ROBOTIC SYSTEM
WITH COMPUTER VISION
A thesis submitted in partial satisfaction of the requirement for the degree of Master of Science in Electrical and Computer Engineering by
Soo-Man Lee

May 1985
The Thesis of Soo-Man Lee is approved:

Professor Yuh Sun

Professor V. Anderson

Professor Robert Y. Wong, Chairman

California State University, Northridge
To my Eunjin and parents
for their love and patience
# TABLE OF CONTENTS

DEDICATION .......................................................... iii
LIST OF TABLES ...................................................... vi
LIST OF FIGURES .................................................... vii
ABSTRACT .............................................................. ix

CHAPTER

1. INTRODUCTION .................................................... 1
2. ROBOTIC ARM WITH COMPUTER VISION SYSTEM ............. 3
3. VIDEO PROCESSING SYSTEM ................................. 6
   3.1 Video Camera with Lens ................................. 6
   3.2 Video Processing Unit (VPU) ......................... 6
   3.3 Procedure for Detection of Object .................. 14
4. ROBOTIC SYSTEM ............................................... 19
   4.1 Introduction ............................................. 19
   4.2 TeachMover Performance Characteristics .......... 19
   4.3 How the Motors Operate ............................... 21
   4.4 Electronic Circuitry and Interface ............... 25
   4.5 Operation from a Host Computer,
       Apple IIe ............................................. 27
   4.6 Arm Initialization and Calibration ................ 38
   4.7 Coordinate Conversions ............................... 42
   4.8 Procedure for Robot Arm Movement ............... 67
5. APPLE IIe COMMUNICATION TO VPU AND
   TEACHMOVER ............................................... 73
   5.1 Introduction ............................................. 73
   5.2 Hardware Needed ....................................... 73
   5.3 Hardware Setup for VPU ............................. 73
   5.4 Hardware Setup for TeachMover .................... 74
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Serial I/O Pin Assignments</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Switch Settings</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>Motor Steps and Joint Rotations</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>Baud Rate Selection</td>
<td>33</td>
</tr>
<tr>
<td>5</td>
<td>Control Lines</td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>Conversion Factors Between Motor Steps and Revolute Joint Angles</td>
<td>44</td>
</tr>
<tr>
<td>7</td>
<td>Lengths of TeachMover Arm Members</td>
<td>47</td>
</tr>
<tr>
<td>8</td>
<td>Summary of Forward Solution</td>
<td>53</td>
</tr>
<tr>
<td>9</td>
<td>Summary of Backward Solution</td>
<td>65</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>System Structure Configuration</td>
<td>4</td>
</tr>
<tr>
<td>2.</td>
<td>System Arrangement</td>
<td>5</td>
</tr>
<tr>
<td>3.</td>
<td>VPU System Interconnect Diagram</td>
<td>11</td>
</tr>
<tr>
<td>4.</td>
<td>Major Structural Components</td>
<td>20</td>
</tr>
<tr>
<td>5.</td>
<td>Operating Envelope of the TeachMover Arm</td>
<td>22</td>
</tr>
<tr>
<td>6.</td>
<td>The Wrist Joint</td>
<td>23</td>
</tr>
<tr>
<td>7.</td>
<td>Block Diagram of TeachMover Computer and Electronics</td>
<td>28</td>
</tr>
<tr>
<td>8.</td>
<td>Connecting the TeachMover to Apple IIe</td>
<td>30</td>
</tr>
<tr>
<td>9.</td>
<td>Pin Numbering for Serial Port Connectors</td>
<td>31</td>
</tr>
<tr>
<td>10.</td>
<td>Apple IIe-to-TeachMover Serial Connection</td>
<td>32</td>
</tr>
<tr>
<td>11.</td>
<td>Switches for Selecting Serial Transmission Rate</td>
<td>33</td>
</tr>
<tr>
<td>12.</td>
<td>Location of Jumper Connections for Serial Operations</td>
<td>37</td>
</tr>
<tr>
<td>13.</td>
<td>Initialization and Cartesian Paper</td>
<td>41</td>
</tr>
<tr>
<td>14.</td>
<td>Kinematic Model of the TeachMover Arm</td>
<td>45</td>
</tr>
<tr>
<td>15.</td>
<td>Definition of Roll and Pitch Angles</td>
<td>47</td>
</tr>
<tr>
<td>16.</td>
<td>Basic Trigonometric Relationships</td>
<td>49</td>
</tr>
<tr>
<td>17.</td>
<td>Side View of Kinematic Model</td>
<td>50</td>
</tr>
<tr>
<td>18.</td>
<td>Top View of Kinematic Model</td>
<td>52</td>
</tr>
<tr>
<td>19.</td>
<td>Different Hand Orientations</td>
<td>55</td>
</tr>
<tr>
<td>Number</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>20</td>
<td>Top View of Arm with 90 Degree Pitch</td>
<td>57</td>
</tr>
<tr>
<td>21</td>
<td>Top View of Arm</td>
<td>58</td>
</tr>
<tr>
<td>22</td>
<td>Side View of Hand Triangle in Kinematic Model</td>
<td>58</td>
</tr>
<tr>
<td>23</td>
<td>Shoulder-Elbow-Wrist Triangle</td>
<td>60</td>
</tr>
<tr>
<td>24</td>
<td>Simplified Triangle</td>
<td>62</td>
</tr>
<tr>
<td>25</td>
<td>Variation of Hand Length with Hand Opening</td>
<td>66</td>
</tr>
<tr>
<td>26</td>
<td>Conversion of Pixels to Cartesian Coordinate</td>
<td>68</td>
</tr>
<tr>
<td>27</td>
<td>The Conversion from the Cartesian on the Table Top to the Real Cartesian Coordinates for Robot Arm</td>
<td>71</td>
</tr>
</tbody>
</table>
ABSTRACT

ROBOTIC SYSTEM WITH COMPUTER VISION

by

Soo-Man Lee

Master of Science in Electrical and Computer Engineering

This thesis is intended to study the techniques and to solve the problems in the application of video processing to an automatic robotic system. The contents of this thesis are:

a. Description and explanation of the Video Processing Unit, how it is integrated with a television camera, television monitor, and a system controller, Apple IIe.

b. Description and explanation of the Five-Axis Robot, TeachMover, and how it is operated by the system controller, the Apple IIe.

c. Connection and operation of the video camera, video monitor, Video Processing Unit, robotic arm, and the controller, the Apple IIe.
d. Flow chart of controlling program for Apple IIe.
e. Presentation of experimental program.

The basic components of the system are an Apple IIe personal computer with two serial ports, a Video Processing Unit, television camera, television monitor, lighting system, and the TeachMover Robotic Arm.

The system was designed and developed and software programs written. Test results indicated the system operated properly and its performance satisfied the design objectives.
Chapter 1
INTRODUCTION

Pattern recognition and image processing techniques have been developed and applied to automatic visual measurement and inspection. These techniques are used to detect an object and to determine the object's location, size and shape. Industrial automation for assembly, automatic alignment of the assembly for testing and component recognition have demanded the development of a human-like robot. Including the benefit for increasing productivity or improving the quality of life, the life of workers now doing repetitive and sometimes hazardous tasks would be taken care of by using sophisticated robots. Furthermore, using computer vision techniques, the positions and orientations of an object within the field-of-view of the system can be determined.

This thesis involves the design and development of an automated robotic system with computer vision using a Video Processing Unit. The system uses an Apple IIe microcomputer to process data transmitted from the Video Processing Unit (VPU) and sends the command to the robot arm, TeachMover.

First, the data sent from the VPU is processed to analyze where the object is located, its size and orientation.
Second, a set of coordinates and signals were generated and sent to the TeachMover to reach the object.

Finally, the robot arm, moving to that object grasps it from the proper direction and places the object as required by the operations.

Algorithms were developed to program the computer to process the data to recognize the object and to command the robot-arm to perform the proper operations. Cubic and cylindrical objects were used for testing.
Chapter 2

ROBOTIC ARM WITH COMPUTER VISION SYSTEM

A functional block diagram of the system is shown in Figure 1. The basic components of the system are:

a) Apple IIe personal computer with two serial ports
b) DAGE-MTI, INC. MK 11 series vidicon type video camera with 10 mm lens
c) The Rank Videometrix Video Processing Unit
d) Video monitor
e) Robot-arm; Five-Axis Robot Model TCM (TeachMover)
f) 5-1/4 in. single sided, double density floppy disk storage system
g) Okidata printer

Figure 2 shows the physical arrangement of the system.
System Structure Configuration

Figure 1
Experimental Equipment

Video Processing Unit

Monitor

Video Camera

Robot Arm TeachMover

Table Top

Apple IIe Computer

Figure 2
System Arrangement
Chapter 3

VIDEO PROCESSING SYSTEM

3.1 Video Camera with Lens

The optical information is reflected by a surface of an object and its surrounding top into a video camera under a light source. The video camera converts the optical information into an electrical signal. The 10 mm lens was used to cover a 10 x 10 inch field-of-view.

3.2 Video Processing Unit (VPU)

VPU was used to convert analog signal into digital form and analyze the digitized data. The Rank Videometrix Video Processing Unit (VPU) is a general purpose device which was integrated with a TV camera, TV monitor and a system controller-Apple IIe to provide an automatic dimensional measuring system. Its basic function is to process the video signal generated by the camera and extract various edge data which can be used to determine dimensions of the object being viewed. The unit responds to commands received from the system controller and returns various status and measurement data.
3.2.1 Functional characteristics

Measurement Window and Crosshairs -

The VPU superimposes a rectangular "measurement window" and a set of crosshairs on the TV monitor. The location can be changed under software control. The window surrounds that portion of the video scene which the VPU actually "sees." That is, it ignores any part of the scene outside the window. Thus the window can be used to isolate specific areas of the scene for analysis. The horizontal and vertical crosshairs are independently controllable. They enable gathering data along a particular scan line, a feature which is useful in many measurement applications.

The crosshairs are always confined to be within the window and are never allowed to be closer than 8 pixels or lines from a window side. The left and right window slides cannot be closer than 12 pixels. The same is true for the top and bottom. The VPU will override system controller commands that violate these constraints. At full size the window sides are located as follows:

\[
\begin{align*}
\text{TOP} & = 0 \text{ lines} \\
\text{BOTTOM} & = 400 \text{ lines} \\
\text{LEFT} & = 8 \text{ pixels} \\
\text{RIGHT} & = 503 \text{ pixels}
\end{align*}
\]

Edge Detection

In a sense, the VPU is an analog-to-digital converter. The continuous analog video signal generated by the
camera is sampled at a specific time interval and compared with a pre-set threshold. Sampled voltages above the threshold are declared to be "1" and those below are "0". The result is "digitized" bi-level video, that is, pure black and white with no gray. By saving all the 0's and 1's in memory it would be possible to digitize the entire scene. For most measurement applications, however, this is neither required or desirable. Typically, edge transitions form the basis for measurement. Therefore, the VPU was designed to "remember" only the location of the first edge transition (or alternately the last) that it encounters on each horizontal TV scan line. Similarly, it remembers edge transitions in the vertical direction yielding, in effect, a two-dimensional outline of the image. There are 400 usable scan lines and each is divided into 500 elements by the VPU. The data is stored in a table in its computer memory. Upon command the VPU writes to the table one of four types of data:

- 400 X (horizontal) leading edges (first transitions)
- 400 X trailing edges (last transitions)
- 500 Y (vertical) leading edges
- 500 Y trailing edges

The terms "X", "Y", "leading" and "trailing" are used frequently throughout the remainder of this manual. The VPU can transfer all this data to the system controller upon demand, which is useful in some cases. More typically, however, the system controller would request only the minimum of all the edge values, or the maximum,
or the one coinciding with the current crosshair location, etc. Appendix A describes all the various possibilities under "Data Gathering Commands."

**Centroids and Areas**

Besides defining edge locations the VPU can compute the centroid and area of the image in the window. This feature is useful in finding the area or centroid (area moment) of an object that is entirely within the measurement window. The process is performed entirely in hardware and runs at the video scan rate, that is, a centroid/area can be computed thirty times a second. Unlike the portion of the VPU which does edge detection, the centroid/area hardware uses all edge transitions, not just leading and trailing. The result is a true area/area moment. The centroid is referenced to the upper left hand corner of the window (when at its maximum size). This corner always represents 0,0.

**Thresholding**

The analog video signal voltage for a given scene covers a range representing the blackest black to the whitest white. The VPU contains peak detectors which in effect remember these extremes over the entire frame. It is then able to compare the intensity of every other point in the scene relative to these peaks in making its "0" or "1" determination as previously described under "EdgeDetection." This process is called thresholding.
The comparator setting can be anywhere from 0 to 100% of the range defined by the peak detectors and is under software control. A typical setting is 50% but sometimes various lighting and surface conditions require some experimentation to find the proper setting. Appendix A describes the threshold setting command.

3.2.2 System interconnects

Figure 3 shows cable interconnects for a the system consisting of the Video Processing Unit, TV camera, monitor, and a system controller. Video cables were the coaxial, shielded type.

Table 1 shows the pin assignments for the RS-232 connector (DB-25S) on the rear panel, which were connected to Apple IIe with serial port. And, Table 2 shows the switch settings on the first pc board inside the VPU.

3.2.3 RS-232 specifications

The VPU configuration used with the Apple IIe was as follows:

2400 Baud
7 Bit Characters (ASCII Standard)
Even Parity
1 Stop Bit

The VPU software does not support the following RS-232 functions:

Clear to Send
Request to Send
Data Terminal Ready
Data Set Ready
VIDEO PROCESSING UNIT
REAR PANEL

Figure 3

VPD System Interconnect Diagram

SERIAL I/O

SYSTEM CONTROLLER
Apple IIe

TV MONITOR

tv camera

MONITOR

VIDEO

VERT DRIVE

HORIZ DRIVE
### SERIAL I/O PIN ASSIGNMENTS

<table>
<thead>
<tr>
<th>PIN NO.</th>
<th>SIGNAL NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Protective Ground</td>
</tr>
<tr>
<td>2</td>
<td>Received Data</td>
</tr>
<tr>
<td>3</td>
<td>Transmitted Data</td>
</tr>
<tr>
<td>4</td>
<td>✓ Clear to Send</td>
</tr>
<tr>
<td>5</td>
<td>✓ Request to Send</td>
</tr>
<tr>
<td>6</td>
<td>✓ Data Terminal Ready</td>
</tr>
<tr>
<td>7</td>
<td>Signal Ground</td>
</tr>
<tr>
<td>8-19</td>
<td>Unused</td>
</tr>
<tr>
<td>20</td>
<td>✓ Data Set Ready</td>
</tr>
<tr>
<td>21-25</td>
<td>Unused</td>
</tr>
</tbody>
</table>

Connector - DB-25S

Table 1
## SWITCH SETTINGS

<table>
<thead>
<tr>
<th>Bit No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPU EOL Sequence</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Daisy/Non-daisy</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baud Rate</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**EOL Sequence (bit 4)**

- 0 = CR on VPU output
- * 1 = CR/LF

**Daisy/Non-daisy (bit 5)**

- * 0 = Standard I/O
- 1 = Daisy-chained I/O

**Baud Rate (bits 6-8)**

<table>
<thead>
<tr>
<th>Baud Rate</th>
<th>19200</th>
<th>9600</th>
<th>4800</th>
<th>2400</th>
<th>1200</th>
<th>600</th>
<th>300</th>
<th>150</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 011</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>300</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* : setting for experimental program

Table 2
3.3 Procedure for Detection of Object

1. If the width of the object is less than 100 pixels, it is less than two block lengths. Therefore, it never exceeds the two blocks (one block is 100 pixel-wide).

2. The object is adjusted to be wider than 30 pixels.

3. Then, the step of Y should be less than 30 pixels. Let Y step be 20.

4. a) Set y(I) at first 20 of a selected 200 x 200 square window.
   For I = 20 to 200 step 20
   Find IE/X/L and IE/X/T

   b) If (XT-XL) << 30
      That is, if (XT-XL) < 5 then it's just one portion of arc of an object. Then, save it and continue to measure the remaining arc or line.

   20 < width < 100
c) If $(XT-XL) > 5$ then, this block might hold the object then treat it as the object and go to step d)
If $(XT-XL) > 1000$ then, that point may contain noise and go to e.

d) Accumulate the number
\[ ps(I,J) = p(I,J) + 1 \]
e) Go to next block (move the window) and check as above.

![Diagram]

Window $ps(2,2)$  Window $ps(2,3)$

\[ ps(2,2) < ps(2,3) \]

5. The maximum number of $ps(I,J)$ is implemented as a holding block. Therefore, $n$th block is holding the object.
Save $I,J$ number into EX and EY and set up that window, and analyze the data using commands to get the data for the object and compute where it is placed and how much it is rotated if it is cubic.
6. In order to detect where it is located, locate the window around the object:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>T</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td>R</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>AG/T/B/L/R</td>
</tr>
</tbody>
</table>

**a)** Now with EX, EY, compute T, B, L, R to locate the window; AG/T/B/L/R.

**b)** To get the top point of the object compute MI/Y/L.

To get the bottom point of that object, compute MX/Y/T.

To get the left-most point of the object, compute MI/X/L.

To get the right-most point of the object, compute MX/X/T

**c)** From I=T to B step 5

Compute IE/X/L and IE/X/T.

Find Max. (XT-XL)

IE/Y/L and IE/Y/T

**d)** See if these values matche 6(b).

If almost same or less than 10% error then, there is no noise and data is good. And go to e. Else, go to c) and check data or check threshold again.
e) If the difference between X of the top point and X of the bottom point is less than 5, and the difference between Y of the left-most and Y of the right-most point is less than 5, THEN it is 45 degree rotated cubic or cylinder top, and go to next step 7. Else, it is surely cubic go to 7-c to check the degree of rotation.

7.

\[ \frac{(X - L_1)}{2} + L_1 \]

a) Using CC or CN/B, get the center of that top surface of the object. Save it to \((X_C, Y_C)\). After getting the point \((L_1, Y)\), and \((L_2, Y)\) calculate \((X_C - L_1)/2 + L_1\) and set up the crosshair using the command CS/V/\(((X_C - L_1)/2 + L_1)\). Next, get the point \((X, T_3)\) and \((X, T_4)\)

b) If \(ABS((T_4 - T_3) - (L_2 - L_1)/2) < (L_2 - L_1)/4\) THEN the object is cubic and rotated degree is 45 degrees else, it is a cylinder.

c) If it is cubic then, get the degrees the cubic is rotated.
1) First, get the centroid.
Second, calculate the length \( W_2 \) from the top to the horizontal crosshair which is set up through \((X_C, Y_C)\).
Third, calculate \( W_1 \).
Finally, using \( \theta = \tan^{-1}\left(\frac{W_2}{W_1}\right) \)

2) If the left-most point is higher than the centroid point, then again set the horizontal crosshair via that point and get \( \theta = \tan^{-1}\left(\frac{W_2}{W_1}\right) \). The data needed to command Robot-Arm is compensated and corrected if there is an error and checked again by slightly different method in program.
Chapter 4

ROBOTIC SYSTEM

4.1 Introduction

The robotic system used for this project is the TeachMover robot arm which is a microprocessor-controlled, six-jointed mechanical arm designed to provide an unusual combination of dexterity. This project is using Serial Interface Mode, in which the TeachMover arm can be controlled by a host computer, Apple IIe via one of two built-in RS-232C asynchronous serial communications lines. Major structural components are shown in Figure 4.

4.2 TeachMover Performance Characteristics

4.2.1 General

<table>
<thead>
<tr>
<th>Configuration</th>
<th>5 revolution axes and integral hand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drive</td>
<td>Electric stepper motors - Open loop control</td>
</tr>
<tr>
<td>Controller</td>
<td>6502A microprocessor with 4K bytes of EPROM and 1K bytes of RAM located in base of unit</td>
</tr>
<tr>
<td>Interface</td>
<td>Deal RS-232C asynchronous serial communications interfaces (baud rates in switch-selectable between 110, 150, 300, 600, 1200, 2400, 4800, and 9600 baud)</td>
</tr>
<tr>
<td>Teach Control</td>
<td>14 key - 13 function keyboard, 5 output and 7 input bits under computer control</td>
</tr>
</tbody>
</table>
Major Structural Components

Figure 4.
Power Requirement  12 to 14 volts, 4.5 amps DC

4.2.2 Performance

Resolution  0.011 in. (0.25 mm) maximum on each axis
Load Capacity  16 oz. (445 gm) at full extension
Gripping Force  3 lbs. (13 Nextons) maximum
Reach  17.5 in. (444 mm)
Velocity  0-7 in./sec. (0-178 mm/sec.) with controlled acceleration

4.2.3 Detailed Performance

<table>
<thead>
<tr>
<th>Motion</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>±90 degrees</td>
</tr>
<tr>
<td>Shoulder</td>
<td>+144, -35 degrees</td>
</tr>
<tr>
<td>Elbow</td>
<td>+0, -149 degrees</td>
</tr>
<tr>
<td>Wrist Roll</td>
<td>±360 degrees</td>
</tr>
<tr>
<td>Wrist Pitch</td>
<td>±90 degrees</td>
</tr>
<tr>
<td>Hand</td>
<td>0-3 in. (0-75 mm)</td>
</tr>
</tbody>
</table>

4.2.4 Physical characteristics

Arm Weight  8 lbs. (4 kg)
Teach Control  3.75 ft. (1150 mm)
Cable Length

4.3 How the Motors Operate

Each of the cable drives is controlled by a stepper motor. The motors used have 4 coils, each driven by a power transistor. The drive is digital with the transistors either turned on or turned off to obtain the desired
NOTE: Dimensions in inches.

Operating Envelope of the TeachMover Arm

Figure 5.
The wrist joint
Figure 6.
pattern of currents in the motor windings. By changing the pattern of currents, a rotating magnetic field is obtained inside the motor that causes the motor to rotate in small increments or steps.

Stepper motors are not the only kind of motors used in robot arms. Some arms use servo motors with electronic feedback loops for precise position control. Unlike stepper motors, these servo motors cannot develop slip page. This advantage must be weighed against the servo motor's far greater cost.

Stepper motors are easier to control from a computer than are servo motors.

Now, in order to turn a stepper motor in the TeachMover, a particular sequence of binary phase patterns is output to the desired motor, one pattern per step. In order to change motor direction, the order in which the phase patterns are output is simply reversed. The particular phase patterns used in the TeachMover generate a sequence known as "half-stepping;" the steps are half the size specified by the motor manufacturer. (The motors used to drive the TeachMover are specified by the manufacturer at 48 steps per revolution, but are actually stepped at 96 steps per revolution.) Compared to full stepping, halfstepping produces smoother slow-speed motions, reduces the power requirement, and improves the arm resolution by a factor of two.
The relationship between motor steps and actual joint rotation is given in Table 3.

4.4 **Electronic Circuitry and Interface**

4.4.1 **On-board computer and memory**

A circuit card houses all the internal electronics, including the 6502A Microprocessors. In technical terms, this microprocessor is an 8-bit, 2MHz chip. It is the same chip used in the Apple, Atari, and PET computers; it is used in the TeachMover to coordinate all joint motions and handle all input and output.

TeachMover firmware (permanently built-in software) is contained in another chip housing 4K bytes of read-only memory (ROM); this firmware interprets the commands given to the arm, converting these to electrical signals the arm can obey.

The circuit card also includes chips containing 1K bytes of random-access memory (RAM). This is enough RAM to store an arm-motion program of up to 53 steps. It is possible to "piggy-back" a second set of RAMs on the first, thereby extending the program capacity to 126 steps.

4.4.2 **Serial ports**

Two serial interface ports allow the connections to the TeachMover to a host computer, printer, or terminal. Serial transmission speed is selectable with eight
## MOTOR STEPS AND JOINT ROTATIONS

<table>
<thead>
<tr>
<th>Motor</th>
<th>Joint</th>
<th>Steps per degree</th>
<th>Steps per radian</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Base</td>
<td>19.64</td>
<td>1125</td>
</tr>
<tr>
<td>2</td>
<td>Shoulder</td>
<td>19.64</td>
<td>1125</td>
</tr>
<tr>
<td>3</td>
<td>Elbow</td>
<td>11.55</td>
<td>672</td>
</tr>
<tr>
<td>4</td>
<td>Right wrist</td>
<td>4.27</td>
<td>241</td>
</tr>
<tr>
<td>5</td>
<td>Left wrist</td>
<td>4.27</td>
<td>241</td>
</tr>
</tbody>
</table>

Motor Steps and Joint Rotations

Table 3
standard speeds available from 110 to 9600 Baud. The 9600 Baud speed is used for Apple IIe.

4.4.3 User inputs and outputs

The computer card also contains an auxiliary parallel input/output port. Interfaces from the TeachMover to external equipment is done through a 16-conductor flat ribbon cable. Five TTL compatible user output bits can be set (to 1) or cleared (to 0) under program control to turn other equipment on or off when a given arm motion is complete. Seven TTL compatible input bits can be used to control an arm sequence when a given external condition is met.

A block diagram of the TeachMover's electronic circuitry is shown in Figure 7.

4.5 Operation from a Host Computer, Apple IIe

Connecting the TeachMover arm to a host computer or a terminal greatly extends the unit's capabilities.

4.5.1 Configuring the serial ports

"Configuring the serial ports" refers to making sure that the computer and the TeachMover can "talk" to one another. This requires taking care of the following:

1. electrical connections
2. transmission rate
3. data format
4. settings for standard interface signals
Block Diagram of TeachMover Computer and Electronics

Figure 7.
5. opening the port
6. testing the configuration

4.5.2 Electrical connections

The two serial ports perform the following functions:

- Signals that enter the left port (P2) always pass through to the right port (P1) unchanged.
- Signals that enter the right port pass through to the left port unchanged, unless the signals are a series of characters beginning with an "@" sign and terminating with a <CR> (carriage return); these signals are not passed through, but are interpreted as arm commands.

Thus, to operate the arm from a host computer or a terminal, connect the computer or terminal to the Teach-Mover's right serial port.

4.5.3 Transmission rate

The TeachMover is configured to operate at a transmission rate of 9600 baud (9600 bits per second), for both send and receive. You can change this rate to any of seven other standard rates by means of three switches located on TeachMover computer card (Figure 11). The available rates and the corresponding switch settings are given in Table 4.
Connecting the TeachMover to Apple IIe

Figure 8.
Pin Numbering For Serial Port Connectors

Figure 9.
Apple IIe-to-Teach Mover Serial Connection

Figure 10.
### Baud Rate Selection

#### Table 4

<table>
<thead>
<tr>
<th>Baud</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>150</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>300</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>600</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>1200</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>2400</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>4800</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>9600</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
</tbody>
</table>

*NOTE: SW4 is not used.*

---

Swatches for Selecting Serial Transmission Rate

*Figure 11.*
These switches should be changed when power is off, since the switch settings are read by TeachMover firmware on power-up only.

4.5.4 **Data format**

The TeachMover uses the following data format:

- word length = 8 bits
- 1 start bit
- 1 stop bit
- no parity bit
- full duplex

Especially, Apple IIe rather than specify a word length of 8, it is necessary to specify a word length of 7 plus a parity bit equal to zero. This is because this computer uses a most significant bit equal to 1 when processing 8-bits words. (It is possible to use the @ARM command to allow the robot to recognize an "@" with the eighth bit = 1, but in order to execute this command the robot must recognize the first "@" in this @ARM command. To do this the robot must receive an "@" character with the most significant bit = 0.).

4.5.5 **Standard interface signals**

Some computers and terminals require logic levels on certain pins to indicate the following status conditions:

- Data Terminal Ready
- Clear to Send
- Carrier Detect
- Request to Send

The TeachMover does not use these signals, but does pass them through when it is placed in series between a computer and a terminal.
4.5.6 Serial interface commands

Ten different commands can be issued to the Teach-Mover over the serial lines. (A concise summary of all ten commands is given in Appendix D.)

Note: All commands can be abbreviated to an "@" sign plus the first three characters--@CLO for @CLOSE, etc.

- All characters and numeric values are decimal ASCII (industry-standard character format).
- Once a serial command is executed, the teach control is left in TRAIN mode, with two exceptions:
  - @RESET leaves it in MODE mode.
  - @RUN simply runs the arm until another command stops it.
- However, the indicator lights will remain as they were before the serial command was executed. (Example: If MODE light is on, and then, say, a @CLOSE command is executed, the Teach Control will then be in TRAIN mode but with the MODE light still on.)
- To change the status of the indicator lights, use the @STEP command with all parameters set to zero except the "OUT" value (see below). No other serial command affects the status of the lights (except the closed light which
<table>
<thead>
<tr>
<th>Left Port Pin No.</th>
<th>Description</th>
<th>Right Port Pin No.</th>
<th>Jumper</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Data Carrier Detect</td>
<td>8</td>
<td>W1</td>
</tr>
<tr>
<td>1,7</td>
<td>Ground</td>
<td>1,7</td>
<td>--</td>
</tr>
<tr>
<td>3</td>
<td>Transmit from TeachMover</td>
<td>2</td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td>Receive by TeachMover</td>
<td>3</td>
<td>--</td>
</tr>
<tr>
<td>4</td>
<td>Request to Send</td>
<td>4</td>
<td>W4</td>
</tr>
<tr>
<td>5</td>
<td>Clear to Send</td>
<td>5</td>
<td>W3</td>
</tr>
<tr>
<td>20</td>
<td>Data Terminal Ready</td>
<td>20</td>
<td>W2</td>
</tr>
</tbody>
</table>

Control Lines

Table 5
Location of Jumper Connections for Serial Operations

Figure 12.
always indicates the state of the gripper switch).

The ten commands are as follows:

1. @STEP            6. @ARM
2. @CLOSE           7. @DELAY
3. @SET             8. @QDUMP
4. @RESET           9. @QWRITE
5. @READ            10. @RUN

4.6 Arm Initialization and Calibration

The computer in the robot keeps track of the arm by using the starting position as a reference. To run a program to operate the robot arm, the starting position was as specified in the program.

With the origin of the coordinate system located at the axis of rotation of the base of the robot arm, the location of the front edge of the base of the arm is defined as \( x = 1\frac{5}{8} \). The base is centered on the \( y \)-axis. The gripper is brought to rest, barely touching, on the spot PO, shown on the grid at \( x=5 \) and \( y=0 \).

Because of the proper window of the VPU for the experimental program, two relative cartesian coordinates are used. One is called table top cartesian coordinates and the other called real cartesian coordinates.

The table top cartesian coordinates is placed on the table top which is 8 inch higher than the bottom where the base of the robot arm is placed. And the 0 point of
the X axis (X=0) is 4 inches farther placed from the axis of rotation of the base of the robot arm.

The real cartesian coordinates is for robot arm, with origin located at the axis of rotation of the base of the robot arm.

After all the data was calculated using the table top cartesian coordinates from the VPU data, it was converted into the real cartesian coordinates as follows:

Let XR, YR, and ZR be the variables for VPU and X, Y and Z be those for robot arm.

Then, X=XR+4
Y=YR
Z=ZR+8

This conversion is shown in Figures 13 and 26.

Following the above explanation, the initialization point is:

XR=5
YR=0
ZR=0

Therefore,
X=9
Y=0
Z=8

The hand must be perpendicular to the work surface and parallel to the front edge of the base of the robot. The gripper opens as the arm is brought to this position, and then closes the gripper as the last step in
setting the initial position. The gripping force is applied by the motor as the point is closing.

The arm can be brought to this starting position by moving it manually with the power off or by using the teach control with the power on.

Specifically, the cartesian coordinates of the initial configuration are:

\[
\begin{align*}
XR &= 4, \; X = 9 \text{ inches} \\
YR &= 0, \; Y = 0 \text{ inches} \\
ZR &= 0, \; Z = 8 \text{ inches} \\
\text{Pitch} &= -90^\circ \\
\text{Roll} &= 0^\circ \text{ (see Note 1, below)} \\
\text{Grip} &= \text{Closed} \text{ (see Note 2, below)}
\end{align*}
\]

**Note 1:** Because the hand can turn through many revolutions of "roll," it is difficult to tell by simply looking at the hand whether the "roll" has been set to \(0^\circ\). Yet, it is important that the roll initially be \(0^\circ\) in order that the wrist cables be allowed their full range of motion.

**Note 2:** For experimental program it is important that the initial position be very precise. Initialization point is shown in Figure 13.
This is the bottom, 8 inches lower than the right side.

This is the table top 8 inches higher than the left side (bottom).

Initialization point:

\( X = X_R + 4 \)

Initial point:

\( X = X_R = 0 \)

\( X = X_R = 3 \)

\( X = X_R + 5 \)

\( X = X_R = 7 \)

\( X = 4 \)

\( X = 6 \)

\( X = 9 \)

\( X = 11 \)

1 block is 1x1 inch.
4.7 Coordinate Conversions

It is often advantageous to be able to describe the configuration of a robot arm in more than one coordinate system. The two most commonly used systems are:

- **Joint Coordinates** (the joint angles of the arm).
  These are most convenient for controlling the arm directly from a computer.

- **Cartesian Coordinates** (X, Y, Z, pitch, and roll).
  These are more convenient for describing an assembly task on a flat table top.

For practical work, a set of formulas is needed to mathematically convert from one coordinate system to the other.

- **The Forward Solution** converts from joint angles to Cartesian coordinates.

- **The Backward Solution** converts from Cartesian coordinates to joint angles.

This section describes how both of these coordinate systems are defined, and how the forward and backward solutions may be derived and implemented.

### 4.7.1 Kinematic model of arm

Before formulating the arm solutions, the relationship between the different parts of the arm must be specified. This can be done in terms of the kinematic model shown in Figure D-1. The kinematic model indicates
how each joint is articulated, how the joint angles are measured, and the distances between joints.

θ is used to indicate joint angles in mathematical expressions. The symbols \( \theta_1, \theta_2, \theta_3, \theta_4, \) and \( \theta_5, \) respectively, are proportional to the joint expressions \( J_1, J_2, J_3, J_4, \) and \( J_5 \) used in the computer command discussed in Chapter 7. The \( \theta_s, \) measured in degrees or radians, are related to the \( J_s, \) measured in motor steps, as shown in Table D-1. There are 360 degrees or 2 radians in one complete revolution.

The distances between joints (lengths of arm members) are indicated by the constants, \( H, L, \) and \( LL \) shown in Figure 14. \( H \) is the distance from the table top to the shoulder joint centerline; \( L \) is the distance from shoulder joint to elbow joint, which equals the distance from elbow joint to wrist joint; and \( LL \) is the distance from the wrist joint to the center point between the two fingertips, with the fingertips separated by 1.5 inches. Values for these distances are given in Table 7.

The pitch angle, \( P, \) and the roll angle, \( R, \) are given by the following equations.

\[
P = .5 (\theta_5 + \theta_4) \quad (1)
\]

\[
R = .5 (\theta_5 - \theta_4) \quad (2)
\]

where \( \theta_4 \) and \( \theta_5 \) are right and left wrist angles. The angles \( P, \theta_4, \) and \( \theta_5 \) are all measured from the horizontal as shown in Figure 15.
CONVERSION FACTORS BETWEEN MOTOR STEPS
AND REVOLUTE JOINT ANGLES

<table>
<thead>
<tr>
<th>Motor</th>
<th>Joint</th>
<th>Steps in Revolution</th>
<th>Steps per Radian</th>
<th>Steps per Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Base</td>
<td>7072</td>
<td>1125</td>
<td>19.64</td>
</tr>
<tr>
<td>2</td>
<td>Shoulder</td>
<td>7072</td>
<td>1125</td>
<td>19.64</td>
</tr>
<tr>
<td>3</td>
<td>Elbow</td>
<td>4158</td>
<td>672</td>
<td>11.55</td>
</tr>
<tr>
<td>4</td>
<td>Right wrist</td>
<td>1536</td>
<td>241</td>
<td>4.27</td>
</tr>
<tr>
<td>5</td>
<td>Left wrist</td>
<td>1536</td>
<td>241</td>
<td>4.27</td>
</tr>
</tbody>
</table>

Conversion Factors Between Motor Steps and Revolute Joint Angles
Table 6
KINEMATIC SYMBOLS USED

Hinge Joint

Swivel Joint

Differential Joint

Figure 14.

Kinematic Model of the TeachMover Arm
### Lengths of TeachMover Arm Members

<table>
<thead>
<tr>
<th>Segments</th>
<th>Length (inches)</th>
<th>Length (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>7.68</td>
<td>195.0</td>
</tr>
<tr>
<td>L</td>
<td>7.00</td>
<td>177.8</td>
</tr>
<tr>
<td>LL</td>
<td>3.80</td>
<td>96.5</td>
</tr>
</tbody>
</table>

Lengths of TeachMover Arm Members

Table 7
Definition of Roll and Pitch Angles

Figure 15.
4.7.2 Forward arm solution

This section shows how to determine $P$, $R$, and the $X$, $Y$, and $Z$ coordinates of the end point from the joint angles $\theta_1$, $\theta_2$, $\theta_3$, $\theta_4$, and $\theta_5$. The coordinates and joint angles are defined in Figure 14. This solution relies on the trigonometric relationships given in Figure 16 for reference.

The first step is to determine $Z$, the height of the end point above the table top, and an intermediate variable $RR$, the horizontal distance from the base pivot to the end point. The situation is summarized in Figure 17. Summing the vertical contributions from each link gives the following expression for $Z$:

$$Z = H + L \sin \theta_2 + L \sin \theta_3 + LL \sin P$$  \hspace{1cm} (3)

Summing the horizontal contributions gives:

$$RR = L \cos \theta_2 + L \cos \theta_3 + LL \cos P,$$  \hspace{1cm} (4)

where pitch angle $P$ is given by

$$P = .5(\theta_5 + \theta_4).$$  \hspace{1cm} (5)

The second step is to determine the $X$ and $Y$ coordinates of the end point from the intermediate variable, $RR$, as shown in Figure 18. By inspection, the coordinates are:

$$X = RR \cos \theta_1$$  \hspace{1cm} (6)
ANGLE FORMULAS:
\[ \alpha + \beta + \gamma = 180^\circ \]
for a right triangle \( \gamma = 90^\circ \)
and \( \alpha + \beta = 90^\circ \)

PYTHAGOREAN THEOREM:
\[ C^2 = A^2 + B^2, \quad \text{or} \]
\[ C = \sqrt{A^2 + B^2} \quad \text{or} \quad A = \sqrt{C^2 - B^2} \]

RATIOS OF SIDES:
\[ \sin \alpha = \frac{A}{C} \quad \text{or} \quad A = C \sin \alpha \]
\[ \cos \alpha = \frac{B}{C} \quad \text{or} \quad B = C \cos \alpha \]
\[ \tan \alpha = \frac{A}{B} \quad \text{or} \quad A = B \tan \alpha \]

ANGLE DEFINED BY INVERSE FUNCTION:
\[ \alpha = \tan^{-1} \left( \frac{A}{B} \right) \]

Basic Trigonometric Relationships

Figure 16.
Side View of Kinematic Model

Figure 17.
\[ Y = RR \sin \theta_1 \] 

A summary of this forward solution is given in Table D-3. A BASIC program implementing this solution is given in Figure D-12 (Statements 460 to 510). The program's variables \( T_1, T_2, \ldots, T_5 \) correspond to the angles \( \theta_1, \theta_2, \ldots, \theta_5 \).

4.7.3 Backward arm solution

This section shows how to determine the joint angles \( \theta_1, \theta_2, \theta_3, \theta_4, \) and \( \theta_5 \) required to position the end point at a desired \( X, Y, Z \) position and with desired values of pitch and roll. The coordinates referred to are shown in Figure 14. A review of the formulas used is given in Figure 16.

A. Specifying position/orientation - \( X, Y, Z, P, \) and \( R \)

Before starting the backward solution it is necessary to specify the desired position and orientation of the end point. The position of the end point is defined by the following three distances:

- **X**: The distance of the desired end point in front of the arm, measured from the base pivot along the X-axis.
- **Y**: The distance of the desired end point to the left of the arm, measured from the base pivot along the Y-axis.
Top View of Kinematic Model

Figure 18.
Summary of Forward Solution

<table>
<thead>
<tr>
<th>Step</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$P = (\Theta_5 + \Theta_4)/2$</td>
</tr>
<tr>
<td>2</td>
<td>$R = (\Theta_5 - \Theta_4)/2$</td>
</tr>
<tr>
<td>3</td>
<td>$RR = L \cos \Theta_2 + L \cos \Theta_3 + LL \cos P$</td>
</tr>
<tr>
<td>4</td>
<td>$X = RR \cos \Theta_1$</td>
</tr>
<tr>
<td>5</td>
<td>$Y = RR \sin \Theta_1$</td>
</tr>
<tr>
<td>6</td>
<td>$Z = H + L \sin \Theta_2 + L \sin \Theta_3 + LL \sin P$</td>
</tr>
</tbody>
</table>

Table 8
Z: The vertical height of the desired end point above the table top.
The units of these distances (inches or millimeters) are match the units of the segment lengths shown in Table 7.

The orientation at the end point is defined by the following two angles (see Figure 19):

P: The desired pitch angle, measured in degrees
R: The desired roll angle, measured in degrees

In practice it is difficult to distinguish between positive and negative roll angles (as +90° and -90°, or +45° and -135°) by looking at the hand. It is helpful to mark the top of the hand when it is at 0° to eliminate this ambiguity. The 0° position corresponds to the orientation when the wrist cable turnbuckles are aligned.

B. Specifying roll in cartesian frame, R'

Sometimes it is useful to express "roll" with respect to a Cartesian frame rather than with respect to the arm. One way to do this is to use P = -90° (hand point down) as a reference orientation, and measure the "Cartesian roll" with respect to the x-axis, as indicated in Figure 20. The formula relating to roll measured with respect to the arm (R) and the roll measured with respect to the Cartesian frame (R') is then simply:

\[ R' = R - 0° \]
Different Hand Orientations

Figure 19.
In the backward solution, we introduce a special variable, \( R_l \), that enables us to write equations that are valid regardless of whether roll is measured with respect to the arm or with respect to the Cartesian frame.

\[ R_l = 1 \text{ if roll is with respect to Cartesian frame.} \]

\[ R_l = 0 \text{ if roll is with respect to arm frame.} \]

With this new variable, Equation (8) can be modified to express both normal and Cartesian roll as follows:

\[ R' = R - \theta_1 R_l \] \hspace{1cm} (9)

Solving for \( R \) gives:

\[ R = R' + \theta_1 R_l \] \hspace{1cm} (10)

C. **Backward solution, step-by-step**

The first step of the backward solution is to determine the base angle, \( \theta_1 \), and the radius vector, \( RR \), from the base to the end point as shown in Figure 21.

Using the Pythagorean Theorem:

\[ RR = \sqrt{x^2 + y^2} \] \hspace{1cm} (11)

\[ \theta_1 = \tan^{-1}(Y/X). \] \hspace{1cm} (12)

The second step is to find \( \theta_4 \) and \( \theta_5 \) from \( P \) and \( R \).

Using Equation (1) and Equation (2) of the wrist
Top view of arm with Pitch $= 90$ degrees showing roll in Cartesian frame ($R'$) and roll with respect to the arm ($R$).

Figure 20.
Top View of Arm

Figure 21.

Side View of Hand Triangle in Kinematic Model

Figure 22.
differential previously described, and substituting \((R' + 1R_l)\) for \(R\) using Equation (10) gives:

\[
5 = P + R' + \theta_1 R_l \\
4 = P - R' - \theta_1 R_l.
\]

[Note: From here on, the prime will be dropped and use \(R\) for roll in all cases, remembering to set \(R_l = 0\) when roll is measured with respect to the arm, and \(R_l = 1\) when roll is measured with respect to the Cartesian frame.]

The third step is to work back from the coordinates of the end point to those of the wrist. As in the forward solution, we use the side of the kinematic model shown in Figure 17. Distances in this view are measured vertically along the \(Z\) axis and horizontally along the radius from the base (\(r\) axis). Letting \(R_e\) and \(Z_e\) be the coordinates of the end point in this plane, we can calculate the coordinates of the wrist \((R_w, Z_w)\) by using the triangle shown in Figure D-9. From this triangle the coordinates of the wrist are:

\[
R_w = R_e - LL \cos P \\
Z_w = Z_e - LL \sin P
\]

The fourth step is to define the shoulder-elbow-wrist triangle so that \(\theta_2\) and \(\theta_3\) can be determined. For this purpose, the translated coordinate system introduced in Figure 23 is used. The origin \((0,0)\) is at the shoulder and the coordinates of the wrist are now \((R_0, Z_0)\).
Shoulder-Elbow-wrist Triangle

Figure 23.
The distance from the shoulder to the wrist, $R_0$, is the same as $R_w$ previously determined in Equation (15). This is expressed as:

$$R_0 = R_e - LL \cos \theta $$ (17)

The height of the wrist above the shoulder, $Z_0$, is just the height of the wrist above the table top, $Z_w$, less the height of the shoulder, $H$. Thus,

$$Z_0 = Z_w - H $$ (18)

substituting for $Z_w$ using Equation (16) gives

$$Z_0 = Z_e LL \sin \theta - H $$ (19)

The fifth step is to solve the shoulder-elbow-wrist triangle for $\theta_2$ and $\theta_3$. Three new angles: $\alpha$, $\beta$, and $\phi$, are introduced to simplify this solution. First solve for $\alpha$, $\beta$, and $\phi$.

Since $\tan \beta = (Z_0/R_0)$, we obtain:

$$\beta = \tan^{-1}(Z_0/R_0). $$ (20)

Pivoting the shoulder-elbow-wrist triangle about the shoulder by $\beta$ gives the simplified triangle shown in Figure 24. The length of the base of the simplified triangle is given by $\sqrt{Z_0^2 + R_0^2}$ (Pythagorean Theorem, using the right triangle at the bottom of Figure 23). As shown in Figure 24, the simplified triangle can be partitioned into two congruent right triangles. The base, $b$, of each of these smaller triangles is then given by:

$$b = .5\sqrt{Z_0^2 + R_0^2} $$ (21)
Simplified Triangle

Figure 24.
The height \( h \) (again using the Pythagorean Theorem) is
\[
h = L^2 - b^2.
\] (22)

Since the tangent of \( \alpha \) is \( h/b \),
\[
\alpha = \tan^{-1}(h/b).
\] (23)

Substituting for \( h \) in Equation (23) by using Equation (22) gives
\[
\alpha = \tan^{-1} \frac{\sqrt{L^2 - b^2}}{b}.
\] (24)

Substituting for \( b \) in Equation (24) using Equation (21) gives
\[
\alpha = \tan^{-1} \frac{\sqrt{\frac{4}{R_o^2 + Z_o^2} - 1}}{L}.\] (25)

The sixth step is to use \( \phi \) and \( \Theta_2 \) to determine \( \Theta_2 \) and \( \Theta_3 \). The following three relations are first set up and then solved. At the shoulder (see Figure D-10),
\[
\Theta_2 + \phi + \Theta_3 = 180^\circ.\] (27)

Summing the internal angles of the simplified triangle (Figure 24) gives \( \phi + \alpha + \alpha = 180^\circ \), or
\[
\phi = 180^\circ \alpha - 2.\] (28)

Substituting the value of \( \Theta_2 \) from Equation (26) and the value of \( \phi \) from Equation (28) into Equation (27) gives
\[
\Theta_3 = \alpha - \beta.\] (29)
Note however, that the elbow angle, $\theta_3$, is defined as the angle above the horizontal and hence we must change the sign of $\theta_3$.

In summary, the results of the sixth step are:

$$\theta_2 = \alpha + \beta .$$  \hspace{1cm} (30)

$$\theta_3 = \beta - \alpha .$$  \hspace{1cm} (31)

thus completing the backward solution. A summary of the backward solution is given in Table 9.

4.7.4 Variation of hand length with hand opening.

The opening of the hand is proportional to the number of steps of the hand drive motor. The constant of proportionality is:

$$S_6 = 371 \text{ steps/inch (14.6 steps/mm)}. $$

Although the length of the hand, $LL$, has been treated as a constant in the previous calculations, it varies slightly with hand opening, as shown in Figure 25. The effect is small, $\pm 0.10$ in. ($\pm 2.5$ mm), but for more precise work it may be necessary to take this into account.

The hand length, $LL$, may be expressed as the sum of a fixed length, $L_1$, and a varying length that depends on hand opening, $G$, by the following formula:

$$LL = L_1 + \sqrt{L_2^2 - \frac{(G - G_0)^2}{2}}$$  \hspace{1cm} (32)
# Summary of Backward Solution

## Step 1 Operation

1. Determine arm constants H, L, LL

2. Determine the desired X, Y, Z, R, P, and R1 coordinates of the endpoint

3. \[ 1 = \tan^{-1}(Y/X) \]

4. \[ RR = X^2 + Y^2 \]

5. \[ 5 = P + R + R1 \]

6. \[ 4 = P - R - R1 \]

7. \[ R_0 = RR - LL \cos P \]

8. \[ Z_0 = Z - LL \sin P - H \]

9. \[ = \tan^{-1}(Z_0/R_0) \]

10. \[ = \tan^{-1} \frac{4L^2}{(R_0^2 + Z_0^2)} - 1 \]

11. \[ 2 = + \]

12. \[ 3 = - \]

---

**Summary of Backward Solution**

**Table 9**

---

65
Variation of Hand Length with Hand Opening

Figure 25.
where:

\[ L_1 = 1.884 \text{ in (47.9 mm)} \]
\[ L_2 = 1.700 \text{ in (43.2 mm)} \]
\[ G_0 = 1.520 \text{ in (38.6 mm)} \]

The hand opening, \( G \), may be converted to motor steps and vice-versa by using the proportionality constant, \( S_0 \), given above.

Varying hand length may be taken into consideration in both the forward and backward solutions. Before starting either solution, the correct value of \( LL \) would be computed from the hand opening using Equation (32).

4.8 Procedure for Robot Arm Movement

In order to grab the object correctly and place it where we want, those data that were determined and calculated in procedure for detection of object should be converted into cartesian coordinates for robot arm movement as below.

1. Conversion of pixels to cartesian world coordinate for Robot shown in Figure 26.

As compared in Figure 26, the coordinate for \( X,Y \) in cartesian paper can be calculated in the following way,

\[ X = (430 - XC)/41 \]
\[ Y = (YC - 200)/41 \]

where \( X \) and \( Y \) are in inch, and \( XC \) and \( YC \) are in pixels.
Cartesian Paper in Inch for Robot

$X, Y$ coordinates in pixels for VPU

Conversion of pixels to Cartesian Coordinate

Figure 26
2. Save the data of the object into some rooms.

\[ R = \text{transmitted data for rotated degree of the object} \]
\[ GP = \text{the width of the object} \ (=HG) \]
\[ XR = \text{X coordinate of the center or centroid of the object} \]
\[ YR = \text{Y coordinate of the center or centroid of the object} \]
\[ ZR = \text{height from the table top to the tip of the hand grip} \]
\[ P = \text{pitch in degree} \]

3. Initialize the robot arm using cartesian coordinates. To initialize the robot arm, use the Hand Held Control to place the tip of the hand grip.

\[ XR = 5 \text{ inch} \]
\[ YR = 0 \text{ inch} \]
\[ ZR = 0 \]
\[ P = -90 \text{ degrees} \]
\[ R = 0 \text{ degree} \]
\[ GP = 0 \]

Then, calculate the real cartesian coordinates for robot arm. Since the table top is 8 inches higher than TeachMover base and the center of the table top \((5,0)\) is 4 inches farther placed from the real cartesian coordinate for TeachMover:
\[ X = XR + 4 = 9 \]
\[ Y = YR = 0 \]
\[ Z = ZR + 8 = 8 \]
\[ P = -90 \text{ degrees} \]
\[ R = 0 \]
\[ GP = 0 \]

This conversion can be easily understood in Figure 27.

4. Move the arm to the stand-by position to avoid the blocking of lens' sight for the 10-in. x 10-in. table top cartesian paper.

\[ XR = 2; \quad X = XR + 4 = 6 \]
\[ YR = 0; \quad Y = YR = 0 \]
\[ ZR = 9; \quad Z = ZR + 8 = 17 \]
\[ P = 45 \text{ degrees} \]
\[ R = 0 \]
\[ GP = 0 \]

5. Then, obtain data of the position of the object.

6. Move the arm to the 1-in. higher than the top of that object and ready to be picked up by opening the hand and rotate in proper degrees if it is cubic.

7. Lower the hand to have the object between the hand grips.

8. Hold the object.

9. Lift up the object by one inch.

10. Place it as programmed. If it is cubic, XR = 2, YR = 3, ZR = 0, P = -90 degrees, R = 0 and GP = 0. If
The conversion from the cartesian on the table top to the real cartesian coordinates for robot arm

Figure 27
it is cylinder, XR = 2, YR = -3, ZR = 1, P = -90 degrees, R = 0 and GP = 0.

11. Go back to the origin, XR = 5, YR = 0, ZR = 0, P = -90 degrees, R = 0, GP = 0.

12. Stop the program.
Chapter 5
APPLE IIE COMMUNICATION TO VPU AND TEACHMOVER

5.1 Introduction
For the Apple Computer to communicate with the Teach-Mover and VPU it is necessary to interconnect the two via a serial card and a cable.

5.2 Hardware Needed
1. Super Serial Card, manufactured by Apple Computer, Inc.
2. RS-232-C cable with a 25-pin male connector at each end.

5.3 Hardware Setup for VPU
The Apple II Super Serial Card (SSC) needs to be prepared first in accordance with Chapter 1 of the SSC manual. Then proceed with the settings below. For reference, the TeachMover uses the SSC in the Communications Mode.

1. The reversible jumper block on the SSC should have the white triangle pointed at "MODEM."
2. Switch No. 1 (SW1) Settings
3. Switch No. 2 (SW2) Settings

<table>
<thead>
<tr>
<th>POSITION</th>
<th>SETTING</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Off</td>
<td>These settings are 2400 BAUD</td>
</tr>
<tr>
<td>2</td>
<td>On</td>
<td>transmission rate</td>
</tr>
<tr>
<td>3</td>
<td>Off</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>On</td>
<td>Communications mode RS-232-C</td>
</tr>
<tr>
<td>7</td>
<td>On</td>
<td>signals</td>
</tr>
</tbody>
</table>

4. The SSC was plugged into slot #2 in the Apple.

5.4 Hardware Setup for TeachMover

The Apple II Super Serial Card (SSC) needs to be prepared first in accordance with Chapter 1 of the SSC manual. Then proceed with the settings below. For reference, the TeachMover uses the SSC in the Communications Mode.
1. The reversible jumper block on the SSC.

2. Switch No. 1 (SW1) Settings

<table>
<thead>
<tr>
<th>POSITION</th>
<th>SETTING</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Off</td>
<td>These settings are 9600 BAUD transmission rate, and match the factory setting of the TeachMover.</td>
</tr>
<tr>
<td>2</td>
<td>Off</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Off</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>On</td>
<td>Communications mode</td>
</tr>
<tr>
<td>7</td>
<td>On</td>
<td>RS-232-C signals</td>
</tr>
</tbody>
</table>

3. Switch No. 2 (SW2) Settings

<table>
<thead>
<tr>
<th>POSITION</th>
<th>SETTING</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>On</td>
<td>Sets 1 stop bit</td>
</tr>
<tr>
<td>2</td>
<td>Off</td>
<td>Sets 7 data bits</td>
</tr>
<tr>
<td>3</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Off</td>
<td>Odd parity</td>
</tr>
<tr>
<td>5</td>
<td>Off</td>
<td>No LF after CR</td>
</tr>
<tr>
<td>6</td>
<td>Off</td>
<td>Interrupts off</td>
</tr>
<tr>
<td>7</td>
<td>Off</td>
<td>RS-232-C signals</td>
</tr>
</tbody>
</table>

4. The SSC was plugged into slot 4 in the Apple.

5. RS-232-C cable was plugged into the right connector when looking from the rear of the TeachMover.
Chapter 6

FLOW DIAGRAM FOR ROBOTIC SYSTEM
WITH COMPUTER VISION

START

Set the proper threshold of VPU

Initialize the robot arm and move it to the stand-by position

Set the overall window size
20-429 pixels for X;
0-400 pixels for Y coordinate

Check every window which sign if 200x200 pixels
if there is an object

Set the window 200x200 pixels in which the object is placed

Get the data of center if it is cylinder, or centroid if it is cubic. Get the data of rotated degree if it is cubic, the width of that object to set up the robot hand grip
Details and algorithms are explained in 3.3 including Procedure for Detection of Object, and in 4.8, the Procedure for Robot Arm Movement. The experimental program is listed in Appendix E.
Chapter 7

CONCLUSION

Image processing was found to be useful in detecting an object. Analyses were made to get the proper information for robot movement using a controller such as the Apple IIe computer. Even though the usage of personal computers was limited in the past, it was found to be useful as a robotic system controller.

The system was found to be capable of distinguishing cubic and cylindrical objects of different sizes. Programs written were capable of retrieving these objects and placing them in the proper location.

As a future project, a sensor for object detection can be attached to the robot itself to simplify coordinate transformations. Filtering techniques can be applied to images of the object using video image processing to improve the quality of the images when operating at poor lighting environments.
REFERENCES


3. R. Y. Wong, "Computer Pattern Classification and Scene Matching," published by the Department of Electrical and Computer Engineering, California State University, Northridge.


APPENDIX A

System Set-Up
APPENDIX A

System Set-up

Power Supply (for the video camera)

ON/OFF switch

This switch turns the camera system power on and off.

Video Camera

1. Camera Connector
   Connect the camera cable connector. Be sure the cable connector if fitted securely to each Vertical, Horizontal, and Video connector which should be properly connected to VPU input. And, the monitor is to be connected to the monitor input.

2. For proper size and picture
   Adjust Lens focus for the sharpest picture. Adjust the apertature for the proper threshold of the VPU. Adjust the distance from the 8-inch high table top to have the entire picture within the 409x400 pixel window of the VPU.

Videometrix

1. Connector coming from camera should be properly connected to camera as explained above in Video Camera section.

2. The connector for monitor to TV monitor
Serial Port (for VPU)
1. 21-pin connector goes to the serial port which resides in slot #2 of the Apple Computer.
2. The switch on the first layer inside the VPU is set up as below. Switch 6 is OFF, 7 is ON, and 8 is ON for 2400 baud rate. Switch 4 is ON for CR/LF and 5 is OFF for standard I/O. The rest of the switch left does not affect any situation.

Robot Arm
1. DC Power Supply should be connected to the robot arm, TeachMover using the two leg connector which red leg should be plugged into red female plug.
2. The DB-24S coming from the Apple IIe connector should be connected to the right female connector attached to the back of the base.
3. The center of the robot-arm should be placed at X=0 and Y=0 point in real cartesian world coordinates.

Apple IIe
1. Super Serial Port for VPU should reside in slot #2 for experimental program.
2. Super Serial Point for TeachMover is in slot #4.
3. Disk Drive has ribbon cable connected to its controller card in slot #6, drive 1 of the Apple IIe computer.
APPENDIX B

System Operating Instruction

1. Turn on power supply for the video camera.
2. Turn on the lighting system.
3. Turn on the monitor.
4. Turn on the VPU power switch.
5. Turn on computer power and load "project" program from disk drive 2.
6. Run the program, following the instructions below.
   6.1 Initialize the robot arm with X=5, Y=0, Z=0, P=90 degrees, R=0, GP=0 at the center of the table top cartesian paper, which is actually X=9, Y=0, Z=8, P=90 degrees, R=0, GP=0 in real cartesian coordinates for robot calculation and movement.
   6.2 Set the proper threshold.
   6.3 Return to get the data for the object location and move the robot arm to place the object where we want. If it is cubic, place it at X=6, Y=0 and if it is cylinder, place it at X=6, Y=0 in real cartesian coordinates.
7. Check the movement and the result.
APPENDIX C

Functional Difference of VPU Software
APPENDIX C

Functional Difference of VPU Software

1. Required Changes in Host Software

   A. The following commands no longer return output

   CS )
   CM )
   GS ) All Gate & Crosshair
   GM ) Manipulation commands
   WS )
   WM )
   AG )

   TH    Threshold Set
   DV    Digitized Video Enable/Disable
   CI    Comparator Input Select
   RS    Reset
   HO    Horizontal Offset
   DM    Display Mode
   US    Update Aspect Ratio
   WP    Write to a Port

   B. Ensure that VPU switch settings are correct for
      application (settings now control baud rate, daisy/non-daisy, and VPU output terminator
      characters.

   C. Spaces may no longer be used as delimiters in
      polled VPU command strings.

2. Enhancements

   A. New Commands

   AN    Vertex & Cosine of an angle
   RP    Read a parallel port
   WP    Write to a parallel port
   EL    Equation of a least squares fit line
   MF    Median filter enable/disable
   CT    Crosshair tracking enable/disable
   FN    Surface or edge focus
B. Fixed or Enhanced Commands

RS  Reset now works
CS/W  Set Xhairs to center of current window
CC  Can now use many points (up to 10,000 can be specified)
AG  Now works reliably
WS  Now works reliably
CS  Now center crosshairs to precise center of maximum window

3. Power-Up Conditions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparator Input</td>
<td>Video</td>
</tr>
<tr>
<td>Digitization</td>
<td>Off</td>
</tr>
<tr>
<td>Video Threshold</td>
<td>50% (Relative)</td>
</tr>
<tr>
<td>Gradient Threshold</td>
<td>50% (Absolute)</td>
</tr>
<tr>
<td>Window Setting</td>
<td>Full window</td>
</tr>
<tr>
<td>Crosshair Setting</td>
<td>Centered</td>
</tr>
<tr>
<td>Aspect Ratio</td>
<td>1.25 (5X:4Y)</td>
</tr>
</tbody>
</table>
APPENDIX D

Serial Interface Commands
APPENDIX D
Serial Interface Commands

Notes: 1. \(<\text{CR}>\) = Carriage Return
2. Arm returns [0<\text{CR}>] if command has a syntax error, [1<\text{CR}>] after command is executed (except for @RUN), [2<\text{CR}>] if STOP button was pressed before execution was completed (@STEP and @CLOSE only)

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Syntax/Details of Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>@ARM</td>
<td>Specifies recognition character to use instead of &quot;@&quot; sign</td>
<td>(@\text{ARM} \ &lt;\text{CHAR}&gt; \ &lt;\text{CR}&gt;) where CHAR is any character except a carriage return.</td>
</tr>
<tr>
<td>@CLOSE</td>
<td>Close gripper until grip switch is activated</td>
<td>(@\text{CLOSE} \ &lt;\text{SP}&gt; \ &lt;\text{CR}&gt;) where SP = optional speed value (see item 4 in this appendix).</td>
</tr>
<tr>
<td>@DELAY</td>
<td>Inserts a delay between transmitted characters</td>
<td>(@\text{DELAY} &lt;\text{CR}&gt;) Where N = proper delay value, determined by trial and error.</td>
</tr>
<tr>
<td>@QDUMP</td>
<td>Uploads entire current program from TeachMover to host computer</td>
<td>(@\text{DUMP} &lt;\text{CR}&gt;) Returns character string comprising 8 two-byte values for each program step. See Table 9 in chapter 7 for details.</td>
</tr>
<tr>
<td>@QWRITE</td>
<td>Downloads a program step from host computer to TeachMover</td>
<td>(@\text{WRITE}&lt;N&gt;,&lt;L1&gt;,&lt;L2&gt;,...,&lt;L7&gt;&lt;\text{CR}&gt;) where N = Step number to which program step is to be written. L1-L7 = two-byte values as in @QDUMP command. See chapter 7 for details.</td>
</tr>
</tbody>
</table>
**Command**  
@READ

**Function**  
Reads value of the internal position registers, gives last key pressed on teach control, and tells which input bits are on.

**Syntax/Details of Operation**  
@READ <CR>
Arm returns:
<K1>,<K2>, ..., <K6>,<I><CR>
where K1-K6 = values of internal position registers
I = Last key *256 + Input Byte where "Last key" values are defined below:

<table>
<thead>
<tr>
<th>&quot;Last key&quot; Value</th>
<th>Key Pressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TRAIN</td>
</tr>
<tr>
<td>2</td>
<td>PAUSE</td>
</tr>
<tr>
<td>3</td>
<td>GRIP</td>
</tr>
<tr>
<td>4</td>
<td>OUT</td>
</tr>
<tr>
<td>5</td>
<td>FREE</td>
</tr>
<tr>
<td>6</td>
<td>MOVE</td>
</tr>
<tr>
<td>7</td>
<td>MODE</td>
</tr>
<tr>
<td>8</td>
<td>STEP</td>
</tr>
<tr>
<td>9</td>
<td>POINT</td>
</tr>
<tr>
<td>10</td>
<td>JUMP</td>
</tr>
<tr>
<td>11</td>
<td>CLEAR</td>
</tr>
<tr>
<td>12</td>
<td>ZERO</td>
</tr>
<tr>
<td>13</td>
<td>SPEED</td>
</tr>
<tr>
<td>14</td>
<td>REC</td>
</tr>
</tbody>
</table>

and:
"Input Byte" = decimal number whose binary equivalent specifies which of the eight input bits are set to 1 (see "jump condition" numbers under hand-held teach control JUMP command, above).

@RESET

**Function**  
Zeros the internal position registers and turns off motor currents

**Syntax/Details of Operation**  
@RESET<CR>

@SET

**Function**  
Sets subsequent arm speed and activates joint control keys on hand-held teach control

**Syntax/Details of Operation**  
@SET<SP><CR>
where SP = optional speed value (see item 4 in this appendix). Control returns to host when REC or MODE key is pressed.
@STEP <SP>, <J1>, <J2>, ..., <J6>, <OUT><CR>

where SP = speed value (see item 4 in this appendix).

J1-J6 = Number of motor half-steps
J1 = Base swivel (positive counterclockwise)
J2 = Shoulder (positive downwards)
J3 = Elbow (positive downwards)
J4 = Right wrist (positive downwards)
J5 = Left wrist (positive downwards)
J6 = Hand (positive open)
OUT = Optional decimal number whose binary equivalent specified the value of the output bits (see Appendix F, item G).
APPENDIX E

Program Listing
10 MUME
10 US = CHR$(4)
20 US = CHR$(4)
40 REM
50 GOSUB 7000
55 PRINT "HS"
60 E = 0
70 T = 0: B = 400: L = 20: R = 429
80 GOSUB 6000
90 PRINT "TV/E"
90 PRINT "D2/E"
95 PRINT "CB/W"
96 PRINT "TH/E"
100 GOSUB 8000
101 GOTO 20000
102 REM FROM 270B2
103 PRINT "****************************"
104 PRINT "INITIALIZATION"
105 PRINT "--- (5,0) AT CENTER --- (0,0) AT RIGHTMOST CENTER--- (10,0)
106 PRINT "AT LEFTMOST CENTER --- "
107 GOTO 160
110 PRINT "TYPE IN COMMAND AND, 0 OR, 1"
120 INPUT L,E
130 GOSUB 7000
140 GOSUB 10000
145 PRINT "WANT MORE COMMAND? Y-0,N-1"
146 INPUT ZV
147 IF ZV = 0 GOTO 110
150 PRINT "TYPE IN THRESHOLD VALUE, JUST NUMBER"
150 PRINT "TYPE IN COMMAND AND, 0 OR, 1"
160 INPUT TN
170 GOSUB 7000
180 PRINT "TH/1T"
185 GOSUB 8000
190 PRINT "AGAIN? Y-0,N-1"
200 INPUT ZV
210 IF ZV = 0 THEN GOTO 160
220 GOSUB 7000
230 IF (C = 0:V) = 200: LD = 20: RD = 223
240 M = INT(M)
250 M = INT(M)
260 DIM X(M), Y(T), Y(M)
260 DIM X(M), Y(T), Y(M)
270 DIM X(M), Y(T), Y(M)
270 DIM X(M), Y(T), Y(M)
280 DIM X(M), Y(T), Y(M)
290 PRINT "CN/B"
310 PRINT "CN/B"
310 PRINT "CN/B"
320 PRINT "CN/B"
330 INPUT XC, YC, RC
340 IF ABS(XC - XCM) > 50 OR ABS(YC - YCM) > 50 THEN PRINT "** FAILED!
350 PRINT "CN/B"
360 PRINT "CN/B"
370 IF XC (60 THEN L = 20: GOTO 380
375 L = INT(XC - 60)
380 IF XC > 369 THEN R = 429: GOTO 400
390 H = INT(XC + 60)
400 IF VC (60 THEN T = 0: GOTO 420
410 F = INT(YC - 60)
420 IF VC (340 THEN B = 400: GOTO 440
430 B = INT(YC + 60)
440 M = INT ((F - T) / 5 + 1)
450 GOTO 5000
3280 FOR NJ = 1 TO 3
3280 FOR NI = 1 TO 3
3280 T = TD + (NJ - 1) * 100
3280 B = BD + (NI - 1) * 100
3280 L = LD + (NI - 1) * 100
3280 H = HD + (NI - 1) * 100
3280 PRINT "AS:/";T;"/";B;"/";L;"/";R
3280 J = 0
3330 FOR I = 1 TO B STEP 5
3380 J = J + 1
3390 PRINT "CS/W:";I
3400 PRINT "IE/X/L"
3400 INPUT XL(J), Y(J)
3410 PRINT "IE/X/T"
3410 INPUT XT(J), Y(J)
3420 NEXT J
3430 FOR J = 1 TO M
3440 PRINT "AT Y=";Y(J);" XL=";XL(J);" XT=";XT(J)
3450 DF(J) = XT(J) - XL(J)
3460 NT(J) = XT(J) - XT(J - 1)
3470 NL(J) = XL(J) - XL(J - 1)
3480 PRINT "DIFFERENCE XT-XL=";DF(J)
3490 REM (XT(J) - XL(J)) * 5 AND (XT(J) - XL(J)) (110 THYN
3500 P $S(NI,NJ) = PS(NI,NJ) + 1
3510 IF DF(J) > 5 AND DF(J) (110 AND ABS (NL(J)) (18 AND ABS (NT(J))
3520 THEN PS(NI,NJ) = PS(NI,NJ) + 1
3530 IF XL(J) < 0 OR XL(J) (S03 THEN NS(NI,NJ) = NS(NI,NJ) + 1
3540 IF BS(J) 11 OR BS(J) (S03 THEN NS(J) = NS(J) + 1
3550 IF BS(NJ) > 18 OR BS(NJ) (S03 THEN PD(NI,NJ) = PD(NI,NJ) + 1
3560 IF S(NI,NJ) = 0 PRINT " TOO BIG"
3570 NEXT J
3580 NEXT NI
3590 NEXT NJ
3600 PRINT "---------------------"
3610 PRINT " POSSIBLE ::::: TOO BIG"
3620 FOR J = 1 TO 3
3630 PRINT "PS(";I;",";J;")=";PS(I,J);" PE(";I;",";J;")=";PE(I,J)
3640 PRINT "BS(";I;",";J;")=";BS(I,J)
3650 PRINT "NS(";I;",";J;")=";NS(I,J)
3660 NEXT I
3670 NEXT J
3680 REM FIND MAXIMUM PS(I,J)
3690 REM IF PS(I,J) IS MAXIMUM THEN SAVE I,J
3690 EX = 0
3700 DX = 0
3710 FY = 0
3720 FOR J = 1 TO 3
3730 FOR I = 1 TO 3
3740 IF NS(I,J) = 20 AND PS(I,J) ) EX THEN GOTO 4064
3750 IF FY = 0 GOTO 4070
3760 IF PE(I,J) = 21 THEN EX = PS(I,J); IX = I; JX = J
3770 IF PE(I,J) ) FX THEN FX = PE(I,J); MX = I; MX = J
3780 NEXT J
3790 NEXT I
3800 REM MINIMUM=30, 3W/5=.6, LEAST NO.
3810 IF EX = 4 THEN GOTO 5000
3820 REM FIND CENTER
3830 REM IF IT IS TOO BIG
3840 PRINT "PLACE OBJECT, IF THERE IS AN OBJECT THEN IT IS TOO SMALL TO
3850 GIVE UP"
3860 PRINT "AFTER PLACE OBJECT, PUSH RETURN"
3870 PRINT "STARTING POINT"
3880 PRINT NI, NJ
3890 INPUT BS#
3900 UDOT 100 REM 3200 STARTING POINT
3910 NP = OBJ.CENTER AND RADIUS
1) REM OBJECT IS IN THIS BLOCK
2) REM FUM FROM 450
3) PRINT "ALL"/"TH"/"TH"/"TH"/<R
4) PRINT "ALL"/"TH"/"TH"/"TH"/"TH"
5) REM WINDOW EXPANSION
6) PRINT "### ( ) USING COMMAND-CC ***
7) IF KP = 0
8) PRINT "TH"/"TH"
9) PRINT "CL"
10) INPUT WX,WY,RK
11) PRINT "CNT"/"TH"
12) INPUT XL,YC
13) IF ABS (XC - WX) < 10 OR ABS (YC - WY) < 10 THEN GOTO 5:38
14) IF KP = KP + 1 THEN PRINT "### TRY AGAIN WITH DIFFERENT TH ###": GOTO 5:28
15) PRINT "###": GOTO 5:12
16) GOSUB 4500
17) PRINT "-------"
18) PRINT "THE OBJECT IS PLACED AT X";XC:";YC:" WITH RADIUS = ";RK"
19) PRINT "-------"
20) PRINT "### ( ) USING MI, MX ###
21) GOSUB 7000
22) PRINT "MI/X/L"
23) GOSUB 7000
24) PRINT "MX/X/T"
25) GOSUB 7000
26) PRINT "MX/Y/T"
27) GOSUB 7000
28) PRINT "BX,BY"
29) GOSUB 7000
30) PRINT "--------
31) PRINT "LEFT MOST POINT(";XL:";YL:")
32) PRINT "RIGHTMOST POINT(";XR:";YR:")
33) PRINT "TOP POINT(";TX:";TY:")
34) PRINT "BOTTOM POINT(";Bx:";By:")
35) YO = H/0 (RY - LY) / 10 = ABS (BX - TX)
36) PRINT "RY-LY=";YO:" BX-TX=";XD
37) PRINT "--------
38) PRINT "CENTER BETWEEN LEFT AND RIGHT = (";XI:";YI;"
39) PRINT "CENTER BETWEEN TOP AND BOTTOM = (";XI:";YI;"
40) PRINT "CENTER KC,YC WAS = (";XCI:";YC;"
41) REM IF (X,Y) AND (X,Y) ARE 100 DIFFERENT
42) REM THEN TRY--10/X/L--10----
43) GOSUB 10
44) PRINT "TH"/"TH"
45) J = 0
46) FOR I = 1 TO 10 STEP 5
47) J = J + 1
48) PRINT "GCLI": I

X1 = (WX - LX) / 2 + LX
Y1 = (HY - LY) / 2 + LY
X2 = (BX - TX) / 2 + TX
Y2 = (BY - TY) / 2 + TY
PRINT "--------
PRINT "CENTER BETWEEN LEFT AND RIGHT = (";XI:";YI;"
PRINT "CENTER BETWEEN TOP AND BOTTOM = (";XI:";YI;"
PRINT "CENTER KC,YC WAS = (";XCI:";YC;"
REM IF (X,Y) AND (X,Y) ARE 100 DIFFERENT
REM THEN TRY--10/X/L--10----
PRINT (X1,Y1) = 0
PRINT PT = 0:XP = 0:YP = 0:MT = 0:XM = 0:YM = 0
GOSUB 7000
PRINT "TH"/"TH"
J = 0
FOR I = 1 TO 10 STEP 5
J = J + 1
PRINT "GCLI": I
GOSUB WTRA: PRINT "TH/":IN
5260 YC = INT (YC)
5265 PRINT "CB/H":YC
5270 PRINT "IE/Y/L": INPUT L1,T1
5275 PRINT "IE/Y/T": INPUT L2,T2
5280 R1 = XC - L1:R2 = LC - XC
5285 LC = (XC - L1) / 2 + L1
5290 LL = INT (LC)
5295 PRINT "CB/":LLC
5300 PRINT "IE/Y/L": INPUT L3,T3
5305 PRINT "IE/Y/T": INPUT L4,T4
5310 GOSUB 5800
5315 PRINT "TH":IN
5320 IF LL (43 THEN HS = LL / 43: GOTO 5920
5330 HS = (LL - 43) / 54 + 1
5340 REM FROM 5800:
5345 PRINT "A/:DX (4 / ABS (DR)) : GOTO 5920 PRINT "CURVE NL" CYLINDER
5350 GOSUB 5920
5360 PRINT "CYLINDER", SM = 1:SM
5370 IF LL (43 THEN HS = LL / 43: GOTO 5920
5380 HS = (LL - 43) / 54 + 1
5390 PRINT "IBR": DES "HBR": INCH
5395 GOSUB 7000
5400 PRINT "TH":IN
5405 PRINT "CB/H": INT (YC)
5410 PRINT "BS/": INT (XC)
5415 GOSUB 9999
5420 IF SM = 0 THEN PRINT "CUBIC **********":INCH
5425 INPUT ABS
5430 PRINT "DR"; "": PRINT DR "": "":"]/"": "" 
5435 GOTO 5920
5440 PRINT "HBR": "": HBR "": "":"]/"": ""
5445 RETURN
DIM U(7,40) : REM ROOM FOR 40 STEPS
GOTO 26500
D1 = 0 : D2 = - 500 : D3 = + 1162 : D4 = + 384 : D5 = D4 : D6 = 0
DIM LINE 133 IS THE NUMBER OF JOINT STEPS FROM T1=0,T2=0,T3=0,T4=0,
AND J=0, TO X=5,Y=0,Z=0,P=90,R=90, AND J=0.
REM
REM HEAD IN FIRST LINE FOR INITIALIZATION
READ X,Y,Z,P,R,0,5
PRINT "SET ARM TO THE FOLLOWING POSITION & ORIENTATION"
PRINT "USING TEACH CONTROL PENDANT, PRESS MODE KEY WHEN FINISHED"
PRINT "X" : INPUT X : INCHES"
PRINT "Y": INPUT Y : INCHES"
PRINT "Z": INPUT Z : INCHES"
PRINT "P": INPUT P : DEGREES"
PRINT "R": INPUT R : DEGREES"
PRINT "H": INPUT H : DEGREES"
PRINT "M": INPUT M / SSI: INCHES"
PRINT "K": INPUT K / SSI: INCHES"
PRINT "#SET 200" : INPUT 1
PRINT "#SET 201" : INPUT 1
PRINT "#SET 201" : INPUT 1
HOME = VTAB 71 ; HTAB 5
U = 0
GOTO 26120
**REM BACKWARD SOLUTION CALCULATIONS**

```plaintext
24000 REM *** TEST 12 LI:="1" PI=":PI:" LL=":LL"
24010 LL = 3.281 = 7.05" REM USED VARIABLE IN VPU
24015 P = V / C \ H = H / C
24010 IF X = 0 THEN T1 = SGH (Y) * PI / 2
24010 IF X = 1 THEN T1 = ATN (Y / X)
24010 IF T1 (0 THEN PRINT "PRIN ------------------" ; T1)
24010 HX = SGH (X * X + Y * Y)
24010 IF HX (2.25 AND 1 (15 THEN PRINT "PRIN HAND TOO CLOSE TO END Y, RR=" ; RR
24010 IF HX = 17.3 THEN PRINT "REACH OUT OF RANGE, RR=" ; RR
24010 HC = RR - LL * COS (P)
24010 IF X (2.25 AND Z (1.25 AND HC (3.5 THEN IF P ( - 90 / C THEN
24010 PRINT "PRIN HAND INTERFERENCE WITH BASE.")
24010 REM NOTE THAT THE ABOVE STATEMENT MAY BE ALTERED TO ACCOMMODATE Y
24010 REM VES CLOSE TO THE BASE
24010 Z = Z - LL * SIN (P) - H
24010 IF R0 = 0 THEN PRINT "PRIN HAND OUT OF RANGE.")
24010 IF R0 = 0 THEN PRINT "PRIN HAND OUT OF RANGE FOR SHOULDER
24010 AND ELBOW." ; GOTO 25500
24010 REM A = ATN (LDH (A))
24010 REM T2 = A + B
24010 REM T3 = A
24010 IF X (1.44 / C OR T2 ( - 35 / C THEN PRINT "SHOULDER OUT
24010 OF RANGE. T2=" ; T2
24010 IF X = T3 (0 OR T2 - T3) 149 / C THEN PRINT "ELBOW OUT
24010 OF RANGE. T3=" ; T3
24010 IF X (270 / C OR R ( - 270 / C THEN IF P ( 90 / C + T3) - (R + 270 / C) OR P ( (- 90 / C + T3) - (R - 270 / C)) THEN PRINT
24010 "PRIN ELBOW OUT OF RANGE. PITCH=" ; P
24010 IF P (90 / C + T3) OR P ( - 90 / C + T3) THEN PRINT "PRIN"
24010 "PITCH OUT OF RANGE. PITCH=" ; P
24010 IF R (360 / C - ABS (P - T3)) OR R ( - 360 / C + ABS (P - T3)) THEN PRINT "ROLL OUT OF RANGE, ROLL=" ; R
24010 REM "**** CHECK #170 ******
24020 REM CORRECT COORDINATES
24030 W1 = INT (S1 + T1 + .5) - P1
24030 W2 = INT (S2 + T2 + .5) - P2
24030 W3 = INT (S3 + T3 + .5) - P3
24030 W4 = INT (S4 + T4 + .5) - P4
24030 W5 = INT (S5 + T5 + .5) - P5
24040 RETURN
24040 RETURN
```

The code snippet provided calculates backward solutions for a robotic arm or similar mechanical system, checking for various conditions such as reaching out of range or interference with the base. It involves calculations for angles, distances, and ensuring that the system remains within specified limits.
20:10 INPUT " ROLL":" R
20:11 INPUT " HAND":" GP
20:12 INPUT " SPEED":" B
20:15 INPUT " HIT RETURN TO GO ON: " AS
20:16 IF SS = 0 GOTO 20180
20:18 X = X + 417 = Z + B
20:19 IF X (0 THEN 21000
20:20 GOSUB 240000 REM SHOW COORDINATES
20:20 GOSUB 25000
20:22 PRINT "(1,1)" = " UU(1,1)" UU(2,1) = UU(2,1)
20:23 PRINT " UU(3,1) = UU(3,1)" UU(4,1) = UU(4,1)
20:24 PRINT " UU(5,1) = UU(5,1)" UU(6,1) = UU(6,1)
20:25 PRINT " W1 = W1" W2 = W2 W3 = W3 W4 = W4
20:26 PRINT " W5 = W5" GP = GP S = S
20:27 PRINT " @STEP "1S"," W1 = UU(1,1)" ," W2 = UU(2,1)" ," W3 = UU(3,1)"
20:28 PRINT " W4 = UU(4,1)" ," W5 = UU(5,1)" ," W6 = UU(6,1)" INPUT I
20:29 U = U + 1
20:31 UU(1,1) = W1
20:32 UU(2,1) = W2
20:33 UU(3,1) = W3
20:34 UU(4,1) = W4
20:35 UU(5,1) = W5
20:36 UU(6,1) = W6
20:37 UU(7,1) = B
20:38 PRINT " @STEP : S = 0,0,0,0,0,0,0." : GP: INPUT I
20:39 IF U = 1 THEN GOTO 27070
20:44 IF U = 2 THEN GOTO 27500
20:46 IF U = 3 THEN GOTO 27540
20:48 IF U = 4 THEN DD = 01: GOTO 27750
20:50 IF U = 5 THEN UU = 01: GOTO 27900
20:52 IF U = 6 THEN UU = 01: GOTO 28000
20:54 IF U = 7 THEN UU = 01: GOTO 28060
20:56 IF U = 8 THEN UU = 01: GOTO 28200
20:58 IF U = 9 THEN UU = 01: GOTO 28400
20:60 IF U = 10 THEN UU = 01: GOTO 28600
20:62 IF U = 11 THEN UU = 01: GOTO 28660
20:64 GOTO 25161
20:66 REM COMPARER 20090 THRU 20320 WITH OTHERS DIFFERENT RESULT?
20:69 CY = 0
20:71 PRINT " RETURN TO INITIAL POSITION"
21002 PRINT " @READ": INPUT I
21003 INPUT A, B, C, D, E, F, G
21004 PRINT " @CLOSE 242": INPUT I
21005 PRINT " @STEP 242": I = H; I = I; I = C; I = I; I = C; I = C; I = C
21006 PRINT " C: INPUT I
21007 PRINT " RUN THE PROGRAM"
21008 FOR I = 1 TO U
21009 PRINT " @STEP ": UU(7,1) = UU(7,1) = UU(7,1) = UU(7,1)
21010 IF UU = 01: (0 THEN 21060
21011 PRINT " @STEP 240,0,0,0,0,0,0; UU(6,1): INPUT N
21012 GOTO 21100
21013 PRINT " @CLOSE245": INPUT N
21014 PRINT " @STEP 240,0,0,0,0,0,0; " : UU(6,1): INPUT N
21100 NEXT I
21101 CY = CY + 1
21102 PRINT " CYCLE "CY
21120 .300000
21121 W = R
21150 NEW DISPLAY COORDINATES
21151 PRINT " THE ROLL ARM IS MOVING TO THE COORDINATES:
21200 ON "COORDINATES"
21201 ON = 1 PRINT " X = " X = " INDEX"
21206 IF " X = " X = " INDEX"
XC>ucci: i'n11.90
01:00\, \text{UU}(5,0) = 0
01:00\, \text{UU}(6,0) = -931
01:00\, \text{UU}(7,0) = 6395
01:10\, \text{GOTO} 20133
01:10\, \text{REM} \quad \text{FROM INITIALIZATION} \quad 20160
01:10\, X = \text{BY} \cdot 0.7 = 17
01:10\, Y = 45; K = \text{GP} = 0
27030 \quad \text{GOSUB} \quad 24000
27040 \quad \text{GOSUB} \quad 25000
27050 \quad \text{GU} = 1 \quad \text{REM} \quad \text{TO BE COMMING BACK}
27060 \quad \text{GOTO} \quad 20202
27070 \quad \text{GU} = 0
27080 \quad \text{PRINT} \quad ;
27090 \quad \text{GU} = 1
27100 \quad \text{IF} \quad 0 = 0 \quad \text{THEN} \quad \text{DP} = 3 \quad \text{GOTO} \quad 20161
27110 \quad \text{GU} = 0 \quad \text{REM} \quad \text{FROM} \quad 20161 \quad \text{THEN} ---
27120 \quad \text{REM} \quad \text{GET DATA OF} \quad \text{VPU}
27130 \quad \text{HUME} = \text{GOTO} 102 \quad \text{REM} \quad \text{GOING TO VPU}
27140 \quad \text{REM} \quad \text{FROM} \quad 5555 \quad \text{VPU FINISHED}
27150 \quad XH = (450 - XC) / 41
27160 \quad YH = (YL - 200) / 41
27170 \quad \text{ZH} = \text{HS} + 1 \quad \text{REM} \quad \text{OR} \quad ZH = (XH - 1) / 2 \quad \text{IF} \quad \text{LUBIC}
27180 \quad \text{GOSUB} = \text{P}
27190 \quad \text{PRINT} \quad "XH": XH; \quad "YH": YH; \quad "ZK": ZK
27200 \quad \text{PRINT} \quad ": \quad \text{DR}"; \quad \text{DR} = \text{DR} + 20 \quad \text{DR}" = \text{DR} + 20
27210 \quad \text{REM} \quad \text{CONVERSION FROM} \quad \text{CARTESIAN} \quad \text{TO} \quad \text{ROBUIT} \quad \text{REAL} \quad \text{CARTESIAN}
27220 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27500
27230 \quad \text{IF} \quad XH \leq X \quad \text{THEN} \quad \text{GOTO} \quad 27494
27240 \quad \text{IF} \quad XH = 0 \quad \text{THEN} \quad \text{GOTO} \quad 27446
27250 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27260 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27270 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27280 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27290 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27300 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27310 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27320 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27330 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27340 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27350 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27360 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27370 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27380 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27390 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27400 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27410 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27420 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27430 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27440 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27450 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27460 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27470 \quad \text{IF} \quad XH \geq -3 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27480 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27490 \quad \text{IF} \quad XH \geq -2 \quad \text{THEN} \quad \text{GOTO} \quad 27496
27500 \quad \text{IF} \quad XH = \text{ZH} \quad \text{THEN} \quad \text{GOTO} \quad 27519
27510 \quad \text{Y} = \text{YH} = \text{YR} = \text{YR} + 4
27520 \quad \text{L} = \text{LH} + \text{A}
27530 \quad \text{R} = \text{DR}
27540 \quad \text{V} = \text{DR}
27550 \quad \text{GP} = \text{GP} + 1
27560 \quad \text{LL} = 3.6: \text{REM} \quad \text{LL WAS LENTH IN} \quad \text{VPU} \quad \text{LL}=3.6 \quad \text{IS} \quad \text{FOR ROBOT}
27570 \quad \text{B} = \text{B}
27580 \quad \text{PRINT} \quad "XH": XH; \quad "YH": YH; \quad "ZK": ZK; \quad "R": R; \quad "LH": LH;
27590 \quad \text{PRINT} \quad "A": \quad \text{GP} = \text{GP} + 1 \quad \text{GP} = \text{GP} + 1 \quad \text{GP} = \text{GP} + 1 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20
27600 \quad \text{PRINT} \quad "=\text{GP} + 1 \quad \text{GP} = \text{GP} + 1 \quad \text{GP} = \text{GP} + 1 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20 \quad \text{DR} = \text{DR} + 20
27610 \quad \text{B} = \text{B}
27620 \quad \text{GOTO} \quad 27000
27630 \quad \text{RE}: \quad \text{FINISHED}
27640 \quad \text{PRINT} \quad "\text{FINISHED}"

\text{or} \quad \text{FINISHED}"
REM INSTEAD OF 2020:
PRINT "CLOSE E20"
INPUT I
REM FROM 2770
I = 3.415 = 22016P = 0
X = 6.1Y = 4.1P = -99
BUSH = 24000
LET P = -60
GO = 5
GOTO 20210
REM UP ON THE GOAL PLACE
IF SM = 0 THEN I = 2.415 = 22016P = 0:X = 6:Y = 4:R = 0:G = -99
REM CLOSE HAND
PRINT "CLOSE E35"
INPUT I
REM: GO BACK TO ORIGIN
X = 9:Y = 0:Z = 10.2
B = 15:K = 230:R = 0
P = -99
BUSH = 24000
LET R = 24000
LET Z = 9
REM GOTO ORIGIN Z=8.2.2
X = 9:Y = 0:Z = 10.2
P = 15:K = 230:R = 0
P = -99
BUSH = 24000
BUSH = 24000
LET Z = 10
GOTO 20210
REM FROM 28550
LET Z = 81P = -9016P = 0
LET R = 24000
LET Z = 11
25000 GOTO 20210
25010 REM FROM 20259
25020 REM FROM 27702
25030 PRINT "------------------------"  
25040 PRINT " YOU DID A GOOD JOB "THANK YOU !"  
25050 VZ = 1
25060 IF VZ = 0 GOTO 20161
25070 END
25080 DATA 5, 0, 0, -90, 0, 0, 130  
25090 DATA 8, 0, 2, -90, 0, 90, 242  
25100 DATA 8, 0, 0, 5, -50, 0, -50, 242  
25110 DATA 8, 0, 2, -90, 0, 0, 240  
25120 DATA 6, 5, 2, -90, 0, 0, 240  
25130 DATA 6, 5, 0, 5, -90, 90, 300, 2-2  
25140 DATA 6, 5, 2, -90, 90, -1, 2-2  
25150 DATA 8, 0, 2, -90, 0, 800, 2-0  
25160 DATA -999, 0, 0, 0, 0, 0, 0  
25170 END