THE SIMULATION PROGRAM OF TRAINING ROBOT SLR 1500

Poppeová, V.; Urícek, J. & Máca, M.

Abstract: *For a simulation of robot is needed the information about the dimension and the character of his kinematic elements. After the simplification of complex simulation problem was encountered following main sub-tasks: determination of kinematic structure end points in the space, 3D representation of individual robot components on the computer screen, interpolation solution for robot 3D movement, problem of program editing in special language, its writing, saving, loading and execution of the simulation. For the robot component representation was used polygonal bodies.*

1. INTRODUCTION

The training robot SLR 1500 was developed at the University of ? ilina and the firm IQM Zvolen (Slovakia) to serve for the teaching of robotics at high schools and universities. Its design allows to assist with the programming, the technical cybernetics, solving the drive control problems, the design of control system for industrial robots and operation of NC machines.

The training robot SLR 1500 has angular design. It has five degrees of freedom and three of them serve to position the end effector in the space and the remaining two determine the its orientation with respect to the object being manipulated. Kinematic structure of the robot SLR 1500 consist of rotational waist, two arms and wrist mechanism. All parts are connected with rotational joints in such a way that the arms are not in the same plane but are moved with respect to each other. Such design allows extremely large rotational range of individual arms. Robot SLR 1500 has 5 degrees of freedom.

As a control system of this robot can be used any 16 or 32 bit computer with RS 232C interface. For this computer a special software must be developed to represent functions of the control system. The needs of such control systems were satisfied with a programming language RAMAS [VUKOV Prešov, 1985]. Its commands can be divided into several groups: editing commands, move instructions and commands, technological, control, arithmetical, logical, waiting, interruption serving, help commands and other.

2. SIMULATION

By the simulation we generally understand observation and study of a system behaviour on its model. The system consist of ordered pair of sets, where one of the sets represents the objects (items, components and so on) and the other describes the relations between them.

A graphical model which would represent the behaviour of a real robot up to smallest details would be very difficult and also ineffective. What is important for a robot simulation is therefore the depiction its element position with respect to activities the robot is performing. Either picking up of an object, moving the end element from a point to another (with no trajectory specification or with linear interpolation) or it is simulation of its manual control.

The movement is executed after entering a program instruction with co-ordinate parameters of the movement end point. From these co-ordinates the robot co-ordinates are calculated first (rotations of individual joints) and then all the elements start to move with the maximal speed in the appropriate direction.

If the trajectory is specified then it is generated by interpolator. The interpolation replaces given trajectory with a system of incremental movements in the direction of co-ordinate axes in such a way that the resulting trajectory is in maximal accordance with the trajectory originally requested. This trajectory must definitely satisfy the requirements for precision. Moreover, from the interpolation method we expect the reliability, accuracy, simplicity and speed of interpolation.

For the simulation of object manipulation it must be possible to ensure opening and closing of the end effector. For the representation of this operation we use value of internal flag, which will stand for opening / closing of the gripper and at the same time, release / pick up of the object.

The simulation of manual control gives an idea about the robot movement in the working area. It is the robot rotation around horizontal axis, the individual arm rotation around their axes, the gripper mechanism rotation around lateral and transverse axes and also opening and closing of the gripper.

Further, the manual control of the robot allows to imagine, the robot working range and its boundary positions. The movement in individual direction is realised by pressing keys on the computer keyboard.

The algorithm creation follows the practice of problem decomposition into smaller and simpler sub-problems, which solution is easier to find. The main simulation program was supposed to be able to execute this tasks. Later, these tasks were in detail worked out in PASCAL programming language.

Since the controlling language is RAMAS and language instruction for simulation program are not compatible we have a problem of program interpretation. A RAMAS program is a ASCII text file edited in arbitrary text editor.

We have had to develop component which would serve for adaptation (translation) of the RAMAS program to a list of instruction understandable for simulation program. The algorithm for this translator is based on a flowchart. On the basis of this flowchart we have created a subprogram in PASCAL language, which reads the RAMAS program, line after line, detects the instruction and following its character and parameters it calls appropriate procedure, which executes an action particularly for this instruction.

As a interpolation procedure which best suits our needs was chosen the method DDA (Digital differential interpolation). Its advantage with respect to direct functional calculation is that DDA interpolator directly calculates the function with constant trajectory increments. With this approach is the movement derived from impulses sent to individual axes with relation to their frequency.

If we have two points A(x_A, y_A, z_A) and B(x_B, y_B, z_B) the interpolator must generate the trajectory between them. It divides the trajectory into n segments. In the same way it also divides into n segments a trajectory in individual axes. The dimensions of individual axes increments i_X, i_Y, i_Z are determined. After the execution of appropriate number of steps in each direction reaches the interpolator the end point of movement. The higher number of impulses per time unit, the smaller is the unit step (increment) and the movement precision is then also higher.

The control system must also solve the inverse transformation from Cartesian co-ordinates if the interpolation to robot co-ordinates with higher frequency. Less complicated case is when the robot end element is supposed to reach the end point without trajectory specification. For starting and ending point the robot local co-ordinates are calculated A($?_{1A}$, $?_{2A}$, $?_{3A}$, $?_{4A}$, $?_{5A}$) and B($?_{1B}$, $?_{2B}$, $?_{3B}$, $?_{4B}$, $?_{5B}$). For each joint the direction and the angle of the movement are determined. If value of a variable reaches the value of the angle in the end position, the movement in this point is terminated.

For the robot programming we need to know the relation between its local co-ordinates and global co-ordinate system and also the other way around. From the known global co-ordinates we need to determine position of its kinematic elements - local robot co-ordinates. The first case is called the direct transformation and the other one is inverse transformation.

For direct transformation we need to use the relations between joint angles and Cartesian co-ordinates and to express end element position in co-ordinate system fixed with the robot base. We start from robot kinematic structure. The dimensions of individual components and relative joint rotation angles are important for the transformation.

The inverse transformation of the robot is more complicated and the solution is possible only for certain class of relatively simple kinematic structures. It is possible to prove that they are mainly those, which three axes satisfy one of the following conditions: they are transitional, they are rotational with one intersection - a spherical joint, they are rotational with parallel axes.

These constraints influence the design of mechanical components of contemporary manipulators. For the required position and orientation of end element of the mechanical system in the co-ordinate system connected with the origin it is necessary to determine robot local co-ordinates. The methods how one can realise a graphical simulation can be different: The simplest one is a symbolical projection of the robot to three planes from the front, the side and the top. A more realistic representation of 3D object is a axonometric representation.

Kinematic components of the robot can be replaced by the basic 3D bodies. Such bodies are frequently the prisms or bodies specified by finite number of plane surfaces - polygons. If these bodies are complex and their axonometric representation is unclear, it is necessary that the model is represented realistically respecting the visible and the invisible edges and surfaces.

For graphical depiction of 3D object to 2D space (such as computer screen) we use several representation methods. Most of the time they are centre, parallel and right angle projection. The kinematic components of SLR 1500 robot are the bodies of general shape bounded by planar and curve surfaces.

If the simulation is running in the real time, we must allow several simplification: planar surfaces are represented by polygons with arbitrary number of vertices, curved surfaces (spherical and cylindrical) are represented by arbitrary number of planar surfaces. If we want to present sufficient information about the body, we need to encode each of its sides, edges and vertices. With a help of such system we can create structure of input information about the body. Therefore is the structuring important for computer graphics.

In order to keep the graphical model moving in the same way as real robot, we need to determine kinematic relations of graphical objects representing the arms.

Because we wanted to keep the visibility of the bodies only the vertices which orientation in the projection is same as the orientation of input data (only visible sides). The point which create individual planes are first recalculated (according to given viewpoint) to the plane of the screen (3D to 2D). For the depicting of the whole model we have to decide about the visibility and then draw 61 polygons to the screen. The speed of drawing depends on the speed of the computer and size and speed of its video memory.

Fig.1. The simulation of training robot SLR 1500

3. CONCLUSIONS

The simulation program allows to create the programs for training robot SLR 1500 either by executing RAMAS language commands or by manually teaching the robot movement. Such a program can be later edited in the program editor.

Since this simulation program works with programming language RAMAS, which is used in majority of robots of Slovak production it is possible to use it for teaching of industrial robots programming. The simulation is realised in axonometric view and respects the visibility of the edges and surfaces, which helps to visualise the concrete situations.

After the development of hardware control system for robot SLR 1500 is finished it is possible to extend this simulation software with communication module. This would allow to send the created, with a simulation verified programs directly to the memory of the control system via serial interface.

4. REFERENCES

VUKOV Prešov (1985). Programovací jazyk RAMAS. VUKOV, Prešov

Mièieta, B.& Král,J. (1998). Plánovanie a riadenie výroby. ES ?U, ?ilina

Tolnay, M.(1983). Priemyselné roboty a manipulátory. SVŠT, Bratislava.

Kolíbal, Z. (1992). Prumyslové roboty I. VUT, Brno

Matièka,R. & Talácko,J.(1990). Mechanizmy manipulátoru a prumyslových robotu. SNTL, Praha

Authors: *Ing. Viera Poppeová, PhD.,Ing. Juraj Uríèek,PhD., Department of Measurement and Automation, University of ?ilina, Velký diel, 010 26 ?ilina, Slovak Republik, Phone: 00421-89-52806, Fax: 00421-89-652940, e-mail: poppeova@ fstroj.utc.sk*

Ing.Milan Máca, Eindhoven University of Technology, Faculty of Mechanical Engineering, Netherlands