Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

17th International Conference on Process Control 2009

Hotel Baník, Štrbské Pleso, Slovakia, June 9 – 12, 2009

ISBN 978-80-227-3081-5

http://www.kirp.chtf.stuba.sk/pc09

Editors: M. Fikar and M. Kvasnica

Šťastný, J., Motyčka, A.: Robots Control by Means of Object Oriented Technology, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 17th International Conference on Process Control '09*, Štrbské Pleso, Slovakia, 548–552, 2009.

Full paper online: http://www.kirp.chtf.stuba.sk/pc09/data/abstracts/095.html

ROBOTS CONTROL BY MEANS OF OBJECT ORIENTED TECHNOLOGY

J. Šťastný *, A. Motyčka **

Mendel University in Brno, Faculty of Business and Economics,
Department of Informatics, Zemědělská 1, 613 00 Brno, CR,
** tel.: +420 -545132728, e-mail: stastny@node.mendelu.cz*
*** tel.: +420 -545132200, e-mail: mot@mendelu.cz*

Abstract: The paper deals with problems of the industrial robots control on a trajectory given with a view to the 6-axial anthropomorphic robots. The control program which has been created by means of modern object oriented technology applied to the design and implementation of the program data structure and to the chosen vector method in the calculation of an inverse kinematics problem. The principles and algorithms given below have been used in an application that was developed at Brno University of Technology and Mendel University in Brno.

Keywords: Robots control, object oriented technology, class, data structure.

1 INTRODUCTION

In general a robot consists of four main sections:

- manipulator (positioning, orientation).
- end effector (tentacle, Schoop gun, welding head etc.).
- drive (electric, pneumatic, hydraulic).
- control system (software, sensors, camera system).

Control of robot motion is usually solved in three and more axis. In industry control in 6-axis is mostly used, three for positioning (RRR) and three for orientation of end effector (RRR). The most important part of the robot is control system that contains control software. The control program computes all values that are necessary for implementation of the project given. It means for example the motion control of end effector on a trajectory given including orientation change or displacement of the position in space with regard to orientation of end effector [Bělohoubek (1999)].

For the control of robot motion or more precisely its end effector between two node points it is important to determine if it is necessary to consider the trajectory of end effector or end point of the robot tool. Accordingly the motion control of end effector is divided into control with CP method (Continuous Point) or PTP method (Point To Point). Within the motion between node points not only position of end effector can be changed but also its orientation. In general this change can proceed along one of axis of robot system of coordinates in which position of end effector is changed or along general threedimensional axis that is defined by point and rotation angle [Burkardt (2005)].

The created control program uses CP method for motion control of end effector for 6-axial anthropomorphic robot. CP method is in the program implemented as class *INTERPOLATOR_CP* that performs integrating linear interpolation of the trajectory between node points [Šťastný (1999)].

2 PROGRAM DATA STRUCTURE

Class hierarchy of program data structure is displayed in fig. 1.

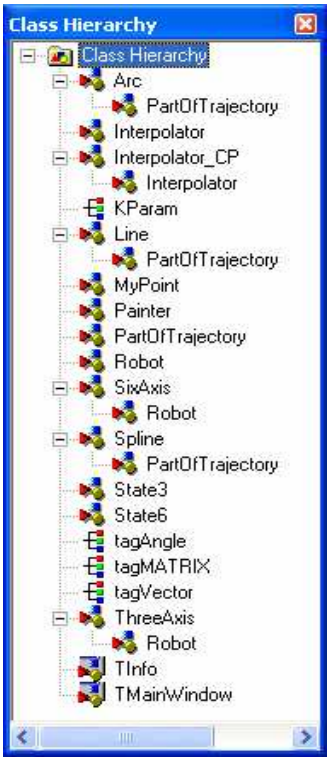

Fig. 1 Class hierarchy

Program data structure uses advantages of object technology for simple representation of class *Robot* (see fig. 2) as abstract data class that is not determined for making instances of this class but for the next inheritance of members representing individual robot types (*SixAxis*, *ThreeAxis*).

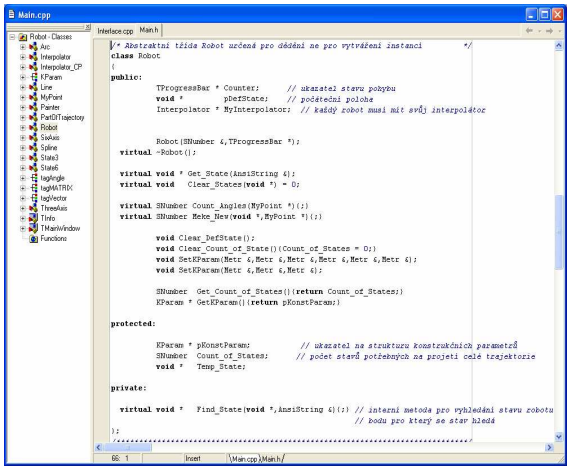

Fig. 2 Abstract data class Robot

As members of class *Robot* in the program there are implemented classes *SixAxis* and *ThreeAxis* representing 6-axial and 3-axial robots with the corresponding virtual functions. The parental class *Robot* uses parametric constructor *Robot::Robot(SNumber & ,TProgressBar *)* and members of this class gain parameters of their parent's constructor.

```
SixAxis::SixAxis(SNumber &
ptr,TProgressBar * p):
Robot(ptr,p)
{
  pKonstParam = new KParam();
  pDefState = 0;
}

Robot::Robot(SNumber &
ptr,TProgressBar * cc)
{
  Temp_State = NULL;
  Counter = cc;
  pDefState = 0;
  MyInterpolator = new
Interpolator_CP(ptr,Counter);
  pKonstParam = 0;
  Count_of_States = 0;
}
```

With paternal class *PartOfTrajectory* and its members *Line*, *Arc* and *Spline* it is the same. From these classes only class *Line* is used on, the other two are presented like possibilities for the next modification of the program:

```
Line::Line(Second & ptr,SNumber &
_count,TProgressBar * _prog):
PartOfTrajectory("Line ",ptr,_count,_prog)
{
}

PartOfTrajectory::PartOfTrajectory(AnsiString
name,Second & ptr,SNumber &
_serial,TProgressBar * _prog)
{
  Name = name;
  Prog = _prog;
  DeltaT = ptr;
  StartP = new MyPoint(Name +
IntToStr(_serial) + "  start point");
  EndP   = new MyPoint(Name +
IntToStr(_serial) + "  end point");
  Vec_a = new Vector();
  Nx = 0;Sx = 0;Ax = 0;
  Ny = 0;Sy = 0;Ay = 0;
  Nz = 0;Sz = 0;Az = 0;
  pCn = new tagMATRIX();
  pCo = new tagMATRIX();
  pCon = new tagMATRIX();
  Kind_Rot = 0;
  Gama = 10000;  // inicializacni hodnota
  Axe = 0;
  rx = ry = rz = 0;
  Serial = _serial;
 }
```

This method is also used in implementation of the interpolator. Class *Interpolator* is proposed as

parental class and class *Interpolator_CP* is implemented like the only member:

```
Interpolator_CP::Interpolator_CP(SNumber
& ptr,TProgressBar * _op):
Interpolator(ptr,_op)
{
 for(int i = 0;i < 100;i++)
  Trajectory[i] = 0;
 CountOfTrajParts = 0;
 Points = 0;
}
```

Entities necessary for computing and saving results are implemented in the program as objects represented by classes or structures and their methods (see tab. 1).

Data type	Implementation
Kparam	typedef struct Kparam { …
MyPoint	class MyPoint { …
Painter	class Painter { …
State3	class State3 { ...
State6	class State6 { ...
tagAgle	typedef struct tagAngle { ...
tagMATRIX	typedef struct tagMATRIX { ...
tagVector	typedef struct tagVector { ...

Tab. 1. Data types defined

The defined data types used in the program are in tab. 2.

Data type	Title
typedef int	Snumber
enum	Axe_def
typedef double	Number
typedef Number	Metr
typedef Number	Second
typedef unsigned short int	Speed
typedef Number	MetrPerSec
typedef unsigned short int	Y_N
const Number	PI

Tab. 2. Data types defined by the program

3 CONTROL PROGRAM ENVIRONMENT

The control program consists of (see fig. 3):

- Input program unit.
- Computational program unit (includes data type definition program unit)
- Animation program unit.

The control program was created in *C++* language in Borland Builder development environment with *vcl.h* default library for Microsoft Windows platform [Prata (2001), Prosise (2000)]. The program uses default mathematical libraries of *C* language, for example *math.h* library for goniometric functions implementation [Kent (2004)].

In the program it is possible to add simply the other robot types, interpolation methods and threedimensional trajectories. In the control program they are processed:

- 6-axial anthropomorphic robot and 3-axial robot.
- trajectory consisted of more space vectors.
- orientation change of end effector rotation.

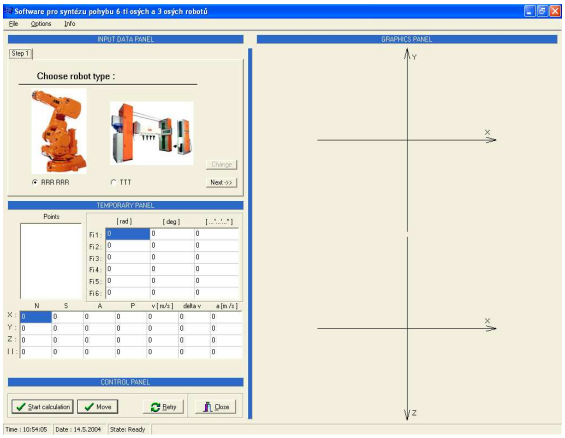

Fig. 3 Control program interface

4 SETTING PARAMETERS POSSIBILITIES

The user can select two robot types in the program: 6-axial anthropomorphic robot with structure of kinematics RRR_RRR or 3-axial robot with structure of kinematics TTT (see fig. 4).

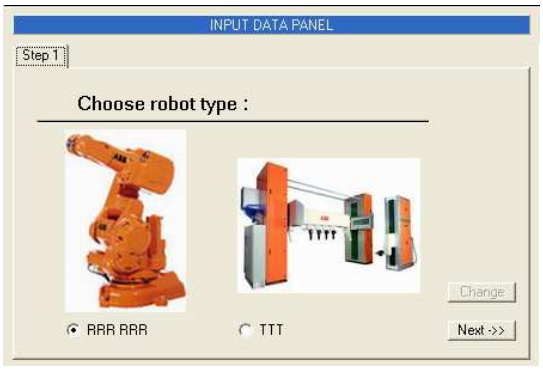

Fig. 4 Robot type selection

Design parameters of robots are set in metres (see fig. 5). In program design parameters of robot ALR–1 are predefined [Bělohoubek (1999)].

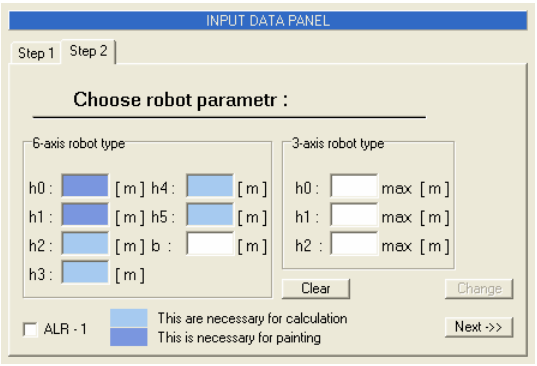

Fig. 5 Robot parameters setting

The trajectory is defined as space curve consisted of vectors (see fig. 6). In each part of the trajectory it is possible to enter different constant Δt of servo-drive timing. It influences the precision of interpolation in the part of trajectory given.

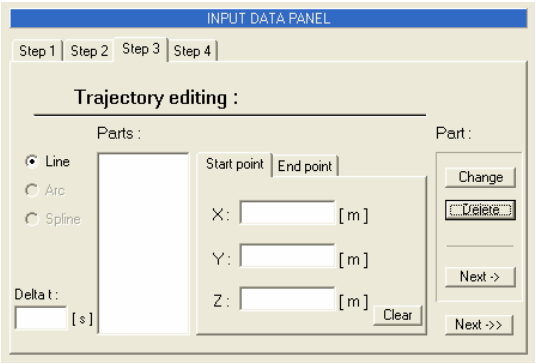

Fig. 6 Trajectory editing

The change of orientation in a linear part of trajectory can be done by rotation around one of axis of robot system of coordinates (see fig. 7) or by rotation around general threedimensional axis (see fig. 8).

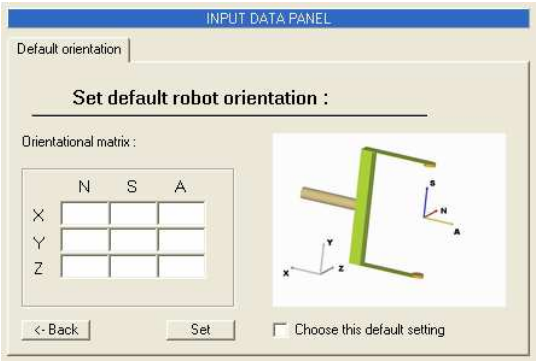

Fig. 7 Robot orientation setting

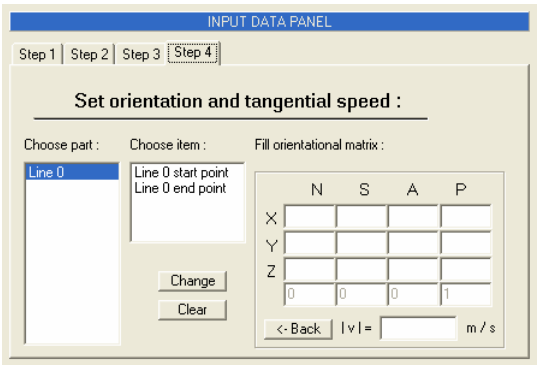

Fig. 8 Trajectory modification editing

5 COMPUTATION AND ANIMATION

After making all input steps the computation is dependent on the choice of design parameters variation (see fig. 10) and on the precision of computation (see fig. 9). It is possible to change both factors of computation in program setting before starting computation and animation (see fig. 11).

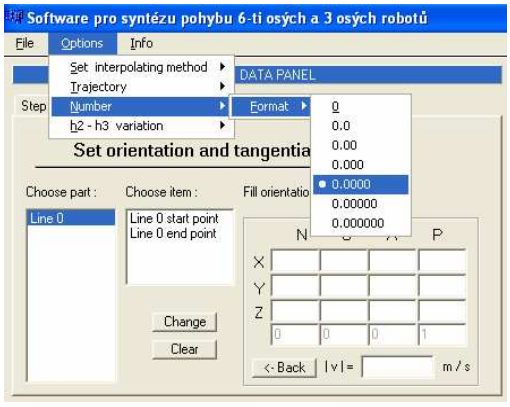

Fig. 9 Computation precision setting

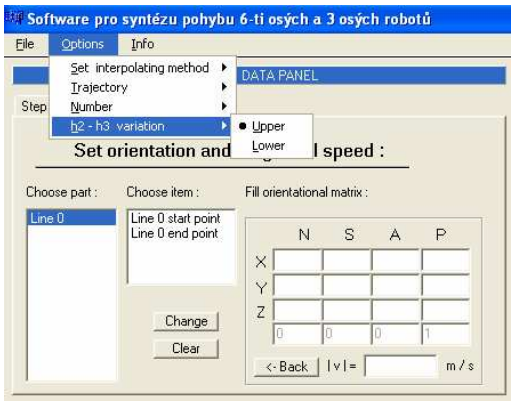

Fig. 10 Design parameters setting

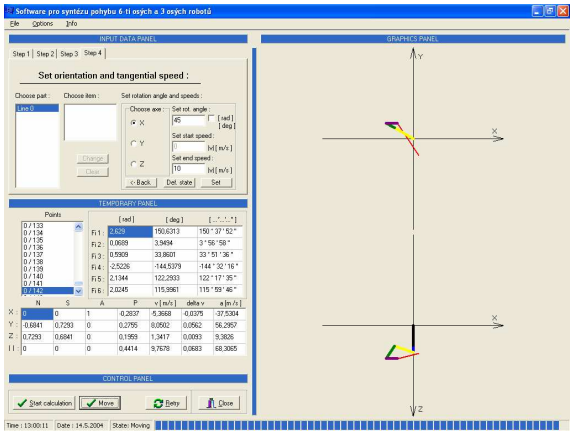

Fig. 11 Computation and animation

6 CONCLUSION

The contribution deals with problems of the industrial robots control on a trajectory given with a view to the 6-axial anthropomorphic robots. Robot design parameters are over possibilities of many existing control programs. Therefore it is very important to target the development of control software that will not be dependent on platform, controlled robot, structure of kinematics, place of using etc.

The most important feature of the created control program is his simple management and modification. With using object oriented technology it is possible simply include a new method or algorithm as independent program unit to the control program.

ACKNOWLEDGMENTS

This work has been supported by the grants:

MSM 6215648904/03 Research design of Mendel University in Brno

MSM 0021630529 Intelligent systems in automation (Research design of Brno University of Technology)

No 102/07/1503 Advanced optimisation of communications systems design by means of neural networks. The Grant Agency of the Czech Republic (GACR)

Refernces

P. Bělohoubek. *Matematický aparát pro řízení 6-ti osých robotů.* Výzkumná zpráva VUT, Brno, 1999.

J. Burkardt. *Geometry: Geometric calculation.* SCS, Florida State University, Florida, USA, 2005.

J. Kent. *C++ Demystified: A self teaching guide.* McGraw-Hill/Osborne, Emeryville, California, USA, 2004.

J. Prosise. *Programování ve Windows pomocí MFC.* Computer Press, Praha, 2000.

J. Schmidtmayer. *Maticový počet a jeho použití v technice.* SNTL, Praha, 1974.

J. Šťastný. Simulation of Control System. In *FARANA, R. & SMUTNÝ, L. Proceedings of XXIII. ASR Seminary '99 "Instruments and Control",* vol. 2, 6 pp., VŠB TU, Ostrava, 1999.

S. Prata. *Mistrovství v C++.* Computer Press, Praha, 2001.