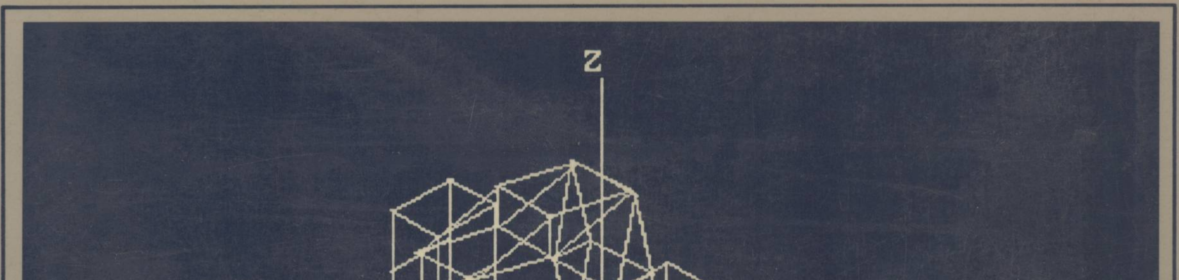
DISK INCLUDED

ROBOTIC MANIPULATION

PROGRAMMING AND SIMULATION STUDIES WITH SOFTWARE



obert J. Schilling / Robert B. White

TP242
S334-2

9261890

ROBOTIC MANIPULATION

Programming and Simulation Studies

E9261890

ROBERT J. SCHILLING
Associate Professor
Clarkson University

ROBERT B. WHITE
Manhattan, Kansas

华南理工大学
图书馆藏书

In Memory of

Albert, Amanda, and Gretta

© 1990 by Prentice-Hall, Inc.
A Division of Simon & Schuster
Englewood Cliffs, New Jersey 07632

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

0-13-344441-4

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

PREFACE

Robotics is an application-oriented field of study. Many of the concepts which form the foundation of robotics are amenable to testing and verification in a laboratory setting. Practical *hands-on* experience of this nature provides a good test of the student's understanding of the underlying theoretical concepts, some might say the ultimate test. The focus of this book is the investigation of robotic manipulation techniques through programming and simulation studies. This book is designed as a supplement to the textbook *Fundamentals of Robotics: Analysis and Control* by Schilling (1990) which includes a more complete and formal presentation of topics presented here.

The most important elements of this book are the programming and simulation projects that appear at the end of each chapter. Many of the projects are designed around a robotic workstation that includes a personal computer, an educational robot, and an educational computer vision system. The robot that is featured is a popular five-axis articulated-coordinate table-top robot called the Rhino XR-3 which is manufactured by Rhino Robots, Inc. Additional articulated robots are also used in some of the simulation studies. The computer vision system is a MetraByte MV2 frame grabber with a Javelin JE2362 CCD camera, an inexpensive system supplied by MetraByte Corp. Other binary vision systems can also be used.

Although most of the programming and simulation projects are designed around a robotic workstation, *all* of the projects can be completed and tested without the benefit of special laboratory facilities. What is needed is an IBM PC/XT/AT compatible computer with a graphics adaptor card, plus the robot simulator software on the distribution disk included with this book. The graphics robot simulator program included on the disk is described in detail in Chapter 2. With the simulator installed, users can develop and test robot control programs off line without a physical robot present using the language of their choice. The simulator is a terminate and stay resident program that resides in the background and monitors the serial COM1 device that would normally be connected to the robot controller. It intercepts commands destined for the robot controller and responds to them as if it were the controller. The status of the simulated robot is available through a 3D graphics display and also through a one-line text window at the top of the screen. Simulator options can be activated through the keyboard or from within a user program. The physical characteristics of the robot to be simulated are contained in a data file that is specified when the simulator is installed. A second data file is used to describe the workcell or environment within which the robot is to operate. The workcell data file, which can be customized by the user, includes an overhead camera and a set of blocks which can be viewed by the camera and manipulated by the robot.

The material in this book is organized into eight chapters. Chapter 1 is an introduction which summarizes the characteristics of the hardware to be used, and briefly outlines the sequence of programming and simulation projects. There are a total of twelve projects summarized in Table 0.1. Chapter 2 is devoted to a discussion of the graphics robot simulator. The modes of operation and simulator options are discussed, and examples of using the simulator with several popular programming languages are presented. Chapter 3 focuses on direct kinematics, the problem of determining the position and orientation of the robot tool or hand given values for the joint angles. This lays the foundation for Chapter 4 which is concerned with inverse kinematics, the problem of finding suitable values for the joint angles given a desired position and orientation for the tool. Complete kinematic solutions are presented for the Rhino XR-3 robot in Chapters 3 and 4. Chapter 5 is concerned with trajectory planning including pick-and-place operations, straight-line motion, and the use of part feeders. Control of robots through resolved-motion rate control methods is the topic of Chapter 6. Also included in Chapter 6 is an analysis of robot arm stiffness and tool-tip deflections generated by contact with the environment. More sophisticated robot control techniques based on complete dynamic models and torque control methods are the subject of Chapter 7 which features a two-axis planar articulated robot. Chapter 8 is devoted to robot vision. A number of analysis techniques which allow the user to determine the identity, position, and orientation of objects in the workspace are investigated.

Chapter	Projects
2	GRASP, MANUAL, HOME
3	DIRKIN
4	INVKIN
5	PICK, THREAD
6	RATE, STIFF
7	TORQUE
8	IMAGE, VPICK

Table 0.1: Programming and Simulation Projects

We would like to express our gratitude to a number of institutions and individuals whose support helped make this book possible. The programming and simulation projects were developed and tested in the Robotics Instructional Laboratory of Clarkson University. The work on the graphics robot simulator by R. White was sponsored by a grant from the Westinghouse Foundation. R. Nelson and D. Delorie of the Educational Computing Service at Clarkson developed the low-level graphics routines that are used in the simulator. Finally, we thank the students in EE551 and EE552 who helped with early versions of the projects through their suggestions and often inventive solutions.

Robert J. Schilling
Potsdam, NY

Robert B. White
Manhattan, Kansas

此为试读，需要完整PDF请访问：www.ertongbook.com

Trademarks

All brand and product names are trademarks or registered trademarks of their respective holders. IBM PC/XT/AT and PS/2 are trademarks of International Business Machines Corp.; Turbo Pascal and Turbo C are trademarks of Borland International; Rhino XR-3 is a trademark of Rhino Robots, Inc.; MetraByte is a trademark of MetraByte Corp.; MS-DOS, Microsoft BASIC, GW-BASIC and BASICA are trademarks of Microsoft Corp.; Hercules is a trademark of Hercules Computer Technology Corp.; Intelledex 660 is a trademark of Intelledex, Inc.; Adept One is a trademark of Adept Technologies, Inc.

Limited Warranty, Disclaimer

All software supplied on the distribution disk is protected by a copyright and is intended for exclusive use by the purchaser. With the exception of an archive copy, the disk may not be duplicated, and the software may not be resold, in whole or in part, or sold as part of another product. The physical media on the distribution disk is warranted to be free of defects for a period of 60 days from the time of purchase. All other warranties, expressed or implied, are disclaimed including, but not limited to, fitness of the software for any particular application, use, or purpose.

CONTENTS

此为试读，需要完整PDF请访问：www.ertongbook.com

Chapter 1

Introduction

The purpose of this book and its accompanying software is to provide students with an opportunity to obtain some *hands-on* experience with robotic manipulation. This book is designed as a supplement for the textbook, *Fundamentals of Robotics: Analysis and Control* (Schilling, 1990). Many important theoretical concepts in the field of robotics are amenable to programming and simulation studies. These concepts are presented here, in concise form, in order to lay a foundation for a series of robotic manipulation laboratory projects. Most of the laboratory projects described in this book are designed around a popular family of educational table-top robots, the XR-3 series of robotic arms manufactured by Rhino Robots, Inc. To make effective use of this book, the reader does not have to have direct access to a Rhino XR-3 robot or any other physical robot. Instead, a three-dimensional graphics robot simulator program (White et al., 1989) is supplied on a distribution disk. The simulator runs on IBM PC/XT/AT personal computers and true compatibles. With the use of the simulator, students can develop and test robot control programs off line without a robot present using the language of their choice. The use of the simulator is illustrated with several popular programming languages including BASIC, Pascal, and C.

1.1 An Educational Robot: Rhino XR-3

In recent years a number of educational table-top robots have appeared in the market place. Many of these robots rely on open-loop position control with small stepper motors, while others employ closed-loop control with feedback from sensors. Knowing the position of the robot from sensors is important because otherwise it is difficult, if not impossible, to perform *reliable* manipulations of objects in the environment. Disturbances, such as a heavy payload or collisions with obstacles, can cause an error in the robot position which will go undetected in the absence of sensor feedback. One popular family of educational robots which employs position feedback using optical encoders is the XR Series of robotic arms manufactured by Rhino Robots, Inc. The most recent member of the series, the Rhino XR-3 robot, is shown in Figure 1.1.

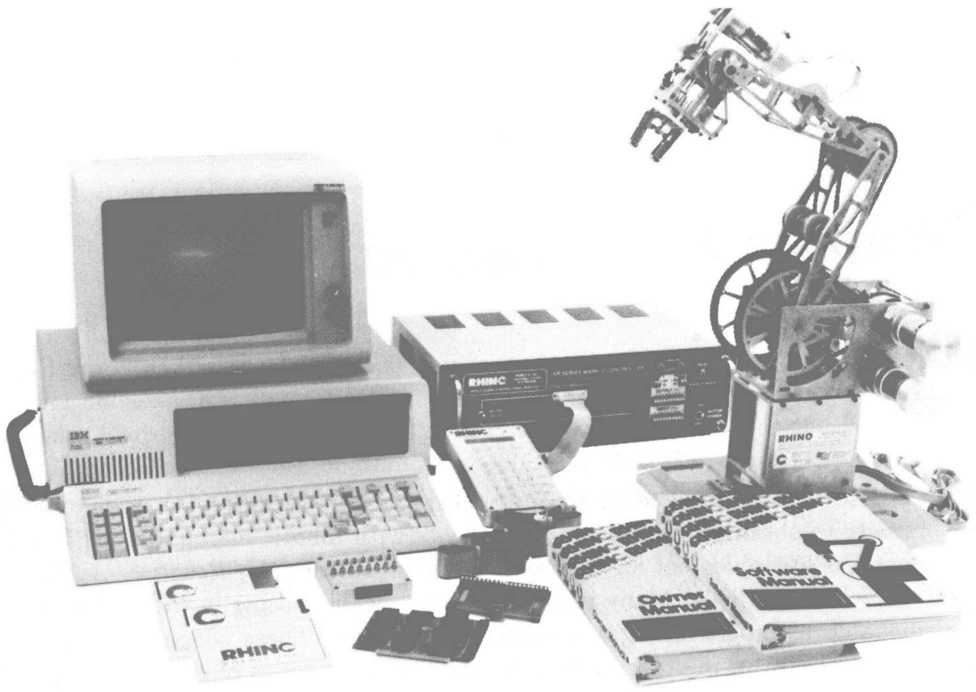


Figure 1.1: Rhino XR-3 Robotic System (Courtesy of Rhino Robots, Inc., Champaign, IL, Robotic arm developed and manufactured in the U.S.A.)

The Rhino XR-3 robot is a five-axis articulated-coordinate robotic arm with joints at the base, shoulder, elbow, tool pitch, and tool roll. It is an electric-drive robot that employs permanent magnet DC servomotors with incremental position encoders. One of the useful educational features of this robot is its *open design* which allows students to examine its mechanical construction. In addition, the architecture of the robot controller is also open which means that users can modify or enhance it in various ways such as adding new commands to the controller firmware (Schilling and Williams, 1988).

The basic specifications of the Rhino XR-3 are summarized in Table 1.1. The load carrying capacity, maximum tool-tip speed, and repeatability are rough estimates based on use of this robot in an instructional laboratory. The reach and stroke specifications are measured with respect to the tool tip assuming the standard 1.5X fingers are used for the tool.

An important specification for programming purposes is the load shaft precision for each of the five axes. Each joint is driven by a 20 volt DC servo motor that has an incremental encoder attached to the high speed shaft. The encoder resolves the high speed position to 60 degrees. Each motor has a built-in gear head with a turns ratio of either 66.1:1 or 96:1. Finally each joint has its own set of sprockets and chains, which then determine the joint angle precision. The resulting precision in degrees/count for each joint is summarized in Table 1.2.

Characteristic	Value	Units
number of axes	5	—
load carrying capacity	1	kg
maximum tool tip speed	25	cm/sec
horizontal reach/stroke	62.23	cm
vertical reach/stroke	88.27	cm
tool pitch range	270	deg
tool roll range	infinite	deg
repeatability	±0.1	cm

Table 1.1: Rhino XR-3 Specifications

Motor	Joint	Precision
F	base	0.2269
E	shoulder	0.1135
D	elbow	0.1135
C	tool pitch	0.1135
B	tool roll	0.3125

Table 1.2: Joint Precision of the Rhino XR-3 Robot (degrees/count)

1.2 Programming Commands

The Rhino XR-3 robot comes with an eight-channel controller which interfaces to a host microcomputer through a serial interface. Five of the channels are used to control the position of the five axes of the robot, one channel is used to open and close the tool, and the remaining two channels are spare channels that can be used to control optional peripheral items of equipment such as an indexing carousel, a conveyor transport, or an xy table. The interface between the microcomputer and the Rhino XR Series controller is an RS-232C serial interface requiring cable connections to pins 2,3,7. Microcomputer communications can be adjusted to transmission rates of 300, 1200 or 9600 bits/second with a BAUD rate switch inside the controller. The firmware supplied with the controller is designed to execute a set of basic robot control commands. Complex manipulation tasks are then built up using appropriate sequences of these commands. The basic controller commands are summarized in Table 1.3. All commands should be terminated with a carriage return character. Here a stands for a letter in the range A-H indicating a channel or motor letter, n stands for an integer in the range −95 to 95 specifying encoder counts, and m is an integer in the range 1 to 8 specifying an output line.

Command	Syntax	Description
Move	an	Turn motor a a total of n counts
Question	a?	Read the error count for motor a
Stop	aX	Zero the error count for motor a
Inquiry	IA	Read limit switches C-H
	JA	Read limit switches A-B, input lines 1-4
	KA	Read input lines 5-8
Auxiliary	LA	Turn Aux #1 port ON
	MA	Turn Aux #1 port OFF
	NA	Turn Aux #2 port ON
	OA	Turn Aux #2 port OFF
Output	PmA	Set output line m HIGH
	RmA	Set output line m LOW
Reset	QA	Reset controller

Table 1.3: Basic Rhino Controller Commands

The syntax of the Rhino controller commands in Table 1.3 specifies the ASCII character strings that must be sent to the robot controller through a serial interface. This can be done with any programming language that supports communication with a serial port. For certain commands which return status information, a one-byte response must be subsequently read from the serial port. Specific examples of sending robot controller commands and receiving responses using BASIC, Pascal, and C are presented in Chapter 2.

1.2.1 Move command: *an*

The *move* command consists of sending the controller a character string starting with a letter which specifies the motor or channel (A through H), followed by a + or − sign, followed by a one or two digit number in the range 0 to 95. Motor A activates the tool, motors B through F correspond to the five joints as specified in Table 1.2, and motors G and H are spare channels for peripherals. For positive numbers the + sign is optional. Each motor has an 8-bit *error counter* associated with it. Whenever the count in the error counter is nonzero, the controller activates the motor and turns it in the direction which reduces the magnitude of the error. The move command loads this error counter by algebraically adding the number n to the current contents of the error counter. For example the command "D−45" turns the elbow motor −45 counts which, from Table 1.2, corresponds to −5.1 degrees. Note that since the maximum number of counts in the move command is $n = 95$, long moves have to be implemented at the software level on the host computer as a sequence of short moves.

1.2.2 Question command: a?

The *question* command is implemented as a two-step command. The first step sends a motor letter in the range A through H followed by the question mark "?". The controller then places a byte in the serial port buffer which indicates the current contents of the error counter of the specified motor. Of course, if the motor is turning (nonzero error), the value returned for the error count is valid only for a short time. The error count itself is not sent back to the host computer. Instead, the value returned is 32 plus the *magnitude* of the error count.

$$x = |\text{error}| + 32 \tag{1.1}$$

Thus the values sent are in the range $32 \leq x \leq 127$. The controller never sends an ASCII code in the range 0 to 31 back to the computer. These codes are control codes such as line feed, form feed, and bell, which can be misinterpreted by some computer terminals. For *all* commands that return status information, the single-byte responses always have 32 added to them before they are transmitted.

Depending on the speed of the computer used, it may be necessary to introduce a small *delay* between sending the question command and reading the response. This is because the Rhino XR-3 robot controller has no handshaking, so the user may attempt to read the response before it is ready. This is the case with *all* controller commands which return responses.

1.2.3 Stop command: aX

The *stop* command consists of sending a motor letter to the serial port followed by the letter "X". This zeros the error counter for the specified motor. Hence a turning motor can be stopped with this command. This is useful, for example, when sending commands to close the tool. By repeatedly invoking the question command, one can determine whether or not the tool has grasped an object because the magnitude of the error count will no longer continue to decrease. At this point the stop command can be sent to the tool to cancel the motor *stall* condition. When a motor stalls, it draws increased current and if the problem is left uncorrected the drive circuit can become overheated to the point where it sustains permanent damage. A stall condition can also be cleared with a move command.

1.2.4 Inquiry commands: I,J,K

The *inquiry* commands are useful for resetting the robot to a known position at the beginning of a sequence of manipulations. They can also be used to read input lines which monitor the status of workcell equipment. The first inquiry command, called the I-inquiry command, is invoked by sending the letters "IA" to the serial port. The controller then sends back a single-byte response which specifies the status (open or closed) of each of the six limit switches for motor channels C-H. The value returned, x, is computed as follows.

$$x = 32 + s_C + 2s_D + 4s_E + 8s_F + 16s_G + 32s_H \tag{1.2}$$

Here s_a denotes the status of the limit switch for channel a with $s_a = 0$ if switch a is closed and $s_a = 1$ if switch a is open. Consequently, if $x - 32$ is expressed in base 2, the result is as follows.

$$(x - 32)_2 = 00 s_H s_G s_F s_E s_D s_C \tag{1.3}$$

As with the question command, a small delay may have to be inserted before the response is read. The second step of the I-inquiry command reads the response and subtracts 32 from it. This yields a number in the range from 0 to 63. By masking off the 6 least significant bits and examining their values, the states (open or closed) of the six limit switches can be determined. A 63 corresponds to all six switches open, which is the normal state. When a switch goes from open to closed, this corresponds to the joint being at its *hard home* position. The appropriate masks for the limit switches of the Rhino XR-3 robot are summarized in Table 1.4.

Status	Bit	Mask
limit switch C	0	1
limit switch D	1	2
limit switch E	2	4
limit switch F	3	8
limit switch G	4	16
limit switch H	5	32

Table 1.4: Status Returned by I-Inquiry Command

此为试读,需要完整PDF请访问: www.ertongbook.com