รายการอ้างอิง

Amstead, B.H. , Ostwald,P.F. and Begeman, M.L. Manufacturing Process. 8th ed. U.S.A : John-Wiley & Sons. ,1987.

Berhardt, R. ,Schreck, G. and Willnow, C. Accuracy Enhancement in Offline Programming for Industrial Robot. I.E.E. Institution of Electrical Engineers, 1994.

Billingsly , J. Robot and Automated Manufacture. United Kingdom: Short Run Press. ,1985.

Chan, S.F. , Weston, R.H. and Case, K. Robot Simulation and Off-line Programming. Computer-Aided Engineering Journal , August,1988.

Dassault Systems, CATIA. Cell Design and Robot Programming , User 's Guide. ,1996.

Dassault Systems, CATIA. Cell Design and Robot Programming , User 's Referece Manual. ,1996.

Groover, M.P. Automation,Production System, and Computer Integrated Manufacturing. U.S.A. :Prentice-Hall. ,1987.

Luggen, William W. Flexible Manufacturing Cell and System. U.S.A.: Prentice-Hall. , 1991.

Maus, R. and Allsup, R. Robotics : A Manager 's guide. U.S.A. : John Wiley & Sons., 1986.

Mitsubishi Industry , Industrial Micro -Robot System Model RV-M1 Move Master EX Industrial Manual. BFP-A5191E-B.

ภาคผนวก

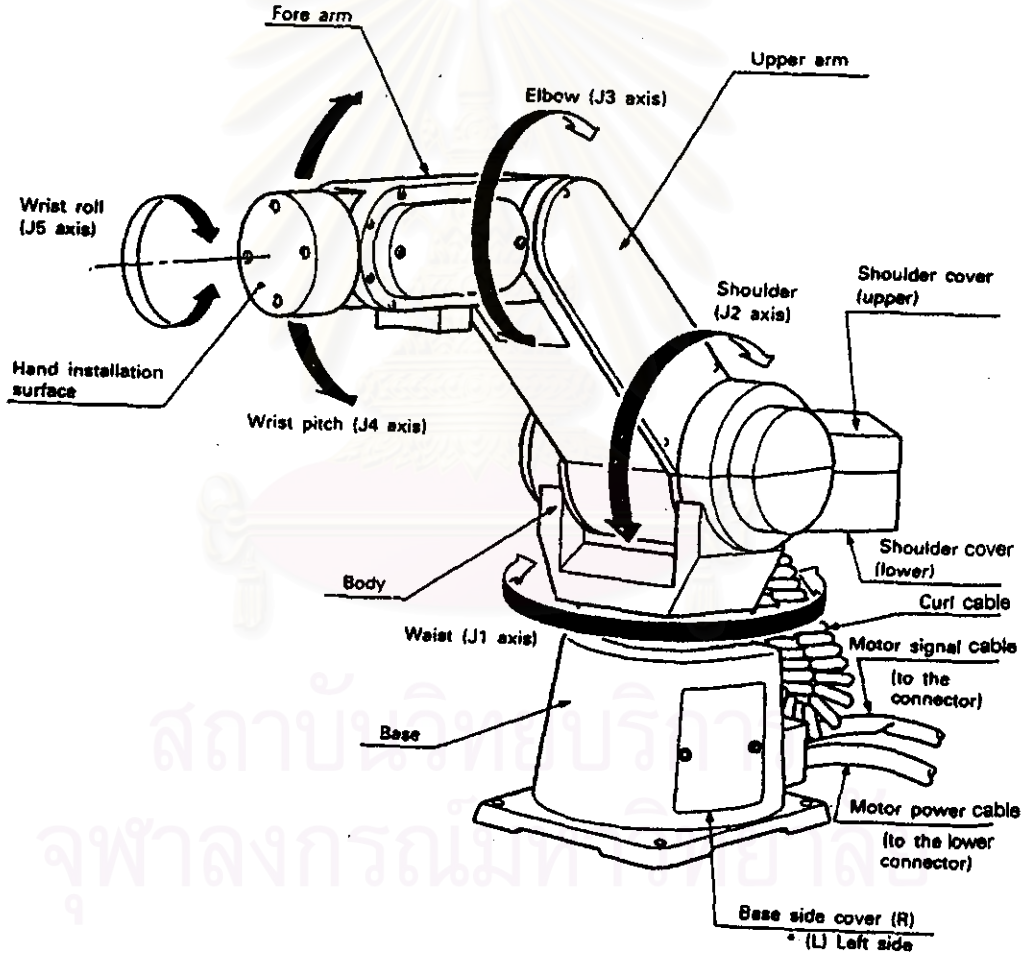
ภาคผนวก ก.

ลักษณะของหุ่นยนต์ที่ใช้ในการทำวิทยานิพนธ์ (MITSUBISHI ROBOT RV-M1 (MOVEMASTER EX)

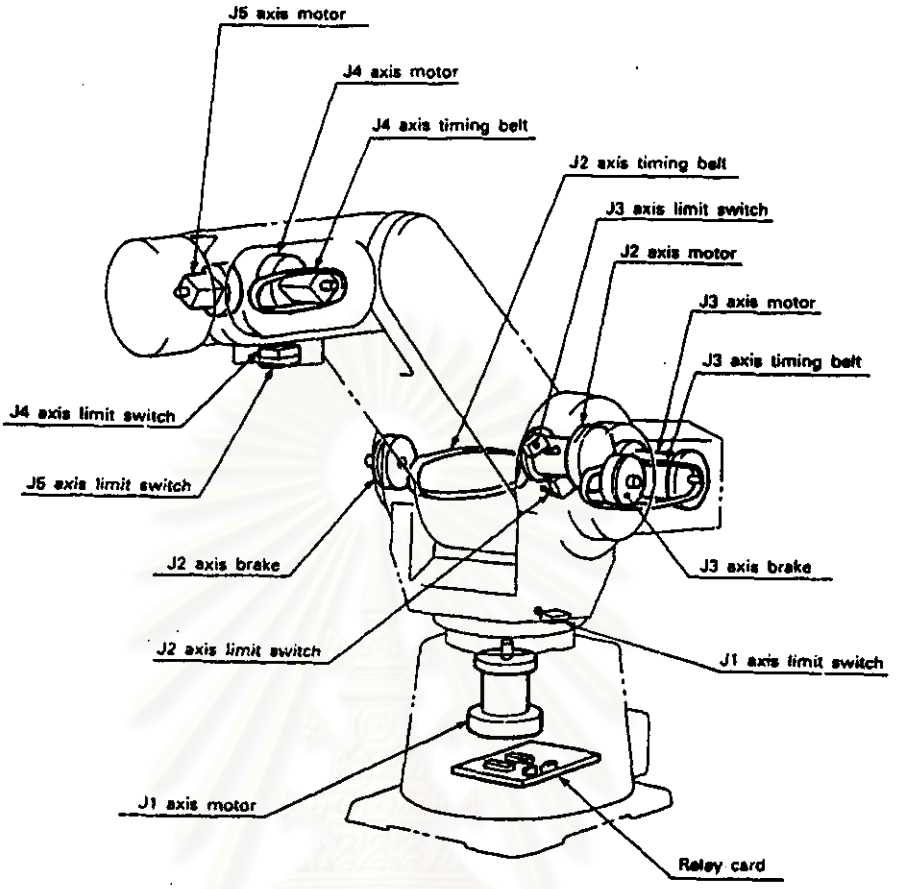
1) รายละเอียด (Specifications)

1.1) ลักษณะของหุ่นยนต์ (Robot specitications)

1.1.1) วิธีการกำหนดชื่อ (Nomenclature)

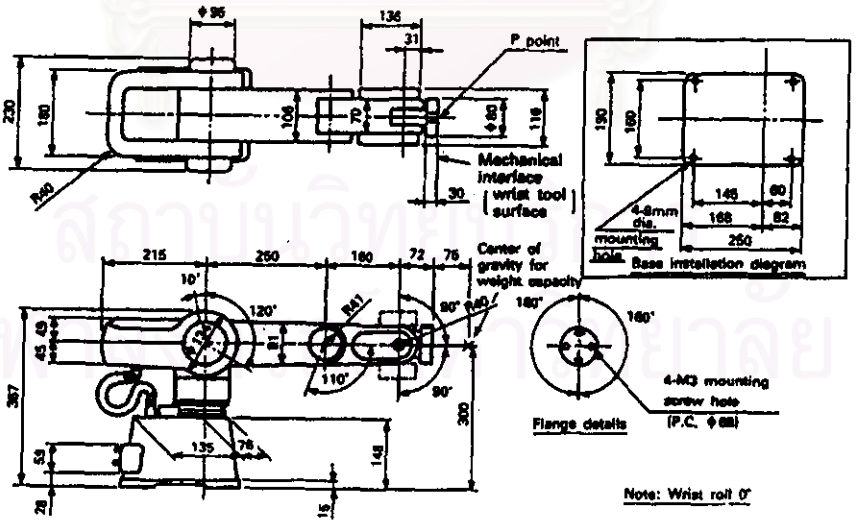


รูปที่ ก-1 แสดงวิธีการกำหนดชื่อ (ลักษณะภายนอก)



รูปที่ ก-2 แสดงวิธีการกำหนดชื่อ (ลักษณะภายใน)

1.1.3) ขนาดมิติภายนอก (External Dimensions)



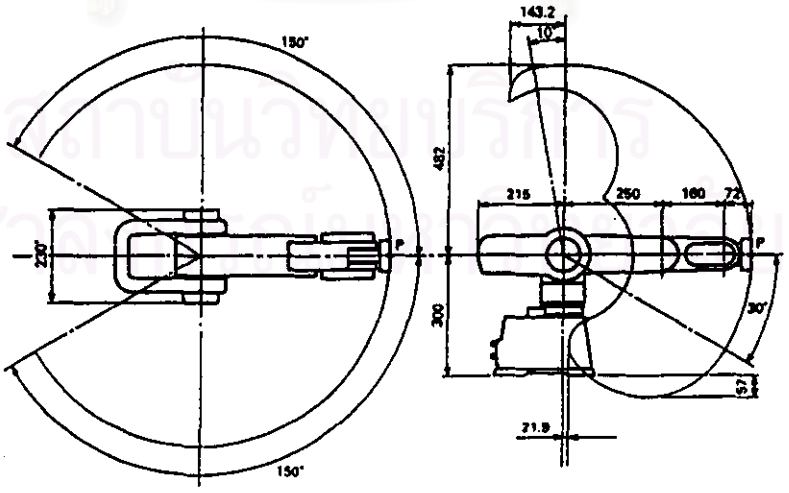
รูปที่ ก-3 แสดงขนาดมิติภายนอก

1.1.2) รายละเอียดมาตรฐาน (Standard Specification)

Item		Specifications	Remarks
Mechanical Structure		5 degrees of freedom, vertical articulated ro bot	
Operation range	Waist rotation	300 (max.120 /sec)	J1 axis
	Shoulder rotation	130 (max.72 /sec)	J2 axis
	Elbow rotation	110 (max.109 /sec)	J3 axis
	Wrist pitch	90 (max.100 /sec)	J4 axis
	Wrist roll	180 (max.163 /sec)	J5 axis
Arm length	Upper arm	250 mm	
	Fore arm	160 mm	
Weight capacity		Max . 1.2kgf (including the hand weight)	75mm from the mechanical interface (center of gravity)
Maximum path velocity		1000 mm/sec (wrist tool surface)	Speed at point P in Fig. n3
Position repeatability		0.3mm (roll center of the wrist tool surface)	Accuracy at point P in Fig. n3
Drive system		Electrical servo drive using DC servo motors	
Robot weight		Approx. 19kgf	
Motor capacity		J1 to J3 axes: 30W; J4, J5 axes: 11W	

ตารางที่ ก-1 แสดงรายละเอียดมาตรฐาน

1.1.4) ขอบเขตการปฏิบัติการ (Operation Space)



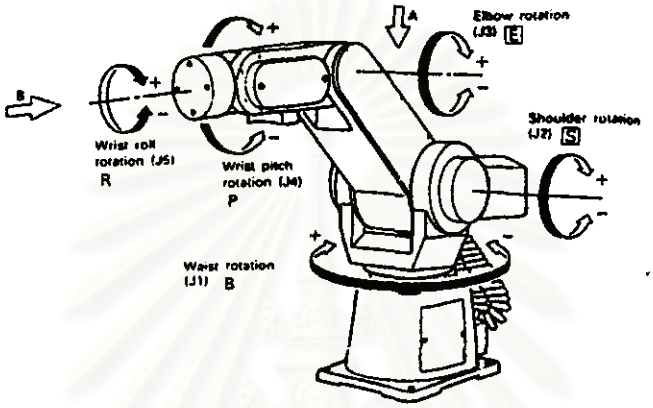
รูปที่ ก-4 แสดงขอบเขตการปฏิบัติการ

โดยมี<u>ข้อกำหนด</u>ดังต่อไปนี้

- ขอบเขตการปฏิบัติการที่แสดงในรูปที่ นั้นจะเป็นขอบเขตเมื่อหุ่นยนต์ไม่ได้ติดอุปกรณ์จับยึด

- การปฏิบัติการจะต้องกระทำอย่างระมัดระวังเป็นพิเศษโดยเฉพาะการกระแทกระหว่างข้อมือ และฐานของ หุ่นยนต์ หรือพื้นระนาบ

1.1.5) การปฏิบัติการพื้นฐาน (Basic Operation)

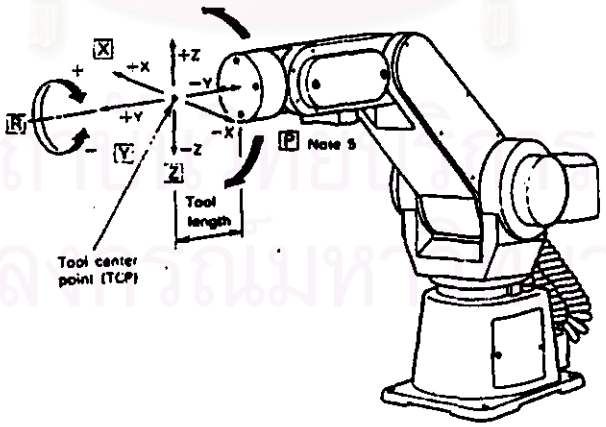


รูปที่ ก-5 แสดงการปฏิบัติการในระบบข้อต่อ (Articulated System)

<u>ข้อกำหนด</u>

- การปฏิบัติการในแกน J1 และ J5 จะมีทิศทางเป็นบวกเมื่อหมุนตามเข็มนาฬิกาจากการมองตามลูกศร A และ B

- การปฏิบัติการในแกน J2, J3 และ J4 จะมีทิศทางเป็นบวกเมื่อมีทิศขึ้น จากแขน (arm) และข้อมือ (wrist)

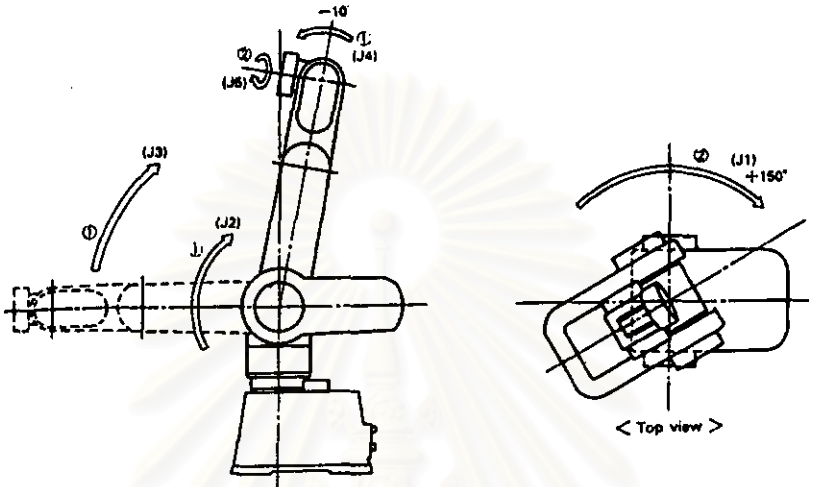


รูปที่ ก-6 แสดงการปฏิบัติการในระบบพิกัดคาร์ทีเซียน (Cartesian Coordinate System)

ข้อกำหนด

- จุด TCP (Tool Center Point) จะเคลื่อนที่เป็นเส้นตรงในระบบพิกัดคาร์ทีเซียน
- ความยาวของอุปกรณ์ (Tool Length) จะถูกกำหนดโดยค่าพารามิเตอร์

1.1.6 ตำแหน่งเริ่มต้น (Origin Set)



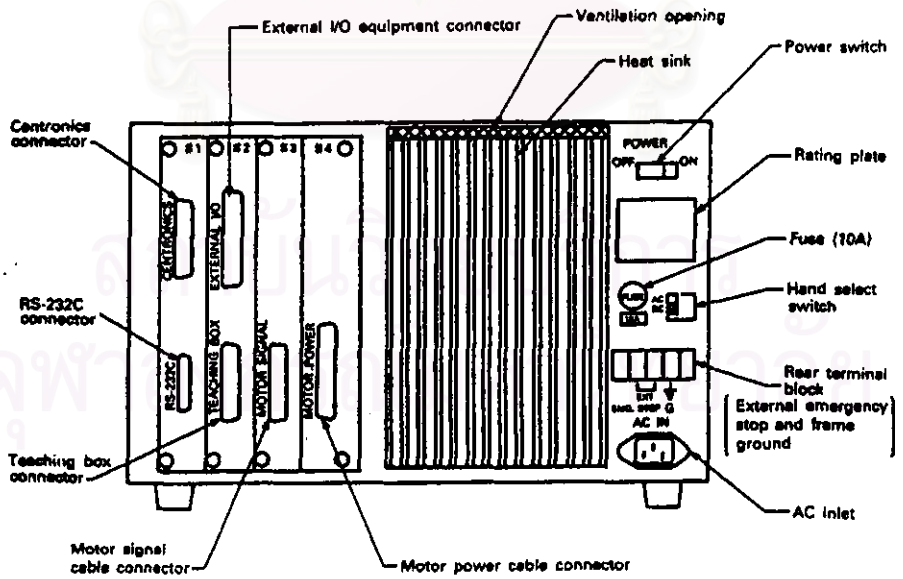
รูปที่ ก-7 แสดงตำแหน่งเริ่มต้นของหุ่นยนต์

ข้อกำหนด

- หุ่นยนต์จะต้องเคลื่อนที่ไปจุดเริ่มต้น หลังจากการเปิดเครื่องทุกครั้ง

1.2) ลักษณะของส่วนควบคุม (Drive Unit Specifications)

1.2.1) วิธีการกำหนดชื่อ (Nomenclature)



รูปที่ ก-8 แสดงวิธีการกำหนดชื่อ (ส่วนควบคุม)

1.2.2) รายละเอียดมาตรฐาน (Standard Specifications)

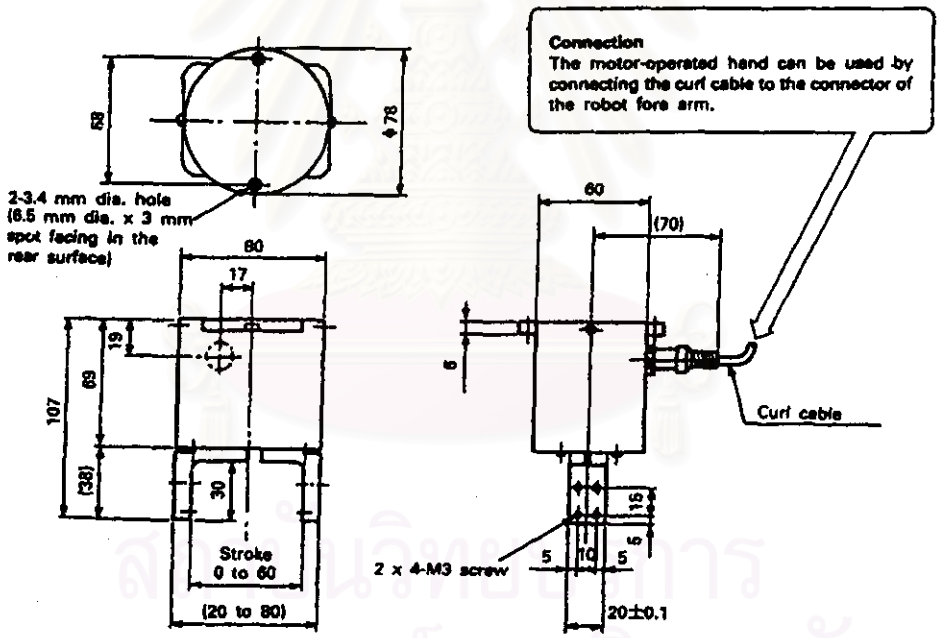
Item	Specifications
Teaching method	Programming language system (63 commands), MDI (using a personal c omputer)
Control method	PTP position control system using DC servo motors
Number of control axes	5 axes (+1 optional axis)
Position detection	Pulse encoder system
Return to Origin (Origin setting)	Limit switches and pulse encoders (Z phase detection method)
Interpolation function	Articulate interpolation , linear interpolation
Speed setting	10 steps (max. 1000mm/sec)
Number of positions	629 (8KB)
Number of program steps	2048 (16KB)
Data storage	Write to EP-ROM using the built-in EP- ROM writer or storage in the battery- backed static RAM(the battery is optional and backs up the RAM for about 2 years)
Position teaching equipment	Teaching box (option) or personal computer
Programming equipment	Personal computer
External I/O	General-purpose I/O, 8 points each (16-point type available) General-purpose synchronous signals (STB,BUSY,ACK,RDY) No dedicated I/O (dedicated I/O of 3 points each available) Power for external I/O should be prepared by the user (12V to 24V dc)
Interface	1 parallel interface (conforming to Centronics) 1 serial interface (conforming to RS-232C)
Emergency stop	Using any of the front control switch, teaching box switch, and rear terminal block (N/C contact terminal)
Hand control	Motor-operated hand or pneumatically-operated hand using AC solenoid)
Brake control	J2 axis (shoulder) , J3 axis (elbow)
Power source	120V/220V/230V/240V AC 10%, 0.5KVA
Ambient temperature	5 C to 40 C
Humidity	10 to 85%, non-condensing.
Weight	nrox. 23kgf
Size	380 (W) x 331 (D) x 246 (H) mm

ตารางที่ ก-2 แสดงรายละเอียดมาตรฐานของส่วนควบคุม

1.3) อุปกรณ์จับโดยใช้มอเตอร์ (Motar-operated Hand)

Item	Specification	Remarks
Type	HM-01	
Drive system	DC servo motor drive	
Opening/closing stroke	0 to 60 mm	The gripping force can be set in 12 st eps
Grip power	Max. 3.5kgf	4<starting/retained gripping force<15
Ambient temperature	5 to 40 C	(see command GP)
Service life	More than 300,000 times	
Weight	600gf	

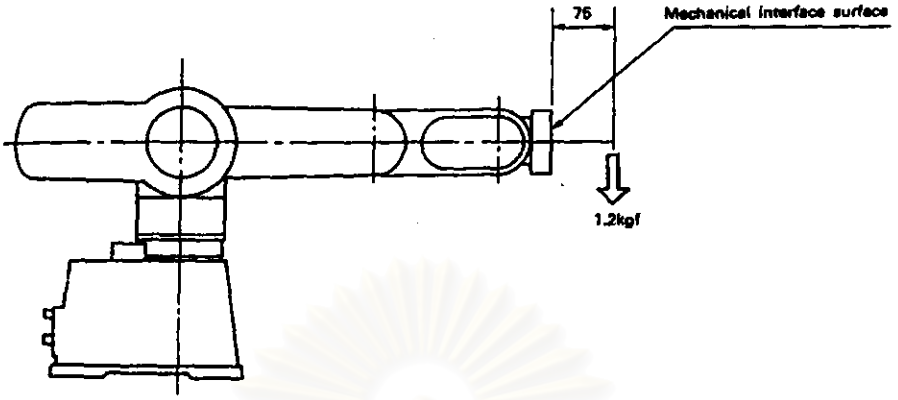
ตารางที่ ก-3 แสดงลักษณะของอุปกรณ์จับโดยใช้มอเตอร์



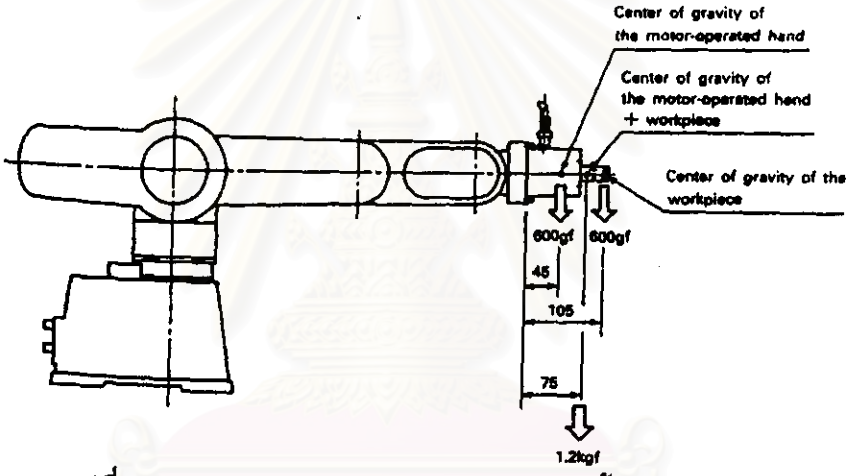
รูปที่ ก-9 แสดงขนาดมิติภายนอก (อุปกรณ์จับโดยใช้มอเตอร์)

1.4) ความสามารถในการรับน้ำหนัก (Weight Capacity)

1.4) ความสามารถในการรับน้ำหนัก (Weight Capacity)



รูปที่ ก-10 แสดงความสามารถในการรับน้ำหนักของหุ่นยนต์ซึ่งกำหนดอยู่ในคู่มือการใช้งาน



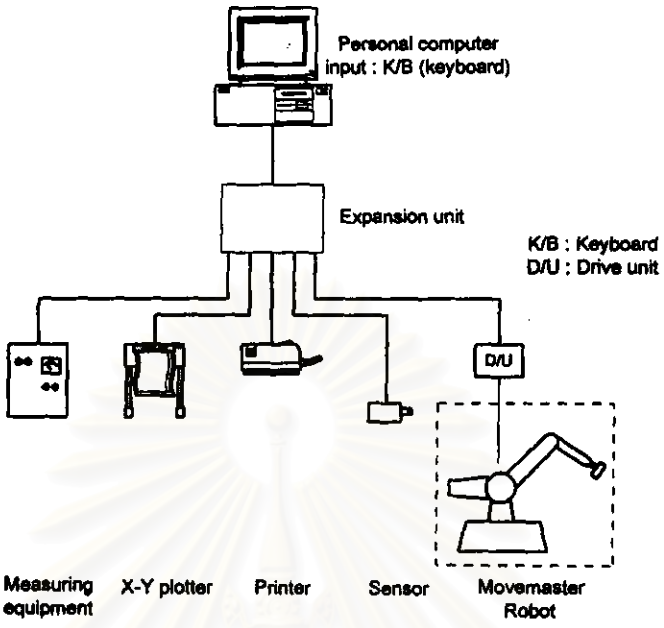
รูปที่ ก-11 แสดงความสามารถในการรับน้ำหนักของหุ่นยนต์ เมื่อติดตั้งอุปกรณ์จับโดยใช้มอเตอร์

2) การปฏิบัติการ (Operation)

2.1) ลักษณะการจัดวางระบบ (System Configuration)

2.1.1) ลักษณะการจัดวางระบบโดยมีคอมพิวเตอร์ส่วนบุคคลเป็นศูนย์กลาง (System configuration centering around a personal computer)

ระบบนี้จะจับวางโดยมีคอมพิวเตอร์ส่วนบุคคลเป็นตัวควบคุมการเคลื่อนที่ของหุ่นยนต์ โดยใช้ภาษาเฉพาะสำหรับหุ่นยนต์เป็นคำสั่งควบคุม ซึ่งในระบบนี้คอมพิวเตอร์จะทำหน้าที่เสมือนเป็นสมอง ในการจัดการให้หุ่นยนต์ทำงานลักษณะต่างๆ เช่น การประกอบ หรือการทดลองต่างๆ เป็นต้น

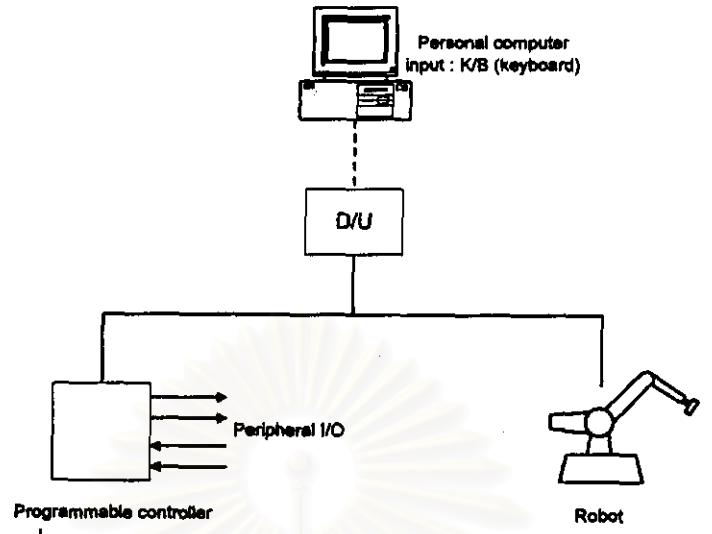


รูปที่ ก-12 แสดงลักษณะการวางระบบโดยมีคอมพิวเตอร์ส่วนบุคคลเป็นศูนย์กลาง

2.1.2) ลักษณะการวางระบบโดยมีส่วนควบคุมเป็นศูนย์กลาง

(System Configuration centering around the drive unit)

การวางระบบลักษณะนี้จะใช้ส่วนควบคุมเป็นตัวควบคุมการทำงานของหุ่นยนต์ โดยที่คอมพิวเตอร์ส่วนบุคคลจะใช้เพื่อจุดประสงค์ในการเขียนโปรแกรมเท่านั้น โดยที่โปรแกรมที่เขียนขึ้นดังกล่าวจะถูกโอนไปยังส่วนควบคุม เพื่อที่จะสั่งให้หุ่นยนต์ทำงานในภายหลัง นั่นก็คือไม่มีความจำเป็นในการติดตั้งคอมพิวเตอร์ส่วนบุคคลในพื้นที่การผลิต การแลกเปลี่ยนสัญญาณระหว่างหุ่นยนต์ และอุปกรณ์รอบนอกอื่นๆเช่น ลิมิตสวิตช์,รีเลย์,LEDS และ Programmable Logic Controllers จะถูกส่งผ่านพอร์ตส่งข้อมูลภายนอก (external I/O port) ภายในส่วนควบคุม เราจะสามารถเปลี่ยนแปลงโปรแกรมที่เก็บไว้ใน built-in EPROM โดยการแทนที่โปรแกรมเดิมด้วยโปรแกรมใหม่ การประยุกต์ใช้งานจะเกี่ยวข้องกับงานในสายการผลิต สถานีตรวจสอบต่างๆ ภายในโรงงาน



รูปที่ ก-13 แสดงลักษณะการวางระบบโดยมีส่วนควบคุมเป็นศูนย์กลาง

2.2) การเชื่อมต่อระหว่างหุ่นยนต์และคอมพิวเตอร์ (Robot-Computer link)

2.2.1) Centronics intertace

เป็นมาตรฐานการต่อขนานเริ่มต้นสำหรับเครื่องพิมพ์ ซึ่งถูกคิดค้นโดยบริษัท Centronics Corporation เครื่องพิมพ์และเครื่องพล็อตเตอร์ X-Y ส่วนใหญ่จะสนับสนุนมาตรฐานนี้ โดยคอมพิวเตอร์จะส่งข้อมูล 8 บิตไปพร้อมๆ กัน หรือขนานกัน โดยมีสายส่งข้อมูลเป็นตัวควบคุมการไหลของข้อมูล

ด้วยข้อจำกัดเกี่ยวกับระยะทางในการส่งข้อมูล (1-2 เมตร) การส่งข้อมูลแบบขนานนี้จะมีความเร็วในการส่งที่ค่อนข้าสูง และไม่ต้องการการติดตั้งอุปกรณ์กรณ์อย่างอื่นเป็นพิเศษ ซึ่งทำให้สะดวกในการใช้งาน ในการต่อเชื่อมกับหุ่นยนต์จะมี interface ที่คล้ายคลึงกับเครื่องพิมพ์ ซึ่งหมายความว่า ข้อมูลจะถ่ายเทไปในทางเดียวจาก คอมพิวเตอร์ไปยังหุ่นยนต์ดังนั้นการใช้คำสั่งบางคำสั่ง (คำสั่งในการอ่านข้อมูลจากหุ่นยนต์ เช่น WH, PR, LR) จะไม่สามารถใช้งานได้

การส่งข้อมูลจะทำโดยการใช้ข้อความ LPRINT ในภาษา BASIC

2.2.2) RS-232C interface

เป็นมาตรฐานดั้งเดิมในการส่งข้อมูลโดยใช้สายโทรศัพท์ และพัฒนามาเป็นมาตรฐานการส่งข้อมูล อนุกรมสำหรับคอมพิวเตอร์และอุปกรณ์ประกอบอื่นๆ

ในการส่งข้อมูลแบบอนุกรม ข้อมูลจะส่งไปตามสายข้อมูลเดียวซึ่งจะทำให้ส่งข้อมูลไปได้ 1 บิตในเวลาใดเวลาหนึ่ง ดังนั้นการส่งข้อมูลประเภทนี้จะช้ากว่าการส่งข้อมูลแบบ

ขนานถ้าอัตราบอด (Baud rate) มีค่าต่ำ การกำหนดค่าต่างๆ ของหุ่นยนต์จะต้องสอดคล้องกับคอมพิวเตอร์ และก็ไม่สามารถจะต่อกับคอมพิวเตอร์ได้ทุกเครื่อง

เพราะว่าความสามารถในด้านการส่งข้อมูลได้ 2 ทิศทาง ในขณะที่คอมพิวเตอร์สามารถอ่านข้อมูลภายในหุ่นยนต์ได้ การส่งข้อมูลแบบอนุกรมจึงสามารถส่งข้อมูลได้ไกลกว่าแบบขนาน โดยส่งได้ตั้งแต่ 3-15 เมตร และสามารถใช้งานได้ถึงแม้ว่าพอร์ตขนานของคอมพิวเตอร์จะต่ออยู่กับเครื่องพิมพ์หรืออุปกรณ์ประเภทอื่น ในภาษา BASIC จะสามารถส่งข้อมูลได้โดยใช้ข้อความ OPEN, PRINT #, และข้อความ LINE INPUT #

2.3) รูปแบบการควบคุม (Control Modes)

2.3.1) รูปแบบคอมพิวเตอร์ส่วนบุคคล (Personal Computer mode)

รูปแบบนี้จะใช้คอมพิวเตอร์ส่วนบุคคลในการจัดการคำสั่งโดยตรงไม่ว่าจะเป็นการเขียน , การถ่ายโอนโปรแกรม และการเริ่มต้นโปรแกรมโดยการส่งคำสั่งไปยัง RAM ของส่วนควบคุม (วิธีนี้จะตรงกับลักษณะการวางระบบโดยมีคอมพิวเตอร์เป็นศูนย์กลางในหัวข้อที่ 2 .1.1)

การปฏิบัติการจะแบ่งเป็น 3 ส่วน โดยจะแสดงในรูปภาษา BASIC

(1) การกระทำโดยตรง (Direct execution)

เป็นการกระทำคำสั่งของหุ่นยนต์โดยตรง เช่น เคลื่อนที่หุ่นยนต์ไปยังจุดที่สอนไว้ (ตำแหน่ง 1) โดยใช้คำสั่ง "MO" (move) รูปแบบกำลังจะเป็น

"MO 1" (เคลื่อนที่ไปยังตำแหน่งที่ 1)

โดยจะสั่งเป็น ASCII Codes โดยจะตรงกับ

LPRINT "MO 1" ในการเชื่อมต่อแบบ Centronics

และรูปแบบ

PRINT # 1, "MO 1" ในการเชื่อมต่อแบบ RS-232 C

คำสั่งต่างๆ เหล่านี้จะถูกส่งไปเป็นส่วนๆ อย่างต่อเนื่อง และทำทีละบรรทัด โดยจะไม่มีการสร้างเป็นโปรแกรมเก็บไว้ ในส่วนควบคุม

(2) การสร้างโปรแกรม (Program generation)

เป็นการสร้างโปรแกรมที่เกิดจากคำสั่งของหุ่นยนต์ โดยที่โปรแกรมจะถูกเก็บไว้ใน RAM ของส่วนควบคุม ยกตัวอย่างในกรณีเดียวกับข้างบน รูปแบบคำสั่งจะเป็น

"10 MO 1"

โดย "10" จะแสดงถึงบรรทัดของโปรแกรม ซึ่งจะเป็นการกำหนดขั้นตอนการจัดเก็บในส่วนความจำ จำนวนบรรทัดจะมีค่าได้ตั้งแต่ 1 จนถึง 2048 และเมื่อเทียบกับข้างบนจะได้ว่า

LPRINT "10 MO 1" ในการเชื่อมต่อแบบ Centronics และ

PRINT # 1 "10 MO 1" ในการเชื่อมต่อแบบ RS-232C

(3) การกระทำโปรแกรม (Program execution)

เป็นการสั่งให้โปรแกรมที่ถูกเก็บไว้ใน RAM ของหุ่นยนต์ทำงานโดยการใช้คำสั่ง "RN" (RUN) โดยที่ในรูปแบบ Centronics จะเป็น

LPRINT "RN" และในรูปแบบ RS-232C จะเป็น

PRINT #1 "RN"

ตัวอย่างที่ 1 การกระทำโดยตรง (Centronics)

```
100  LPRINT "NT"          การกำหนดจุดเริ่มต้น
110  LPRINT "SP 7"        กำหนดความเร็วระดับ 7
120  LPRINT "MO 10, 0"    เคลื่อนไปยังตำแหน่ง 10 โดยมือจับเปิด
130  LPRINT "GC"          ปิดอุปกรณ์จับยึด
140  LPRINT "MO 11, C"    เคลื่อนที่ไปยังตำแหน่ง 11 โดยมือจับปิด
150  END                  สิ้นสุดโปรแกรม BASIC
RUN                       สั่งให้โปรแกรม BASIC ทำงาน
OK
```

คำสั่ง "RUN" จะกระทำโปรแกรมภาษา BASIC โดยจะเปลี่ยนเป็นการกระทำโปรแกรมของหุ่นยนต์โดยตรงทีละบรรทัด

ตัวอย่างที่ 2 การสร้างโปรแกรมให้ทำงาน (Centronics)

```
100  LPRINT "10 NT"
110  LPRINT "12 SP 7"
120  LPRINT "14 MO 10, 0"
130  LPRINT "16 GC"
140  LPRINT "18 MO 11, C"
150  LPRINT "20 ED"       สิ้นสุดโปรแกรมหุ่นยนต์ Movemaster
160  END                  สิ้นสุดโปรแกรม BASIC
RUN                       สั่งให้โปรแกรม BASIC ทำงาน
OK
LPRINT "RN"               สั่งให้โปรแกรมหุ่นยนต์ Move master ทำงาน
```

คำสั่ง "RUN" จะสั่งให้กระทำโปรแกรมภาษา BASIC โดยจะเปลี่ยนเป็นการถ่ายโอนโปรแกรมของหุ่นยนต์ลงในส่วนควบคุม ในตอนนี้หุ่นยนต์ยังไม่เคลื่อนที่เมื่อคำสั่ง "RN" ถูกถ่ายโอนจะส่งผลให้หุ่นยนต์เคลื่อนที่ตามต้องการ

ตัวอย่างที่ 3 การสร้างโปรแกรมให้ทำงาน (RS-232C)

```
100 OPEN "COM 1 : 9600, E, 7,2"AS # 1, for Mitsubishi MULTI 16.
110 PRINT # 1, "10 NT"
120 PRINT # 1, "12 SP 7"
130 PRINT # 1, "14 MO 10"
140 PRINT # 1, "16 MO 11"
150 PRINT # 1, "18 MO 12"
160 PRINT # 1, "20 TI 30"
170 PRINT # 1, "22 GR 14"
180 END                    ; จบโปรแกรม BASIC
RUN                        ; สั่งให้โปรแกรม BASIC ทำงาน
OK
```

2.3.2 รูปแบบส่วนควบคุม (Drive unit mode)

รูปแบบนี้จะเป็นการเก็บบันทึกโปรแกรมไว้ใน EPROM หรือ RAM ของหุ่นยนต์เพื่อใช้ในการดำเนินการ จะใช้สวิตช์ควบคุมด้านหน้าของส่วนควบคุมในการเริ่ม หรือหยุดโปรแกรม

ตัวอย่างที่ 4 ตัวอย่างโปรแกรมที่ใช้ (Movemaster program)

```
10 NT          กำหนดจุดเริ่มต้น
12 SP 7        กำหนดความเร็วระดับ 7
14 MO 10, 0    เคลื่อนไปยังตำแหน่ง 10 โดยมือจับเปิด
16 MO 11, C    เคลื่อนไปยังตำแหน่ง 11 โดยมือจับเปิด
18 MO 12, C    เคลื่อนไปยังตำแหน่ง 12 โดยมือจับเปิด
20 TI 30       หยุดเป็นเวลา 3 วินาที
22 GT 14       ข้ามไปยังบรรทุดที่ 14
```

3) คำอธิบายตัวอย่างคำสั่ง (Description of the command)

คำสั่งต่างๆ ของหุ่นยนต์จะสามารถ แบ่งเป็นหมวดหมู่ขึ้นอยู่กับลักษณะการทำงานดังต่อไปนี้

- คำสั่งที่ควบคุมตำแหน่งและการเคลื่อนที่ (Position/motion control instruction) 24 คำสั่ง

เป็นคำสั่งที่เกี่ยวกับตำแหน่งและการเคลื่อนที่ของหุ่นยนต์ ซึ่งจะประกอบไปด้วย การกำหนดรูปแบบ , การคำนวณรูปแบบการเคลื่อนที่ที่เกิดจากผลของการเคลื่อนที่แบบ การคำนวณเชิงเส้น (linear interpolations motion) หรือการเคลื่อนที่แบบทางเดินต่อเนื่อง (continuous-path motion) รวมถึงการกำหนดความเร็ว , การกำหนดจุดเริ่มต้น

- คำสั่งที่ควบคุมโปรแกรม (Program control instruction) 19 คำสั่ง

เป็นคำสั่งที่ควบคุมขั้นตอนการทำงานของโปรแกรม ซึ่งจะประกอบไปด้วยส่วนโปรแกรมย่อย (subroutines) , ลูปการทำซ้ำ (repetitive loops) และการเลือกแบบมีเงื่อนไข (conditional jumps) นอกจากนี้ยังรวมถึงรูปแบบการทำงานของเคาน์เตอร์ และการกำหนดการทำงานของอินเทอรัปต์ เมื่อมีสัญญาณควบคุมจากภายนอก

- คำสั่งควบคุมมือจับ (Hand control instuction) 4 คำสั่ง

เป็นคำสั่งที่ควบคุมมือจับของหุ่นยนต์ , ควบคุมแรงที่ใช้ในการจับชิ้นงาน และเวลาที่ใช้ในการจับ หรือ ปล่อยของอุปกรณ์จับชิ้นงาน

- คำสั่งควบคุมการติดต่อกับอุปกรณ์ภายนอก (I/O control instrution) 6 คำสั่ง

เป็นคำสั่งที่เกี่ยวกับการสื่อสารข้อมูลเข้าและออกผ่าน อินพุต/เอาท์พุต พอร์ต ซึ่งข้อมูลจะสามารถเปลี่ยนแปลงอย่างพร้อมกัน (Synchronously) หรือไม่พร้อมกัน (asynchronously) และอาจจะสามารถทำงานที่ละบิตข้อมูลหรือหลายบิตข้อมูลพร้อมกัน (parallel)

- คำสั่งที่ใช้ในการอ่าน RS 232C (RS-232C read instructions) 6 คำสั่ง

เป็นคำสั่งที่จะทำให้คอมพิวเตอร์ สามารถอ่านข้อมูลจากหน่วยความจำของหุ่นยนต์ ซึ่งข้อมูลดังกล่าวจะประกอบไปด้วย ข้อมูลตำแหน่ง , ข้อมูลโปรแกรม, ข้อมูลเคาน์เตอร์, สัญญาณจากภายนอก และสภาวะความผิดพลาด

- คำสั่งทั่วไป (Miscellancous) 4 คำสั่ง

เป็นคำสั่งที่เกี่ยวกับการรีเซทค่าผิดพลาด , คำสั่งการอ่านหรือเขียนของผู้ปฏิบัติการ และคำสั่งที่เกี่ยวกับการเขียนคำอธิบาย

ตัวอย่างการอธิบายความหมายของคำสั่ง (บางคำสั่งที่ใช้งาน)

3.1) คำสั่งที่ใช้ควบคุมตำแหน่ง และการเคลื่อนที่

3.1.1) HO (Home)

ความหมาย เป็นการกำหนดจุดอ้างอิงในระบบพิกัดฉาก

รูปแบบ HO

คำอธิบาย คำสั่งนี้จะเป็นการกำหนดจุดอ้างอิง (ค่า X, Y, Z และมุมพิทช์ , โรล) ในระบบแกนพิกัดฉากโดยที่การเคลื่อนที่ต่างๆ จะเทียบกับจุดอ้างอิงดังกล่าว

ตัวอย่างคำสั่ง 10 LPRINT "HO"

3.1.2) MC (Move Continuous)

ความหมาย สั่งให้หุ่นยนต์เคลื่อนที่อย่างต่อเนื่องโดยผ่านจุดภายในระหว่างจุด 2 จุดที่กำหนดตำแหน่งการเคลื่อนที่

รูปแบบ MC <Position number (a)>, <Position number (b)>

โดยที่ $1 \leq$ Position number (a), (b) ≤ 692

$1 \leq$ Position number (a) - Position number (b) $1 \leq 99$

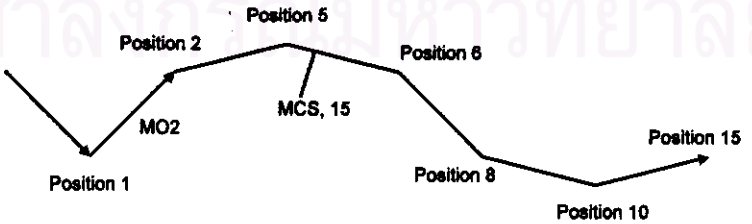
คำอธิบาย

- คำสั่งนี้จะส่งให้หุ่นยนต์ เคลื่อนที่ด้วยความเร็วที่กำหนด จากจุดที่อยู่ปัจจุบัน (ตำแหน่ง a) ไปยังจุดปลาย (ตำแหน่ง b) โดยเคลื่อนที่อย่างต่อเนื่องผ่านจุดภายใน (Intermediate points) ที่กำหนดไว้ระหว่างตำแหน่ง a และ b

- สภาวะเปิด/ปิด ของมือจับก่อนที่หุ่นยนต์เคลื่อนที่จะมีค่าคงเดิมตลอดการเคลื่อนที่ โดยที่คำสั่งเปิด/ปิด ของมือจับจะไม่มีผลที่จุดภายในเหล่านั้น

ตัวอย่างคำสั่ง

```
10 LPRINT "MO 1"
20 LPRINT "MO 2"
30 LPRINT "SP 3"
40 LPRINT "MC5, 15"
```



รูปที่ ก-14 แสดงลักษณะทางเดินแบบต่อเนื่อง

บรรดทัดที่ 10 และ 20 จนเป็นการสั่งให้หุ่นยนต์เคลื่อนไปตำแหน่ง 1 และ 2 ตามลำดับ และบรรทัดที่ 40 จะเป็นการสั่งให้หุ่นยนต์เคลื่อนที่อย่างต่อเนื่อง ผ่านจุดภายใน 6, 8, 10 ระหว่างจุด 5 และ 15 โดยจะหยุดที่ตำแหน่ง 15

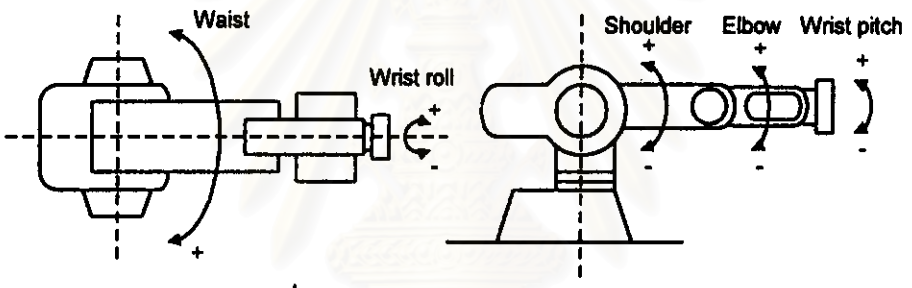
3.1.3) MJ (Move Joint)

ความหมาย เป็นคำสั่งที่สั่งให้หุ่นยนต์หมุนข้อต่อ (Joint) ด้วยมุมที่กำหนดจากตำแหน่งปัจจุบัน

รูปแบบ MJ [< Waist turing angle >] , [< Shoulder turning angle >] , [< Elbow turning angle >] , [< Pitch angle >] , [< Roll angle >]

คำอธิบาย

- ค่าของมุมแตกต่างที่น้อยที่สุดมีค่าเท่ากับ 0.1
- ทิศทางบวก หรือลบ ของแกนในการเคลื่อนที่ แสดงดังรูป



รูปที่ ก-15 แสดงทิศทางการหมุนของข้อต่อ

ตัวอย่างคำสั่ง

```
10 LPRINT "MJ + 90, 0, 0, 0, 0"
20 LPRINT "MJ 0, - 30, 0, 0, 0"
30 LPRINT "MJ 0, 0, 0, +20, 0"
```

บรรทัดที่ 10 จะสั่งให้หมุน waist ไปด้วยมุม 90 ในทิศทางบวก บรรทัดที่ 20 จะสั่งให้หมุน Shoulder ไปด้วยมุม 30 ในทิศทางลบ และบรรทัดที่ 30 ซึ่งให้หมุน writst ด้วยมุม 20" ในทิศทางบวก

3.1.4) MO (Move)

ความหมาย เป็นคำสั่งที่ให้หุ่นยนต์เคลื่อนที่ ปลายของมือจับไปตำแหน่งที่ต้องการ

รูปแบบ MO (Position number) [,(O or C)]

โดยที่ 1 ≤ (Position number) ≤ 629

คำอธิบาย

-คำสั่งนี้จะใช้เพื่อเคลื่อนปลายของมือจับไปยังตำแหน่งที่ได้กำหนดไว้ โดยใช้วิธี "articulated interpolation" ตำแหน่งของส่วนปลายของมือจับนั้นจะสามารถหาได้จากระยะอุปกรณ์ (tool length) ที่ได้กำหนดไว้

- ถ้ามีการกำหนดสภาวะ ปิด/เปิดของมือจับ (0 : Open ; C : Closed) หุ่นยนต์จะทำคำสั่งเกี่ยวกับมือจับดังกล่าวก่อนที่จะเคลื่อนที่ต่อไป

ตัวอย่างคำสั่ง

```
10 LPRINT "SP 5"        กำหนดความเร็วระดับ 5
20 LPRINT "MO 20, C"    เคลื่อนที่ไปยังตำแหน่งที่ 20 ด้วยสภาวะมือจับปิด
30 LPRINT "MO 30, 0"    เคลื่อนที่ไปยังตำแหน่งที่ 30 ด้วยสภาวะมือจับเปิด
```

3.1.5) MP (Move Position)

ความหมาย เป็นคำสั่งที่ใช้ในการเคลื่อนปลายของมือจับไปยังตำแหน่งที่กำหนดในระบบแกนพิกัดฉาก (ตำแหน่งและมุม)

รูปแบบ MP [(X-axis coordimate)] , [(Y-axis coordinate)] , [(Z-axis coordinate)] , [(Pitch angle)] , [(Roll angle)]

คำอธิบาย

- สามารถกำหนดเลขนัยสำคัญได้สูงสุดเท่ากับ 0.1 mm หรือ 0.1
- สภาวะเปิด/ปิดของมือจับจะไม่เปลี่ยนแปลงก่อนและหลังการเคลื่อนที่
- ตำแหน่งของปลายของมือจับจะสามารถหาได้จากคำสั่ง TL

ตัวอย่างคำสั่ง

```
10 LPRINT "PD 1, 0, 380, 300, -70, -40"
20 LPRINT "MO 1"
30 LPRINT "MP 0, 380, 300, -70, -40"
```

บรรทัดที่ 10 เป็นการกำหนดตำแหน่งในระบบพิกัดฉาก และสั่งให้เคลื่อนที่ไปยังตำแหน่งดังกล่าวด้วยคำสั่งในบรรทัดที่ 20 และบรรทัดที่ 30 เป็นคำสั่งให้เคลื่อนที่มือจับลงในแนวแกน Z โดยที่สภาวะเปิด/ปิดของมือจับไม่มีการเปลี่ยนแปลง

3.1.6) MT (Move Tool)

ความหมาย เป็นคำสั่งที่ใช้ในการเคลื่อนปลายของมือจับ จากตำแหน่งปัจจุบันไปยังตำแหน่งที่กำหนด โดยบอกเป็นค่าระยะทางที่เปลี่ยนไปในทิศทางของเครื่องมือ (tool direction)

รูปแบบ MT (Position number), [(Travel distance)] [,(O or C)]

โดยที่ 1 ≤ Position number ≤ 629

คำอธิบาย

- สามารถกำหนดเลขนัยสำคัญได้สูงสุดเท่ากับ 0.1 mm

- เมื่อใส่ค่าระยะทางที่เปลี่ยนเป็นบวก จะทำให้ปลายของมือจับเคลื่อนที่ไปตามทิศทางของเครื่องมือ ในขณะที่ถ้าระยะทางเป็นลบ ปลายของมือจับจะเคลื่อนที่ไปในทิศตรงข้ามกับทิศทางของเครื่องมือ

- ถ้ามีการกำหนดภาวะของมือจับ (เปิด/ปิด) หุ่นยนต์จะทำตามคำสั่งควบคุมมือจับดังกล่าวก่อนจะเคลื่อนที่

ตัวอย่างคำสั่ง

```
10 LPRINT "HE 1"
20 LPRINT "MT 1, + 30, C"
```

บรรทัดที่ 20 เป็นการกำหนดให้เคลื่อนที่จากตำแหน่งอ้างอิงปัจจุบันเป็นระยะ 30 mm ในทิศทางของเครื่องมือ (tool direction) โดยสภาวะมือจับปิด

3.1.7) PD (Position Define)

ความหมาย เป็นการกำหนดค่าโคออร์ดิเนท (ตำแหน่งและมุม) ให้กับตำแหน่งต่างๆ

รูปแบบ PD (Position number) , [(X-axis coordinate)] , [(Y-axis coordinate)] , [(Z-axis coordinate)] , [(Pitch angle)] , [(Roll angle)]

โดยที่ 1 ≤ Position number ≤ 629

คำอธิบาย

- สามารถกำหนดเลขนัยสำคัญได้สูงสุดเท่ากับ 0.1 mm หรือ 0.1

- คำสั่งนี้จะสามารถใช้ได้ ก็ต่อเมื่อค่าโคออร์ดิเนทของปลายของมือจับ หาค่าจากการกำหนดทิศทางของคำสั่งเครื่องมือให้อยู่ตามแนวแกน Z (ทิศทางของพื้นผิวสัมผัสของหุ่นยนต์)

ตัวอย่างคำสั่ง

```
100 LPRINT "GF 1"
110 LPRINT "PO 10, 0, 380, -70, -40"
120 LPRINT "PO 20, 0, 0, 20, 0, 0"
130 LPRINT "MO 10"
```

ในบรรทัดที่ 110 จะเป็นการกำหนดตำแหน่งที่ 10ซึ่งมีสภาวะมือจับปิด ในขณะที่บรรทัดที่ 120 จะเป็นการกำหนดตำแหน่งที่ 20 และบรรทัดที่ 130 จะเป็นการเคลื่อนหุ่นยนต์ไปยังตำแหน่งที่ 10 ตามที่ได้กำหนดไว้

3.1.8) SP (Speed)

ความหมาย เป็นการกำหนดค่าความเร็ว และ เวลาที่ใช้ของ ความเร่ง/ความหน่วง ในการทำงานของหุ่นยนต์

รูปแบบ SP (Speed level) , [(H or L)]

โดยที่ 0 ≤ Speed level ≤ 9

คำอธิบาย

- ความเร็วจะมีระดับทั้งหมด 10 ระดับ โดยที่ระดับ 9 จะเป็นระดับที่ค่าความเร็วมากที่สุด และ 0 เป็นระดับที่มีค่าความเร็วต่ำที่สุด ส่วนในการกำหนดค่าเวลาที่ใช้สำหรับการเร่งหรือหน่วงจะสามารถเลือกได้ระหว่าง H และ L เวลาที่ใช้ในการเร่งสำหรับ H คือ 0.35 วินาที และสำหรับ L คือ 0.5 วินาที ส่วนเวลาที่ใช้ในการหน่วงสำหรับ H คือ 0.4 วินาที และสำหรับ L คือ 0.6 วินาที เมื่อเลือกค่า H หรือ L ค่าของความเร่งและความหน่วง จะมีค่าคงที่จาก SP0 ไปถึง SP9 (ดูหัวข้อ "ความสัมพันธ์ระหว่างตัวแปรอัตราเร็วและความเร็ว)

- เมื่อมีข้อต่อเคลื่อนที่มากกว่า 2 ข้อต่อ คำสั่งนี้จะกำหนดให้ความเร็วในการทำงานของข้อต่อ มีค่าตามจำนวนพัลส์สูงสุดของมอเตอร์

- เมื่อมีการกำหนดความเร็วและเวลาที่ใช้สำหรับการเร่งหรือหน่วง ระยะทางที่ใช้ดังกล่าวจะต้องมีค่าที่สอดคล้อง กล่าวคือ ถ้าระยะทางในการเคลื่อนที่น้อย จะทำให้ความเร็วในการเคลื่อนที่มีค่าไม่ถึงความเร็วที่กำหนด

ตัวอย่างคำสั่ง

10 LPRINT "SP 3"	กำหนดความเร็วที่ระดับ 3
20 LPRINT "MO 10"	เคลื่อนไปยังตำแหน่งที่ 10
30 LPRINT "SP 6, 2"	กำหนดความเร็วที่ระดับ 6 และเวลาเป็น L
40 LPRINT "MO 12"	เคลื่อนไปยังตำแหน่ง 12
50 LPRINT "MO 15"	เคลื่อนไปยังตำแหน่ง 15

3.1.9) TI (Timer)

ความหมาย เป็นคำสั่งที่หยุดการเคลื่อนที่ของหุ่นยนต์ ด้วยระยะเวลาที่กำหนด

รูปแบบ TI (Timer counter)

โดยที่ O ≤ Timer counter ≤ 3276

คำอธิบาย

- คำสั่งนี้จะทำให้หุ่นยนต์หยุดการเคลื่อนที่ ในระยะเวลาที่กำหนด โดยกำหนดจากค่าตัวแปรเคาน์เตอร์ของเวลา(timer counter value) x 0.1 วินาที (มากที่สุด 3276.7 วินาที)

- คำสั่งนี้สามารถใช้เพื่อกำหนดค่าเวลาหน่วง (time delay) ก่อนและ/หรือขณะทำการจับชิ้นงาน

ตัวอย่างคำสั่ง

```
10 LPRINT "MO 1, O"    เคลื่อนที่ไปยังตำแหน่ง 1 ด้วยสภาวะมือจับเปิด
20 LPRINT "TI 5"       กำหนดเวลา 0.5 วินาที
30 LPRINT "GC"         ปิดมือจับ (จับชิ้นงาน)
40 LPRINT "T15"        กำหนดเวลา 0.5 วินาที
50 LPRINT "MO 2, C"    เคลื่อนที่ไปยังตำแหน่ง 2 ด้วยสภาวะมือจับปิด
```

3.1.10) TL (Tool)

ความหมาย เป็นคำสั่งที่ใช้ในการกำหนดระยะระหว่างหน้าสัมผัส และจุดปลายของมือจับ

รูปแบบ TL (Tool length)

โดยที่ $0 \leq$ tool length $\leq +300.0$ (mm)

คำอธิบาย

- เมื่อมีการเปลี่ยนแปลงค่าระยะชิ้นงาน (tool length) ตำแหน่งจุดปัจจุบันจะมีค่าเปลี่ยนแปลงตามไปด้วย อย่างไรก็ดีจะไม่เกี่ยวข้องกับการเคลื่อนที่ของหุ่นยนต์ (ค่าเริ่มต้นของระยะชิ้นงานคือ 107 mm)

- จุดที่กำหนดด้วยคำสั่งนี้จะเป็นข้อมูลที่จะใช้คำนวณหาตำแหน่งปัจจุบัน , การบังคับหุ่นยนต์ รวมถึงคำสั่งต่างๆ ที่เกี่ยวข้องกับระบบแกนพิกัดฉาก ดังนั้นจะต้องกำหนดระยะชิ้นงานให้ถูกต้องตามชิ้นงานที่ใช้นั้นๆ

- เมื่อจะสั่งให้โปรแกรมทำงาน จะต้องตรวจสอบค่าระยะชิ้นงานที่ตั้งไว้ และที่เป็นค่าจริงให้ตรงกันเสมอ

ตัวอย่างคำสั่ง

```
10 LPRINT "TL 120"
20 LPRINT "HE 1"
30 LPRINT "TL 100"
40 LPRINT "MO 1"
```

บรรทัดที่ 30 จะเป็นการเปลี่ยนค่าระยะชิ้นงาน และบรรทัดที่ 40 จะเป็นการเคลื่อนปลายมือจับไป 20 mm ตามทิศทางของชิ้นงาน (tool direction)

3.2) คำสั่งใช้ควบคุมมือจับชิ้นงาน

3.2.1) GC (Grip Close)

ความหมาย เป็นคำสั่งที่ใช้ในการสั่งให้มือจับปิด

รูปแบบ GC

คำอธิบาย

- สำหรับมือจับที่ทำงานด้วยมอเตอร์ : เป็นคำสั่งที่สั่งให้มือจับปิดด้วยรูปแบบของแรงบีบ (gripping force waveform) ที่กำหนดด้วยคำสั่ง GP (Grip pressure)

- สำหรับมือจับที่ทำงานด้วยระบบนิวแมติก : เป็นคำสั่งที่จะทำให้วาวล์โซลินอยด์ปล่อยพลังงานเพื่อปิดมือจับ (หรือสัมผัสชิ้นงาน)

ตัวอย่างคำสั่ง

```
10  LPRINT "100 MO 10, 0"    เคลื่อนไปตำแหน่งที่10 (ด้วยมือจับเปิด)
20  LPRINT "110 TI 5"        กำหนดระยะเวลา 0.5 วินาที
30  LPRINT "120 GC"          สั่งให้มือจับปิด (จับชิ้นงาน)
40  LPRINT "130 TI 5"        กำหนดระยะเวลา 0.5 วินาที
50  LPRINT "140 MO 15, C"    เคลื่อนไปตำแหน่งที่ 15 (ด้วยมือจับปิด)
```

3.2.2) GO (Grip Open)

ความหมาย เป็นคำสั่งที่ใช้ในการสั่งให้มือจับเปิด

รูปแบบ GO

คำอธิบาย เหมือนกับคำอธิบายของคำสั่ง GC

ตัวอย่างคำสั่ง

```
10  LPRINT "100 MO 10, C"
20  LPRINT "110 TI 5"
30  LPRINT "120 GO"          สั่งให้มือจับเปิด(ปล่อยชิ้นงาน)
40  LPRINT "130 TI 5"
50  LPRINT "140 MO 15, 0"
```

4) ภาคผนวก (Appendices)

4.1) ความสัมพันธ์ระหว่างตัวแปรอัตราเร็วและความเร็ว

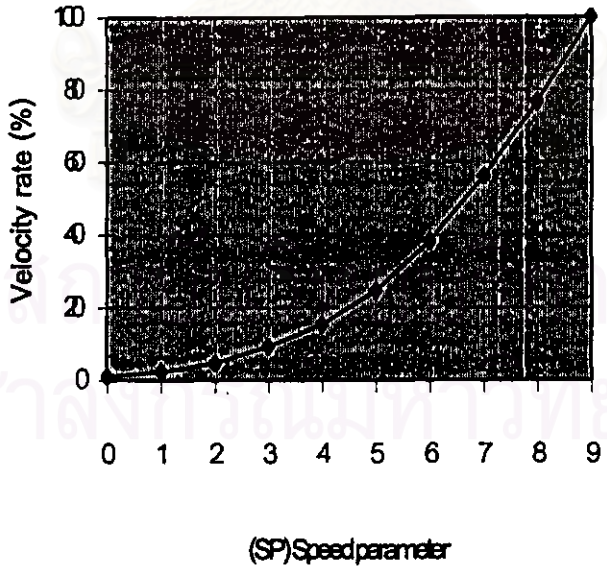
(Relation between Speed Parameters and Velocitics)

จากรูปที่ ก-16 และรูปที่ ก-17 แสดงถึงความสัมพันธ์ระหว่าง ตัวแปรอัตราเร็วของคำสั่ง SP (Speed) และอัตราความเร็ว (มีค่าเป็น % โดยเทียบกับความเร็วในการหมุนสูงสุดของแต่ละแกนของข้อต่อของหุ่นยนต์)

เมื่อมีการดำเนินการกับแกนมากกว่า 2 แกนขึ้นไป ตัวแปรอัตราเร็วที่กำหนดจะมีค่าระหว่างการหมุนของแกน ซึ่งมีค่าของจำนวนพัลส์ของการเคลื่อนที่มากที่สุด ความเร็วที่กำหนดจะมีค่าไม่ถึงค่าที่กำหนด ถ้าระยะทางที่ใช้ในการเคลื่อนที่น้อยเกินกว่าระยะทางสำหรับความเร่ง/ความหน่วง

SP	0	1	2	3	4	5	6	7	8	9
%	1.2	2.4	4.7	9.4	15.3	24.7	38.3	56.0	76.5	100.0

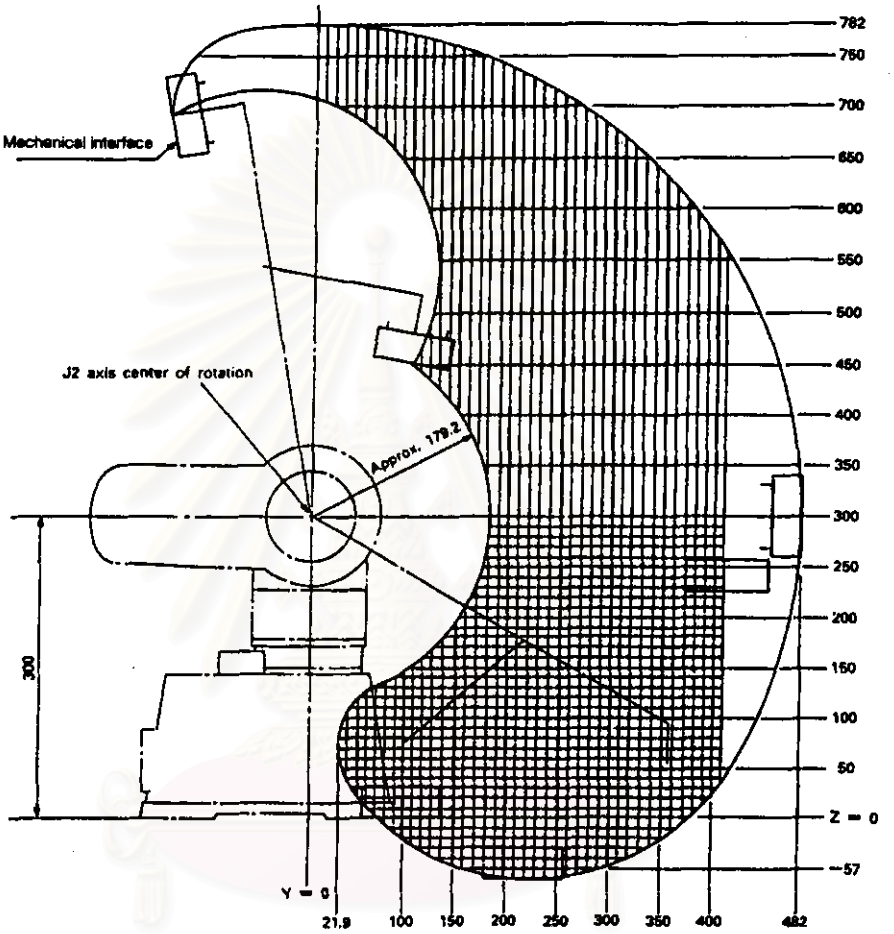
รูปที่ ก-16 รูปแสดงความสัมพันธ์ระหว่าง Speed Parameters และ Velocities



รูปที่ ก-17 รูปแสดงกราฟความสัมพันธ์ระหว่าง Speed Parameters และ Velocities

4.2) แผนผังขอบเขตการทำงาน (Operational Space Diagram)

จากรูปที่ ก-18 จะแสดงถึงแผนงานขอบเขตการทำงาน ซึ่งจะสามารถนำไปใช้เพื่อตรวจสอบ การวางอุปกรณ์พ่วงต่อ และการทำ pallet



รูปที่ ก-18 รูปแสดงขอบเขตการทำงาน (Operational Space Diagram)

ภาคผนวก ข.

โปรแกรมโพสต์โปรเซสเซอร์ที่เขียนขึ้นเพื่อใช้ในวิทยานิพนธ์

```
/*****************************************************************************/
/******* THESIS : DEVELOPEMENT OF POST-PROCESSORS FOR *******/
/***************** AUTOMATIC MANUFACTURING SYSTEM **************/
/*****************************************************************************/
/****** POST-PROCESSOR FOR MITSUBISHI model RV-M1 ROBOT *******/
/*****************************************************************************/
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <dos.h>
float phi[100],teta[100],psi[100],test[200];
float zeta[100],pitch[100],roll[100],aaa,bbb,ccc;
//float x[100],y[100],z[100],xx[100],yy[100];
int   i,j,k,m,n,o,j0,jj0,jx,jjx,jy,jjy;
int   jjz,jjq,com[100];
int   ii,init,start,*vx,*vy,c,check;
char  ch,*st1,*st2,*st3,*s,*var1,*var2,label[20];
char  name1[20],name2[20],name3[20],name4[20],name5[20],name6[20];
char  *out1="pong",out2[]="pong",out3[]="adv";
//char  VAR1,VAR2;
main(void)
{   FILE *fw1,*fw2,*fw3,*fr1,*fr2,*fr3,*fopen();
    float z[200],y[200],x[200],xx[200],yy[200];
    char cl1[20],cl2[20],motion[20],initial[20],status[20];
// float phi[100],teta[100],psi[100],test[200];
    int l;
    char *joint="joint";
    char *cart="cart";
    char *advance="advance";
    char ext1[ ]=".pos",ext2[ ]=".mrl";
    char gra[ ]="GC",rel[ ]="GO";
   char VAR1,VAR2;
```

```
clrscr();
cprintf(" ================================================\r\n");
cprintf(" POST-PROCESSOR FOR MITSUBISHI model RV-M1 ROBOT\r\n");
cprintf(" ================================================\r\n");
/*************************************************************/
/***************** Define for input files ********************/
/*************************************************************/
 cprintf("=======================================\r\n");
 cprintf("The output file from CATIA has 2 types\r\n");
 cprintf("=======================================\r\n");
 cprintf("      1) Basic level file : <<no logical statement>>\r\n");
 cprintf("      2) Advance level file : <<logical statement>>\r\n");
 cprintf("What type of file which you select ==>1,2:  ");
 scanf("%d",&check);if(check==1){
 cprintf(" Enter name of input file\r\n  ");
 cprintf("  (Cartesian Position) : cart");
 scanf("%19s",&name1);strcat(cart,name1);
 fr1=fopen(cart,"r");
 if(fr1==NULL)cprintf("Cannot open %s for reading.\r\n",cart);
 cprintf(" Enter name of input file\r\n  ");
 cprintf("    (Joint Position)   : joint");
 scanf("%19s",&name2);strcat(joint,name2);
 fr2=fopen(joint,"r");
 if(fr2==NULL)cprintf("Cannot open %s for reading.\r\n",joint);
 /**************** Define for basic output files ******************/
 cprintf(" ============================================================\r\n");
 cprintf(" PROGRAM ASSIGNMENT FOR MITSUBISHI Model RV-M1 ROBOT\r\n");
 cprintf(" HAS 2 PARTS: 1) Position lists (file with extension:.pos)\r\n");
 cprintf("              2) Program list (file with extension:.mrl)\r\n");
 cprintf(" ============================================================\r\n");
  cprintf("  \r\n") ;
  cprintf(" ----------------------\r\n");
  cprintf(" Enter name of output file(Position lists<*.POS>): pong");
  scanf("%19s",&name3);
  strcat(out1,name3);
  strcat(out1,ext1);
  printf("out1----->%s\n",out1);getch();
  cprintf("  \r\n") ;
  cprintf(" ----------------------\r\n");
```

```
    cprintf(" Enter name of output file(Program lists)<*.MRL>: pong");
    scanf("%19s",&name4);
    strcat(out2,name4);
    strcat(out2,ext2);
    printf("out2------>%s\n",out2);
    cprintf(" Please open file to 'CHECK'==><<%s>> and <<%s>>\r\n",out1,out2);
    getch();   }
    else{
   cprintf(" Enter name of input file\r\n  ");
   cprintf("    (Advance Position)   : advance");
   scanf("%19s",&name5);strcat(advance,name5);
   fr3=fopen(advance,"r");
   if(fr3==NULL)cprintf("Cannot open %s for reading.\r\n",advance);
   /************** Define for advance output files *****************/
    cprintf(" Enter name of output file(Program lists<*.MRL>): adv");
    scanf("%19s",&name6);
    strcat(out3,name6);
    strcat(out3,ext2);
    printf("out3------>%s\n",out3);
    getch();   }
if(check==1){
/*****************************************************************************/
/********* Open 1st file : "Basic Cartesian Positions File"***********/
/*****************************************************************************/
i=0;      c=0;
      do{
            fscanf(fr1,"%s",st1);
            /******** (1) Check for"Catesian Absolute Motion" ******/
            /********       and "Joint Absolute Motion"      ******/
             if((strcmp(st1,"CARA")==0)||(strcmp(st1,"JNTA")==0))
                  { i++;
                 fseek(fr1,18L,1); fscanf(fr1,"%f",&x[i]);     /* x-coordinates */
                                   fscanf(fr1,"%f",&y[i]);     /* y-coordinates */
                                   fscanf(fr1,"%f",&z[i]);     /* z-coordinates */
                                   fscanf(fr1,"%f",&phi[i]);   /* euler angle about z-axis */
                                   fscanf(fr1,"%f",&teta[i]);  /* euler angle about x-axis */
                                   fscanf(fr1,"%f",&psi[i]);   /* euler angle about z-axis */
                                   fgets(s,0,fr1);
                     j=ftell(fr1);
```

```
        com[i]=0;
 printf("PD%d  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f\n",i,x[i],y[i],z[i],
 phi[i],teta[i],psi[i]);    }
/******** (2) Check for"Catesian Relative Motion" ******/
/********        and "Joint Relative Motion"       ******/
         if((strcmp(st1,"CARR")==0)||(strcmp(st1,"JNTR")==0))
        {   i++;
fseek(fr1,18L,1); fscanf(fr1,"%f",&x[i]);     /* x-coordinates */
                  fscanf(fr1,"%f",&y[i]);     /* y-coordinates */
                  fscanf(fr1,"%f",&z[i]);     /* z-coordinates */
                  fscanf(fr1,"%f",&phi[i]);   /* euler angle about z-axis */
                  fscanf(fr1,"%f",&teta[i]);  /* euler angle about x-axis */
                  fscanf(fr1,"%f",&psi[i]);   /* euler angle about z-axis */
                  fgets(s,0,fr1);
        j=ftell(fr1);
        com[i]=0;
 printf("PD%d  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f\n",i,x[i],y[i],z[i],
 phi[i],teta[i],psi[i]);
        }
/******** (3) Check for"Catesian Curve Motion" ******/
else  if(strcmp(st1,"CARC")==0)
   { i++; k=0;
        fgets(s,80,fr1);
        fseek(fr1,18L,1);
         fscanf(fr1,"%f",&x[i]);                /* x-coordinates */
                  if(k==0) { j0=ftell(fr1);}
                  fscanf(fr1,"%f",&y[i]);       /* y-coordinates */
                  fscanf(fr1,"%f",&z[i]);       /* z-coordinates */
                  fscanf(fr1,"%f",&phi[i]);     /* euler angle about z-axis */
                  fscanf(fr1,"%f",&teta[i]);    /* euler angle about x-axis */
                  fscanf(fr1,"%f",&psi[i]);     /* euler angle about z-axis */
                  fgets(s,80,fr1);
                  com[i]=1;
                  printf("PD%d  %7.3f  %7.3f  (%7.3f)  %7.3f  %7.3f
%7.3f\n",i,x[i],y[i],z[i],phi[i],teta[i],psi[i]);
        do {     i++;  k++;
fseek(fr1,1L,1); fscanf(fr1,"%f",&x[i]);      /* x-coordinates */
                  jx=ftell(fr1);
                  fscanf(fr1,"%f",&y[i]);     /* y-coordinates */
```

```
                    fscanf(fr1,"%f",&z[i]);      /* z-coordinates */
                    fscanf(fr1,"%f",&phi[i]);    /* euler angle about z-axis */
                    fscanf(fr1,"%f",&teta[i]);   /* euler angle about x-axis */
                    fscanf(fr1,"%f",&psi[i]);    /* euler angle about z-axis */
                    fgets(s,80,fr1);
            j=ftell(fr1);
                if( (jx-j0)/k >= 82)
                {
                com[i]=1;
                printf("PD%d  %7.3f  %7.3f  (%7.3f)  %7.3f  %7.3f  %7.3f\n",i,x[i],y[i],z[i],
                phi[i],teta[i],psi[i]);
                }
                else {
                  fseek(fr1,-81L,1);jy=ftell(fr1);
                  fscanf(fr1,"%s",var1);fseek(fr1,-4L,1);
                  fscanf(fr1,"%s",VAR1);
                  printf("1)Loop variable==>>>>>%s\n",VAR1);
                   if(strcmp(var1,"HOME")==0)
                       {
                       x[i]= -0.001;    /* x-coordinates */
                       y[i]= 589.000;   /* y-coordinates */
                       z[i]= 300.000;   /* z-coordinates */
                       phi[i]=  0.000;  /* euler angle about z-axis */
                       teta[i]= 0.000;  /* euler angle about x-axis */
                       psi[i]=  0.000;  /* euler angle about z-axis */
                       com[i]=0;
                       printf("===========>home position\n");
                       printf("PD%d  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f  %7.3f\n",i,x[i],y[i],z[i],
                       phi[i],teta[i],psi[i]);
                       printf("============================\n");
                       i++;
                       }
                  }
              } while( (jx-j0)/k >= 82);i--;getch();
            }
/******* (4) Check for command "Home Motion" *******/
        else  if(strcmp(st1,"HOME")==0)
            {
            i++;
```

```
                    x[i]= -0.001;    /* x-coordinates */
                    y[i]= 589.000;   /* y-coordinates */
                    z[i]= 300.000;   /* z-coordinates */
                    phi[i]=  0.000;  /* euler angle about z-axis */
                    teta[i]= 0.000;  /* euler angle about x-axis */
                    psi[i]=  0.000;  /* euler angle about z-axis */
                com[i]=0;
                printf("---------->home position\n");
                printf("PD%d  %7.3f  %7.3f  (%7.3f)  %7.3f  %7.3f  %7.3f\n",i,x[i],y[i],z[i],
                phi[i],teta[i],psi[i]);
                printf("-------------------------\n");
                getch();
                }
          /***** (5) Check for command "Grasp" and "Release" *****/
                 else if(strcmp(st1,"GRA")==0)
                   {
                    i++;
                    com[i]=2;
                   }
                 else if(strcmp(st1,"REL")==0)
                   {
                    i++;
                    com[i]=3;
                   }
           else
                {
                  fgets(s,80,fr1);
                 }
        } while(!feof(fr1));
  }
     if(check==1){
/*************************************************************************/
/************** Open 2nd file : "Basic Joint Positions File ************/
/*************************************************************************/
             i=0;
          do{
             fscanf(fr2,"%s",st2);
           /******** (1) Check for"Catesian Absolute Motion" ******/
           /********       and "Joint Absolute Motion"       ******/
```

```
if((strcmp(st2,"CARA")==0)||(strcmp(st2,"JNTA")==0))
        { i++;
fseek(fr2,38L,1); fscanf(fr2,"%f",&xx[i]);      /* pitch angle */
                  fscanf(fr2,"%f",&yy[i]);      /* roll angle */
                  fgets(s,0,fr2);
         printf("PD%d  %7.3f  %7.3f\n",i,xx[i],yy[i]);
        }
  /******** (2) Check for"Catesian Relative Motion" ******/
  /********       and "Joint Relative Motion"       ******/
   if((strcmp(st2,"CARR")==0)||(strcmp(st2,"JNTR")==0))
        { i++;
fseek(fr2,38L,1); fscanf(fr2,"%f",&xx[i]);      /* pitch angle */
                  fscanf(fr2,"%f",&yy[i]);      /* roll angle */
                  fgets(s,0,fr2);
         printf("PD%d  %7.3f  %7.3f\n",i,xx[i],yy[i]);
        }
  /******** (3) Check for"Catesian Curve Motion" ******/
  else if(strcmp(st2,"CARC")==0)
   { i++;  k=0;
    fseek(fr2,38L,1);fscanf(fr2,"%f",&xx[i]);  /* 4th axis angle */
                     fscanf(fr2,"%f",&yy[i]);  /* 5th axis angle */
                     fgets(s,0,fr2);
        printf("PD%d  %7.3f   %7.3f\n",i,xx[i],yy[i]);
        i++;
    fseek(fr2,1L,1);fscanf(fr2,"%f",&test[i]);
                  if(k==0)
                  {jj0=ftell(fr2);}
    fseek(fr2,20L,1);fscanf(fr2,"%f",&xx[i]);   /* 4th axis angle */
                     fscanf(fr2,"%f",&yy[i]);   /* 5th axis angle */
                     fgets(s,0,fr2);
                  printf("PD%d  %7.3f   %7.3f\n",i,xx[i],yy[i]);
       do { i++;  k++;
       fseek(fr2,1L,1); fscanf(fr2,"%f",&test[i]);
                     jjx=ftell(fr2);
          if((jjx-jj0)/k >=82)
                {
    fseek(fr2,20L,1);fscanf(fr2,"%f",&xx[i]);   /* 4th axis angle */
                     fscanf(fr2,"%f",&yy[i]);   /* 5th axis angle */
```

```
                    fgets(s,0,fr2);
                }
    if((jjx-jj0)/k >=82)
{ printf("PD%d  %7.3f   %7.3f\n",i,xx[i],yy[i]);}
    else
        {
        fseek(fr2,-8L,1);jjy=ftell(fr2);
        fscanf(fr2,"%s",var2);
        jjz=ftell(fr2);
        fseek(fr2,-8L,1);
        fscanf(fr2,"%s",VAR2);
        printf("2)Loop variable==>>>>>%s\n",VAR2);
            if(strcmp(var2,"HOME")==0)
            {
            xx[i]= 0.000;                   /* 4th axis angle */
            yy[i]= 0.000;                   /* 5th axis angle */
            printf("==========>home position\n");
            printf("PD%d   %7.3f    %7.3f\n",i,xx[i],yy[i]);
            printf("============================\n");
            i++;
            }
        }
} while ((jjx-jj0)/k >= 82);i--;getch();
    }
/******* (4) Check for command "Home Motion" *******/
else  if(strcmp(st2,"HOME")==0)
   {
    i++;
            xx[i]= 0.000;   /* 4th axis angle */
            yy[i]= 0.000;   /* 5th axis angle */
   printf("----------->home position\n");
   printf("PD%d   %7.3f    %7.3f\n",i,xx[i],yy[i]);
   printf("--------------------\n");
   }
   /***** (5) Check for command "Grasp" and "Release" *****/
    else if(strcmp(st2,"GRA")==0)
     {
      i++;
     }
```

```
                else if(strcmp(st2,"REL")==0)
                  {
                   i++;
                  }
                else
                {
                fgets(s,80,fr2);/* printf("No data on line no %d\n",i);       */
                }
          }while(!feof(fr2));
    }
/*********************************************************************/
/*************** Open file : "Advance level File"*****************/
/*********************************************************************/
   if(check!=1){         /** check basic or advance ? **/
     fw3=fopen(out3,"w");
     if (fw3==NULL)
         {
         printf("Cannot open %s for creating output file.\n",out3);
         }
         i=0;
         fprintf(fw3,"\n");
         fprintf(fw3,"110 CP 1\n");
         fprintf(fw3,"120 EQ 0,150\n");
         fprintf(fw3,"130 OT 1         /* nest drive ? */\n");
         fprintf(fw3,"140 GT 280\n");
         do{
            fscanf(fr3,"%s",st3);
           /********** (1) Check for status output *********/
            if(strcmp(st3,"IF")==0)
                 { i++;
                 fscanf(fr3,"%s",&initial);
                  if(strcmp(initial,"L#INIT.EQ.0")==0){ /* initial position */
                   fprintf(fw3,"150 CP 2         /* initial position ? */\n");
                   fprintf(fw3,"160 EQ 0,230\n");
                   fprintf(fw3,"170 CP 3         /* part gripped ? */\n");
                   fprintf(fw3,"180 EQ 0,210\n");
                     do{
                          fscanf(fr3,"%s",&status);
                     if(strcmp(status,"L#STATUS.EQ.1")==0){ /* part gripped */
```

```
                fprintf(fw3,"190 OT 3        /* initial position , part gripped */\n");
                fprintf(fw3,"200 GT 280\n"); }

             if(strcmp(status,"L#STATUS.EQ.0")==0){ /* no part gripped */
                fprintf(fw3,"210 OT 2        /* initial position , no part gripped */\n");
                fprintf(fw3,"220 GT 280\n"); }
            } while ((strcmp(status,"ENDC")!=0));
          }
          if(strcmp(initial,"L#INIT.EQ.1")==0){     /* no initial position */
            fprintf(fw3,"230 CP 3        /* part gripped ? */\n");
            fprintf(fw3,"240 EQ 0,270\n");
             do{
             fscanf(fr3,"%s",&status);
              if(strcmp(status,"L#STATUS.EQ.1")==0){ /* part gripped */
                fprintf(fw3,"250 OT 5        /* no initial position , part gripped */\n");
                fprintf(fw3,"260 GT 280\n"); }
              if(strcmp(status,"L#STATUS.EQ.0")==0){ /* no part gripped */
                fprintf(fw3,"270 OT 4        /* no initial position , no part gripped */\n");
                fprintf(fw3,"280 RT\n"); }
            }while ((strcmp(status,"ENDI")!=0));
          }
/********** (2) Check for" Logic Condition" **********/
/******************* (L#2 AND L#3)  *****************/
        if(strcmp(initial,"L#3.EQ.1")==0){      /* check sensor L#3 */
        fprintf(fw3,"340 EQ  4 , 620\n");
            do{
                fscanf(fr3,"%s",&status);
             if(strcmp(status,"N#1.EQ.5")==0){   /* scrap-part */
                fprintf(fw3,"370 EQ  7 , 820\n");  }
             if(strcmp(status,"N#1.EQ.10")==0){  /* red-part */
                fprintf(fw3,"380 EQ  8 , 870\n");  }
             if(strcmp(status,"N#1.EQ.15")==0){  /* black-part */
                fprintf(fw3,"390 EQ  9 , 920\n");  }
             if(strcmp(status,"N#1.EQ.20")==0){  /* metal-part */
                 fprintf(fw3,"400 EQ 10 , 970\n");  }
             if(strcmp(status,"N#1.EQ.25")==0){  /* sorting-station */
                 fprintf(fw3,"360 EQ  6 , 770\n");  }
            } while ((strcmp(status,"ELSE")!=0));
          }
```

```
        if(strcmp(initial,"L#2.EQ.1")==0){      /* check sensor L#2 */
           fprintf(fw3,"330 EQ  3 , 550\n");
           fprintf(fw3,"350 EQ  5 , 720\n");
           do{
           fscanf(fr3,"%s",&status);
          }while ((strcmp(status,"END!")!=0));
         fprintf(fw3,"410 EQ 11 , 1050\n");
         fprintf(fw3,"420 NE 16 , 440\n");
         fprintf(fw3,"430 SC 4,0\n");
         fprintf(fw3,"440 GT 1050\n");
         fprintf(fw3,"\n");
         fprintf(fw3,"450 NT\n");
         fprintf(fw3,"460 SC 1,0\n");
         fprintf(fw3,"470 GT 480\n");
         }
      }
/******** (3) Check for" Motion in sub program " ******/
/******  " LABEL 5 : SCRAP PART & Processing (up) *****/
 if(strcmp(st3,"LAB")==0)
 { i++;  o=0;
     fscanf(fr3,"%s",&label);  /* check label */
        if(strcmp(label,"LABEL1")==0){    /* initial position */
         fprintf(fw3,"\n");
         fprintf(fw3,"480 CP 3\n");        /* part gripped */
         fprintf(fw3,"490 EQ 0,520\n");
         fprintf(fw3,"500 MO 4,C\n");
         fprintf(fw3,"510 GT 530\n");
         }
         if(strcmp(label,"LABEL2")==0){    /* initial position */
         fprintf(fw3,"520 MO 4,O\n");      /* no part gripped */
         fprintf(fw3,"530 SC 2,1\n");
         fprintf(fw3,"540 GT 1050\n");
         }
         if(strcmp(label,"LABEL5")==0){       /* check label-5 */
          printf("<<label-5 = %s>>\n",label);
          do{
              fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
               if(strcmp(motion,"CARA")==0){ o++;
                 if(o>=7){                  /* put down at magazine1 */
```

```
                    if(o==7){
                        fprintf(fw3,"\n");
                        fprintf(fw3," * put down at magazine 1 *\n");
                        fprintf(fw3,"820 MO 10,C\n");}
                    if((o-7)==1) fprintf(fw3, "830 MO  5,C\n");
                    if((o-7)>1){
                        fprintf(fw3, "850 MO 10,O\n");
                        fprintf(fw3, "860 GT 1020\n");}
                  printf("motion is -->'CART ABSOLUTE(-x-)',%d\r\n",o);}
                  if(o<7){                  /* pick up from processing */
                    if(o==1){
                        fprintf(fw3,"\n");
                        fprintf(fw3," * pick up from processing station *\n");
                        fprintf(fw3,"620 MO  3,O\n");}
                    if((o-1)==1) fprintf(fw3, "630 MO  2,O\n");
                    if((o-2)==1) fprintf(fw3, "640 MO  1,O\n");
                    if((o-4)==1) fprintf(fw3, "650 MO  2,C\n");
                    if((o-5)==1) {
                        fprintf(fw3, "670 MO  3,C\n");
                        fprintf(fw3, "680 GT 690\n");}
                   printf("motion is -->'CART ABSOLUTE(-y-)',%d\r\n",o);}}
                 if(strcmp(motion,"GRA")==0){ o++;
                   if(o>=7){                /* grasp at magazine1 */
                    printf("motion is -->'GRASP(-x-)',%d\r\n",o);}
                   if(o<7){                 /* grasp from processing */
                    fprintf(fw3,"670 GC \n");
                    printf("motion is -->'GRASP(-y-)',%d\r\n",o);}}
                 if(strcmp(motion,"REL")==0){ o++;
                   if(o>=7){                /* release at magazine1 */
                    fprintf(fw3,"840 GO \n");
                    printf("motion is -->'RELEASE(-x-)',%d\r\n",o);}
                   if(o<7){                 /* release from processing */
                    printf("motion is -->'RELEASE(-y-)',%d\r\n",o);}}
              }while ((strcmp(motion,"BRAN")!=0));getch();
          }
/******** (4) Check for" Motion in sub program " ******/
/******  " LABEL 10 : RED PART & Processing (up) *****/
          if(strcmp(label,"LABEL10")==0){     /* check label-10 */
          printf("<<label-10>> = %s \n",label);
```

```
            fprintf(fw3,"\n");
            fprintf(fw3," * put down at magazine 2 *\n");
                do{
                fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
                 if(strcmp(motion,"CARA")==0){  o++;  /* put down at magazine2 */
                   if(o>=5){
                    if(o==5) fprintf(fw3,"870 MO 11,C\n");
                    if((o-5)==1) fprintf(fw3, "880 MO  6,C\n");
                    if((o-6)>1)  fprintf( fw3, "900 MO 11,O\n");
                   printf("motion is -->'CART ABSOLUTE',%d\r\n",o);}}
                 if(strcmp(motion,"GRA")==0) {  o++;  /* grasp at magazine2 */
                   if(o>=5){
                   printf("motion is -->'GRASP',%d\r\n",o);}}
                 if(strcmp(motion,"REL")==0) {   o++;  /* release at magazine2 */
                   if(o>=5){
                   fprintf(fw3,"890 GO \n");
                   printf("motion is -->'RELEASE',%d\r\n",o);}}
              }while ((strcmp(motion,"BRAN")!=0));getch();fprintf(fw3,"910 GT 1020\n");
            }
/******** (5) Check for" Motion in sub program " ******/
/******  " LABEL 15 : BLACK PART & Processing (up) *****/
             if(strcmp(label,"LABEL15")==0){      /* check label-15 */
            printf("<<label-15>> = %s \n",label);
            fprintf(fw3,"\n");
            fprintf(fw3," * put down at magazine 3 *\n");
                do{
                fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
                 if(strcmp(motion,"CARA")==0){  o++;   /* put down at magazine3 */
                   if(o>=5){
                    if(o==5) fprintf(fw3,"920 MO 12,C\n");
                    if((o-5)==1) fprintf(fw3, "930 MO  7,C\n");
                    if((o-6)>1)  fprintf( fw3, "950 MO 12,O\n");
                   printf("motion is -->'CART ABSOLUTE',%d\r\n",o);}}
                 if(strcmp(motion,"GRA")==0){  o++;    /* grasp at magazine3 */
                   if(o>=5){
                   printf("motion is -->'GRASP',%d\r\n",o);}}
                 if(strcmp(motion,"REL")==0){  o++;     /* release at magazine3 */
                   if(o>=5){
                   fprintf(fw3,"940 GO \n");
```

```
                printf("motion is -->'RELEASE',%d\r\n",o);}}
            }while ((strcmp(motion,"BRAN")!=0));getch();fprintf(fw3,"960 GT 1020\n");
        }
/******** (6) Check for" Motion in sub program " ******/
/******  " LABEL 20 : METAL PART & Processing (up) ****/
        if(strcmp(label,"LABEL20")==0){     /* check label-20 */
        printf("<<label-20>> = %s \n",label);
        fprintf(fw3,"\n");
        fprintf(fw3," * put down at magazine 4 *\n");
            do{
            fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
             if(strcmp(motion,"CARA")==0){  o++;   /* put down at magazine4 */
               if(o>=5){
                 if(o==5) fprintf(fw3,"970 MO 13,C\n");
                 if((o-5)==1) fprintf(fw3, "980 MO  8,C\n");
                 if((o-6)>1)  fprintf( fw3, "1000 MO 13,O\n");
                printf("motion is -->'CART ABSOLUTE',%d\r\n",o);}}
             if(strcmp(motion,"GRA")==0){  o++;   /* grasp at magazine4 */
               if(o>=5){
               printf("motion is -->'GRASP',%d\r\n",o);}}
             if(strcmp(motion,"REL")==0){  o++;    /* release at magazine4 */
               if(o>=5){
               fprintf(fw3,"990 GO \n");
               printf("motion is -->'RELEASE',%d\r\n",o);}}
            }while ((strcmp(motion,"BRAN")!=0));getch();fprintf(fw3,"1010 GT 1020\n");
        }
/******** (7) Check for" Motion in sub program " ******/
/******  " LABEL 25 : SORTING & Processing (up) *******/
         if(strcmp(label,"LABEL25")==0){     /* check label-25 */
        printf("<<label-25>> = %s \n",label);
        fprintf(fw3,"\n");
        fprintf(fw3," * put down at sorting station *\n");
            do{
            fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
             if(strcmp(motion,"CARA")==0){  o++;   /* put down at sorting */
               if(o>=5){
                 if(o==5) fprintf(fw3,"770 MO 19,C\n");
                 if((o-5)==1) fprintf(fw3, "780 MO 20,C\n");
                 if((o-6)>1)  fprintf( fw3, "800 MO 19,O\n");
```

```
                printf("motion is -->'CART ABSOLUTE',%d\r\n",o);}}
              if(strcmp(motion,"GRA")==0){   o++;    /* grasp at sorting */
                if(o>=5){
                printf("motion is -->'GRASP',%d\r\n",o);}}
              if(strcmp(motion,"REL")==0){   o++;     /* release at sorting */
                if(o>=5){
                fprintf(fw3,"790 GO \n");
                printf("motion is -->'RELEASE',%d\r\n",o);}}
            }while ((strcmp(motion,"BRAN")!=0));getch();fprintf(fw3,"810 GT 1020\n");
        }
/******** (8) Check for" Motion in sub program " ******/
/**** " LABEL 35 : "Processing(down) & Buffer(up)" ****/
         if(strcmp(label,"LABEL35")==0){      /* check label-35 */
        printf("<<label-35>> = %s \n",label);
        do{
            fgets(s,80,fr3);fscanf(fr3,"%s",&motion);
             if(strcmp(motion,"CARA")==0){ o++;
               if(o>=6){                  /* put down at processing */
                 if(o==6) {
                     fprintf(fw3,"\n");
                     fprintf(fw3," * put down at processing station *\n");
                     fprintf(fw3,"720 MO 17,C\n");}
                 if((o-6)==1) fprintf(fw3, "730 MO 18,C\n");
                 if((o-7)>1) {
                     fprintf(fw3, "750 MO 17,O\n");
                     fprintf(fw3, "760 GT 1020\n");}
              printf("motion is -->'CART ABSOLUTE(-x-)',%d\r\n",o);}
              if(o<6){                   /* pick up from buffer */
               if(o==1){
                     fprintf(fw3,"\n");
                     fprintf(fw3," * pick up from buffer station *\n");
                     fprintf(fw3,"550 MO 15,O\n");}
               if((o-1 )==1) fprintf(fw3, "570 MO 14,O\n");
               if((o-3)==1) fprintf(fw3, "590 MO 15,C\n");
               if((o-4)==1) {
                     fprintf(fw3, "600 MO 16,C\n");
                     fprintf(fw3, "610 GT 690\n");}
             printf("motion is -->'CART ABSOLUTE(-y-)',%d\r\n",o);}}
            if(strcmp(motion,"GRA")==0){ o++;
```

```
                    if(o>=5){                /* grasp at processing */
                    printf("motion is -->'GRASP(-x-)',%d\r\n",o);}
                    if(o<5){                 /* grasp from buffer */
                    fprintf(fw3,"580 GC \n");
                    printf("motion is -->'GRASP(-y-)',%d\r\n",o);}}
                  if(strcmp(motion,"REL")==0){ o++;
                    if(o>=5){                /* release at processing */
                    fprintf(fw3,"740 GO \n");
                    printf("motion is -->'RELEASE(-x-)',%d\r\n",o);}
                    if(o<5){                 /* release from buffer */
                    printf("motion is -->'RELEASE(-y-)',%d\r\n",o);}}
                }while ((strcmp(motion,"BRAN")!=0));getch();
              }
            }
        /******** (9) check for "initial position ***********/
        if(strcmp(st3,"WAIT")==0)
          {
          fprintf(fw3,"\n");
          fprintf(fw3,"290 IN\n");
          fprintf(fw3,"300 EQ  0 , 1050\n");
          fprintf(fw3,"310 EQ  1 , 450\n");
          fprintf(fw3,"320 EQ  2 , 480\n");
          }
            else
            {
            fgets(s,80,fr3);/* printf("No data on line no %d\n",i);      */
            }
      }while(!feof(fr3));
    }
    fprintf(fw3,"\n");
    fprintf(fw3,"690 SC 2,0          * not in initial position *\n");
    fprintf(fw3,"700 SC 3,1          * part gripped *\n");
    fprintf(fw3,"710 GT 1050\n");
    fprintf(fw3,"\n");
    fprintf(fw3,"1020 SC 2,0         * not in initial position *\n");
    fprintf(fw3,"1030 SC 3,0         * no part gripped *\n");
    fprintf(fw3,"1040 GT 1050\n");
    fprintf(fw3,"\n");
    fprintf(fw3,"1050 RT\n");
```

```
/*************************************************************************/
/********************** Output Command ***********************************/
/*************************************************************************/
  if(check==1){
    fw1=fopen(out1,"w");
    if (fw1==NULL)
     {
     printf("Cannot open %s for creating output file.\n",out1);
     }
     /******** Print "POSITION LISTS" **********/
    l=0;
    for(j=1;j<=i;j++)
    { l++;
          x[j]=(int)(x[j]*10)/10.0;
          y[j]=(int)(y[j]*10)/10.0;
          z[j]=(int)(z[j]*10)/10.0;
          aaa=phi[j];bbb=teta[j];ccc=psi[j];
          transform(j,aaa,bbb,ccc);          /* CALL function trasform() */
          xx[j]=(int)(xx[j]*10)/10.0;
          yy[j]=(int)(yy[j]*10)/10.0;
If((com[j]==0)||(com[j]==1))
     {
          if(l<10)
           {
      printf("PD  %d, %7.3f  %7.3f  %7.3f  %7.3f  %7.3f\n",
      l,x[j],y[j],z[j],pitch[j],yy[j]);
      fprintf(fw1,"PD  %d,%6.1f,%6.1f,%6.1f,%6.1f,%6.1f\n",
      l,x[j],y[j],z[j],pitch[j],yy[j]);
           }
      else
           {
      printf("PD %d, %7.3f   %7.3f   %7.3f  %7.3f  %7.3f\n",
      l,x[j],y[j],z[j],pitch[j],yy[j]);
      fprintf(fw1,"PD %d,%6.1f,%6.1f,%6.1f,%6.1f,%6.1f\n",
      l,x[j],y[j],z[j],pitch[j],yy[j]);
           }
    }
    else if (com[j]==2)
    {printf("<*** number===>%d,---GC\n",j);l--;}
```

```
    else
    { printf("<*** number===>%d,---GO\n",j);l--;}
}
  fclose(fw1);printf("close fw1\r\n");getch()       /** Close 1st written file **/
  fw2=fopen(out2,"w");
   if (fw2==NULL)
   {
   printf("Cannot open %s for creating output file.\n",out2);
   }
    /******** Print "PATH MOTION" **********/
   com[0]=3;
   l=start=0;
   for(j=1;j<=i;j++)
    { l++; init++; start++;
    if(com[j]==0)
        {
         if(j<10)       /* for number of motion(j) < 10 */
           {
            if (com[j-1]==2)
                {
                printf("  %d MO  %d, C\n",start*10,l);
                fprintf(fw2,"  %d MO  %d, C\n",start*10,l);
                }
             else    /* com[j-1]==0 & 3 */
                {
                printf("  %d MO  %d, O\n",start*10,l);
                fprintf(fw2,"  %d MO  %d, O\n",start*10,l);
                }
           }
        else           /* for number of motion(j) > 10 */
         {
          if(com[j-1]==2)
                {
                printf(" %d MO  %d, C\n",start*10,l);
                fprintf(fw2," %d MO %d, C\n",start*10,l);
                }
            else   /* com[j-1]==0 & 3 */
                {
                printf(" %d MO  %d, O\n",start*10,l);
```

```
                    fprintf(fw2," %d MO  %d, O\n",start*10,l);
                    }
                }
            }
        else if (com[j]==1)
            { init=j;
             do{ j++; l++;
             }while (com[j]==1);j--;l--;
             printf(" %d MC %3d,%d\n",start*10,init,j);
             fprintf(fw2, " %d MC %3d,%d\n",start*10,init,j);
            }
else if (com[j]==2)
            {
            printf("  %d GC\n",start*10);fprintf(fw2,"  %d GC\n",start*10);
            l--;
            }
        else   /*com[j]==3*/
            {
            printf("  %d GO\n",start*10);fprintf(fw2,"  %d GO\n",start*10);
            l--;
            }
      }
  if(j>10)
   { start++;
   printf(" %d ED\n",start*10);printf("RN\n");
   fprintf(fw2," %d ED\n",start*10);fprintf(fw2,"RN\n");
   }
   else
   { start++;
   printf("  %d ED\n",start*10);printf("RN\n");
   fprintf(fw2,"  %d ED\n",start*10);fprintf(fw2,"RN\n");
   }
 fclose(fw2);printf("close fw2\r\n");getch();
 fclose(fr1);printf("close fr1\r\n");getch();
 fclose(fr2);printf("close fr2\r\n");getch();
}
 /********************* Advance output file *****************/
 else{
 fclose(fw3);printf("close fw3\r\n");getch();
```

```
    fclose(fr3);printf("close fr3\r\n");getch();
     }
  return 0;
}
/************ Function transform Euler angle Z-X-Z to Z-X-Y *************/
transform(j,aa,bb,cc)
 int j;
 float aa,bb,cc;
{
 float alfa[100],beta[100],gamma[100];
 float zeta_[100],pitch_[100],roll_[100];
 float mat[3][3];
        alfa[j]  =(aa*M_PI)/180;     /* alfa(Z) in radian */
        beta[j]  =(bb*M_PI)/180;     /* beta(X) in radian */
        gamma[j] =(cc*M_PI)/180;     /* gamma(Z) in radian */

        mat[1][1]=cos(alfa[j])*cos(gamma[j])-sin(alfa[j])*cos(beta[j])*sin(gamma[j]);
        mat[1][2]=-cos(alfa[j])*sin(gamma[j])-sin(alfa[j])*cos(beta[j])*cos(gamma[j]);
        mat[1][3]=sin(alfa[j])*sin(beta[j]);

        mat[2][1]=sin(alfa[j])*cos(gamma[j])+cos(alfa[j])*cos(beta[j])*sin(gamma[j]);
        mat[2][2]=-sin(alfa[j])*sin(gamma[j])+cos(alfa[j])*cos(beta[j])*cos(gamma[j]);
        mat[2][3]=-cos(alfa[j])*sin(beta[j]);

        mat[3][1]=sin(beta[j])*sin(gamma[j]);
        mat[3][2]=sin(beta[j])*cos(gamma[j]);
        mat[3][3]=cos(beta[j]);

pitch_[j]= atan2(mat[3][2],sqrt(mat[1][2]*mat[1][2]+mat[2][2]*mat[2][2]));
zeta_[j] = atan2((-mat[1][2]/cos(pitch_[j])),(mat[2][2]/cos(pitch_[j])));
roll_[j] = atan2((-mat[3][1]/cos(pitch_[j])),(mat[3][3]/cos(pitch_[j])));

        zeta[j]  =(zeta_[j]*180)/M_PI;      /** Zeta in degree **/
        pitch[j] =(pitch_[j]*180)/M_PI;     /** Pitch in degree **/
        roll[j]  =(roll_[j]*180)/M_PI;      /** roll in degree **/

        mat[3][2]=(int)(mat[3][2]*1000)/1000.0;
        mat[2][2]=(int)(mat[2][2]*1000)/1000.0;
        printf("mat[3][2]= %f ,mat[1][2]= %f ,mat[2][2]= %f\n",mat[3][2],mat[1][2],mat[2][2]);
```

```
        printf("pitch[%d]= %f\n",j,pitch[j]);
        if(mat[3][2]>=0 && mat[2][2]>=0)      /* Quardant 1*/
        {   pitch[j] =(int)(pitch[j]*10)/10.0; }
        else if(mat[3][2]>=0 && mat[2][2]<0)  /* Quardant 2*/
            { pitch[j] = 180-((int)(pitch[j]*10)/10.0); }
        else if(mat[3][2]<0 && mat[2][2]<0)   /* Quardant 3*/
            { pitch[j] = -180-((int)(pitch[j]*10)/10.0); }
        else if(mat[3][2]<0 && mat[2][2]>=0)  /* Quardant 4*/
            { pitch[j] = (int)(pitch[j]*10)/10.0; }
}
```

ประวัติผู้เขียน

นาย ภาสิต อนุกุลอนันต์ชัย เกิดเมื่อวันที่ 10 ธันวาคม พ.ศ. 2514 ที่อำเภอเมือง จังหวัดขอนแก่น สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตร์บัณฑิต ภาควิชาวิศวกรรมเครื่องกล จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2535 และได้เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีพ.ศ. 2536