SYMBOLIC COMPUTATION

Christian Blume
Wilfried Jakob

Programming Languages for Industrial Robots

Springer-Verlag

Christian Blume Wilfried Jakob

Programming Languages for Industrial Robots

Springer-Verlag
Berlin Heidelberg NewYork
London Paris Tokyo

Authors

Christian Blume
Gerhart-Hauptmann-Straße 5
D-7505 Ettlingen

Wilfried Jakob
Taunusstraße 20
D-1000 Berlin 41

Translator

Klaus Selke
Department of Electronic Engineering
University of Hull, Cottingham Road
Hull HU6 7RX, United Kingdom

Originally published under the title "Programmiersprachen für Industrieroboter" by Vogel-Verlag, Würzburg (Federal Republic of Germany), © 1983 by Vogel-Verlag, Würzburg
Translation, completely revised and extended, by Springer-Verlag 1986

ISBN 3-540-16319-0 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-16319-0 Springer-Verlag New York Berlin Heidelberg

Library of Congress Cataloging-in-Publication Data. Blume, Christian. Programming languages for industrial robots. (Symbolic computation. Artifical intelligence) Rev. translation of: Programmiersprachen für Industrieroboter. Bibliography: p. Includes index. 1. Robots, Industrial. 2. Programming languages (Electronic computers) I. Jakob, Wilfried. II. Title. III. Series. TS 191.8.B5613 1986 670.42'7 86-26000
ISBN 0-387-16319-0 (U.S.)

Media conversion; printing and binding: Appl, Wemding
2145/3140-543210

SYMBOLIC COMPUTATION

Artificial Intelligence

Springer Series
SYMBOLIC COMPUTATION - *Artificial Intelligence*

N. J. Nilsson: Principles of Artificial Intelligence. XV, 476 pages, 139 figs., 1982

J. H. Siekmann, G. Wrightson (Eds.): Automation of Reasoning 1. Classical Papers on Computational Logic 1957-1966. XXII, 525 pages, 1983.

J. H. Siekmann, G. Wrightson (Eds.): Automation of Reasoning 2. Classical Papers on Computational Logic 1967-1970. XXII, 638 pages, 1983.

L. Bolc (Ed.): The Design of Interpreters, Compilers, and Editors for Augmented Transition Networks. XI, 214 pages, 72 figs., 1983.

R. S. Michalski, J. G. Carbonell, T. M. Mitchell (Eds.): Machine Learning. An Artificial Intelligence Approach. XI, 572 pages, 1984.

L. Bolc (Ed.): Natural Language Communication with Pictorial Information Systems. VII, 327 pages, 67 figs., 1984.

J. W. Lloyd: Foundations of Logic Programming. X, 124 pages, 1984.

A. Bundy (Ed.): Catalogue of Artificial Intelligence Tools. XXV, 150 pages, 1984. Second, revised edition, IV, 168 pages, 1986.

M. M. Botvinnik: Computers in Chess. Solving Inexact Search Problems. With contributions by A. I. Reznitsky, B. M. Stilman, M. A. Tsfasman, A. D. Yudin. Translated from the Russian by A. A. Brown. XIV, 158 pages, 48 figs., 1984.

C. Blume, W. Jakob: Programming Languages for Industrial Robots. XIII, 376, 145 figs., 1986.

Preface

Previous works on industrial robots dealt with "programming" and "programming languages" only in passing; no comparison was made between characteristics of the individual programming languages.

This book, therefore, gives a detailed account of industrial robot programming and its environment. After introducing basic concepts special attention is paid to the language constructs relevant to robot programming. The features of various elements of the languages examined are compared. The languages are based on the following concepts:

SRL - high-level programming language based on AL with PASCAL elements
(University of Karlsruhe, F.R.G.)

PASRO - integrated into PASCAL, based on the geometrical data types of SRL
(I.I.-BIOMATIC Informatics Institute, Freiburg, F.R.G.)

AL - derived from the high-level programming language ALGOL
(Stanford University, U.S.A., and University of Karlsruhe, F.R.G.)

AML - high-level programming language, influenced by PL/1
(IBM, U.S.A.)

VAL - language specifically developed for robots
(Unimation, U.S.A.)

HELP - mixture of high-level language elements and robot language elements and real-time processing
(DEA, Italy)

SIGLA - a simple machine language
(Olivetti, Italy)

ROBEX - based on NC programming
(Technical College (RWTH), Aachen, F.R.G.)

RAIL - high-level programming language for industrial robots with elements for graphic processing
(Automatix, U.S.A.)

IRDATA - general software interface between programming and robot controller (Association of German Engineers (VDI), F.R.G.)

Elements of individual languages are given as examples in the discussion on general concepts - where the relevant language relates to the problem discussed. Our knowhow ist based partly on own experience or implementations (SRL, PASRO, AL, VAL, IRDATA) and partly on company data. Since company data does not always give the necessary detailed information, it is possible that certain elements are not discussed although they are part of that language.

This book, however, is not intended as an aid in learning any of the languages introduced, but rather to give a better understanding of the special problems and techniques of robot programming and to instil in the programmer's mind the need for exact and methodical work. Designers and users of robot systems are usually familiar with the peculiarities of robot programming but not with the systematic construction and formulation of algorithms. On the other hand, the computer scientist is conversant with the latter but not with the particular problems and requirements of robot technology. The book is, therefore, meant as an interface between the various disciplines so that both - robot technologist and computer scientist - can communicate at the same level using the same terminology.

Since we are dealing with languages, the presentation of computer science problems takes up more space than the basic robot technology necessary for using industrial robots. However, we start out by explaining important robot-technology terms for the computer scientist. When discussing data in particular, the special requirements of movement programming are presented and new data types are derived from the geometrical presentation.

The German edition of this book was first published in 1983 and was really intended as a manuscript for a lecture on "Programming of Robots" by author Christian Blume. This edition has been completely revised and now also includes the languages SRL, PASRO, and AML as well as the IRDATA software interface and elements for implicit programming such as RODABAS robot database. The reason for this is that the systems discussed have either undergone further technological development or will have an impact on the market, or both.

We are grateful to the following companies and institutes for documentation and information given: To IBM in particular for the demonstration of AML/E in Munich. We thank the German office of Unimation for the detailed explanations of VAL. To DEA and Olivetti go our thanks for kindly letting us have descriptions of the HELP and SIGLA languages. We thank the Machine Tool Laboratory (WZL) of the Technical College Aachen for so readily answering all our intricate questions regarding the ROBEX language developed there. In connection with instructions to a visual system we introduce an extract of RAIL; we are grateful to Automation for sending us the relevant documentation.

We thank all colleagues of the IRDATA working group of the VDI for closely cooperating and allowing us to publish the results.

In connection with the Karlsruhe AL implementation we are grateful for the kind support provided by Stanford University and particularly by Dr. Tom Binford.

For the close cooperation during the PASRO implementation we thank I. I.-BIOMATIC Informatics Institute GmbH and their manager, G. R. Koch.

We are also greatly obliged to K. K. W. Selke, electronics engineer at the University of Hull, U. K., for his translation and for accommodating all our requests for alterations.

And last but not least, we are very much indebted to the Head of the Department of Process Computer Technique at the University of Karlsruhe, Prof. Dr.-Ing. U. Rembold, and PSI GmbH, Berlin, for their kind support of our work. Our thanks go to all colleagues and friends who helped us with their advice and their recommendations.

C. Blume
W. Jakob

Table of Contents

1 Introduction

By now industrial robots are used in many production and manufacturing processes such as handling, welding, or spray-painting. Some of these jobs will continue to be manageable in the future with simple programming methods, e.g., adjustment or teach-in methods. For assembly and more complex handling operations, however, more highly developed programming methods have become necessary. Therefore, work is in progress, on an international level, to develop new programming languages and systems for industrial robots. The emphasis is on integrating sensors into the software of the robot controller, on including data from a CAD system, and on making available programming tools which are fairly simple - in the eyes of the user - but nonetheless provide for flexible programming. This can only be achieved by using industrial robot programming languages which have - contrary to the teach-in methods - the advantages of auto-documentation, communicativeness, being open to corrections, and offering offline programming. The languages discussed here meet the requirements in various ways and more or less completely. The system layouts, their use and the implementation of the individual languages differ greatly and sometimes can hardly be compared.

SRL (Structured Robot Language), with the IRDATA interface (Industrial Robot DATA), was defined by the authors and implemented at the University of Karlsruhe. The procedure and hardware used are described in Chap. 8.

The teaching and demonstration system PASRO (PAScal for RObots) was designed by author Christian Blume for BIOMATIK and has been implemented on a number of computers and under various operating systems and for two demonstration robots; see Chap. 8 as well. PASRO includes PASCAL completely so that for PASCAL no separate comparison is made as in the first German edition. PASRO could be implemented with the aid of MODULA, ADA or (especially for DEC computers) with MicroPower/Pascal extending at the same time it by multi-tasking elements, i.e., the parallel running of programs or subprograms.

AL (Assembly Language) was already developed by Stanford University, in the 1970s and implemented on a computer and robot configuration not suitable for industrial applications. Under a technology transfer in the early 1980s a new implementation was undertaken by the University of Karlsruhe and a number of improvements were made. The basic Karlsruhe implementation was carried out in PASCAL on a PDP 11/34 and was used to program a PUMA 500 robot by Unimation. AL has served as basis for a number of high-level robot programming languages in the United States, Europe, and Japan.

As nearly as in the 1970s, IBM announced a very advanced implicit programming system, AUTOPASS (AUTOmated PArts Assembly System). However, it is not known to what extent this has also been implemented. In the 1980s IBM introduced their own robots, which can be programmed with AML (A Manufacturing

Language), a high-level programming language. AML has a few peculiarities not usual in a robot language and not included in any of the languages discussed in this book. AML is formula-oriented, i.e., each instruction produces a value which may be reused. AML instructions, therefore, may be combined with each other in almost any way (orthogonality), and although this provides for flexible programming it has inherent risks because it permits a vague programming style. AML may also be used interpretatively, which is somewhat unusual for a language with PASCAL-like block structure and strings. On the other hand, the syntax for movement instructions and other commands not used for program structuring are kept very simple, which is the reason why hardly any AML syntax diagrams have been included. AML comprises more than 140 functions, and the demarcation between specific applications subprograms and those of the "system core" are fairly flexible.

There is a simple version of AML available, AML/E (Entry), which has been implemented on an IBM PC. It permits a teach-in by using terminals and a simple graphic presentation of the programmed position in a two-dimensional display of the IBM SCARA robot's working space (see also Chap. 8).

HELP is used by Messrs. DEA (Digital Electronic Automation) to program the PRAGMA robot. On the one hand it comprises robot and specific system functions while, on the other, it also has elements of high level-languages and real-time programming. The PRAGMA robot moves in Cartesian coordination, which means the programmer's positioning instructions can be directly processed by the controller. It is, nevertheless, surprising that no provision is made for data to indicate position and orientation of the robot. A new version is currently under development, and the hardware has also been modified.

Stanford University's close proximity and personal relationships have influenced the development of Unimation's VAL (Variable Assembly Language), a language which initially was determined by the robot's functions. As a consequence the first version of this language had highly sophisticated concepts like frame variables but was severely handicapped by missing enquiry facilities for coordinate values. A new version, VAL-II, consists of VAL-I - with minor limitations - which has been described in the first edition of this book. The present edition includes references to VAL-II elements but no syntax diagrams.

Olivetti's SIGLA (SIGMA LAnguage) is for programming two-armed SIGMA portal robots. These robots are driven by stepping motors, the values for which must be explicitly stated by the programmer. Since the syntax of SIGLA consists only of the statement of a two-lettered keyword and the enumeration of parameters, it is counted among the simple programming languages. Nonetheless, SIGLA can do more than solve simple problems since it is capable of parallel processing of program parts (tasks) and sensor enquiries, although SIGLA programs are not very clear or easy to read.

Besides SRL, ROBEX (ROBot EXapt) of Aachen Technical College is also a German development. It is based on EXAPT (EXtended APT), which is a technological extension of APT (Automatically Programmed Tools). Originally, ROBEX was implemented with severe limitations as far as "standard" EDP is concerned, e.g., lack of variables and arithmetics.

The new version for microcomputers, ROBEX-M, has partly overcome these handicaps, although the restrictive syntax (e.g., 6 characters for a keyword) has been

maintained and the actual NC parts for geometric path calculation are still missing. Current development trends bring ROBEX and SRL closer together since SRL, in turn, is being extended to an implicit system.

In connection with programming algorithms for picture processing on the control computer, Messrs. Automatix have developed RAIL. RAIL is a high-level programming language with structuring facilities similar to those of PASCAL; it is not covered in detail.

Although the IRDATA code (Industrial Robot Data) was designed as an interface between programming and controlling - and not as a new language - it is discussed here. The elements of the robot programming language are transferred into IRDATA code by the language converter and transmitted to the control unit. IRDATA is a logical interface, i.e., control functions and their parameters are stated but not the hardware interface or the transfer protocol. The main advantage of IRDATA is that in future the control unit of a robot may be linked with almost any programming system which can generate IRDATA code.

The comparison of the above languages includes detailed tables as far as this is at all possible with the variety of language elements. For quick reference we conclude with a summary of the most important features of the languages in Appendix L.

Although most of the languages introduced here have no rules for formatting a program text, for reasons of clarity the examples will be indented in accordance with the program structure and reserved words and predefined identifiers will be written in uppercase letters.

2 Fundamentals

In this chapter, basic terms and concepts of robot technology and computer sciencewill be explained. These also include terms already familiar to programmers, such as variable, stack and subroutine, so that engineers with backgrounds in automatic control technology or manufacturing may find an easier introduction to the subject. Also, an explanation of these fundamental concepts is very useful in helping to understand different robot languages.

2.1 Terminology

Since most of the terminology introduced here is fundamental, it will not be formally defined, but just described verbally and explained by simple examples.

Unfortunately, there is no exact definition for the most important term **industrial robot**. Even the name industrial robot, chosen to distinguish it from the robot of science fiction literature, is not universally valid. Terms like *robot, manipulator* or *universal transfer device* are used, and one author even calls it a *robot manipulator.* The VDI (Verein Deutscher Ingenieure, the association of German engineers) has defined an industrial robot as follows (VDI Entwurf 2860 Gründruck):

Industrial robots are universally applicable devices with several axes of motion, which may be freely programmed (i.e., altered without mechanical interference) with respect to their sequence of motion and angles or positions and may be guided by sensors - if applicable. They are equipped with grippers, tools or other means of production and are able to perform manipulation and/or production tasks.

Hence it is important to be able to program the device to perform movements in three-dimensional space with motors actuating axes of motion. Programming without mechanical interference is taken for granted saying in computer science, but there are some simple *pick-and-place devices* which are programmed by repositioning mechanical endstops. An industrial robot may be equipped with sensors (effectively as peripherals). **Sensors** are devices which are able to register physical signals like light waves, pressure or joint positions, and can convert these to electrical signals (analogue or digital) and pass them on to the computer controlling the robot. This sensory information may then be analyzed and the program flow modified accordingly. At the end of the robot, a *gripper, tool* or other *means of production* is attached; these are collectively called **effectors**. These effectors also have to be controlled and may be equipped with sensors themselves. Figure 2.1 shows three typical industrial robots.