บทที่2
หลักการและทฤษฎี

2.1 นิยามของหุ่นยนต์ (Definition of robot)

ในการสร้างนิยามของหุ่นยนต์ ข้อตกลงสำหรับผู้เชี่ยวชาญ ได้ใช้วิธีการหลากหลายลักษณะเพื่อกำหนดนิยาม และเป็นสิ่งที่จะต้องมีความเข้าใจเบื้องต้นของหุ่นยนต์อุตสาหกรรม สามารถจำแนกลักษณะจำเพาะของหุ่นยนต์ได้ ในการเปรียบเทียบคำนิยามระหว่างประเทศ ซึ่งสามารถจำแนกได้ดังนี้

2.1.1 นิยามของหุ่นยนต์จากสถาบันหุ่นยนต์ของสหรัฐอเมริกา

(Definition of robot from The Robot Institute of America)

แขนกลทำงานหลากหลายหน้าที่โดยสามารถโปรแกรมได้ถูกออกแบบเพื่อเคลื่อนวัสดุ ชิ้นส่วนหรืออุปกรณ์พิเศษผ่านการเคลื่อนที่ที่ถูกโปรแกรมสำหรับกระทำงานในหลากหลายลักษณะ (Schlussel , 1985)

2.1.2 นิยามของหุ่นยนต์จากองค์การหุ่นยนต์อุตสาหกรรมญี่ปุ่น

(Definition of robot from Japan Industrial Robot Association)

หุ่นยนต์ถูกแยกประเภทดังต่อไปนี้

1. Manual Maripulators : ถูกควบโดยตรงจากผู้ใช้โดยปราศจากโปรแกรมใด ๆ
2. Fixed Sequence Robot: Handling Device, การซึ่งการเคลื่อนที่นั้นถูกจำกัด การเปลี่ยนแปลงจะทำได้ยาก
3. Variable Sequence Robot : Handling Device, การเปลี่ยนแปลงง่ายและรวดเร็ว
4. Playback Robot: ผู้เขียนโปรแกรมสอนหุ่นยนต์โดยตรงสู่ลำดับการเคลื่อนที่ แต่ละการเคลื่อนที่คือการบันทึกลงในหน่วยความจำ (งานพ่นสี)
5. Numerical Control Robot : Handling Device คล้ายกับเครื่อง NC ลำดับการเคลื่อนที่ถูกโปรแกรมโดยปุ่มสวิตซ์หรือสื่อกลางอื่น ๆ

6. Interlligent Robot: หุ่นยนต์ชนิดนี้เป็นจัดอยู่ในชั้นที่สูงที่สุดถูกใช้กับอุปกรณ์ซึ่งสามารถควบคุมและโปรแกรมโดยการใช้เซนเซอร์เพื่อที่จะปรับปรุงการเคลื่อนที่ในสภาพแวดล้อมหรือของชิ้นงาน

2.1.3 นิยามของหุ่นยนต์อุตสาหกรรมตามมาตราฐาน VDI 2861 Blatt1

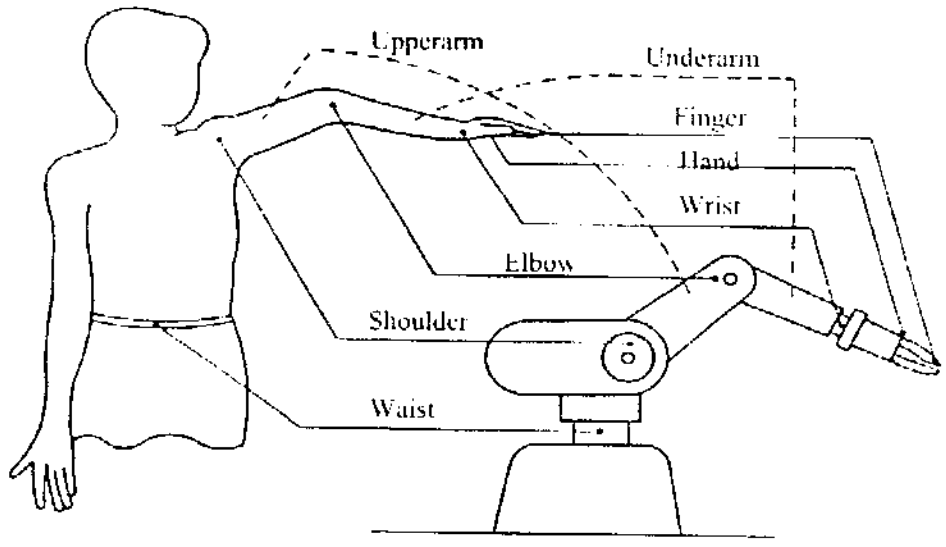
(Definition of Industrial robot by Standard VDI 2861 Blatt1)

หุ่นยนต์อุตสาหกรรมคือ

1. เครื่องจักรอัตโนมัติที่มีแกนหลัก 3 แกน และใช้งานทั่วไป
2. การเคลื่อนที่ที่เกี่ยวข้องกับลำดับ, ขั้นตอนและข้อต่อสามารถโปรแกรมได้อย่างอิสระโดยปราศจากการปรับเปลี่ยนทางกล
3. ถ้าจำเป็นอาจใช้เซนเซอร์
4. หุ่นยนต์ถูกติดตั้งด้วย gripper, tools หรืออุปกรณ์ทาง manufacturing อื่น ๆ และ
5. สามารถจัดการกับงานหยิบยกหรือ manufacturing

2.2 ระบบการทำงานของหุ่นยนต์ (Robot Operation System)

2.2.1 การเปรียบเทียบสรีระของมนุษย์กับหุ่นยนต์



รูปที่ 2.1 แสดงการเปรียบเทียบสรีระของมนุษย์กับหุ่นยนต์

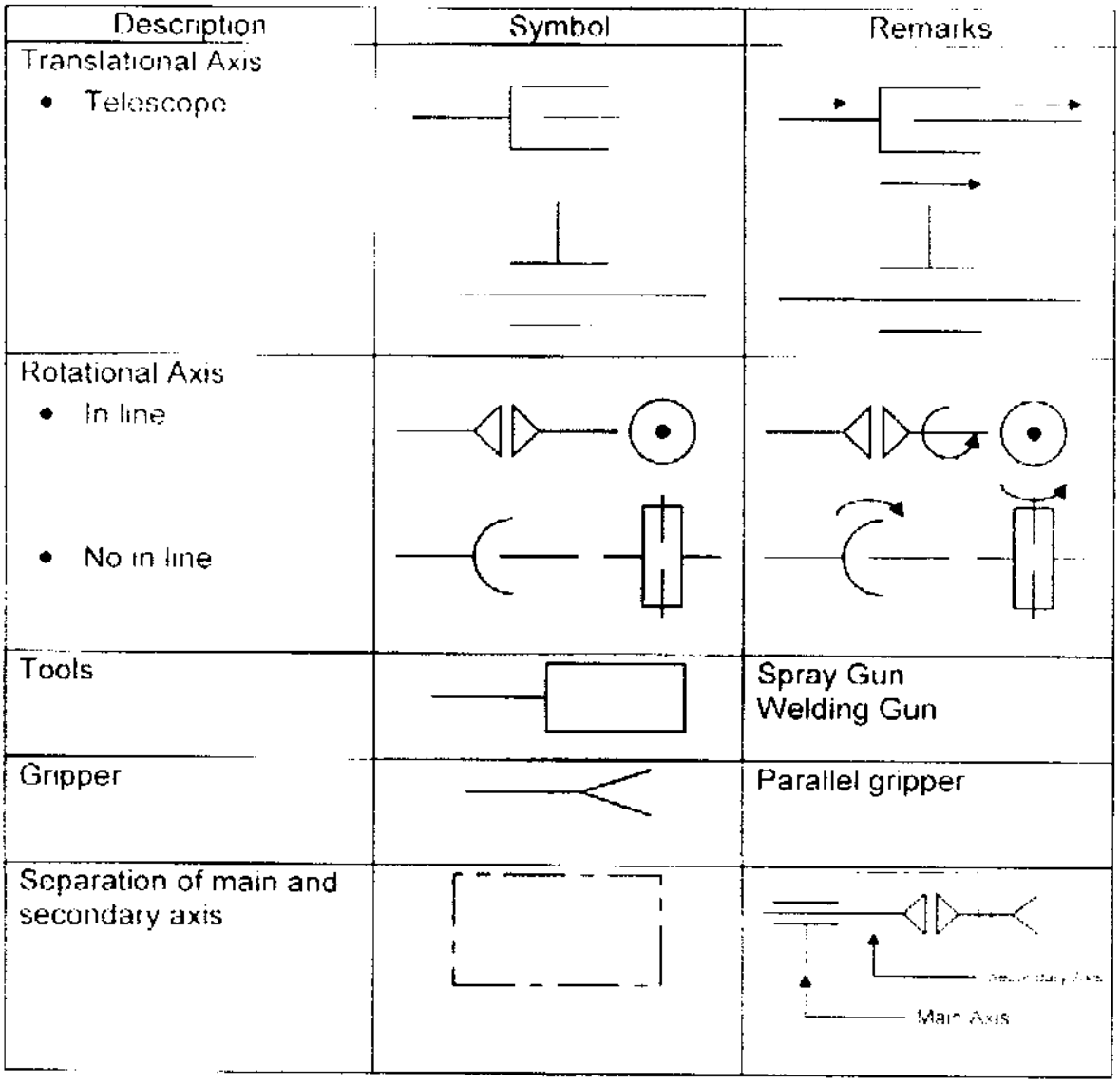
หุ่นยนต์ที่มีลักษณะโครงสร้างมนุษย์นี้อาจกล่าวได้ว่าเป็นหุ่นยนต์ชนิด Anthropomorphic, Revolute, Jointer arm หรือ Articulated industrial robot

2.2.2 องค์ประกอบของหุ่นยนต์(Robot Workcell)

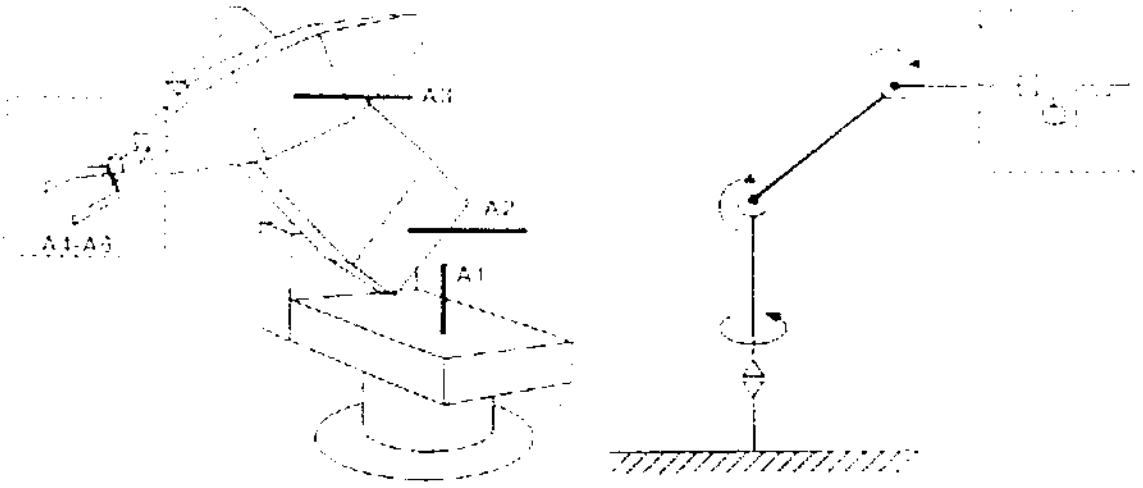
องค์ประกอบของ Robot Workcell ประกอบด้วย

1. Robot ระบบอัตโนมัติที่มีลักษณะการรับรู้และปรับตัวต่อกระบวนการเคลื่อนที่และจัดการงาน
2. Controller ระบบคอมพิวเตอร์ที่จัดการกิจกรรมการเคลื่อนที่ , การโปรแกรม, การเชื่อมต่อกับผู้ใช้และอื่น ๆ
3. Periphery อุปกรณ์ที่อยู่รอบนอก เช่น Working Table ที่จับยึดชิ้นงาน
4. Teach Pendant อุปกรณ์ที่สามารถถือได้ด้วยมือ ใช้สำหรับการเขียนโปรแกรมให้แก่หุ่นยนต์
5. Software โปรแกรมที่ทำการควบคุมการทำงานของคอมพิวเตอร์หรือ Controller ของหุ่นยนต์
6. Sensor ส่วนประกอบทาง Hardware ซึ่งเป็นอุปกรณ์ทางการวัดเพื่อตรวจจับปรากฏการณ์ทางกายภาพ
7. Safety Equipment อุปกรณ์ป้องกันเหตุและเพิ่มความปลอดภัยแก่พนักงาน ในรัศมีการทำงานของหุ่นยนต์
8. Workpiece ชิ้นงานภายใต้กระบวนการผลิต เช่น ตัวถังรถยนต์ที่รอการเชื่อม

2.2.3 สัญลักษณ์โครงสร้างของหุ่นยนต์



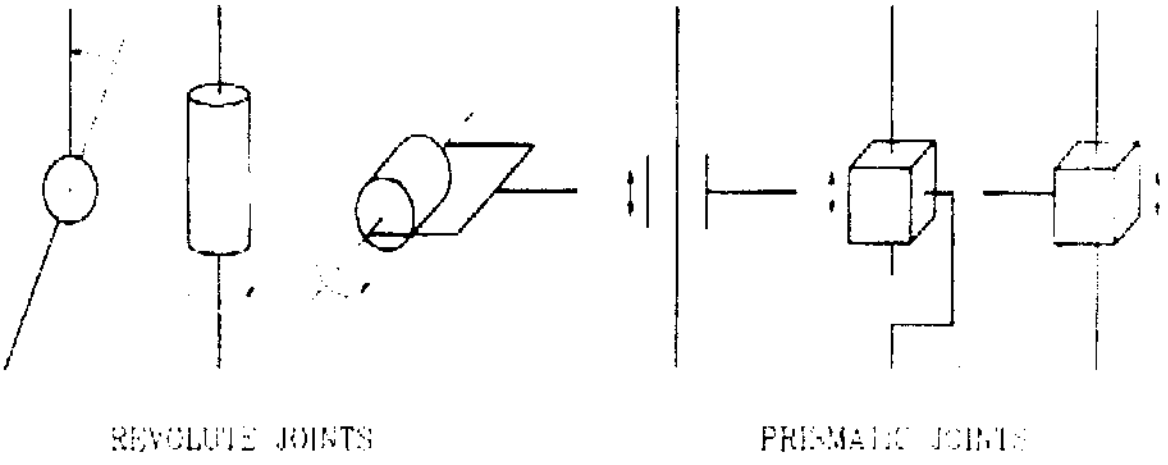
รูปที่ 2.2 สัญลักษณ์โครงสร้างของหุ่นยนต์ตามมาตรฐาน VDI 2861



รูปที่ 2.3 ตัวอย่างการเขียนสัญลักษณ์ kinematic โดยสอดคล้องกับโครงสร้างของหุ่นยนต์จริง

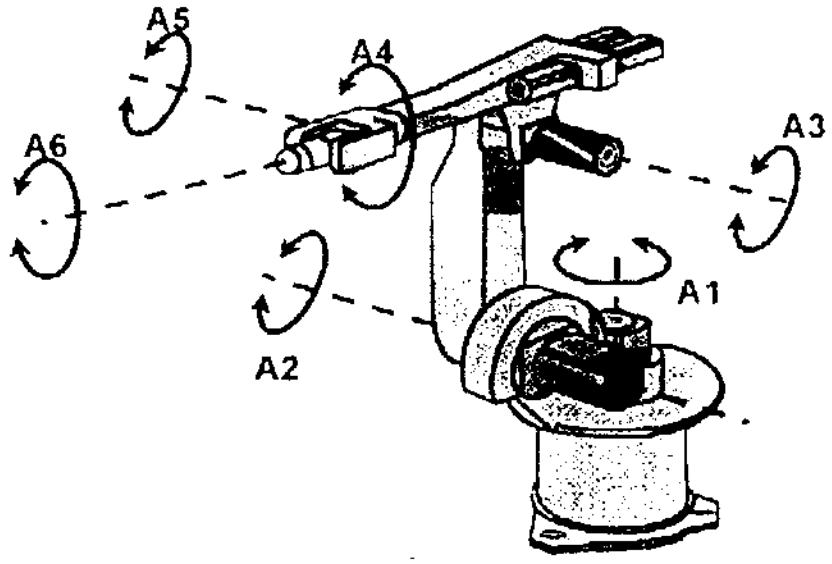
2.2.4 คำนิยามพื้นฐาน

1. ข้อต่อ (Joint) คือจุดเชื่อมต่อที่สามารถเคลื่อนที่ของ 2 สมาชิกใน kinemnatic chain การเคลื่อนที่นั้นสามารถแบ่งออกได้เป็น 2 ชนิดคือ แบบหมุน (Revolute Joints) ซึ่งมีจุดหมุนที่แกน (Axis) ของข้อต่อ และแบบเลื่อนในแนวเส้นตรง (Prismatic Joints)



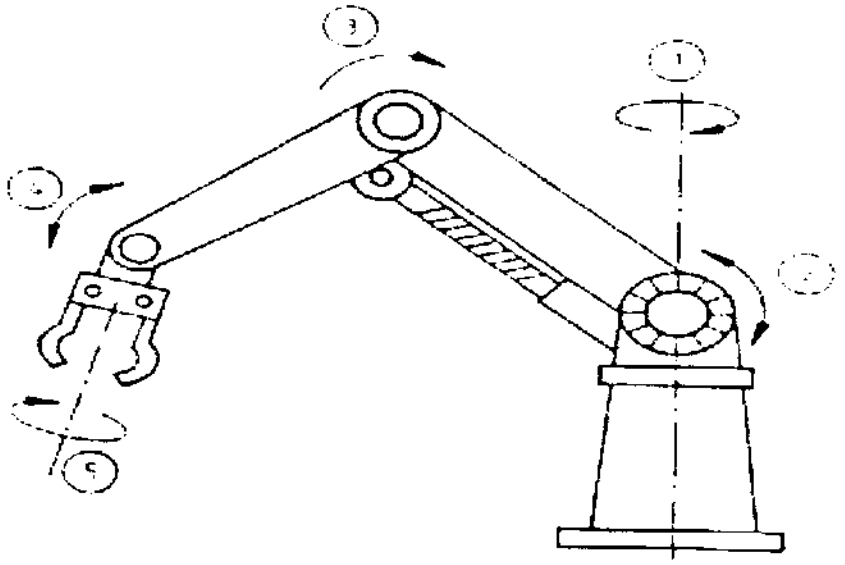
รูปที่ 2 .4 แสดงข้อต่อแบบหมุนและแบบเลื่อนในแนวเส้นตรง

2. แกน (Axis) คือเส้นตรงอ้างอิงที่ข้อต่อ (Joint) ของหุ่นยนต์



รูปที่ 2 .5 แสดงแกนทั้ง 6 แกนของหุ่นยนต์

3. Degree of freedom คือจำนวนแกนการเคลื่อนที่อย่างอิสระของหุ่นยนต์ใน Work Envelope โดยที่การเปิด/ปิดของ gripper จะไม่พิจารณาว่าเป็น degree of freedom



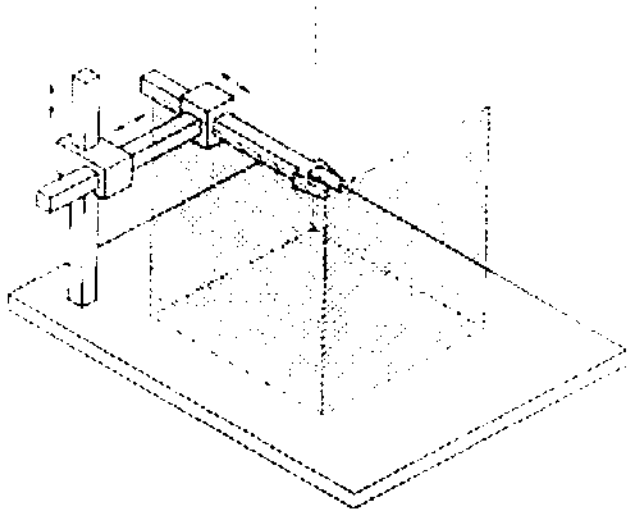
รูปที่ 2.6 แสดงหุ่นยนต์ที่มี Degree of Freedom เท่ากับ *5*

4. End Effector คือส่วนประกอบที่ติดตั้งที่ข้อมือของหุ่นยนต์ที่ใช้ในกระบวนการทำงาน ตัวอย่าง เช่น Gripper และ Tool เป็นต้น

รูปที่ 2.7 แสดง Gripper ชนิดการเปิด/ปิดแบบ Swing

รูปที่ 2.8 แสดง Tool ที่ใช้ในงาน Cutting

5. Work Envelope คือส่วนของสภาพแวดล้อมที่ End Effector สามารถเข้าถึงได้โดยที่ขนาด และ รูปร่างขึ้นอยู่กับโครงสร้างของหุ่นยนต์



รูปที่ 2.9 แสดง Work Envelope ของหุ่นยนต์

2.2.5 โครงสร้างของหุ่นยนต์

Maninpulator คือชุดการเชื่อมต่อทางกล, แขน และข้อต่อที่สามารถเคลื่อนที่ได้หลายทิศทางเพื่อควบคุม end-effector ลักษณะทางเรขาคณิตพื้นฐานอาจแบ่งได้เป็น 4 ชนิดคือ

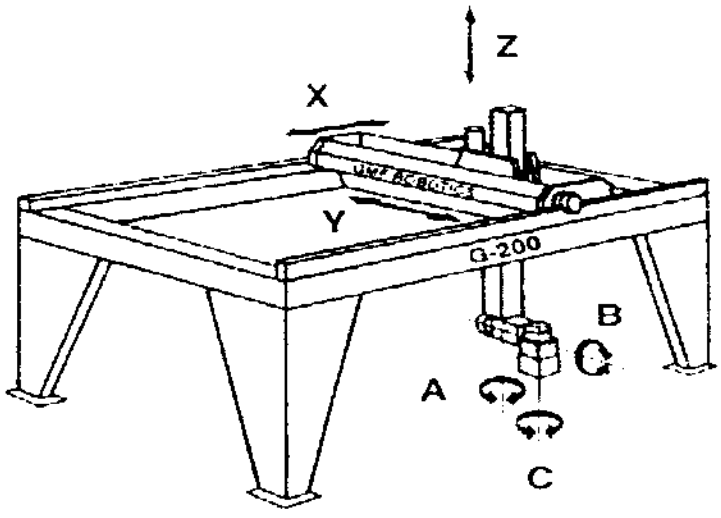
- LLL : Cartesian หรือ Gantry
- LLR : Cylindrical
- LRR : Polar หรือ Spherical
- RRR : Rovolute หรือ Articulated Arm และ SCARA

โดยที่ L ย่อมาจาก Linear Joint คือข้อต่อที่มีลักษณะการเคลื่อนที่แบบเชิงเส้นและ

R ย่อมาจาก Rotary Joint คือข้อต่อที่มีลักษณะการเคลื่อนที่แบบหมุน

2.2.5.1 Gantry

หุ่นยนต์ Gantry เหมือนกับ Overhead Crane โดยมีข้อมือติดตั้งอยู่ดังแสดงในรูปที่2.11หุ่นยนต์นี้มี Work Envelope ขนาดใหญ่ เพราะว่า หุ่นยนต์ถูกติดตั้งเหนือไม่ต้องการพื้นที่วาง Gantry Robot หลายชนิดสามารถถูกติดตั้งเพื่อได้รูปโครงสร้างเดียวกัน



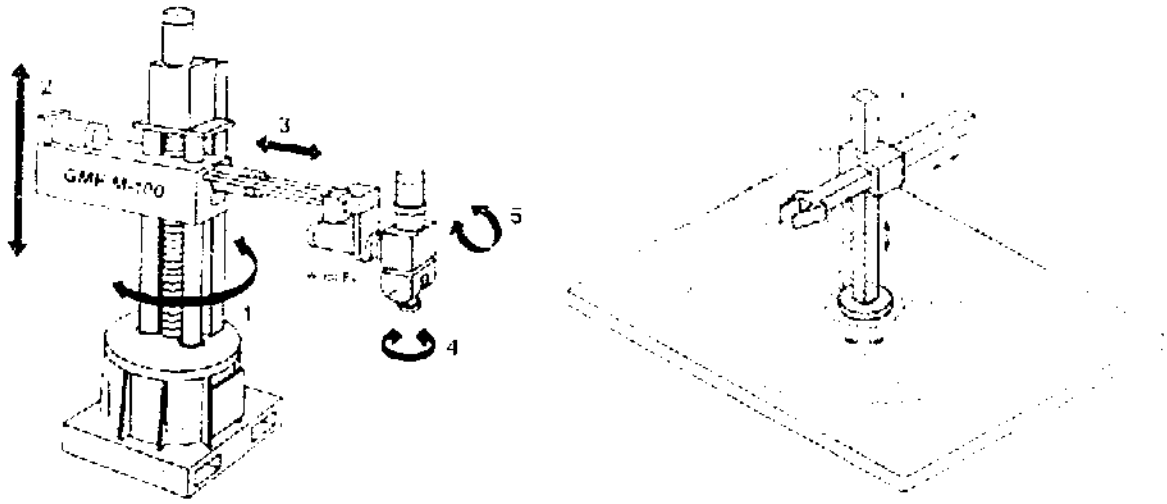
รูปที่ 2.10 แสดงหุ่นยนต์ชนิด Gantry

Gantry ถูกใช้เมื่อยกวัตถุที่มีขนาดใหญ่มากหรือเมื่อเคลื่อนวัตถุเป็นระยะทางยาว การใช้งานทั่วไปคือการเคลื่อนวัตถุระหว่างเครื่องจักรเป็นระยะทางไกล ๆ หรือรับวัตถุโรงเก็บอัตโนมัติ

หุ่นยนต์มีแกนที่มีการเคลื่อนเชิงเส้นแนวระนาบ 2 แกน X และ Y การเคลื่อนที่ในแนวดังกระทำโดยแกน Z อีก 3 แกนที่เพิ่มขึ้น A , B และ C ถูกใช้ในข้อมือเพื่อกำหนดทิศทางของ End Effector

Gantry Robot จะมีความเร่ง และความเร็วที่ช้ากว่าหุ่นยนต์ส่วนใหญ่เพราะมีมวลมาก เมื่อแกน X เคลื่อนที่มวลของหุ่นยนต์ทั้งหมดต้องเคลื่อน

2.2.5.2 Cylindrical



รูปที่ 2 .11 แสดงหุ่นยนต์ชนิด Cylindrical และ Work Envelope

หุ่นยนต์ชนิดนี้จะแพงสำหรับขนาดและ Payload หุ่นยนต์ชนิดนี้จะมีหลากหลายขนาด Payload ซึ่งมีช่วงจาก 15 ถึง 250 ปอนด์, ขึ้นอยู่กับขนาดของหุ่นยนต์และจำนวนของแกนของหัวมือ Work Envelope มีขนาดและ Vertical Stroke โดยทั่วไปจะยาวเท่า ๆ กับ Radial Stroke

หุ่นยนต์แบบ Cylindrical Coordinate ขนาดเล็กจะถูกใช้สำหรับงานประกอบขนาดเล็กที่ต้องการความถูกต้องสูง หุ่นยนต์ขนาดใหญ่จะถูกใช้สำหรับ Material Handling และ Machine Loading and Unloading

หุ่นยนต์ชนิด Cylindrical Coordinat โดยทั่วไปใช้ Ball Screw และ Linear Bearing บนแกนแนวตั้ง (2) และแกน Rodial (3) การหมุนจะถูกบรรจุผลโดยแกนแยก (1) ข้อมือสามารถถูกเพิ่มเพื่อ Provide หนึ่งหรือสอง Additional Axes

การใช้งานของแกนแนวตั้งให้การเคลื่อนที่ที่ราบเรียบและเร็วในทิศทางแกนตั้ง สิ่งที่เป็นต้องการในงานการประกอบและ Material Handling ดังที่อธิบายใน SCARA Robot Radial Axes (3) จะให้Robot หดหรือยืดอย่างรวดเร็ว ถ้าหุ่นยนต์Reach สู่ Low-Clearance Machine เช่น Staming Press ลักษณะของRadial Stroke เป็นที่ต้องการ หุ่นยนต์สามารถถูกจัดทิศทางเพื่อกด ดังนั้นเฉพาะRadil Axis(3) เคลื่อนที่เมื่อเข้าสู่การกด หุ่นยนต์ชนิดอื่นๆ แขนเข้าสู่อุปกรณ์ที่มุมเอียง ดังนั้นต้องการClearance มากกว่า Cylindrical Coordinate Robot

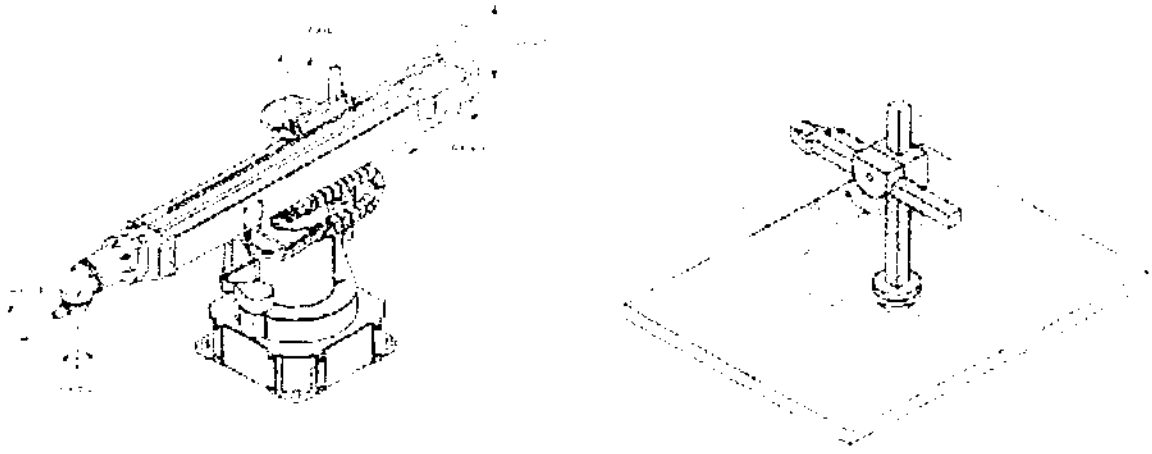
ข้อเสียของหุ่นยนต์แบบทรงกระบอกคือแกน Radial (3) ขึ้นหลังของ Rotbot เครื่องมือไม่สามารถที่จะวางใกล้หุ่นยนต์เพราะเมื่อหุ่นยนต์หมุน 180^0 ส่วนหลังของ Radial Axis Housing จะขันกับอะไรก็ตามที่อยู่ในทิศทางนั้น (SCARA Robot ไม่มีปัญหานี้เพราะไม่มี Housing ของ Radial Axis)

ความเร็วของหุ่นยนต์แบบCylindrical Coordinate คือค่าเฉลี่ย ความเฉื่อยของแกนหมุน (1) มีค่าสูงเนื่องจากโครงสร้างของแกน Radial สิ่งนี้จะลดความเร็วของหุ่นยนต์

Path Control , Repeatability และ Accuracy จะเปลี่ยนแปลงขึ้นกับหุ่นที่กำหนด Cylindrical Coordinate Robot บางชนิดถูกสร้างสำหรับงานประกอบที่ต้องการความเร็วสูงและความถูกต้อง หุ่นยนต์ชนิดนี้มีการควบคุมทิศทาง Repeatability และความแม่นยำที่ดี หุ่นยนต์ชนิดอื่นถูกออกแบบสำหรับงานจับวัตถุที่หนักสำหรับ Machine Loading และ Unloadng หุ่นยนต์นี้มีความสามารถทาง High Plylod แต่ Repeatability และความแม่นยำจะต่ำกว่า

หุ่นยนต์ชนิดนี้จะถูกใช้ในงานประกอบงานเบาที่ต้องการความเร็วสูงในอดีต แต่อย่างไรก็ตามปัจจุบัน SCARA Robot กำลังถูกใช้ทดแทนงานเหล่านี้ Cylindrical Coordinate Robot ยังคงจะเป็น Ideal สำหรับงานยกชิ้นส่วนหนัก ๆ และใช้ในงานที่ต้องการ Work Envelope ขนาดใหญ่

2.2.5.3 Spherical



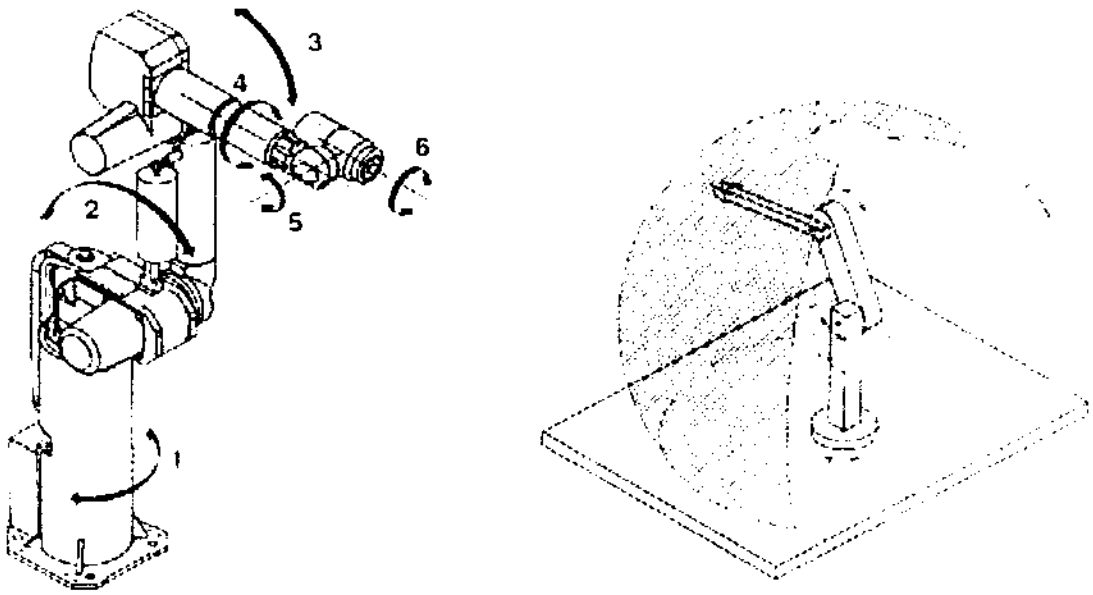
รูปที่ 2.12 แสดงหุ่นยนต์ชนิด Spherical และ Work Enverlope

เป็นหุ่นยนต์อุตสาหกรรมชนิดแรกที่ถูกใช้งานอย่างแพร่หลาย และถูกให้กำลังโดยกระบอกสูบไฮดรอลิกหุ่นยนต์ชนิดนี้มี Work Envelope แบบทรงกลม หุ่นยนต์จะมีจุดยึดที่จุดศูนย์กลางของการหมุนและการเคลื่อนที่แนวตั้งแกน Radial จะเคลื่อนที่เข้าและออกในแนวตรงที่จุดศูนย์กลางของหุ่นยนต์ข้อมือหนึ่งหรือสองแกนถูกใช้เพื่อจัดเรียง Gripper

Work Enverlope แนวตั้งขนาดใหญ่เป็นไปได้ด้วยส่วนทางกลขนาดเล็ก สิ่งนั้นเกิดขึ้นเนื่องจากจุดหมุนของแขนที่เคลื่อนที่ขึ้นและลงสำหรับ Vertical Stroke Vertical Strock สามารถยาวเป็นสองเท่าของแนวราบ

หุ่นยนต์ชนิดนี้ , ผลคือความเร็วต่ำกว่าและความแม่นยำลดลง การเคลื่อนที่เชิงเส้นไม่ราบเรียบเพราะการเคลื่อนที่เชิงเส้นส่วนใหญ่ต้องการ Coordination ของหลายแกน

2.2.5.4 Revolute หรือ Articulated Arm



รูปที่ 2.13 แสดงหุ่นยนต์ชนิด Revolute และ work Envelope

หุ่นยนต์ชนิดนี้ใช้ในงานที่ต้องการ Work Envelope แนวตั้งขนาดใหญ่ และความคล่องแคล่วของข้อมือที่สูง การใช้งานโดยทั่วไปคือ งานเชื่อม , งานทาสี , งาน Sealing , และงานยกวัสดุ

ฐานของหุ่นยนต์หมุนด้วยแกนที่ (1) หุ่นเคลื่อนที่ในระนาบตั้งฉากด้วยแกน(2) และ(3) หุ่นยนต์อาจมีแกนที่ ข้อมือ 2 หรือ 3 แกน ข้อมืออีก 3 แกนจะให้การเคลื่อนที่แกน (4) , (5) และ (6) ในโครงสร้างข้อมือ 2 แกน แกนที่ (5) และ(6) จะถูกใช้

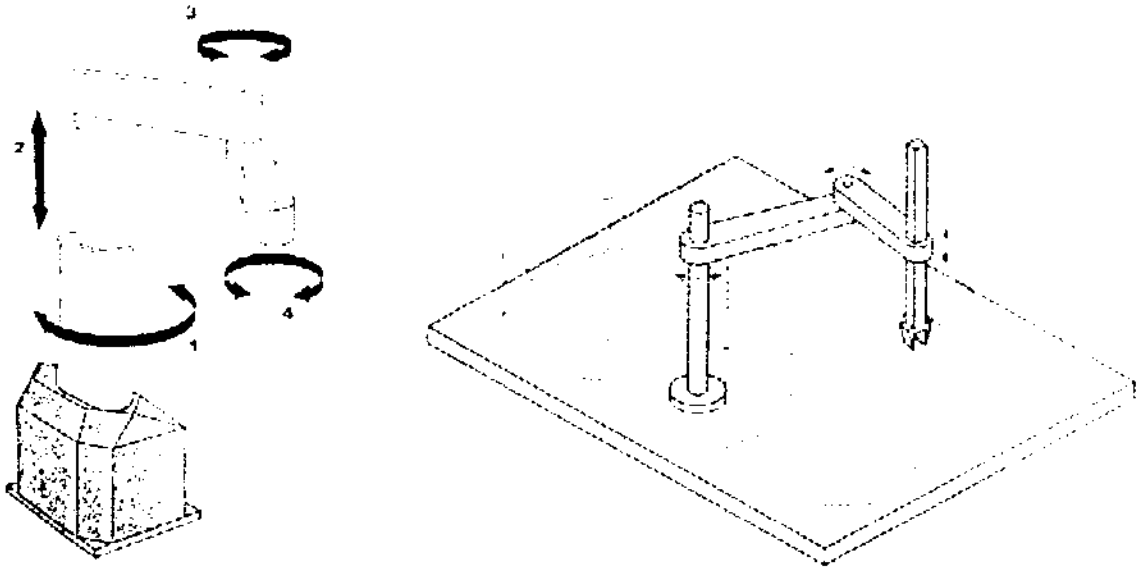
โครงสร้างข้อต่อหมุนให้หุ่นยนต์มี work Envelope ขนาดใหญ่เมื่อเทียบกับขนาดของส่วนทางกล หุ่นยนต์ 6 แกน จะให้การวางตัวของข้อมือ In Virtually ได้ทุก ๆ จุด Repeatability และความแม่นยำของหุ่นยนต์ดี อย่างไรก็ตามอาจจะไม่เพียงพอสำหรับการใช้งานประกอบที่ต้องการความถูกต้องบางอย่าง

ข้อเสียของหุ่นยนต์ชนิดนี้ คือ การสูญเสีย Performance ที่ขอบของ Work Envelope และ Low rigidity To The Frame Payload , ความแม่นยำและ Repeatability ของหุ่นยนต์จะตกลงที่ขอบนอกและขอบในของ Work Envelope แกนหมุนจะไม่ Stiff อย่าง liner Ball Screw Axis ดังนั้น Rigidity จะไม่ดีเมื่อเทียบกับ cylindrical หรือ SCARA Robot

Vertical Articulated Arm Robots ส่วนใหญ่มี Four-Bar Linkage สำหรับแกน (2) และ (3) ดังแสดงในรูปที่2.15 การออกแบบเพิ่มของ Four- Bar Linkage สามารถเพิ่ม Payload ให้แก่หุ่นยนต์ เป็นสองเท่าและเพิ่ม Rigidity ข้อเสียของ Four – Bar Linkage คือมันสามารถที่จะจด work Envelope โดยการกำจัด Angular Stroke ของแกน (2) และ (3)

หุ่นยนต์นี้รู้จักเจาะจงสำหรับการใช้ในงานพ่นสี เพราะหกแกนจัดให้ข้อมือมีการเคลื่อนที่คล่องแคล่วอย่างดีเยี่ยม และแกนหมุนมีการซีลต่อต้านการเสี่ยงต่อการเกิดไฟ หุ่นยนต์ถูกออกแบบพิเศษใช้ในสิ่งแวดล้อมชนิดนี้

2.2.5.5 SCARA (Selective Compliance Assembly Robot Arm)



รูปที่ 2.14 แสดงหุ่นยนต์ชนิด SCARA และ Work Envelope

SCARA มีความเร็วสูงสุดและ Repeatability ดีสุดในบรรดาหุ่นยนต์ทั้งหลาย หุ่นยนต์ชนิดนี้ใช้ในงานความถูกต้อง ความเร็วสูง และงานประกอบที่ไม่หนักจนเกิดไปนัก การใช้งานโดยทั่วไปคือการประกอบชิ้นส่วนของ PCB, ประกอบอุปกรณ์เครื่องกลไฟฟ้าขนาดเล็กและประกอบ Disk Drive ของคอมพิวเตอร์ หุ่นยนต์ชนิดนี้มีแกนตั้งเชิงเส้น (2), หนึ่งแกน แกนหมุนสองแกน (1,3) ซึ่งเคลื่อนแขนของหุ่นยนต์ในแนวระนาบและแกนเดียวที่ 4 สำหรับหมุนส่วนข้อมือของหุ่นยนต์

แกนที่ 1 จะเป็นส่วนการหมุนของหุ่นยนต์ แกนนี้จะถูกใช้อย่างมากในหุ่นยนต์ที่ใช้ในงานประกอบดังนั้นความเร่งและความเร็วของแกนนี้จะมีอิทธิพลต่อ Cycle ของการทำงานหุ่นยนต์แบบ SCARA บางชนิดใช้มอเตอร์ที่ขับเคลื่อนโดยตรงเพื่อเพิ่มความเร็ว ด้วยการขับเคลื่อนโดยตรงเพลาของมอเตอร์จะถูก Couole โดยตรงเข้ากับแกนของหุ่นยนต์สำหรับ Gear Ratio 1:1 ไม่มีการใช้ Gear Reducer หรือรอก

แกนที่ 2 จะถูกใช้ในการเคลื่อนที่ในแนวตั้ง ลักษณะเช่นนี้เป็นที่ต้องการสำหรับหุ่นยนต์ที่ใช้ในงานประกอบเพราะว่า 80% ของงานประกอบจะใช้การเคลื่อนที่ลงแบบแนวตั้ง ด้วยแกนตั้งนี้การเคลื่อนที่ลงจะเรียบและเร็วกว่าการเคลื่อนที่ของคน Coordinate หุ่นยนต์แบบแกน Coordinate เช่น Vertical Articulated arm Sperical Coordinate Robots ต้องเคลื่อน 2 หรือ 3 แกนพร้อมกันเพื่อ

กระทำภาระเคลื่อนที่แบบเส้นตรง ถ้าแกนไม่ได้เคลื่อน Synchrony อย่างสมบูรณ์ หุ่นยนต์ก็จะไม่เคลื่อนที่แบบเส้นตรงอย่างสมบูรณ์แบบ

แกนที่ 3 จะเปลี่ยนการเข้าถึงของหุ่นยนต์ และแกนที่ 4 จะหมุนข้อมือเพื่อปรับมุมองศาในการจับ โครงสร้างนี้จะทำให้มี Repeatability และ Accuracy ที่ดี ความแข็งแรงของ Robot Frame จะสูงมากในแกนแนวตั้งเพราะว่าความโน้มถ่วงของโลกไม่มีผลต่อโหลดของมอเตอร์บนส่วนปรกอบทางแนวราบมอเตอร์ที่รับทอร์คของแกน (1 และ 3) สามารถถูกเปลี่ยนเพื่อให้ Compliance สูงในแนวราบสิ่งนี้เป็นที่ต้องการเมื่อประกอบชิ้นส่วนในแนวตั้ง ถ้ามีการวางทิศทางผิดเพียงเล็กน้อยหุ่นยนต์ก็จะเคลื่อนในแนวราบเพื่อชดเชยทิศทางที่ผิดพลาด เนื่องจาก Rigidity ของแกนแนวตั้ง ชิ้นส่วนของจะยังคงอยู่ในแนวตั้งฉาก

2.2.5.6 Mobile Robot

รูปที่ 2.15 แสดงหุ่นยนต์แบบเคลื่อนที่ได้ (Mobile Robot)

หุ่นยนต์ที่ล้ำสมัยส่วนใหญ่คือ สามารถขับเคลื่อนด้วยตัวเองและสามารถเคลื่อนไปสู่จุดต่าง ๆ ภายใต้กำลังของมันเอง หุ่นยนต์ถูกติดตั้งบน Servo –Controlled Cart คล้ายกับ Automatic Guided Vehicle (AGV) Battery บน Cart ให้กำลังเพื่อขับเคลื่อนมัน สำหรับหุ่นยนต์มาจาก Battery หรือจากเต้ารับ

หุ่นยนต์ถูกนำโดยทางของขดลวดที่ฝังในพื้นหรือโดยระบบเรดาร์ การนำทางโดยใช้ขดลวดเป็นวิธีที่ใช้อย่างแพร่หลายสำหรับ AGV AGV มี Sensor ที่จะค้นหาสายไฟในพื้นและทำการแก้ไขสัญญาณ ถ้า AGV ออกนอกเส้นทางการนำทางโดยเรดาร์เป็นวิธีที่ล้ำสมัย เรดาร์จะรับรู้ตำแหน่งของหุ่นยนต์และสร้างความถูกต้องแก่เส้นทาง ไม่มีความต้องการของสายในพื้นดังนั้นเส้นทางของหุ่นยนต์แบบ Mobile สามารถที่จะเปลี่ยนได้โดยเขียนโปรแกรมใหม่

ตัวควบคุมต้องรู้ตำแหน่งที่ถูกต้องของ Mobile Robot เมื่อเทียบกับ Work Station ดังนั้นมันสามารถละเลยจุดของโปรแกรมเพื่อชดเชยความคลาดเคลื่อนทางตำแหน่งของระบบการขับเคลื่อน วิธีหนึ่งที่ใช้ในการชดเชยความคลาดเคลื่อนคือ การวัดตำแหน่ง Offset ของ Mobile Robot เทียบกับจุดอ้างอิงบน Work Station ป้อนข้อมูลตำแหน่งนั้นสู่ตัวควบคุมหุ่นยนต์ดังนั้นมันสามารถชดเชยค่า Offset

วิธีที่ง่ายกว่าของการทำให้ความคลาดเคลื่อนของตำแหน่งโดยการใช้ Fixture ซึ่งจะกำหนดตำแหน่งอย่างถูกต้องต่อ Mobile Robot ทั้งหมด เมื่อ Mobile Robot มาถึง Work Station The Fixture Clamp Mobile Robot ในตำแหน่งที่รู้ วิธีนี้โดยทั่วไปใช้กลุ่มของเข็มรูปโคนเพื่อกำหนด Robot Cart โดยแทงไปในหลุมอ้างอิงบน Cart และ ยกเพียงเล็กน้อยออกจากพื้นดิน

2.2.6 การประยุกต์ใช้งานหุ่นยนต์ชนิดต่าง ๆ

ตารางที่ 2.1 แสดงการประยุกต์ใช้งานหุ่นยนต์ชนิดต่างๆ

Application Area	Cartesian	Linear	Vertical Articulated	SCARA
Coating			●	●
Spot Welding			●	
Path Welding			●	
Grinding			●	●
Testing		●	●	●
Casting		●		
Press-Interlink		●	●	
Palettizing	●		●	●
Packaging	●		●	
Drilling, Milling	●		●	●
Insertion Device	●	●		●
Screwing				●
Assembly				●
Wiring	●		●	●

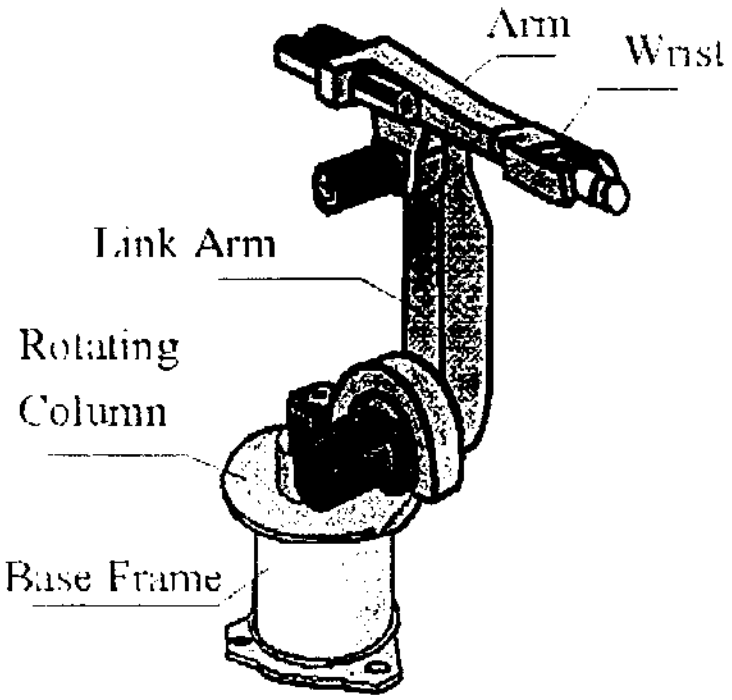
2.2.7 ส่วนประกอบของหุ่นยนต์และหน้าที่การทำงาน

(Robot Components and Functions)

องค์ประกอบของระบบหุ่นยนต์ สามารถแบ่งออกได้ 3 ชนิด คือ

1. ระบบทางกล
2. ระบบทางไฟฟ้าและอิเล็กทรอนิกส์
3. Workcell และการประยุกต์ใช้งาน

2.2.7.1 ระบบทางกล



รูปที่ 2 .16 แสดงส่วนประกอบทางกลของหุ่นยนต์

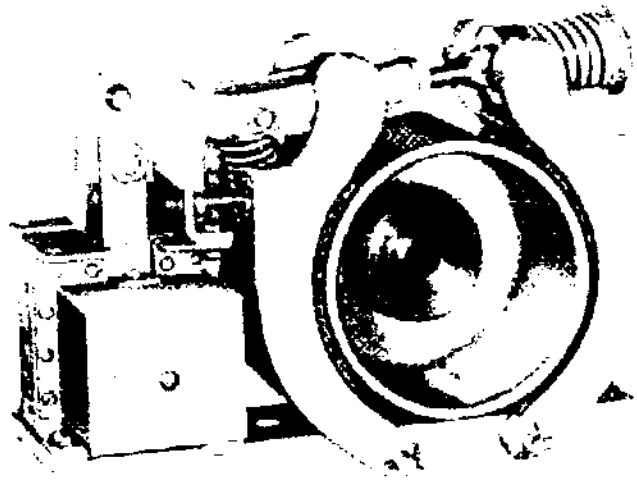
- Base Frame คือส่วนฐานของหุ่นยนต์ที่จะต้องรองรับน้ำหนักของหุ่นยนต์และ dynamic force ของหุ่นยนต์ขณะที่หุ่นยนต์เคลื่อนที่
- Rotating Column คือส่วนแกนที่ 1 ของหุ่นยนต์ที่จะให้การเคลื่อนที่แบบหมุนได้รอบตัว
- Link arm คือส่วนที่เชื่อมต่อระหว่างแขน (Arm)กับ Rotating Column
- Arm คือส่วนแขนของหุ่นยนต์

- Wrist คือส่วนข้อมือของหุ่นยนต์อาจประกอบไปด้วยจำนวนแกน 2 หรือ 3 แกน ก็ได้
- End effector คือ ส่วนปลายสุดของหุ่นยนต์เป็นส่วนที่จะติดตั้งอุปกรณ์เพื่อกระทำตามกระบวนการที่ต้องการ

2.2.7.2 ระบบไฟฟ้าและอิเล็กทรอนิคส์

1. ระบบเบรก

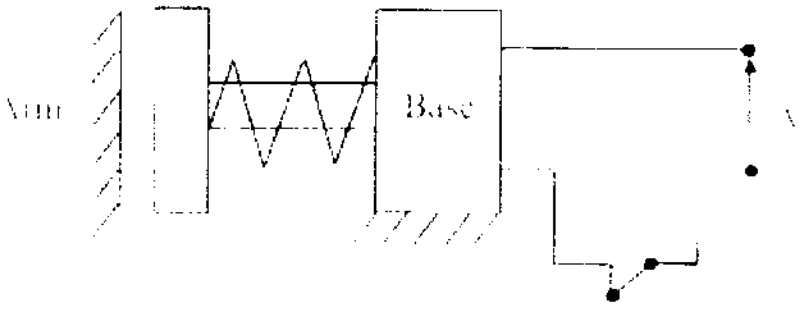
ระบบเบรกทางกล หรือเรียกว่า เบรกแม่เหล็ก หรือเบรกแรงเสียดทาน ทำงานคล้ายเบรกที่อยู่ในรถยนต์ดังรูปที่ 2.17



รูปที่ 2.17 แสดงระบบเบรกทางกล

โดยพื้นฐานแล้วเบรกทางกลจะประกอบด้วย brake shoes จำนวนสองตัว โดยมี Friction coating ติดตั้งอยู่ใกล้กับ drum ซึ่งประกอบติดกับเพลาของมอเตอร์ โดยมี tension spring ที่ shoe ซึ่งจะทำให้ shoe สัมผัสกับ drum ได้ โดยผิวสัมผัสนี้จะก่อให้เกิดการตอบสนองในการเบรก

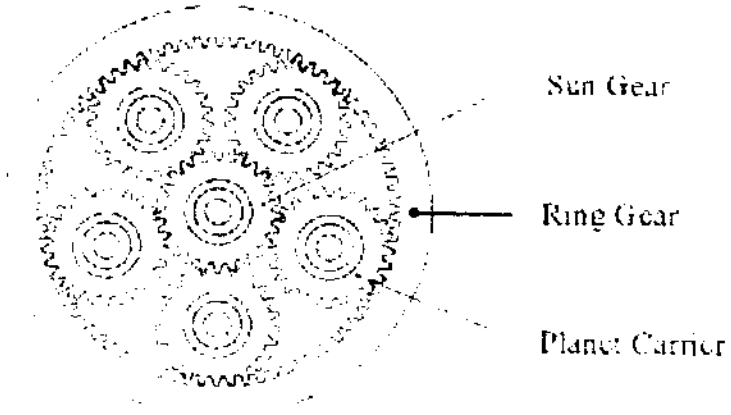
ถ้ามอเตอร์กำลังหมุนเราจะต้องยก shoe ขึ้นจาก drum โดย solenoid ที่ประกอบกับระบบคานจะทำงานนี้เมื่อมอเตอร์ทำงาน solenoid จะถูกกระตุ้น ซึ่งจะเคลื่อนคานเพื่อยก brake shoe และให้มอเตอร์หมุน เมื่อมอเตอร์หยุด solenoid จะไม่กระตุ้น และ tension force ของสปริงจะทำให้ brake shoe สัมผัสกับ drum และส่งผลให้เกิดแรงเสียดทานขึ้นซึ่งจะทำให้มอเตอร์หยุด



รูปที่ 2.18 แสดงตัวอย่างวงจรที่ควบคุมการทำงานของ Brake Shoe

2. ระบบ Gear

- Planetary Gear



รูปที่ 2.19 แสดงส่วนประกอบที่สำคัญของ Planetary Gear

ประกอบด้วย Sun Gear , Ring Gear และ Planet Carrier การทำงานนั้น จะมีลักษณะการเคลื่อนตัวคล้ายดาวเคราะห์และดวงอาทิตย์ในระบบสุริยจักรวาล ซึ่งระบบนี้จะให้ torque ขนาดสูง

รูปที่ 2.20 แสดง Planetary Gear โดยมี Holder Carrier

ตัวอย่าง การขับเคลื่อนในระบบ 1 ดาวเคราะห์ โดยที่ ring gear ถูก fix ให้อยู่กับที่ โดยการขับเคลื่อนเริ่มต้นที่ Sun ring holder ของ planet carrier จะเคลื่อนที่ในทิศทางเดียวกับ Sun ring ซึ่ง planet carrier จะเคลื่อนที่เป็นวงล้ออยู่ระหว่างนั้น โดยความเร็วและการส่งกำลังจะขึ้นอยู่กับจำนวนเป็นของ Sun ring และ ring gear และลักษณะการเคลื่อนที่สามารถดูได้จากตารางที่ 2.2

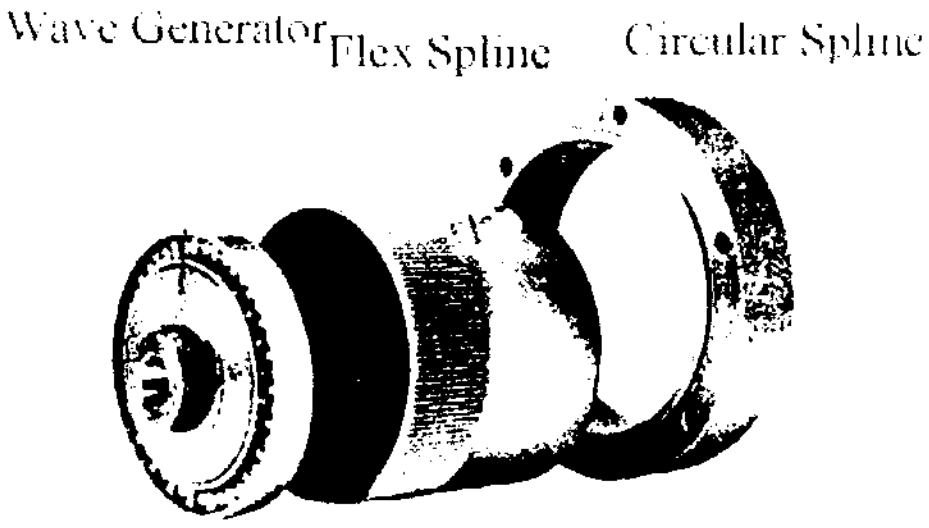
ตารางที่ 2.2 แสดงคุณสมบัติการเคลื่อนที่ของ Planetary Gear

ตัวขับเคลื่อน	ตัวตาม	ตัวหยุดนิ่ง	การหมุน
Sun Gear	Planet Carrier	Ring Gear	เคลื่อนที่ช้าลง ทิศตามกัน
Planet Carrier	Sun Gear	Ring Gear	เคลื่อนที่ช้าลง ทิศตามกัน
Ring Gear	Planet Carrier	Sun Gear	เคลื่อนที่ช้าลง ทิศตามกัน
Planet Carrier	Ring Gear	Sun Gear	เคลื่อนที่เร็วลง ทิศตามกัน
Sun Gear	Ring Gear	Planet Carrier	เคลื่อนที่ช้าลงทิศตรงข้าม
Ring Gear	Sun Gear	Planet Carrier	เคลื่อนที่เร็วลงทิศตรงข้าม

Planetary Gear มีข้อดีมากมายเนื่องจากระบบโครงสร้างเพราะ

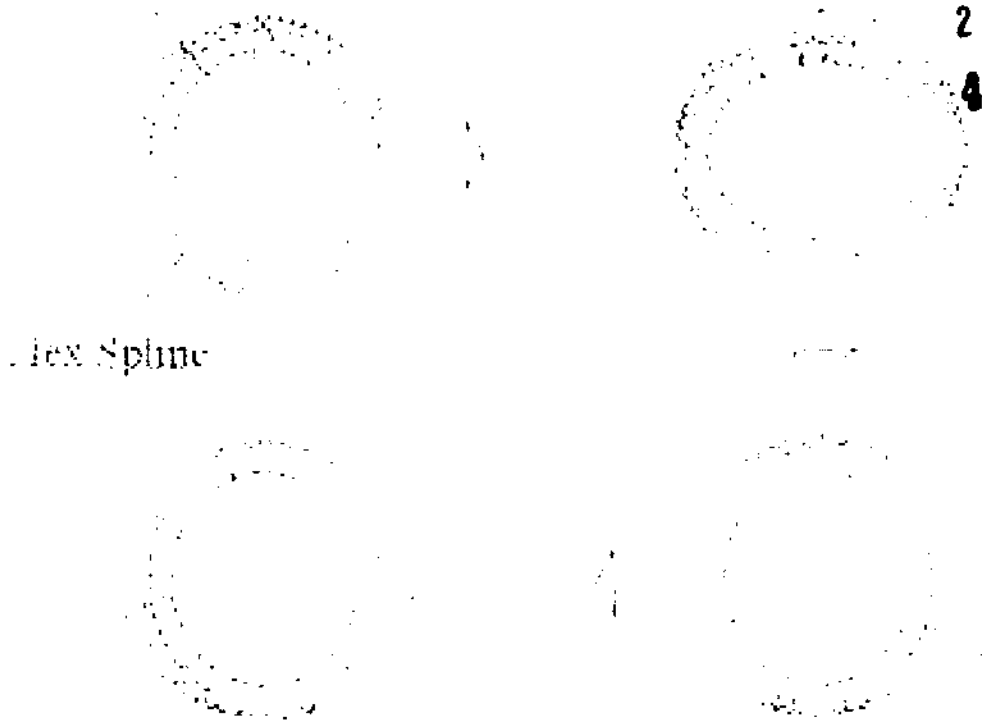
1. มีขนาดโดยมีฟันจำนวนมากในการส่งต่อ torque
2. มีมวลน้อยและสามารถให้กำลังได้สูงแม้จะมีขนาดเล็ก
3. สามารถมีการเปลี่ยน gear ได้โดยฟันเฟืองวงล้อไม่จำเป็นที่จะต้องถูกจับต้อง กล่าวถึงไม่มีการรบกวนการทำงานของฟันเฟืองวงล้อ

- Harmonic Drive



รูปที่ 2.21 แสดงส่วนประกอบที่สำคัญของ Harmonic Drive

ประกอบด้วยแผ่นเดินเครื่องที่เป็นรูปวงรีที่ทำงานโดย bearing แผ่นนี้จะไปกด flex spline ที่มีฟันเฟืองอยู่ภายนอกและจะกดเข้ากับวงแหวนที่อยู่นิ่งอันเนื่องมาจากรูปวงรีทำให้ flex spline ไปสัมผัสกับฟันเฟืองที่อยู่นิ่งเพียง 2 จุดเท่านั้น ฟันเฟืองของ flex spline จะมีฟันน้อยกว่าภายนอกอยู่ 2 ฟัน ถ้าหากว่าจำนวนฟันของ flex spline เท่ากับ 200 Circular spline จะมี 202 Harmonic Drive จะมีโครงสร้างที่ง่ายและมีประสิทธิภาพสูงมากว่า 80% มี Moment of inertia ต่ำ และฟันเฟืองมีความแข็งสูง



รูปที่ 2.22 แสดงลำดับการทำงานของ Harmonic Drive

3. ระบบขับเคลื่อน

เป็นระบบที่ไปขับให้ส่วนต่างๆเคลื่อนที่โดยใช้กันมากก็มี 3 แบบ ด้วยกันคือ

1. ระบบไฮดรอลิก (Hydraulic) ซึ่งเป็นระบบที่ง่ายต่อการดูแลรักษาทนทานและยังทำให้การเคลื่อนที่ทำได้เร็ว
2. ระบบมอเตอร์ไฟฟ้า (Electric Motor) ใช้ในการควบคุมการเคลื่อนที่ของส่วนต่างๆของหุ่นยนต์ระบบนี้จะไม่มีกำลังเหมือนระบบแรก แต่จะมีรายละเอียดไม่ว่าในแง่ของความแม่นยำของตำแหน่งที่เคลื่อนที่ไปหรือการทำซ้ำจะดีกว่า
3. ระบบลมอัด (Pneumatic) ระบบนี้จะใช้กับหุ่นยนต์ขนาดเล็กอีกทั้งระบบไม่ยุ่งยาก

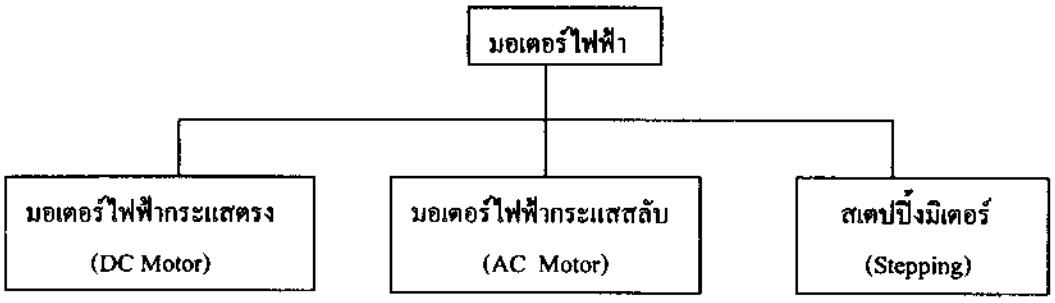
ตารางที่ 2.3 เปรียบเทียบชนิดของ Actuator

	ข้อดี	ข้อเสีย
Hydraulic	• ใช้กับโหลดขนาดใหญ่ • ปลอดภัยกับสภาพแวดล้อมเช่น การพ่นสี	• การรั่วของน้ำมัน • บำรุงรักษามาก • ราคาแพง
Electric	• ความเร็ว • มีความแม่นยำResolution และ Repeatability ดี • ราคาต่ำ • สะอาดและเงียบ • ใช้กับโหลดขนาดใหญ่	• ต้องการ Mechanical brake • การบำรุงรักษาแปรงถ่านจำเป็นกับ DC motor
Pneumatic	• ราคาต่ำ • สะอาด • เร็ว	• ไม่มีความแม่นยำ • ต้องการสารหล่อลื่นให้กระบอกสูบ • มีกำลังน้อย • เคลื่อนโหลดขนาดใหญ่ไม่ได้

4. มอเตอร์ไฟฟ้า

ความหมายของมอเตอร์ไฟฟ้านั้นก็คือ เครื่องกลไฟฟ้าชนิดหนึ่งที่ทำหน้าที่เปลี่ยนพลังงานไฟฟ้าเป็นพลังงานกล โดยมีลักษณะการทำงานเป็นแบบเชิงมุม หรือในลักษณะของการหมุนซึ่งหลักการหมุนในมอเตอร์นั้นเกิดจากการดูดและการผลักกันของขั้วแม่เหล็ก จากหลักการดังกล่าวจึงทำให้มอเตอร์มีส่วนประกอบที่สำคัญอยู่ 2 ส่วน คือ ส่วนที่อยู่กับที่ (Stator) และส่วนที่เคลื่อนที่ (Rotor) โดยขั้วแม่เหล็ก อาจใช้เป็นแม่เหล็กถาวร หรือแม่เหล็กไฟฟ้าก็ได้

มอเตอร์มีอยู่หลายแบบหลายชนิดด้วยกัน โดยในที่นี้จะแบ่งมอเตอร์ออกเป็น 3 ประเภท

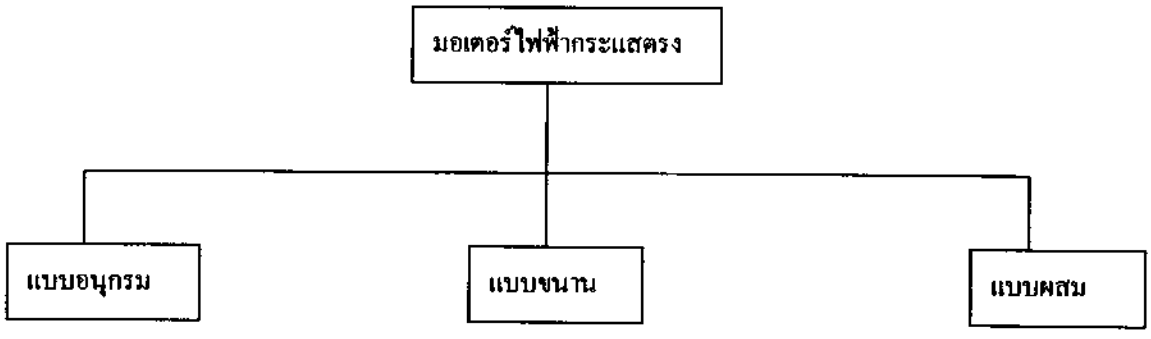


รูปที่ 2.23 การแบ่งประเภทของมอเตอร์ไฟฟ้า

- มอเตอร์ไฟฟ้ากระแสตรง (Direct Current Motor)

มีโครงสร้างและส่วนประกอบที่สำคัญ 2 ส่วน คือ ส่วนที่อยู่กับที่ (Stator) และส่วนที่เคลื่อนที่ (Rotor) ซึ่งส่วนนี้เรียกว่า อาร์เมเจอร์(Armature)

การต่อขดลวดของอาร์เมเจอร์กับขดลวดที่สเตเตอร์ (Field) เข้ากับแหล่งจ่ายไฟกระแสตรง ภายนอกนั้นสามารถทำได้ 3 แบบ คือ



รูปที่ 2.24 การแบ่งประเภทของมอเตอร์ไฟฟ้ากระแสตรง

- มอเตอร์ไฟฟ้ากระแสตรงแบบอนุกรม (series)

มอเตอร์แบบนี้ขดลวดของอาร์เมเจอร์จะต่ออนุกรมกับขดลวดที่สเตเตอร์(field) ข้อดีของมอเตอร์แบบนี้ คือให้แรงบิดขณะเริ่มเดินสูง และมีความเร็วรอบต่ำ เมื่อโหลดมากๆและความเร็วสูงเมื่อโหลดน้อยๆซึ่งอาจจะทำให้มอเตอร์ได้รับอันตรายได้จึงควรต่อ (Coupling) โดยตรงกับโหลด

- มอเตอร์ไฟฟ้ากระแสตรงแบบขนาน (shunts)

มอเตอร์แบบนี้ขดลวดอาร์เมเจอร์จะต่อขนานกับลวดที่สเตเตอร์โดยให้ความเร็วที่ค่อนข้างคงที่ และแรงบิดขณะเดินไม่สูง เมื่อเปรียบเทียบกับมอเตอร์แบบอนุกรม

- มอเตอร์ไฟฟ้ากระแสตรงแบบผสม (Compound)

มอเตอร์แบบนี้จะมีขดลวดที่สเตเตอร์ 2 ชุด และต่อเป็นแบบผสมร่วมกับขดลวดที่อาร์เมเจอร์ หรือกล่าวอีกอย่างหนึ่ง ก็คือ การนำมอเตอร์แบบอนุกรมและขนานมารวมไว้ในตัวเดียวกัน จึงทำให้คุณสมบัติของมอเตอร์แบบนี้อยู่ระหว่างมอเตอร์ทั้ง 2 แบบ

- มอเตอร์ไฟฟ้ากระแสสลับ (Alternating Current Motor)

หลักการทำงานของมอเตอร์ชนิดนี้ หากติดตั้งแม่เหล็กถาวรไว้ที่จุดกึ่งกลางของแม่เหล็ก แล้วหมุนแท่งแม่เหล็ก ขณะที่จ่ายไฟกระแสสลับความถี่ 50 รอบ/วินาที (Hz) เข้าขดลวดของสเตเตอร์แท่งแม่เหล็กถาวรหรือโรเตอร์จะยังหมุนอยู่ต่อไป เนื่องจากการดูดและผลักระหว่างขั้วแม่เหล็กไฟฟ้าของสเตเตอร์และขั้วแม่เหล็กถาวรของโรเตอร์ ขั้วแม่เหล็กไฟฟ้าที่เกิดขึ้นสเตเตอร์จะมีการเปลี่ยนแปลงอยู่ตลอดเวลา โดยขึ้นอยู่กับความถี่ของกระแสไฟที่ป้อนเข้าไป เหตุการณ์ดังกล่าวนี้เรียกว่า สนามแม่เหล็กหมุน ส่วนโรเตอร์จะปรับตัวให้หมุนด้วยความเร็ว 50 รอบ/วินาที หรือเท่ากับ300 รอบต่อนาที เช่นกัน ซึ่งความเร็วดังกล่าวเป็นความเร็วทางทฤษฎีเรียกกันว่า ความเร็วซิงโครนัส

ความเร็วซิงโคนัสของมอเตอร์ไฟฟ้า สามารถหาได้จากสูตรต่อไปนี้

$$Ns = 120\ f/p \qquad (2.1)$$

เมื่อ f คือความถี่ของกระแสไฟฟ้า ส่วน p คือจำนวนขั้วแม่เหล็กของมอเตอร์

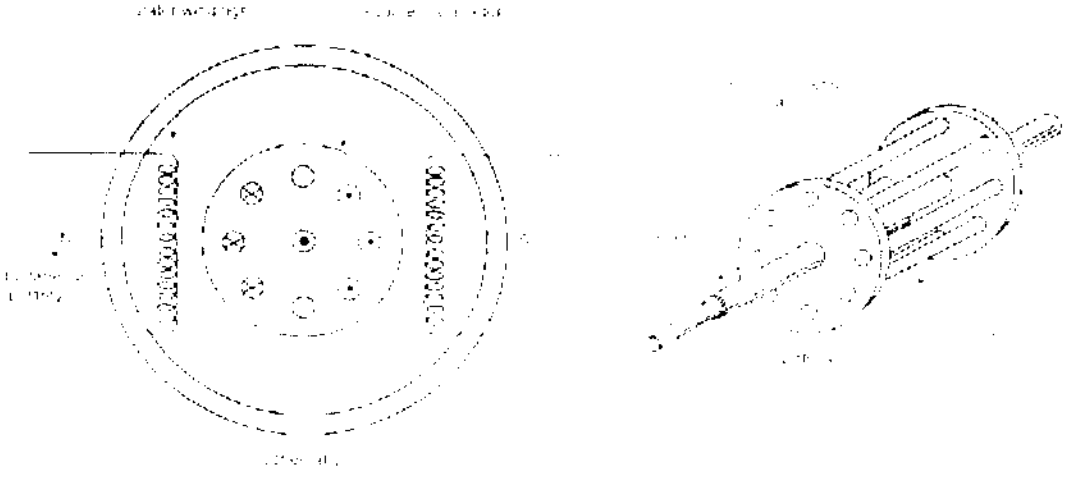
ในความจริงแล้วโรเตอร์ของมอเตอร์ แม้จะหมุนไปในทิศทางเดียวกันกับสนามแม่เหล็กหมุนของสเตเตอร์ (Nr) เรียกว่าสลิป (Slip Speed;Nslip)

$$Nslip = Ns-Nr \qquad (2.2)$$

อัตราส่วนระหว่าง Nslip ต่อ Ns เรียกว่า สลิป (Slip ,S)

$$s = [(Ns-Nr)/Ns]x\ 100 \qquad (2.3)$$

ส่วนโรเตอร์ในทางปฏิบัติจริงๆแล้ว จะไม่ได้ใช้แม่เหล็กถาวร แต่จะใช้โรเตอร์ต่อไปนี้แทนคือ แบบกรงกระรอก (Squirrel Cage) โรเตอร์แบบนี้จะมีแท่งตัวนำฝังอยู่ภายในโรเตอร์คล้ายกรงกระรอก โดยที่ปลายทั้งสองด้านจะถูกลัดวงจรด้วยตัวนำรูปแหวน แสดงดังรูปที่2.33



รูปที่ 2 .25 แสดงโครงสร้างมอเตอร์ไฟฟ้ากระแสสลับเหนี่ยวนำแบบกรงกระรอก

- มอเตอร์ไฟฟ้ากระแสสลับแบบเหนี่ยวนำ (Induction Motor)

การเกิดขั้วแม่เหล็กที่โรเตอร์แบบนี้ จะใช้หลักการของสนามแม่เหล็กหมุนที่สเตเตอร์ตัดผ่านตัวนำในโรเตอร์ ทำให้เกิดแรงดันไฟฟ้าเหนี่ยวนำขึ้น (หลักการคล้ายหม้อแปลงไฟฟ้า) และแรงดันไฟฟ้าเหนี่ยวนำที่ถูกลัดวงจรด้วยวงแหวนตัวนำจะทำให้กระแสไหลครบวงจร เกิดเป็นขั้วแม่เหล็กตรงข้ามกับสเตเตอร์

มอเตอร์แบบนี้ จะมีแรงบิดออกตัวที่ต่ำ จนถึงปานกลาง แต่ความเร็วค่อนข้างคงที่จึงมีใช้กันอยู่ในพัดลมสายพานลำเลียง เครื่องอัดอากาศ ชุดขับปั๊มไฮดรอลิกส์ เครื่องปั๊มโลหะ ซึ่งถ้าเป็นมอเตอร์ขนาดเล็กจะมีการใช้มอเตอร์กรงกระรอกได้ทั่วๆไปในที่ที่มีแหล่งจ่ายไฟฟ้ากระแสสลับสามเฟสอยู่

- มอเตอร์ไฟฟ้ากระแสสลับแบบโรเตอร์ใช้กับไฟกระแสตรง (Synchronous Motor)

มอเตอร์แบบนี้โดยทั่วไป เรียกว่า มอเตอร์ซิงโครนัส (Synchronous Motor) เมื่อป้อนไฟฟ้ากระแสตรงผ่านวงแหวนลื่น (Slip-Ring) เข้าไปในขดลวดของโรเตอร์ จะทำให้เกิดขั้วแม่เหล็กเหนือและใต้ขึ้นบนโครเตอร์จำนวนขั้วจะเท่ากับจำนวนขั้วที่สเตเตอร์ ดังนั้นเมื่อสนามแม่เหล็กของสเตเตอร์หมุน โรเตอร์ซึ่งดูดติดกันอยู่ ก็จะหมุนไปในทิศทางและความเร็วเดียวกัน คือ ความเร็วซิงโครนัส ข้อดีของมอเตอร์แบบนี้ คือ มีความเร็วคงที่ ไม่ว่าจะมีโหลดหรือไม่มีโหลดก็ตาม

- มอเตอร์สเตปปิ้ง (Stepping Motor)

มอเตอร์ไฟฟ้ากระแสตรงหรือกระแสสลับจะให้การหมุนที่ต่อเนื่องแต่ในงานบางลักษณะอาจต้องการหมุนเพียง 2-3 องศา หรือต้องการหยุด ณ ตำแหน่งใด ๆก็ได้ ซึ่งงานในลักษณะดังกล่าวต้องอาศัยมอเตอร์สเตปปิ้งตัวอย่างเช่น หัวอ่าน/เขียนในตัวขับดิสก์ , X-Y Ploter , X-Y Position Table ฯลฯ มอเตอร์สเตปปิ้งที่ใช้กันอยู่ทั่วไปสามารถแบ่งออกเป็น 2 ประเภท คือ แบบโรเตอร์เป็นแบบแม่เหล็กถาวร และโรเตอร์เป็นแกนเหล็กอ่อน

- แบบโรเตอร์เป็นแม่เหล็กถาวร

สเตเตอร์ของมอเตอร์ชนิดนี้ ประกอบด้วยกลุ่มของขดลวดที่ใช้กับไฟกระแสตรงในการปิด-เปิด ไฟไปยังกลุ่มของขดลวดทีละครั้ง ทำให้มอเตอร์หมุนไปที่ละสเตป โดยแต่ละสเตปจะมีองศาเท่ากัน มอเตอร์บางตัวมีสเตปแค่ 1.8 องศา บางตัวอาจสูงถึง 45 หรือ 90 องศา ซึ่งทั้งนี้ขึ้นอยู่กับขั้วแม่เหล็กที่สเตเตอร์และลักษณะการควบคุม ดังตัวอย่างต่อไปนี้

- การควบคุมแบบเต็มสเตป (Full Step)

การควบคุมการทำงานแบบนี้สามารถหาค่ามุมในแต่ละสเตปได้ดังนี้

$$\text{Step Angle} = 360^{0}/P$$

เมื่อ P คือจำนวนขั้วแม่เหล็กที่สเตเตอร์ และจากรูปในตัวอย่าง ดังนั้นในแต่ละสเตปของมอเตอร์ตัวนี้มีค่าเท่ากับ $360^{0}/4$ เท่ากับ 90 องศา

ตำแหน่งของกดสวิทซ์ S1 และ S2 สามารถเขียนเป็นตารางซึ่งมีความสัมพันธ์กับสเตปในการหมุนได้ดังนี้

ตารางที่ 2.4 แสดงความสัมพันธ์กับสเตปในการหมุนแบบทวนเข็มนาฬิกา

	SW1		SW2	
	ขดลวด A	ขดลวด B	ขดลวด C	ขดลวด D
1	1(On)	0(Off)	1(On)	0(Off)
2	0(Off)	1(On)	1(On)	0(Off)
3	0(Off)	1(On)	0(Off)	1(On)
4	1(On)	0(Off)	0(Off)	1(On)

จากรูปและตารางการหมุนในแต่ละสเตปจะหมุนเป็น 90 องศา และเป็นการหมุนแบบทวนเข็มนาฬิกา หากต้องการหมุนแบบตามเข็มนาฬิกา สามารถให้สัญญาณไฟที่ขดลวดโดยการกด SW1 และ SW2 ตามตารางต่อไปนี้

ตารางที่ 2.5 แสดงความสัมพันธ์กับสเตปในการหมุนแบบตามเข็มนาฬิกา

	SW1		SW2	
	ขดลวด A	ขดลวด B	ขดลวด C	ขดลวด D
1	1(On)	0(Off)	1(On)	0(Off)
2	1(On)	0(Off)	0(Off)	1(On)
3	0(Off)	1(On)	0(Off)	1(On)
4	0(Off)	1(On)	1(On)	0(Off)

- การควบคุมแบบครึ่งสเตป (Half Step)

การควบคุมด้วยวิธีนี้ มุมในแต่ละสเตปนั้นสามารถหาได้จากStep Angle = 360^0 / 2P ดังนั้นมอเตอร์ในตัวอย่างที่ผ่านมาจะมีมุมในแต่ละสเตปเท่ากับ 360^0 / 2P เท่ากับ 45 องศา

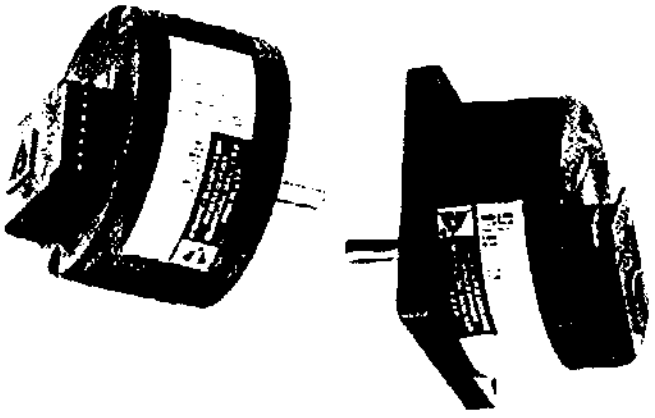
มอเตอร์จะหมุนตามเข็มนาฬิกา ซึ่งสามารถเขียนตารางสัมพันธ์ระหว่างสเตปในการหมุนกับการปิด - เปิดสวิทซ์ไฟเข้าขดลวดแต่ละชุดได้ตามตารางต่อไปนี้

ตารางที่ 2 .6 แสดงสเตปในการหมุนกับการปิด - เปิดสวิทซ์ไฟ

	SW1		SW2	
	ขดลวด A	ขดลวด B	ขดลวด C	ขดลวด D
1	1(On)	0(Off)	1(On)	0(Off)
2	1(On)	0(Off)	0(Off)	0(Off)
3	1(On)	0(Off)	0(Off)	1(On)
4	0(Off)	0(Off)	0(Off)	1(On)
5	0(Off)	1(On)	0(Off)	1(On)
6	0(Off)	1(On)	0(Off)	0(Off)
7	0(Off)	1(On)	1(On)	0(Off)
8	0(Off)	0(Off)	1(On)	0(Off)

5.Transducer

คือ อุปกรณ์ที่แปลงค่าพารามิเตอร์ทางกายภาพ เช่น อุณหภูมิ ความดัน หรือ น้ำหนักเป็นสัญญาณไฟฟ้าหรือที่นำมาใช้ในหุ่นยนต์ก็คือ การแปลงตำแหน่ง และส่งสัญญาณดิจิตอล กลับไปยัง Controller เพื่อทำการประมวลผลและควบคุมตำแหน่งต่อไป



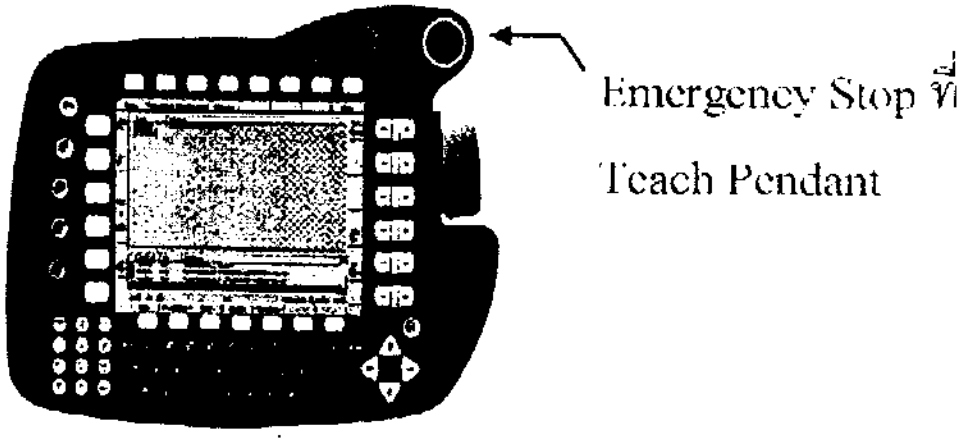
รูปที่ 2 .26 แสดงตัวอย่าง Optical Rotary Encoder

6. Controller

คือ ตัวควบคุมการทำงานของหุ่นยนต์ต่าง ๆ อันประกอบไปด้วยการควบคุม การหยุดทำงานอย่างฉุกเฉิน, การเชื่อมต่อระหว่างอินพุทและเอาท์พุท, การควบคุมแหล่งจ่ายพลังงาน, การติดต่อสื่อสารกับ teach pendant

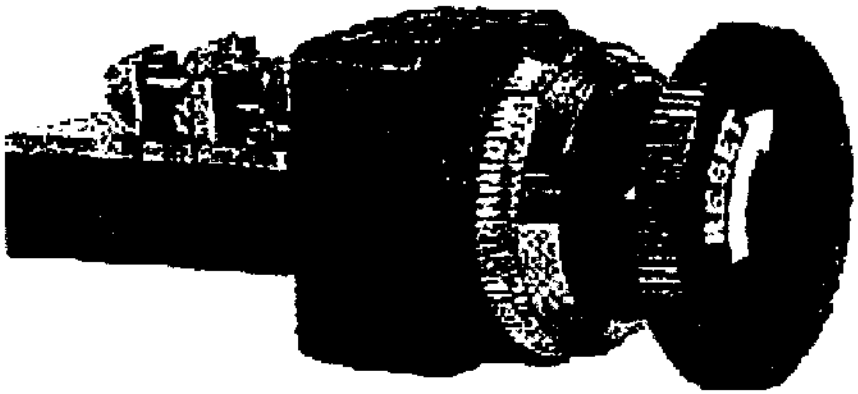
- Emergency Stop

  จะมีหน้าที่ คือ ตัดพลังงานจากแหล่งจ่าย เมื่อ Emergency Stop ทำงาน

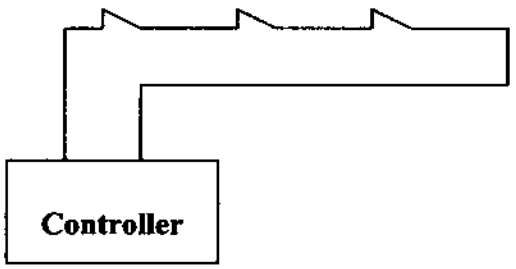


รูปที่ 2 .27 แสดง Emergency Stop ที่ Teach Pendant

เราสามารถติดตั้ง Emergency Stop ภายนอกซึ่งติดตั้งได้ ณ จุดที่ต้องการ เช่น Door Switch



รูปที่ 2 .28 แสดงตัวอย่างของ Emergency Stop ที่อาจติดตั้งภายนอก workcell

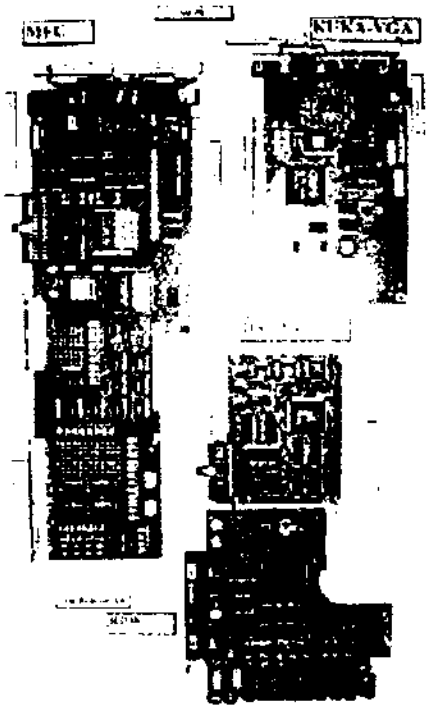


รูปที่ 2 .29 แสดงตัวอย่างของวงจร Emergency Stop

- Control Unit

  จะมีหน้าที่ คือ

  1. Processor ทางการคำนวณตำแหน่งการเคลื่อนที่
  2. เก็บข้อมูลและโปรแกรมของหุ่นยนต์
  3. ควบคุมการทำงานของวงจร Emergency Stop
  4. สื่อสารข้อมูลกับ Teach pendant ควบคุมการทำงานของ Power Module หรือชุดส่งจ่ายกำลัง



รูปที่ 2.30 แสดงส่วนประกอบภายในของ Control Unit

- Input /output Module

จะมีหน้าที่คือ รับสัญญาณ Input เพื่อประมวลผลและส่งสัญญาณ Output สำหรับสัญญาณ Digital หรือ Analog ไปยังอุปกรณ์ภายนอกของ Workcell ตามโปรแกรม

- Teach Pendant

คืออุปกรณ์เชื่อมต่อระหว่างมนุษย์กับเครื่องจักรเพื่อ

1. ควบคุมการเคลื่อนที่ของหุ่นยนต์
2. โปรแกรมหุ่นยนต์
3. เก็บข้อมูลตำแหน่งของหุ่นยนต์
4. แสดงโปรแกรม,ข้อมูล I/O ข้อความและคำเตือน
5. ควบคุม Emergency Stop

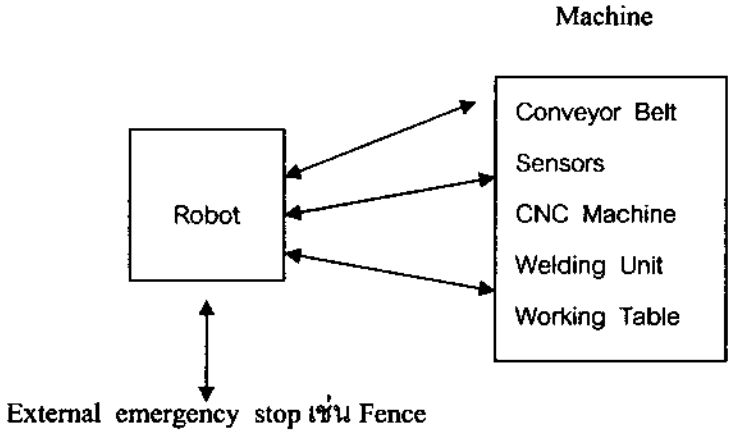
- Programming Devices

  จะมีหน้าที่ คือ

  1. Uplod และ Download โปรแกรมของหุ่นยนต์
  2. ออกระบบโปรแกรมหุ่นยนต์
  3. เก็บข้อมูลโปรแกรมต่างๆบน Harddrive
  4. ทดสอบโปแกรมที่ Simulation Software

2.2.7.3 Workcell / Application

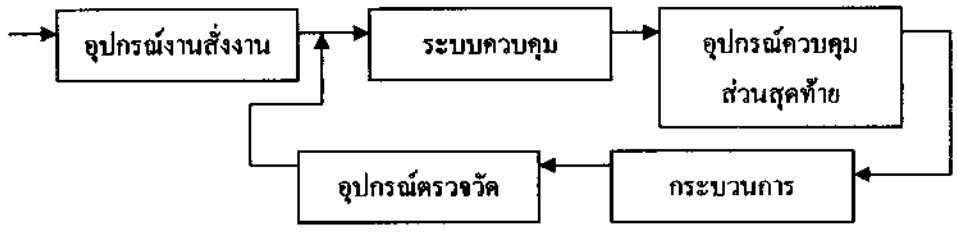
จะเป็นกระบวนการที่ขึ้นอยู่กับอุปกรณ์และการเชื่อมต่อระหว่างหุ่นยนต์กับโปรแกรมโดยสัญญาณ Input และ Output



รูปที่ 2 .31 แสดงองค์ประกอบของ Workcll และ Application

2.2.7.4 องค์ประกอบของการควบคุม

การควบคุมในงานอุตสาหกรรมในรูปแบบที่แตกต่างกันออกไปนั้น จะมีองค์ประกอบหลักๆที่สำคัญและคล้ายคลึงกันดังนี้



รูปที่ 2.32 แสดงองค์ประกอบของการควบคุม

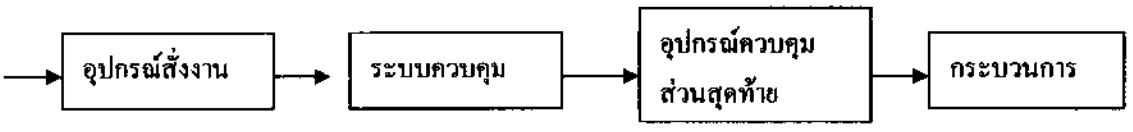
- อุปกรณ์สั่งงาน (Input Element)
  เป็นอุปกรณ์ที่ใช้ในการเริ่มต้น หยุด รวมทั้งตั้งค่าเป้าหมายให้กับระบบการควบคุม ตัวอย่างเช่น สวิทช์ต่างๆเซนเซอร์ ทรานสดิวเซอร์ รวมถึงอุปกรณ์ทางด้านอิเล็กทรอนิกส์ที่นำมาประกอบร่วมกันเพื่อส่งสัญญาณให้กับระบบควบคุม
- ระบบควบคุม (Control System)
  หมายถึง สิ่งที่ทำหน้าที่ออกคำสั่งหรือกำหนดสัญญาณควบคุมตามกฎเกณฑ์ การควบคุมที่กำหนดไว้ล่วงหน้าคำสั่งหรือสัญญาณควบคุมนี้อาจจะสัมพันธ์กับสัญญาณขาเข้าที่ได้รับอุปกรณ์ตรวจวัด
- อุปกรณ์ควบคุมส่วนท้าย (Final Control element)
  สัญญาณที่ได้จากระบบควบคุมในบางครั้งจะมีขนาดที่ค่อนข้างต่ำ รวมทั้งบางครั้งไม่สอดคล้องกับสัญญาณของกระบวนการ จึงจำเป็นต้องใช้อุปกรณ์ในส่วนนี้เพื่อเปลี่ยนแปลงค่าตัวแปรดังกล่าว
- กระบวนการ (Process)
  หมายถึงกระบวนการทางฟิสิกส์ เคมี กายภาพที่ต้องการควบคุมให้มีสภาวะการทำงานตามความต้องการในขณะที่สภาวะการทำงานหรือสภาพแวดล้อมอาจมีการเปลี่ยนแปลง
- อุปกรณ์ตรวจวัด (Measuring Device)
  เป็นอุปกรณ์ที่ให้สัญญาณขาออก ซึ่งมีขนาดสัมพันธ์กับตัวแปรของสิ่งที่ต้องการวัดหรือสั่งงาน ตัวอย่างเช่น เซนเซอร์และทรานสดิวเซอร์ประเภทต่างๆ

2.2.7.5 ลักษณะของการควบคุม

วิธีการที่จะควบคุมพารามิเตอร์หรือสัญญาณต่างๆ ให้เป็นไปตามวัตถุประสงค์ที่ต้องการนั้นสามารถกระทำได้ 2 วิธีด้วยกัน คือ การควบคุมแบบเปิด (Open-Loop Control) และการควบคุมแบบปิด (Closed-Loop Control)

- การควบคุมแบบเปิด (Open-loop Control)

เป็นการควบคุมที่เอาท์พุทของระบบไม่มีผลต่อการควบคุม นั่นคือ เอาท์พุทของระบบจะไม่ถูกวัดหรือถูกป้อนกลับมาเพื่อเปรียบเทียบกับอินพุท

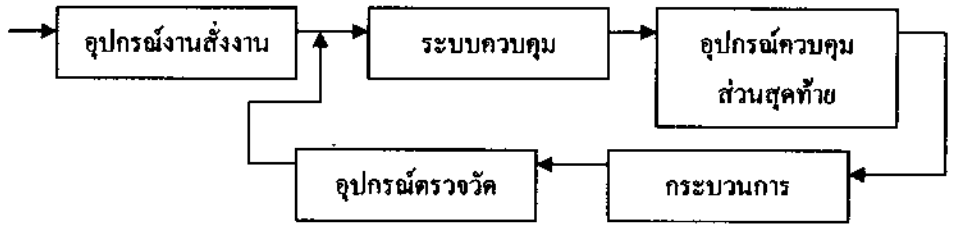


รูปที่ 2.33 แสดง Block Diagram ของระบบการควบคุมแบบเปิด

ในการควบคุมแบบนี้เอาท์พุทไม่ได้นำมาเปรียบเทียบกับอินพุท ดังนั้นความเที่ยงตรงของระบบจะขึ้นอยู่กับการเปรียบเทียบ ในทางปฏิบัติแล้วจะสามารถใช้การควบคุมแบบนี้ได้ถ้าทราบถึงความสัมพันธ์ระหว่างอินพุทและเอาท์พุทของระบบ และระบบควบคุมที่ทำงานตามเวลาที่กำหนดไว้เป็นการควบคุมแบบนี้ ตัวอย่างการควบคุมแบบเปิด ได้แก่ การเปลี่ยนทิศทางการหมุนมอเตอร์

- การควบคุมแบบปิด(Closed- Loop Control)

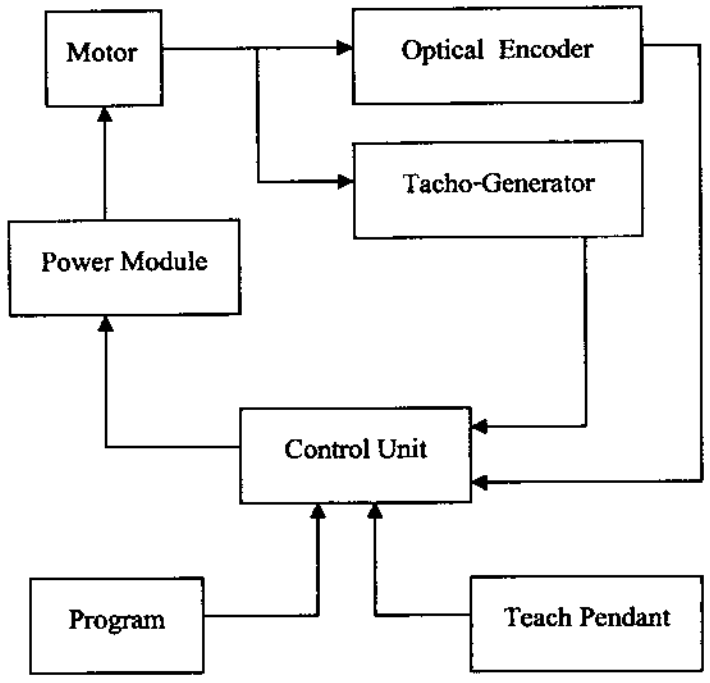
เป็นการควบคุมที่สัญญาณเอาท์พุท มีผลโดยตรงต่อการควบคุม ดังนั้นการควบคุมแบบนี้ก็คือ ระบบควบคุมแบบป้อนกลับ (Feedback Control) นั่นเอง ซึ่งสัญญาณป้อนกลับนี้อาจจะเป็นสัญญาณเอาท์พุทโดยตรง หรือเป็นสัญญาณที่สัมพันธ์หรือสอดคล้องกับสัญญาณเอาท์พุทก็ได้



รูปที่ 2.34 แสดง Block Diagram ของระบบการควบคุมแบบปิด

การควบคุมแบบนี้นั้น สามารถจะพบเห็นได้ทั่วไปทั้งในโรงงานอุตสาหกรรมหรือตามบ้านเรือน ได้แก่ การควบคุมอุณหภูมิ ความเร็ว ตำแหน่ง ความดัน การไหลให้คงที่หรืออยู่ในช่วงตำแหน่งที่ต้องการตลอดเวลาแม้สภาพแวดล้อมจะเปลี่ยนไปก็ตาม

2.2.7.6 วิธีการควบคุมในการกำหนดตำแหน่งของหุ่นยนต์



รูปที่ 2.35 แสดง Block Diagram ของการควบคุมตำแหน่งของหุ่นยนต์

เมื่อมีคำสั่งจาก prodram กำหนดให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งที่ต้องการจะทำให้เกิดคำสั่งให้แก่ Control Unit ซึ่งจะทำการควบคุมให้ Power module จ่ายพลังงานให้แก่มอเตอร์เพื่อให้มอเตอร์หมุนไปยังจุดที่ต้องการ เมื่อมอเตอร์เคลื่อนที่จะมีอุปกรณ์ได้แก่ Optical encoder และ tacho generates ซึ่งวัดค่าตำแหน่งและความเร็วของมอเตอร์ เพื่อป้อนกลับให้ Control Unit ทำการเปรียบเทียบค่าของคำสั่งและค่าจริงที่เกิดขึ้นและให้ Control Unit ทำการควบคุมไปยังค่าของคำสั่งที่ต้องการ

2.3 โปรแกรมการเคลื่อนที่ (Motion Programming)

การเคลื่อนที่ของหุ่นยนต์ (Motion) เป็นสิ่งสำคัญอีกเรื่องหนึ่งผู้เขียนโปรแกรมจะต้องควบคุมการเคลื่อนที่ของหุ่นยนต์โดยใช้คำสั่งการเคลื่อนที่ (motions command) ซึ่งคำสั่งเหล่านี้จะแยกจากคำสั่งที่เป็นภาษาคอมพิวเตอร์ทั่วๆไป

ในการเขียนโปรแกรมควบคุมหุ่นยนต์ จะมีรูปแบบของภาษาที่ใช้หลายรูปแบบด้วยกัน แต่ที่นิยมใช้กันอยู่ได้แก่ รูปแบบของภาษา PASCAL และรูปแบบของภาษา BASIC แม้มีรูปแบบการเขียนโปรแกรมจะมีความแตกต่างกันแต่หลักการเขียนโปรแกรมจะไม่แตกต่างกันมากนักดังนั้นหากเราสามารถเขียนโปรแกรมในรูปแบบใดรูปแบบหนึ่งได้แล้ว รูปแบบอื่น ๆ เราก็สามารถจะประยุกต์เขียนได้ โดยอาจจะอ้างอิงจากคู่มือของหุ่นยนต์นั้นๆ

สำหรับคำสั่งที่การเคลื่อนที่นั้นสามารถแบ่งออกเป็น 2 ส่วนใหญ่ ๆ คือคำสั่งสำหรับการเคลื่อนที่แบบ point-to-Point ซึ่งเป็นแบบง่าย ๆ การเคลื่อนที่ไม่สามารถระบุเส้นทางได้ ขึ้นอยู่กับสรีระของหุ่นยนต์ (Robot kinematics) และคำสั่งการเคลื่อนที่ แบบ continuous-path ซึ่ง end effector จะถูกกำหนดทิศทางไว้แน่นอน เช่น การเคลื่อนที่ในแบบ Linear หรือ arc (ส่วนของวงกลม)

สำหรับตำแหน่งต่าง ๆ นั้นเราสามารถระบุค่าโดยตรงหรือโดยการเคลื่อนที่หุ่นยนต์ไปยังตำแหน่งนั้นเสร็จแล้วก็เก็บตำแหน่งไว้ (teaching) ระบบแกนต่าง ๆ นั้นจะขึ้นอยู่กับ coordinate system สรุปได้ดังนี้

การเคลื่อนที่ Robot tool ไปยังเป้าหมายภายใต้การควบคุมโดยโปรแกรม Robot จะต้องถูกโปรแกรมคำสั่งการเคลื่อนที่โดยคำสั่งนี้จะประกอบด้วยจุดเป้าหมายและค่า parameter อื่นๆขึ้นอยู่กับชนิดการเคลื่อนที่ได้

1. Point-to- point (PTP) การเคลื่อนที่แบบนี้ tool จะเคลื่อนที่ไปตามเส้นทางที่เร็วที่สุดหรือสะดวกที่สุด
2. Linear (LIN) การเคลื่อนที่แบบนี้ tool จะเคลื่อนที่ไปตามเส้นตรง
3. Circular (CIRC) การเคลื่อนที่แบบนี้ tool จะเคลื่อนที่ไปตามส่วนโค้งของวงกลม

การระบุตำแหน่งการเคลื่อนที่แก่หุ่นยนต์สามารถทำได้ 2 วิธี คือ

1. การระบุตำแหน่งโดยใช้ Teach – pendant สามารถทำได้โดยเคลื่อนที่ปลาย Tool ไปยังตำแหน่งที่ต้องการและให้ Controller เก็บตำแหน่งนั้น ๆ ไว้ นอกจากนั้นค่อยเรียกใช้ตอนเขียนโปรแกรม

2. การระบุตำแหน่งโดยการระบุ parameter ต่าง ๆ ของการเคลื่อนที่เช่น ค่า X ,Y , Z , A ,B ,C และค่าอื่น ๆ ที่จำเป็น

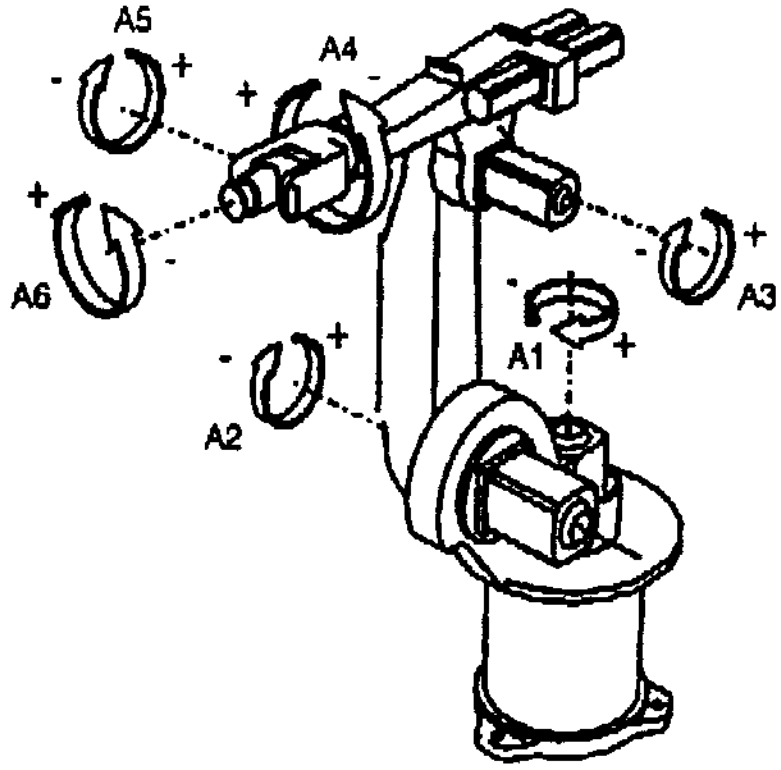
2.3.1 ระบบพิกัดของหุ่นยนต์ (Coordinate system)

2.3.1.1 Joint Coordinate system

คือ การเคลื่อนที่ของแกนแต่ละแกนอย่างอิสระไม่ว่าเป็นการ translation หรือจะเป็นการ rotation

ในระบบ Joint Coordinate System นี้ แต่ละแกนของหุ่นยนต์จะถูกกำหนดโดย Coordinate System ของมันเอง เราสามารถเคลื่อนแต่ละแกนได้อิสระและรวดเร็วโดยมีทิศทาง + และ – เมื่อกด Manual Traversing key ที่ Teach Pendant

ข้อเสียของ Joint Coordinate System คือยากในการเคลื่อนที่ End Effector ไปยังจุดและทิศทางการวางตัว (Orientation) ที่ต้องการเนื่องจากเราควบคุมการเคลื่อนที่ ของหุ่นยนต์ได้จากแกนเพียงแกนเดียวซึ่งมิใช้เป็นการควบคุม การเคลื่อนที่ของ End Effector



รูปที่ 2.36 แสดงการเคลื่อนที่ของหุ่นยนต์ในทิศทาง + และ – ในระบบพิกัดแบบ Joint

2.3.1.2 World Coordinate system

ใน world coordinate system จะมีการอ้างตำแหน่งแบบ Cartesian นั่นคือเป็น Coordinate แบบ X,Y,Z นอกจากนี้ยังมี orientation รอบแกนต่าง ๆ

A คือ rotation รอบแกน Z

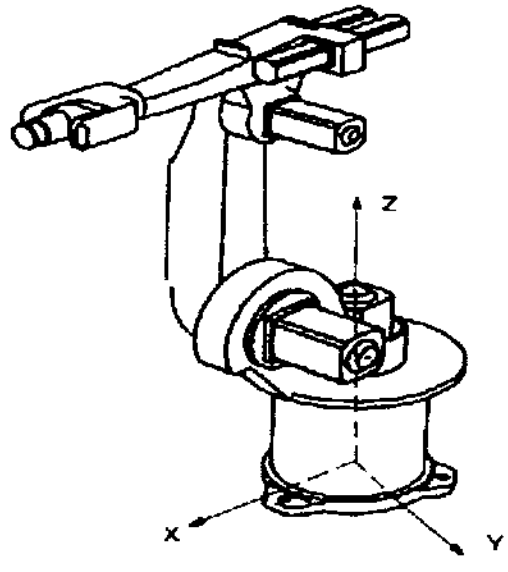
B คือ rotation รอบแกน Y

C คือ rotation รอบแกน X

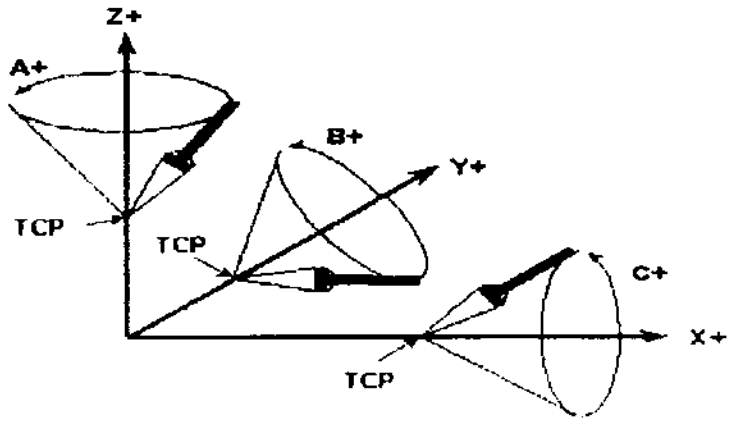
Reference ใน World Coordinate system จะถูก fixed เราไม่สามารถทำการเปลี่ยนแปลงได้ reference ใน Word Coordinate นี้จะเป็น reference ให้ทั้ง robot Cell และ Peripheral equipment Cell

มีการอ้างอิงพิกัด แบบคาร์ทีเซียนใน work cell โดยมีจุดกำเนิดของพิกัดจะอยู่ที่ฐานของหุ่นยนต์ โดยทิศทางแกน X จะพุ่งไปข้างหน้า และแกน Z จะมีทิศทางพุ่งทิศทางพุ่งขึ้นด้านบน ส่วนทิศทาง Y สามารถหาได้โดยกฎมือขวา

ลักษณะการเคลื่อนที่ของหุ่นยนต์ใน World Coordinate System จะเคลื่อนที่ช้าและมีขีดจำกัด ของแกนในการเคลื่อนที่โดยการเคลื่อนที่ของ End Effector จะมีทิศทางขนานกับแกนคาร์ทีเซียน XYZ และสามารถควบคุมทิศทางการวางตัวของ End Effector



รูปที่ 2.37 แสดงจุดกำเนิดและทิศทางการวางตัวของแกนคาร์ทีเซียน XYZ ในระบบพิกัดแบบ World



รูปที่ 2.38 แสดงการควบคุมทิศทางการวางตัวของ End Effector ตามแกนของคาร์ทีเซียน XYZ A , B และ C คือการหมุนหรือ Orientation รอบแกน Z , Y และY ตามลำดับ

2.3.1.3 Robot Coordinate system

การอ้างตำแหน่งเป็นแบบ Cartesian จุดอ้างอิง (reference) ของระบบนี้จะอยู่ที่ฐานของหุ่นยนต์และจะใช้กับส่วนที่เป็น mechanical Coordinate นี้จะมาจาก World Coordinate ซึ่งถูกระบุมาตั้งแต่ผู้ผลิตค่า offset ที่สัมพันธ์กับ World สามารถกำหนดได้โดยใช้ $ROBROOT

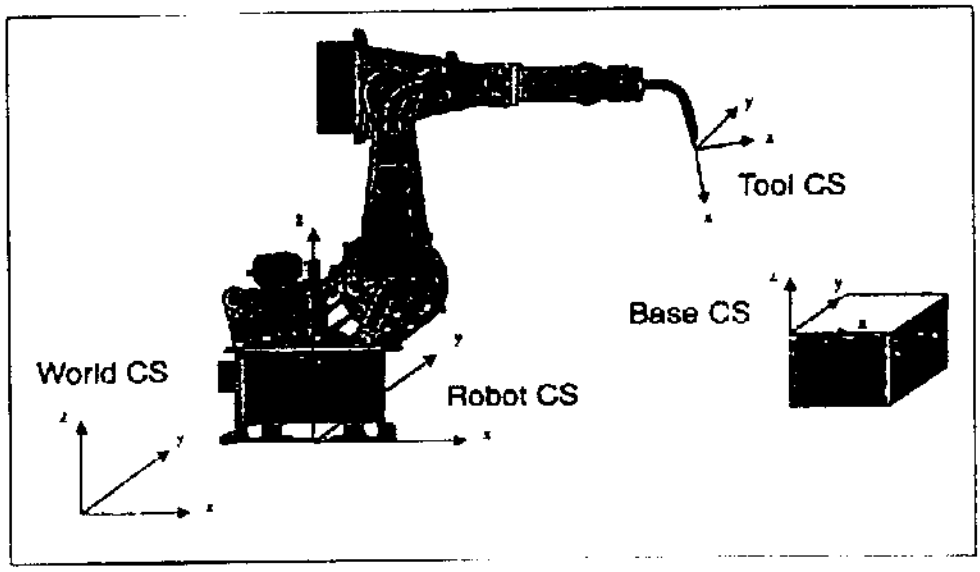
2.3.1.4 Tool Coordinate System

มี origin อยู่ปลายของ Tool (Tool center point : TCP) ถ้า TCP เปลี่ยนแปลง tool Coordinate ก็จะเปลี่ยนแปลงด้วยในหุ่นยนต์ที่พึ่งซื้อมาจะมี TCP อยู่ที่ flange แต่เราสามารถเปลี่ยนได้หลังการทำ Tool Center Point

2.3.1.5 Base Coordinate System ใช้เป็น reference ให้กับชิ้นงาน ตอนซื้อหุ่นยนต์มาใหม่ ๆ $BASE = $World

ตารางที่ 2.7 แสดง Cartesian Coordinated แบบต่าง ๆ

Coordinate system	**System Variable**	**Status**
World coordinate system	$WORLD	Write-protected
Robot coordinate system	$ROBROOT	Write-protected (can be changed in $MASCHINE.DAT
Tool coordinate system	$TOOL*	Writable
Base (work piece) coordinate system	$BASE*	Writable
*In the case of gripper-related interpolation, $TOOL and $BASE are switched over (See below)		

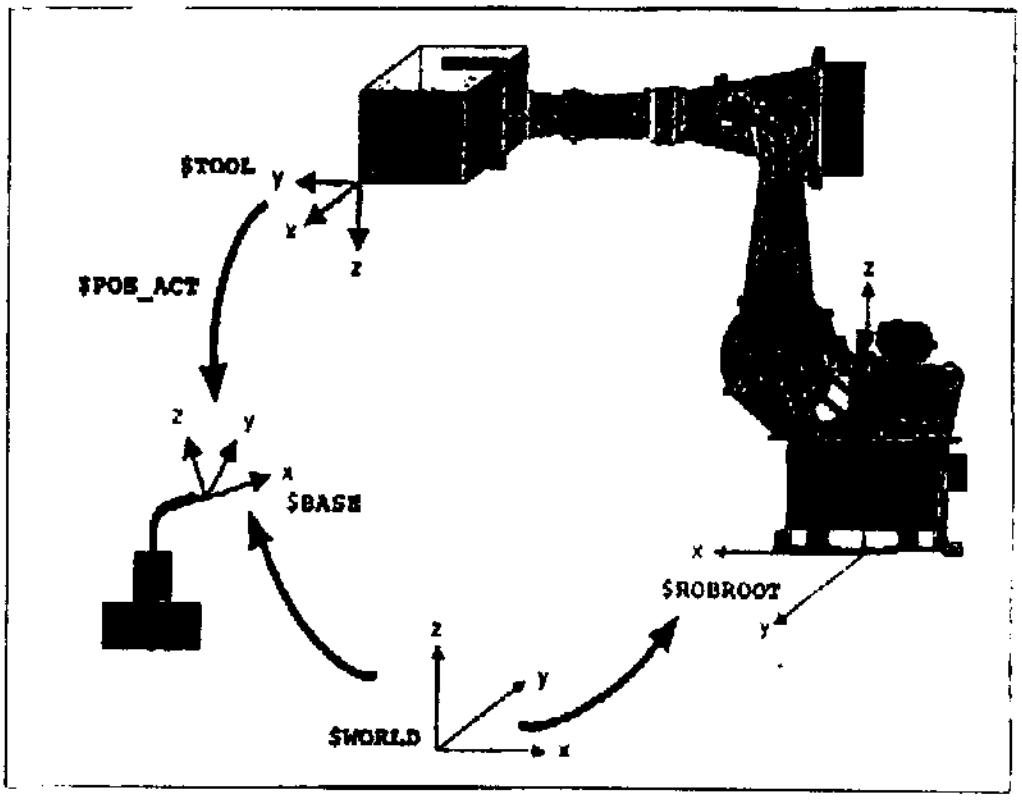


รูปที่ 2.39 Cartesian Coordinate System สำหรับหุ่นยนต์

การเคลื่อนที่ใน Coordinate ต่าง ๆเส้นทางการเคลื่อนที่ (Path) จะถูกคำนวณให้สัมพันธ์กับ Coordinate นั้น ๆ ซึ่งชนิดของการเคลื่อนที่ (interpolation) สามารถระบุได้โดย ตัวแปรระบบ $IP_MODE เช่น

$IP_MODE = # TCP

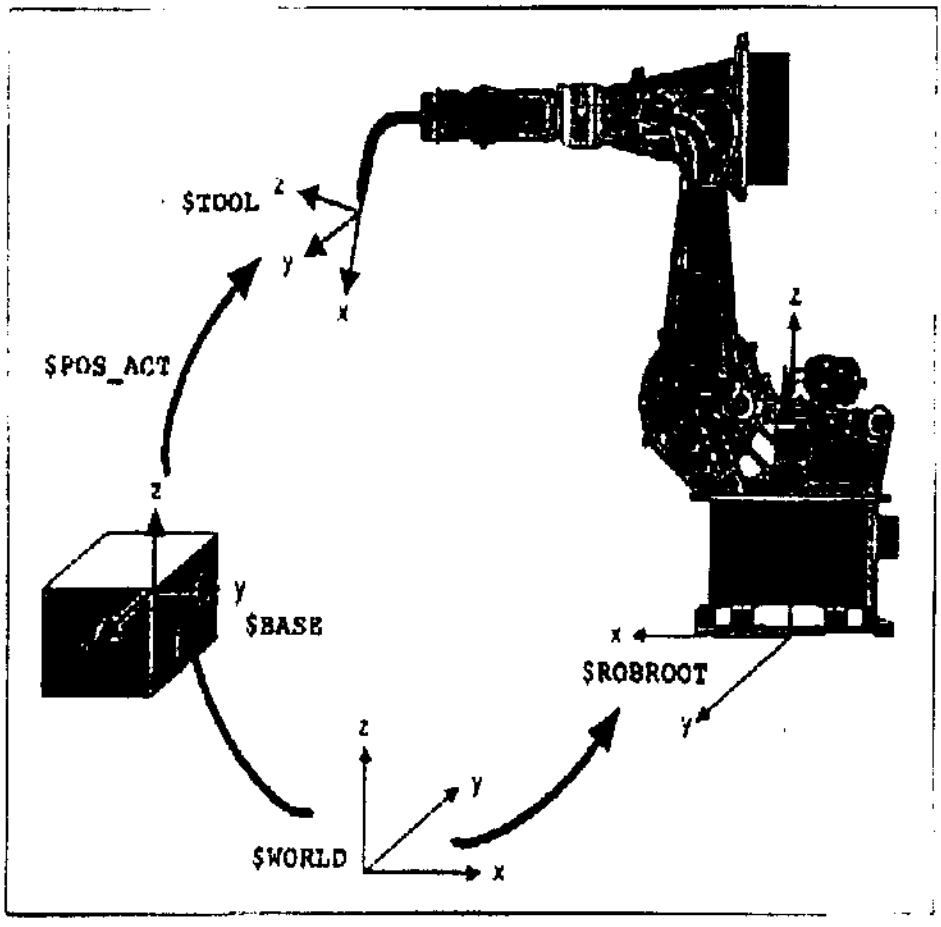
คือ การเคลื่อนที่ในระบบ Tool Coordinate system ค่าตำแหน่งปัจจุบันจะมีตัวแปรระบบที่เป็นตัวชี้ คือ $POS_ACT แสดงดังรูปที่ 2.53



รูปที่ 2.40 แสดงการเคลื่อนที่ใน Tool Coordinate system

$IP_MODE = #BASE

เป็นการ เคลื่อนที่ ในระบบ $Base Coordinate ซึ่งปกติก็จะ set เป็น default ขณะ Controller เริ่มทำงานแสดงดังรูปที่ 2.54



รูปที่ 2.41 แสดงการเคลื่อนที่ใน $BASE Coordinate system

ใน Base Coordinate System ($BASE) จะถูกกำหนดมาจาก World Coordinate System ($WORLD) ปกติจะถูกกำหนดไว้ที่ฐานของหุ่นยนต์ ($ROBROOT) สำหรับ Tool Coordinate System ($TOOL) จะถูกกำหนดเป็น Null frame ($NULLFRAME = {FRAME: X 0,Y 0, Z 0, A 0 ,B 0, C 0})หมายความว่าทั้งหมดนี้จะขึ้นอยู่กับ tool center point ต่อนที่ซื้อหุ่นยนต์มาใหม่ ๆ TCP จะอยู่ที่ flange แต่ถ้าหากมีการติดตั้ง Tool ค่าต่าง ๆ จะต้อง set มาเป็นที่ปลาย tool โดยการทำ Tool Center Point

2.3.2 P-T-P motion

ความเร็วและความเร่ง

การเคลื่อนที่แบบ P-T-P ปลาย Tool จะเคลื่อนที่โดยไม่สามารถระบุเส้นทางการเคลื่อนที่ (Path) ได้แต่ปลาย Tool จะเลือกเส้นทางที่สามารถเคลื่อนที่ไปสู่จุดหมายได้เร็วที่สุด SYSTEM Variable ที่เกี่ยวข้องกับ PTP คือ

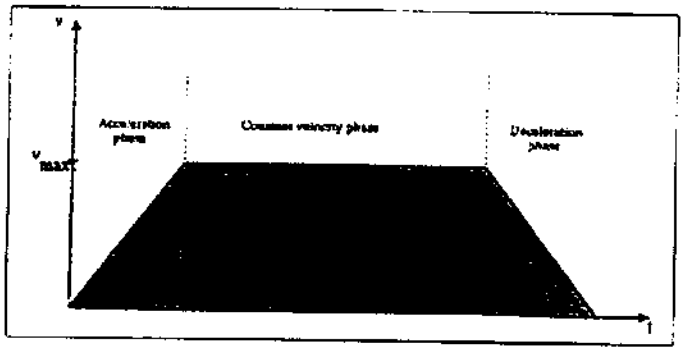
$ VEL_AXIS [axis number] : คือ ความเร็วสูงสุดในการโปรแกรมของแต่ละแกน

$ ACC_AXIS [axis number] : คือ ความเร่งสูงสุดในการโปรแกรม

ทั้งสองค่าจะกำหนดเป็นเปอร์เซ็นต์ของ machine data ถ้าขณะทำการโปรแกรมเราไม่ได้กำหนดความเร็วความเร่งให้กับทุก ๆ แกน (axis) ของหุ่นยนต์จะทำให้เกิด error ขึ้น

การเคลื่อนที่แบบ Synchronous P-T-P

คือ การเคลื่อนที่ของแกนทุกแกนเริ่มเคลื่อนที่และหยุดพร้อมกันนั่น คือ จะมีแกนใดแกนหนึ่งจะเคลื่อนที่ในระยะทางที่ยาวที่สุดเพียงแกนเดียวซึ่งเรียกว่าแกนนำ (Leading axis) ส่วนแกนอื่น ๆ ที่เหลือจะเคลื่อนที่ด้วยความเร็วกว่าทำให้สามารถเคลื่อนที่ถึงจุดหมายในเวลาเดียวกันโดยไม่คำนึงความเร็วและความเร่งใน $ VEL_AXIS [no], ACC_AXIS [no]

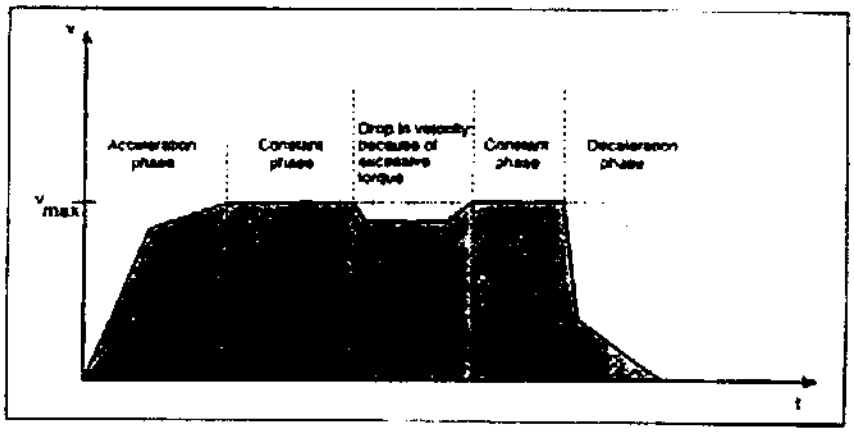


รูปที่ 2.42 Velocity profile สำหรับการเคลื่อนที่แบบ Synchronous P-T-P

ข้อเสีย ของการเคลื่อนที่แบบนี้ คือ คุณสมบัติของ motor ไม่ได้ใช้เต็มประสิทธิภาพ (non optimally) ทุกตัวจะใช้เต็มประสิทธิภาพเพียงตัวใดตัวหนึ่ง

Higher motion profile

Higher motion profile นี้จะใช้กับการเคลื่อนที่แบบ PTP กรณีนี้จะทำ time-optimized motion จากจุดเริ่มไปยังจุดปลายด้วยแต่ละคำสั่ง PTP นั่นคือ มันจะไม่เคลื่อนที่ไปยังปลายทางแบบเร็วที่สุดเพียงอย่างเดียวแต่จะมีการทำ torque optimally ทุก ๆ จุดบนเส้นทางการเคลื่อนที่ เช่น ถ้า torque เกินก็จะทำการลดความเร็วลง (แสดงดังรูปที่ 2.58)



รูปที่ 2.43 แสดงความเร็วของ higher motion profile

2.3.3 Continuous - path motions

ความเร็วและความเร่ง

Continuous path motions นั้นจะมีการเคลื่อนที่ไปตามเส้นทางที่ได้ระบุไว้แน่นอนแล้ว เช่น การเคลื่อนที่แบบ Linear (LIN) หรือ การเคลื่อนที่แบบ Circular (CIRC)

ความเร็ว และอัตราเร่งที่ระบุให้แต่ละแกน (Axis) นั้นไม่ได้สัมพันธ์กับการเคลื่อนที่แบบนี้แต่จะมีการกำหนดความเร็วและอัตราเร่งให้กับ TCP การกำหนดแสดงได้ดังตารางที่ 2.7

ตาราง 2.8 แสดง System Variable สำหรับความเร็วและอัตราเร่งของ CP

	Variable name	Data type	Unit	Function
Velocities	$VEL.CP	REAL	m/s	Path velocity
	$VEL.ORI1	REAL	°/s	Swivel velocity
	$VEL.ORI2	REAL	°/s	Rotational velocity
Accelerations	$ACC.CP	REAL	m/s^2	Path acceleration
	$ACC.ORI1	REAL	$°/s^2$	Swivel acceleration
	$ACC.ORI1	REAL	$°/s^2$	Rotational acceleration

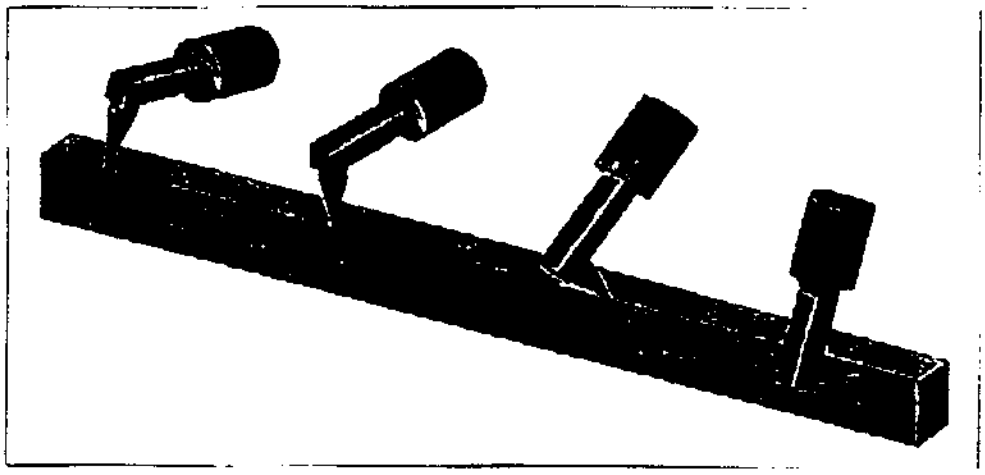
เมื่อตอนเริ่มต้นโปรแกรม Basic package (file BAS.SRC) จะถูกเรียกขึ้นมาเพื่อกำหนดความเร็วและอัตราเร่งของ CP ซึ่งจะ set ที่ค่าสูงสุดที่ได้ระบุใน machine data หรือใน $CONFIG.DAT

Orientation Control

Orientation สามารถset ได้โดยตัวแปรระบบ (system variable) "$ORI_TYPE"

$ORI_TYPE = #CONST ระหว่างการเคลื่อนที่ orientation จะคงที่ถ้าใน program ไม่ได้ set ไว้จะถูก set ให้โดย basic package (BAS) ในขณะ initialization

$ORI_TYPE =# VAR ระหว่างการเคลื่อนที่ orientation จะเปลี่ยนอย่างต่อเนื่องจากจุดเริ่มต้นไปยังจุดปลายทาง



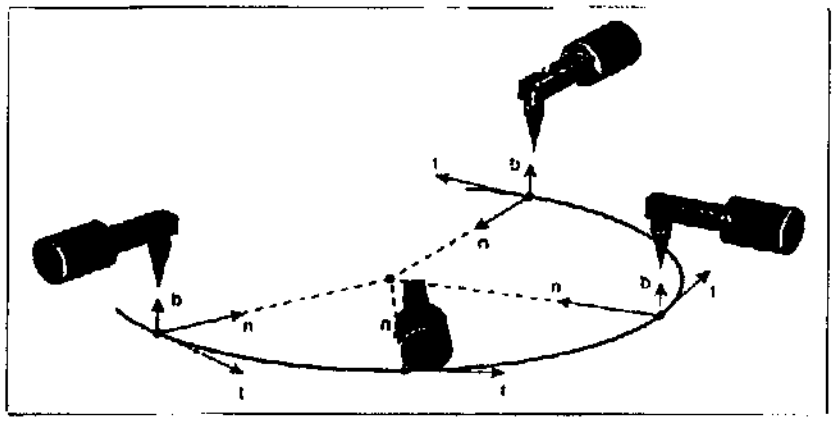
รูปที่ 2.44 แสดงการเปลี่ยนแปลงของ Orientation ใน linear motion ($ORI_TYPE=#VAR)

$ CIRC_TYPE = # จะเคลื่อนที่โดยอ้างอิงทิศทางกับระบบ Base coordinate (Space-related orientation control) ระหว่างการเคลื่อนที่แบบ circular ค่านี้จะถูก set ในขณะ initialization โดย BAS (# INITMOV,O)

$ CIRC_TYPE = # Path จะเคลื่อนที่ตั้งฉากกับเส้นทางการเคลื่อนที่ (Path-related orientation control ระหว่างการเคลื่อนที่แบบ circular

Constant+Path-related ($ORT_TYPE =# CONST, $CIRC_TYPE =# PATH)

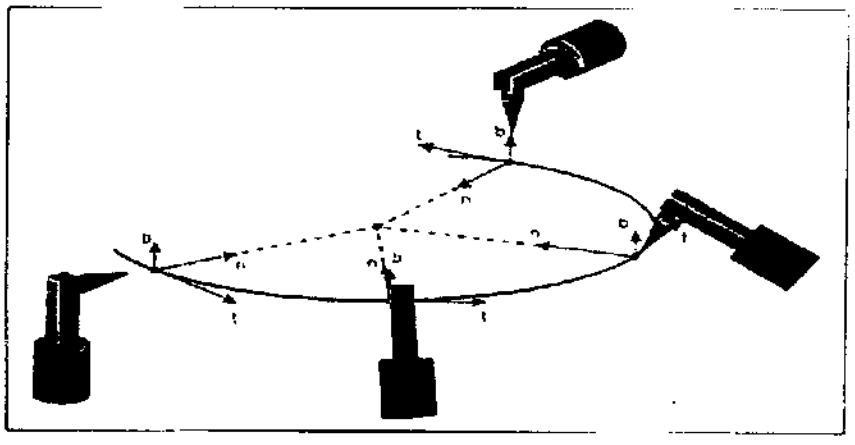
การเคลื่อนที่ของ Tool จะสัมพันธ์กับระนาบและเส้นสัมผัสของวงกลมแสดงดังรูปที่ 2.45 สำหรับการเคลื่อนที่แบบนี้บางทีเรียกว่า Tool-Base moving frame ซึ่งตำแหน่งของ tool จะเคลื่อนที่ไปโดยไม่เปลี่ยนแปลงการ Orientation ลักษณะเช่นนี้จะใช้ในงาน arc welding



รูปที่ 2.45 Constant-path related Orientation Control

Variable + path - related ($SORT_TYPE = # VAR, $CIRC_TYPE = # PATH)

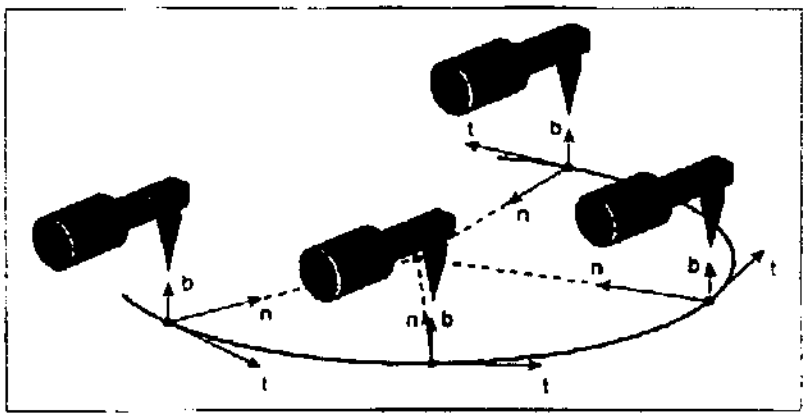
การเคลื่อนที่แบบ tool – based moving frame จะไม่เปลี่ยนแปลง orientation ระหว่างเคลื่อนที่ไป แต่ถ้ามีการเปลี่ยนแปลง Orientation ระหว่างจุดเริ่มต้นและปลายทาง ($ORI_TYPE = # VAR) จะทำให้เกิดการหมุนซึ่งแสดงดังรูป 2.46



รูปที่ 2.46 Variable path-related Orientation Control

Constant + space – related ($ORT_TYPE = # CONST, $CIRC_TYPE =# BASE)

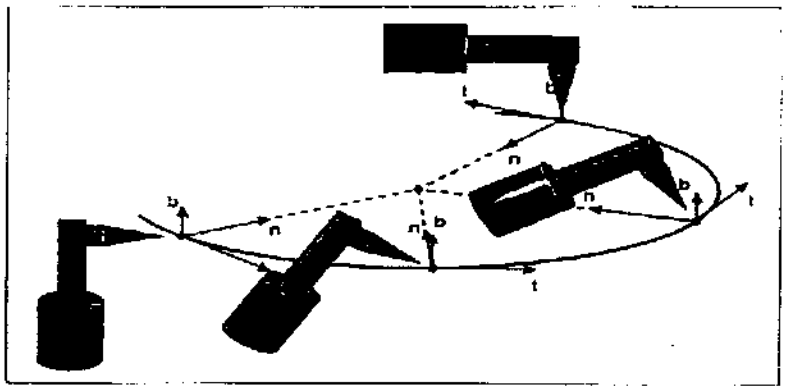
การ Orientation จะถูกควบคุมให้สัมพันธ์กับ base system ($BASE) Space – related Orientation จะมีประโยชน์ในงานที่เป็น path motion เช่นการไกด์ TCP ไปตามเส้นทางวงกลมเป็นกรณีพิเศษที่มีการเปลี่ยนแปลงการ Orientation ระหว่างเริ่มต้นและจุดปลายทางเพียงเล็กน้อยแสดงดังรูป 2.47 เช่น
งานทากาว



รูปที่ 2.47 Constant space-related Orientation Control

Variable + space – related ($ORI_TYPE = # VAR, $ CIRC_TYPE = # BASE)

มีการเปลี่ยนแปลง Orientation ($ORI_TYPE # VAR) และมีการหมุนระหว่างการเคลื่อนที่จากเริ่มต้นไปยังจุดปลายทางโดยสัมพันธ์กับ base Coordinate System แสดงดังรูป 2.48



รูปที่ 2.48 Variable space-related Orientation Control

สำหรับค่า default setting ของ Variable แสดงดังตารางที่ 2.9

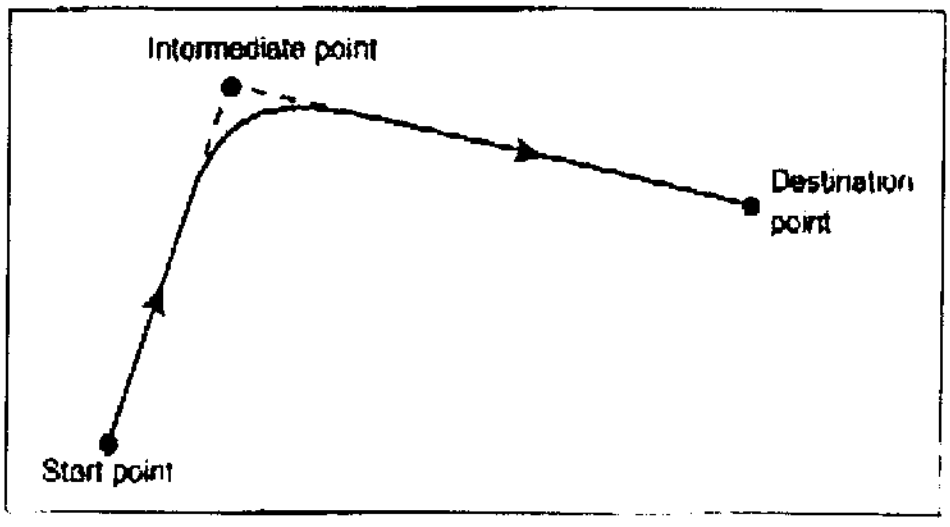
ตารางที่ 2.9 Default Setting of $ORI_TYPE and $ CIRC_TYPE

	In the system	By BAS (#INTIMOV.0)
$ORI_TYPE	# VAR	
$ CIRC_TYPE	# PATH	# BASE

2.3.4 Motion with approximate positioning

ในการเคลื่อนที่ของหุ่นยนต์จะต้องหลีกเลี่ยงการชนกับสิ่งกีดขวางต่าง ๆ ซึ่งสามารถทำได้หลายวิธีและนี่ก็ คือ วิธีหนึ่งที่สามารถช่วยได้

การเคลื่อนที่ไปในบางตำแหน่งไม่มีความจำเป็นที่จะต้องเข้าไปยังจุดนั้น ๆ โดยตรงและอย่างถูกต้องแม่นยำแต่สามารถผ่านเข้าไปในระยะที่ยอมรับได้ก็พอ ซึ่งจะทำให้การเคลื่อนที่ของ Robot มีความรวดเร็วยิ่งขึ้นการ motion with approximate แสดงดังรูป 2.49



รูปที่ 2.49 Approximate positioning ที่จุดกึ่งกลาง

การเคลื่อนที่แบบ PTP Approximate positioning Controller จะคำนวณระยะทางของแต่ละแกน (Axis) ที่จะเคลื่อนที่ไปในย่านตำแหน่ง approximate และจะระบุความเร็วของแต่ละ axis เพื่อให้เคลื่อนที่ไปตามเส้นทางได้

การ Approximate เริ่มต้นเมื่อแกนที่เป็นแกนนำ (Leading) เข้าภายในมุมที่กำหนดให้มีการ Approximate ซึ่งมุมของแต่ละแกนจะถูกกำหนดไว้ก่อนแล้ว คือ

$ APO_DIS_PTP[1] = 90

to

$ APO_DIS_PTP [6] = 90

ในโปรแกรม $APO_CPTP ที่จะให้มีการ Approximate จะถูกระบุเป็นเปอร์เซ็นต์ของมุม maximum เช่น

$ APO.CPTP[1] = 50

คือ การ Approximate จะเริ่มขึ้นเมื่อแกนแรกครอบคลุมตำแหน่งที่จะ Approximate เป็นมุม 45 องศา (50% ของ 90 องศา)

C_PTP

การสั่ง approximate position ใน PTP จะใช้คำสั่ง C_PTP เช่น

PTP POINI4 C_PTP

การ Approximate ในการเคลื่อนที่แบบ PTP จะไปในเส้นทางที่ใช้เวลาน้อยที่สุด

ตัวอย่าง

```
DEL  UEBERPTP()
;.....Declaration section.....
EXT BAS (BAS_COMMAND : IN, REAL : IN)
DECL AXIS  HOME
;.....Initialization of velocities,
                ; accelerations, $BASE, $TOOL, etc.
HOME  = { AXIS : A1 0 , A2-90,A3 90 ,A4 0,A5 0, A6  0}
;..... Main section.....
PTP HOME ; BCO  run
PTP { POS : X  1246.93 , Y –98.86, Z 715, A  125.1,B 56.75 ,C  162.65,S  2,T  10 }
; Approximate positioning  of the point
PTP  { POS: X  1246.93,Y-98.86,Z 715,A  125.1,B 56.75,C  111.66,S 2 , T  10} C_PTP
PTP { POS  : X  1109.41,Y –0.51, Z  715,A  95.44,B 73.45, C  70.29,S 2 , Y  10}
; Approximate  positioning  of two  points
$APO. CPTP = 20
PTP { POS : X  1296.61 , Y 133.41, Z 715,A 114.65, B 50.46, C 84.62, ,S 2, T 11}
C_PTP
PTP {POS : X  988.45 ,Y 238.53, Z 715, A  114.65,B 50.46,  C 84.62, S 5, T 11}
C_PTP
PTP {POS : X 12.9.5, Y  381.09 ,Z 715,A –141.91, B 82.41, C –159.41, S 2 , T 11}
PTP  HOME
END
```

2.3.4.2 LIN-LIN Approximationing

ในการเคลื่อนที่ที่ต้องการให้เป็นไปแบบต่อเนื่องจำเป็นใช้การ Approximate ระหว่าง แต่ละ linear motion ตัว controller จะคำนวณเส้นทางสำหรับผ่านของการทำ Approximate ซึ่งจะเป็นไปโดยให้ความเร่งมีความเหมาะสมดีที่สุด

การเริ่มทำ Approximate positioning.

มี 3 ตัวแปรที่ได้ประกาศไว้สำหรับการทำ Approximate แสดงดังตารางที่ 2.9

ตารางที่ 2.10 System variable ที่ใช้ในการเริ่มต้นทำ Approximate

Variable	Data type	Unit	Meaning	Keyword in the commmand
SAPO.CDIS	REAL	mm.	Transnatioal distance criterion	C_DIS
SAPO.COR[	REAL	0	Orientatoion distance	C_ORI
SAPO.CVEL	REAL	%	Velocity criterion	C_VEL

Distance criterion

ระยะทางที่เปลี่ยนย้ายไม่สามารถกำหนดได้ในตัวแปร $APO.CORI ถ้าการ Approximate ถูก trigger ด้วยตัวแปรนี้ Controller จะสั่งให้หุ่นยนต์ออกจากจุดปลายทางจริงเมื่อเคลื่อนที่มาในย่านระยะทางที่กำหนดในตัวแปร $APO.CDIS

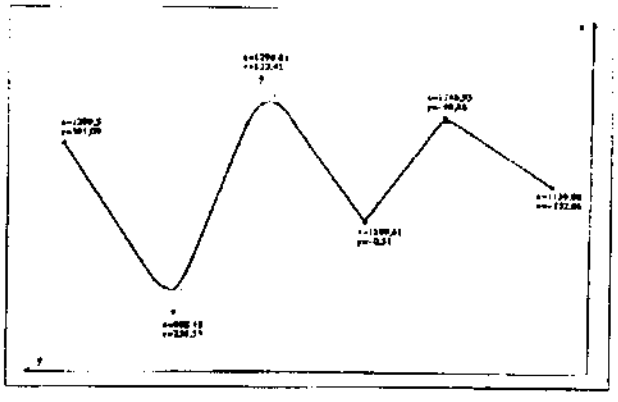
Orientation Criterion

ระยะของการหมุนสามารถกำหนดไว้ได้ในตัวแปร $APO.CORI ในกรณีนี้หุ่นยนต์จะออกจากจุดปลายทางจริงเมื่อมีการหมุนเข้าไปในระยะมุมที่กำหนดใน $ APO.CORI

Velocity Criterion

ค่าความเร็วที่กำหนดใน $APO.CVEL จะเป็นเปอร์เซ็นต์ของค่าความเร็วใน program ($VEL.CP) กระบวนการ Approximation จะเริ่มขึ้นเมื่อแต่ละ Component เคลื่อนที่จนความเร็วถึงช่วงที่กำหนด

```
DEF  UEBERLIN ( )
;.....Declaration section.....
EXT BAS (BAS  COMMAND : IN,REAL : IN)
DECL AXIS HOME
;..... Initialization .....
BAS  (#INTIMOV, 0); Initialization of velocities,
                    ; Accelerations, $BASE, $TOOL, etc.
HOME  =  {AXIS : A1 0,A2 –90 ,A4 0,A5 0 , A6 0 }
;..... Main section.....
PTP { POS : X 1159.08, Y –232.06 , Z 716.38, A 171.85 , B 67.32, C 162.65, S 2 ,T 10 }
; Approximate positioning of the point  using distance criterion $APO. CDIS = 20
LIN { X  124.93, Y –98.86, Z 715, A 125.1, B56.75, C 111.66}  C_DIS
LIN  {  X  1109.41, Y –0.51, Z 715, A 95.44, B 73.45, C 70.95}
; Approximate positioning  of two  points
LIN  {X 1296.61, Y 133.41,  Z 714.99, A 150.32, B 55.07, C 130.23} C_ORI
LIN  { X 988.45, Y 238.53, Z 714.99, A 114.65, B 50.46, C 84.62}  C_VEL
LIN  { X 1209.5, Y 381.09, Z 715, A –141.91, B 82.41, C-159.41}
PTP  HOME
END
```



รูปที่ 2.50 Example of LIN-LIN approximate positioning

สำหรับการกระตุ้นการ Approximate สามารถทำได้โดยการใช้คำสั่ง C_DIS, C_ORI, C_VEL ใน LIN หรือ LIN_REL

2.3.4.3 CIRC-CIRC และ CIRC-LIN Approximate

สำหรับการ Approximate ระหว่าง CIRC block กับ LIN หรือ CIRC นั้นคล้ายกับการ Approximat ระหว่าง LIN ไม่ว่าจะเป็นการหมุนหรือการเคลื่อนย้ายตำแหน่งที่จะทำให้การเคลื่อนที่ต่อเนื่องและราบเรียบขึ้นการเริ่ม Approximate ก็โดยการกำหนดค่าในตัวแปรกระตุ้น Approximate ก็ใช้คำสั่ง C_DIS, C_ORI, C_VEL เหมือนใน LIN เหมือนใน LIN

ตัวอย่าง

```
DEF UEBERCIR ( ); .....Declaration section.....
EXT BAS (BAS COMMAND : IN, REAL : IN)
DECL AXIS HOME; ..... Initialization of velocities,
BAS (# INITMOV, 0) ; Initialization of velocities,
                    ; Accelerations, $BASE, $TOOL, etc.
HOME = {AXIS : A1 0,A2 -90, , A3 90,A4 0,A5 0,A6 0} ;.....Main
Section.....
PTP HOME; BCO run
{PTP POS : X 980, Y-238 , Z718, A 133, B 66, C 146, S 6, T 50 }
;Space-related constant orientation control
; Approximate positioning using distance criterion
$ APO.CDIS = 20
CIRC {X925, Y-285,Z718} , { X 867 , Y –192, Z 718, A 155, B 75, C 160 }
C_DIS
; Space-related constant orientation control
; Destination point defined by angle specification
; Approx. pos. not possible because of adv. Run stop due to
$OUT$ORI_TYPE=#CONST
CIRC {X 982, Y –221, Z 718, A 50 ,B 60,C 0} , {X1061,Y-118, Z 718, A-162, B60, C177} ,
CA
```

```
150 C_ORI
$OUT [3]  = TRUE
; Path –related constant orientation control
; Approximate positioning using orientation criterion
$ORI_TYPE =# VAR
$CIRC_TYPE=# PATH
CIRC {X 963.08, Y –85.39, Z 718} , { X892.05, Y67.25,Z7180.01,A97.4,B57.07, C151.11}
C_ORI
;Relative circular motion
; Approximate positioning  using velocity  criterion
$ APO. CVEL = 50
CIRC_REL  {X-50, Y 50} , { X0,Y 100}  C_VEL
; Approximate positioning using velocity criterion
$APO.CDIS = 40
CIRC_REL {X-50, Y50} , { X0 , Y 100} C_DIS
CIRC_REL X-50 Y50, X0,Y100
PTP HOME
END
```

2.3.4.4 PTP-Continuos Path Approximate Positioning

สำหรับการ Approximate positioning ระหว่าง PTP และ Cartesian CP (LIN หรือ CIRC) นั้น น่าสนใจสำหรับการประยุกต์ใช้ในงาน handling หรือ งาน assembly และงานจำพวก joining palletizing, Clamping ซึ่งจะต้องการเคลื่อนที่ที่สั้น

เริ่ม PTP –CP approximate positioning ด้วยการกำหนด PTP Criterion ใน $APO.CPTP (เหมือนใน PTP-PTP) และกำหนด C_DIS, C_ ORI, C_VEL ในส่วนของ CP (LIN หรือ CIRC)

ถ้าไม่มีการระบุการ approximate ใน PTP block จะต้องระบุใน CP block และ C_DIS จะ default ค่าให้กับ CP block ด้วย

CP-PTP approximate positioning

ค่าการ approximate positioning จะเป็นค่าสำหรับ CP block ขณะที่ System เปลี่ยนไปเป็น $APO. CPTP สำหรับ PTP block ลำดับคำสั่งสามารถเขียนได้ดังตัวอย่างนี้

```
CIRC INTERM_POINT1  C_VEL
PTP  POINT2.
```

ตัวอย่าง

```
DEF  UEBERB_R ( )
;.....Declaration section.....
EXT BAS (BAS_COMMAND IN, REAL: IN)
DECL AXIS  HOME
;.....Initialization .....
BAS (#INITMOV, 0) ; Initialization of velocities,
; Accelerations, $BASE, $ TOOL, etc.
HOME = { AXIS : A 10, A2-90, A 4 0 , A 5 0 , A6 0}
;.....Main section.....
PTP HOME; BCO  run
PTP  { POS : X 1281.55, Y –250.02 , Z716, A79.11, B 68.13, C79.73, ,S 6, T 50}
PTP {POS : X 1209.74, Y-153.44,Z 716, A 79.11, B 68.13 , C 79.73, S 6, T 50} C_PTP C_ORI
LIN  {X 1037.81, Y-117.83, Z716, A 79.11, B 68.13, C 79.73}
$APO. CDIS = 25
LIN  {X 1183.15, Y-152.64, Z 716, A 79.11, B 68.13, C 79.73 } C_DIS
```

PTP HOME

END

2.3.5 การเคลื่อนที่แบบ Absolute และ Relative

เมื่อหุ่นยนต์ถูกเคลื่อนที่ไปยังตำแหน่งต่าง ๆใน แต่ละ AXIS ดังนี้

AXIS1 (A1) = 0^0

AXIS2 (A2) = -90^0

AXIS3 (A3) = -90^0

AXIS4 (A4) = 0^0

AXIS5 (A5) = 0^0

AXIS6 (A6) = 0^0

ตำแหน่งข้างบนนี้นี้คือ Mechanical zero position ซึ่งจะถูก set ไว้ขณะทำการ Master (Calibrate) การสั่งให้หุ่นยนต์เคลื่อนที่สามารถสั่งให้เคลื่อนที่ทีละ Component ได้โดยที่ส่วนอื่น ๆ ไม่เปลี่ยนแปลง เช่น

PTP { A3 45 }

เป็นการเคลื่อนที่เฉพาะแกนที่ 3 (AXIS 3) ไป 45^0

2.3.5.1 การเคลื่อนที่แบบ Absolute

การเคลื่อนที่แบบ Absolute จะอ้างอิง (reference) กับจุดเริ่มต้นของระบบ Coordinate นั้น ๆ เช่น ถ้าให้ world Coordinate จะมีการอ้างอิงจุดเริ่มต้นจากฐานของหุ่นยนต์ไปยังตำแหน่งที่ระบุ เช่น

PTP { X 100 , Y –800}

หรือในระบบ Joint Coordinate System เช่น

PTP { A1 45 } ; rotate Axis 1 45^0

สำหรับคำสั่งของการเคลื่อนที่แบบ linear เช่น

LIN {X 100, Y 55}

เป็นการสั่งให้เคลื่อนที่แบบ Linear ไปยังตำแหน่ง X = 100, Y = 55 อ้างอิงกับจุดเริ่มต้นของ Coordinat System.

คำสั่งการเคลื่อนที่แบบ Circular เช่น

CIRC { X 876, , Y 234}, { X 590, Y 550} , CA 30

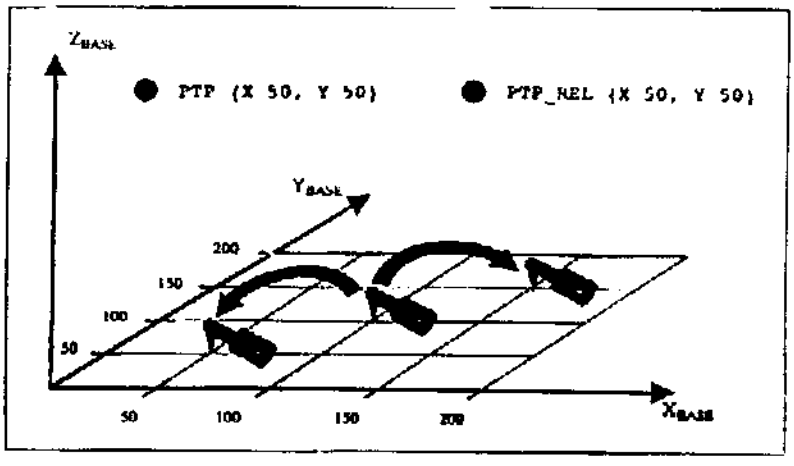
เป็นการสั่งให้เคลื่อนที่แบบ Circular โดยมีจุด midpoint ที่ X = 876, Y = 234 และจุด End point ที่ X = 590, y = 550 โดยมุมของการเคลื่อนที่เท่ากับ 30^0

2.3.5.2 การเคลื่อนที่แบบ Relative

การเคลื่อนที่แบบ Relative นี้จะอ้างอิง (reference) กับตำแหน่งที่อยู่ปัจจุบันของ Tool เช่น

PTP_REL { A 1 35 , A4 35}

คือการเคลื่อนที่แบบ relative PTP, A1 และ A4 หมุนไป 35^0 จากตำแหน่งปัจจุบัน



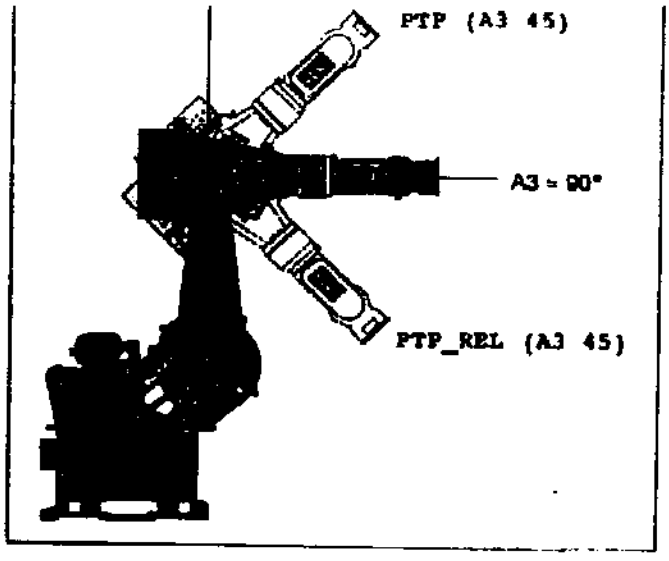
รูปที่ 2.51 แสดงการเคลื่อนที่ PTP แบบ Absolute และ Relative

สำหรับการเคลื่อนที่แบบ linear เช่น

LIN_REL {X 100, Y 55 }

เป็นคำสั่งให้เคลื่อนที่แบบ Linear ไปตามแกน X 100 mm และแกน Y 55 mm จากตำแหน่งปัจจุบัน

CIRC_REL {X 50, Y 20} , { X 200 , Y 300 }, CA 30



รูปที่ 2.52 แสดงความแตกต่างของการเคลื่อนที่แบบ Absolute และ Relative

ข้อดีข้อเสียของการเคลื่อนที่แบบ Absolute

ข้อดี

- ตำแหน่งต่าง ๆ ที่ระบุจะไม่เปลี่ยนแปลงถึงแม้ตำแหน่งปัจจุบันของหุ่นยนต์แตกต่างกัน

ข้อเสีย

- มีความยุ่งยากในการอ้างอิงตำแหน่งกับจุด Origin ของระบบ Coordinate ต่าง ๆ เพราะจะต้องทราบจุด Origin ของ Coordinate ก่อนที่จะสามารถระบุตำแหน่งที่ต้องการได้

ข้อดีข้อเสียของการเคลื่อนที่แบบ Relative

ข้อดี

- มีความง่ายในการกำหนดตำแหน่งต่างๆ เพราะสามารถอ้างอิงกับตำแหน่งปัจจุบัน

ข้อเสีย

- หากตำแหน่งปัจจุบันเปลี่ยนแปลงตำแหน่งที่กำหนดไว้ จะเปลี่ยนแปลงได้ซึ่งอาจทำให้เกิดความเสียหาย เช่น กรณีการเรียกใช้โปรแกรมย่อย หากเรียกโปรแกรมขณะที่หุ่นยนต์อยู่ในตำแหน่งปัจจุบันที่แตกต่างกันจะทำให้ตำแหน่งที่ปลาย Tool จะเคลื่อนที่ไปผิดพลาด