DESIGNING OF ROBOT WORKSPACE BY APPLICATION PROGRAMMING INTERFACE

Miran Brezocnik, Simon Brezovnik, Simon Klancnik, Joze Balic,
Borut Buchmeister and Leo Gusel
University of Maribor, Faculty of Mechanical Engineering,
Smetanova ulica 17, SI-2000 Maribor, Slovenia

ABSTRACT

In this paper, we propose an application programming interface (API) for designing of a robot workspace. The interface is realized inside a SolidWorks computer aided design environment. Via API, the builder of the robot system can check if: a) the current robot mechanism can perform the desired work movements, b) the robot can avoid singularity, and c) the points of the robot trajectory are within the robot workspace. With the proposed approach, a more successful designing of robotic cells is assured.

Keyworks: Robotics, SolidWorks, API interface, Workspace

1. INTRODUCTION

Robotics has become intertwined in recent decades in many industrial processes as an indispensable part of modern, economical and human-friendly technologies.

Without robotic manipulators we cannot imagine things like welding of automotive chasses, insertion of pieces into compactors or spray painting. The appearance of surgical robots or mobile service robots does not surprise us at all. The importance of robotics in the world is growing. This is also clearly evident in the statistics of robots sold in the USA published in the IEEE Robotics & Automation Magazine, Sept 2007th. From 1992 onwards, the growth in sales of robots was over 20% per year.

1999 was the most successful of year and the total market value of the robots in USA alone exceeded one billion U.S. dollars. On this subject it is worth noting that USA is only in third place in the world in the use robots, after Japan and after all the other highly technological Asian countries together. This growth in demand for robots is attributed to the fact that they opened a new, fast-growing areas of use [1].

At the beginning, almost three quarters of robots were used for welding in the automotive industry. Now, they can also be found in other industrial fields, such as electronics, production and processing of foods and beverages, pharmaceuticals, household appliances. The number of robots is steadily growing in Slovenia. We not only have robots in the automotive industry, but they also appear in the product and bulk manufacture, for example, in electric and food industry. We meet them wherever require a high quality of products is required, where the work is monotonous and where health hazards exist.

When designing a robotic cell, the engineers often face the problem of positioning the robot in the robot cell, so that the robot's actions will be within its workspace. It is important that most of robot's movements take place within the optimal working space.

To answer the question presented in this paper, we present one possible solution, with which it is possible to use API interface during the process of planning to determine whether a planned distribution of objects within the robot cell is within the robot's workspace or not.

2. ROBOT DESCRIPTION

In carrying out these robotic applications available, we used an industrial robot ACMA XR701 by Renault Automation, which has six axes of movement (Figure 1). The diameter of the workspace is shown in Figure 1.

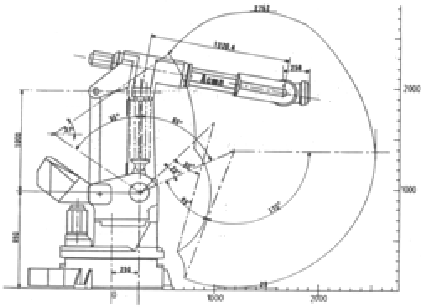

Figure 1. ACMA robot workspace XR701

Characteristics of ACMA XR701:

- Three-phase power synchronous – Brush Less servo motors,
- number of axes: 6, angles of movement (Table 1),
- sensors: resolvers – encoders of rotation position,
- end joint capacity: 125 kg,
- repeatability: r = 0.3 mm.

Table 1. Angle of movement and parameters of ACMA XR701 [2].

Number of axes	Angle of axis movement (°)	Max. speed (rd/s)	Time to reach the maximum speed (ms)
1	360	1.9	690
2	+65/-55	2.2	650
3	+110/-27	2	730
4	400	3	435
5	240	3	435
6	600	4.2	585

We did not have a CAD model for Acma XR701, so we decided to create one by using the reverse engineering process. We used optical digitization. Description of the overall process is shown in reference [4]. In this article we shall therefire limit ourselves to the use of the CAD model, which is the result of work, described in that reference. The development of the API interface follows.

3. API INTERFACE DEVELOPMENT

During the planning process of robotic applications, outlined in chapter two, we faced the question: How to install a robot in a robotic cell and how to adjust the fixture at the location of work?

Robot cell planner can acquire the robot CAD model from the manufacturer, which does not contain the direct and inverse kinematic model, with which it would have been possible to verify the design of workspace during planning and avoid singular positions of the range of work movements. A singular point occurs where it is not possible to calculate the inverse Jacobi matrix and thereby determine the speed of joints from given speeds of robot's top. The consequence is a reduced mobility.

Due to these difficulties, we decided to make a SolidWorks API programming interface with which we can verify the kinematics of movements and the presence of singular points in in the area of work movements of the robot during the robotic cell design. The API interface allows us to automate and adapt the modeling program and opens the possibility to import and export data.

The decision for selecting the SolidWorks 2009 SP0 software was not based on potential advantages over other software which offers similar solutions, like Catia and Pro/ENGINEER, we chose it because of previous experience with the API interface. What constitutes a programming interface API? The SolidWorks Application Programming Interface allows applications to access data from various existing data sources. The standard specification of SolidWorks software allows access to databases on different platforms.

The API programming interface is used to automate a series of commands to be carried out consecutively or at the according to the structure of a macro. A macro is a defined sequence of operations (commands, functions), which function as a set of commands. This definition, however, is expanded when using VBA to connect with the Microsoft Visual Studio 2005 development environment, which means that the programming language can have a very diverse influence on the modeling in SolidWorks 2009. It was this connectivity with the development environment that we used during designing of the API interface to verify the above-mentioned critical operating conditions.

How to create a simple macro? The initial macro can be easily recorded with a macro recorder. Start the Record Macro function, execute successive commands and stop recording.

When we wish to repeat the operations, we simply run the recorded a macro. This approach is used in making macros for simple, repetitive operations. If we want the macro to contain decision, conditions or repetitive operations, and if we want to communicate directly with users or to control the parameters, SolidWorks 2009 SP 0.0 offers a choice of recording a macro in one of the following programming languages: Microsoft Visual Basic (V6), Microsoft Visual J# (Java), Microsoft Visual Basic.Net, Visual C + + and Visual C#. Net. The saved macro can be edited in Microsoft Visual Studio 2005 for Applications, which is already included in the basic installation of SolidWorks 2009 [5]. For the construction of the API interface we chose Visual C#.Net programming language.

Development environment was made in Microsoft Visual Studio 2005, which enables access to all commands, tools and information related to the development of the parts, creation of assemblies or creation of drawings [6,7] via 'SolidWorks.Interop. Sldworks' and 'SolidWorks.Interop. Swconst' libraries.

4. API CONNECTION INTERFACES WITH THE MODEL ROBOT ACMA XR701

In the CAD model of the robot we entered data on restrictions for all six axes (Table 1). We also measured the minimum distance to objects in the area and checked for collision with surrounding objects as well as with its own structure (Figure 2).

Once boundary conditions have been determined, we could check, manually or through scheduled work items, the following: a) can the robot carry out movements in its current position, b) can it avoid singular points, and c) are the work trajectory points located within the robot's workspace [3].

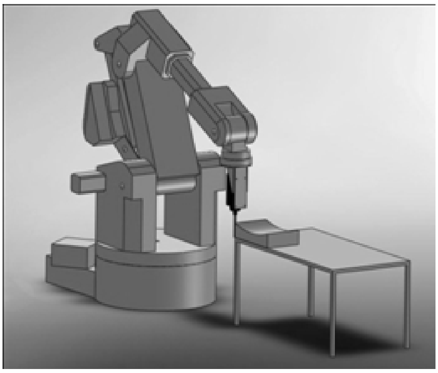

Figure 2: Acma robot model with API application interface for milling

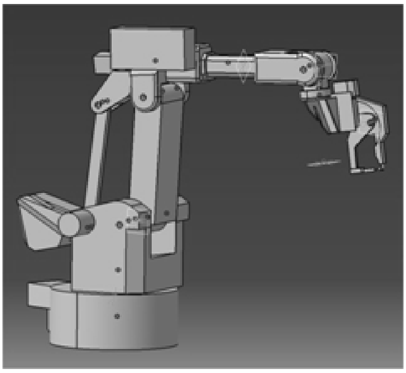

Figure 3: Acma robot model API interface for administration, scoring resistance welding

The workspace is very important for the robot [8]. It tells us, how far the robot can reach. The reachable workspace is the volume robot's top apparently covers when it is at a maximum distance from the apparent center. Figures 2 and 3 show the model of the ACMA robot with the API interface.

5. CONCLUSION

In the future we want to upgrade the API interface to include an automatic control of the available workspace and automatic notification of a nearby singular point. For given boundary conditions of axes rotation measured robot geometry data we will add automatic calculation of direct and inverse kinematic model, with which we wish to graphically show the areas with greater and lesser mobility of the robot. With the help of such tool it will be possible to determine the robot's location within the cell even faster and easier.

6. REFERENCES

[1] Perme T., Kirn B., Skok I.: Renovation of robots for manipulation and assembly, Ventil 11, pp. 52–59, 2005.,

[2] Brezovnik S.: Resistance spot welding robot ACMA Renault XR701, Faculty of Electrical Engineering and Computer Science University of Maribor, University thesis, September, 2006.,

[3] Pahole I., Ficko M.: Programming NC machines, University of Maribor, Faculty of Mechanical Engineering, Maribor, 2004.,

[4] Brezovnik S., Brezocnik M., Pogacar V., Cupar A., Vaupotic B.: Making a virtual model of the robot with digitization for indirect programming, Proceedings, Portoroz, October 2007, Ljubljana, 2007.,

[5] Xiaoshu J., Xichen Y.: School of Machinery and Electron, Institute of Laser Technology, Tianjin Polytechnic University, Tianjin 300160, China, 2009.,

[6] Zlajpah L.: Simulation in robotics, The Jozef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia, 2008.,

[7] Rahmani K., Arezoo B.: A hybrid hint-based and graph-based framework for recognition of interacting milling features, Department of Mechanical Engineering, Amir-Kabir University of Technology, Hafez Ave., P.O. Box 15875-4413, Tehran, Iran, 2007.,

[8] Brezovnik S., Brezocnik M., Klancnik S.: Robot design job with a development environment API, Herakovič N. (editor), Debevec M. (editor)., Adrović E. (editor), Automation Workshop manipulation and editing, 2009, Ljubljana, ASM '09. Ljubljana, Faculty of Engineering, University of Ljubljana, 2009.