstedt, C.F. Martin, Optimal trajectory planning and smooth splines, Automatica 37/7 (2001) 1057-1064.

[9] A. Elnagar, A. Hussein, On optimal constrained trajectory planning in 3D environments. Robotics and Autonomous Systems 33 (2000) 195-206.

[10] I. Dulęba, Algorithms of motion planning for nonholonomic robots, Publishing House of Wrocław University of Technology, Wrocław, 1998.

[11] D. Reclik, G. Kost, The 2½D algorithm in robot workspace analysis, Proceedings of the 4th Conference "Mechatronic Systems and Materials" MSM 2008, Acta Mechanica et Automatica 2/3 (2008) 65-70.

[12] D. Reclik, G. Kost, The comparison of elastic band and B-Spline polynomials methods in smoothing process of collision-less robot trajectory, Journal of Achievements in Materials and Manufacturing Engineering 29/2 (2008) 187-190.

[13] D. Reclik, Planning and the optimizing of the collision-free trajectory of manipulating robot in its workspace, PhD thesis, Silesian University of Technology, Gliwice, 2010 (in Polish).

[14] K. Foit, J. Świder, The project of a platform-independent, off-line programming system for industrial robots, Proceedings of the 7th International Scientific Conference on "Computer Integrated Manufacturing - Intelligent Manufacturing Systems" CIM'2005, Gliwice-Wisła, 2005, 62-65.

[15] J. Świder, K. Foit, G. Wszołek, D. Mastrowski, The off-line programming and simulation software for the Mitsubishi Movemaster RV-M1 robot, Journal of Achievements in Materials and Manufacturing Engineering 20 (2007) 499-502.

[16] K. Foit, The web-based programming interface for the Mitsubishi Movemaster robot, Journal of Achievements in Materials and Manufacturing Engineering 27/2 (2008) 183-186.