รายการอ้างอิง

[1] Craig, J.J. Introduction to Robotics Mechanics and Control. Third edition. USA : Pearson Prentice Hall, 2005.

[2] วิบูลย์ แสงวีระพันธุ์ศิริ. การควบคุมระบบพลศาสตร์.พิมพ์ครั้งที่ 2.กรุงเทพมหานคร : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2548.

[3] มณิขจิต. ศรีสรรพ์อัญมณี. สำนักพิมพ์มติชน, 2546.

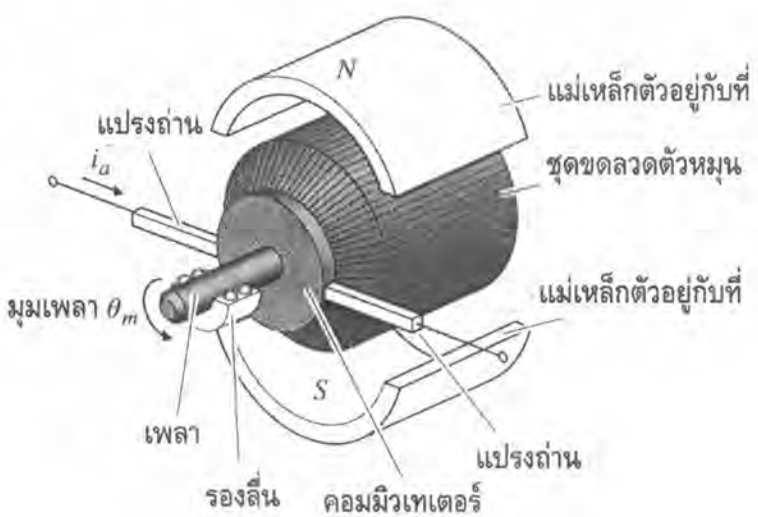
ภาคผนวก

ภาคผนวก ก

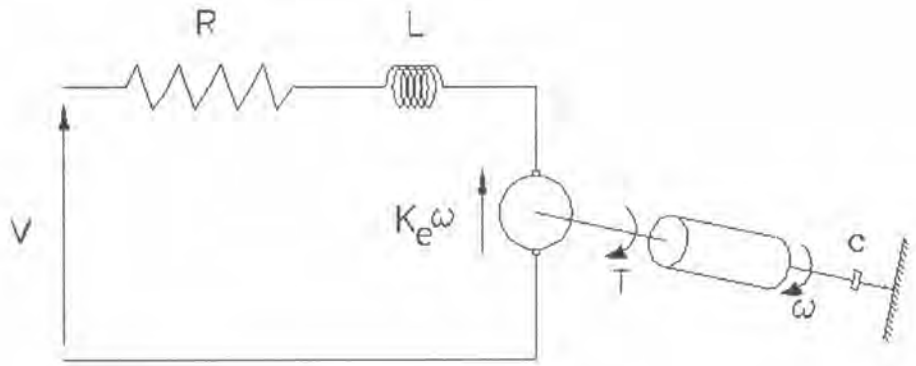
หลักการทำงานของระบบควบคุมการทำงานของเซอร์โวมอเตอร์

ก.1 ระบบการทำงานของเซอร์โวมอเตอร์

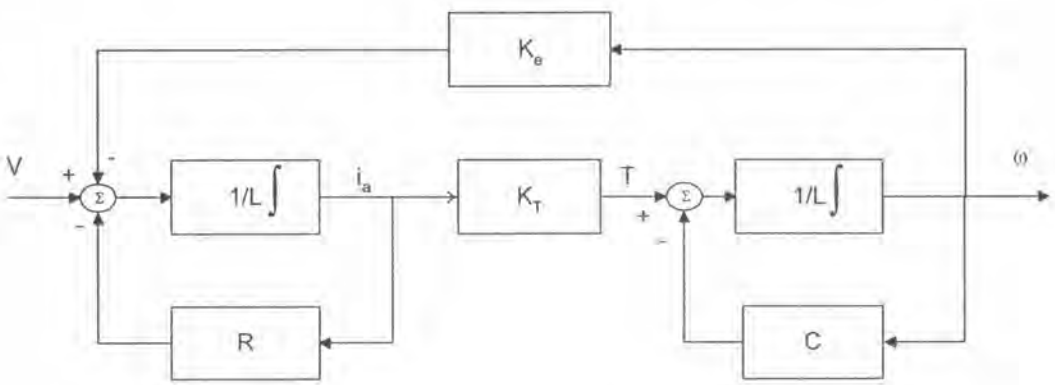
รูปที่ ก.1 เป็นรูปส่วนประกอบภายในที่แสดงชิ้นส่วนต่าง ๆ ของมอเตอร์กระแสตรงแบบเซอร์โว รูปที่ ก.2 เป็นรูปแผนภาพแสดงระบบไฟฟ้าเครื่องกล (electromechanical system) ซึ่งประกอบด้วยมอเตอร์กระแสตรงชนิดแม่เหล็กถาวร (permanent magnet DC servo motor) กับภาระที่มอเตอร์ต้องขับเคลื่อน [2] การควบคุมการทำงานของมอเตอร์กระแสตรงมีด้วยกันสองชนิดคือแบบการควบคุมอาร์เมเจอร์ (armature-controlled) และแบบการควบคุมฟิลด์ (field-controlled) ซึ่งทั้งสองแบบมีลักษณะแตกต่างกันไป ในกรณีของการควบคุมอาร์เมเจอร์ (armature-controlled) จะเหมาะสำหรับงานที่ต้องการควบคุมกระแสหรือแรงบิดที่มอเตอร์จะให้กับระบบ ส่วนในกรณีของการควบคุมฟิลด์ (field-controlled) เหมาะสำหรับงานที่ต้องการควบคุมความเร็ว เพื่อให้ได้ความเร็วตามที่ต้องการ โดยไม่คำนึงถึงกระแสที่อาจจะเพิ่มขึ้นอย่างมากในกรณีที่มีภาระที่ต้องขับสูง ในที่นี้จะกล่าวถึงเฉพาะกรณีของการควบคุมอาร์เมเจอร์ (armature-controlled) เท่านั้นเนื่องจากใช้กับเครื่องเจียระไนพลอยอัตโนมัติ ในการควบคุมนี้ค่าแรงดันไฟฟ้า (voltage) ที่ขดลวดฟิลด์ (field coil) จะคงที่ซึ่งมีค่าค่อนข้างสูง ในขณะเดียวกันค่าความเฉื่อย (inertia) ของตัวอาร์เมเจอร์ (armature) ควรจะมีค่าต่ำที่สุดเท่าที่จะเป็นไปได้ วิธีการหาสมการของการควบคุมอาร์เมเจอร์ (armature-controlled) พร้อมทั้งบล็อกไดอะแกรม (block diagram) ได้แสดงไว้ในรูปที่ ก.3



รูปที่ ก.1 ส่วนประกอบภายในของมอเตอร์กระแสตรงแบบเซอร์โว



รูปที่ ก.2 รูปแผนภาพแสดงระบบไฟฟ้าเชิงกล (Electromechanical System)



รูปที่ ก.3 แผนภาพบล็อกของระบบไฟฟ้าเชิงกล (DC servo motor)

แรงดันไฟฟ้า (Voltage) ที่ตกคร่อมวงจรอาร์เมเจอร์ (armature circuit) สามารถเขียนได้ดังนี้คือ

$$V - K_e\omega = i_aR + L\frac{di_a}{dt} \qquad \text{ก.1}$$

เมื่อ

K_e = แรงดันไฟฟ้าคงตัวหรือ $\left(\frac{\text{volt}}{\text{rad/sec}}\right)$ (back e.m.f constant)

i_a = กระแสไฟฟ้า (amp)

R = ค่าความต้านทานที่ขดลวดอาร์เมเจอร์ (ohm)

L = ค่าความเหนี่ยวนำที่ขดลวดอาร์เมเจอร์ (henry)

ω = ความเร็วเชิงมุมของเพลามอเตอร์ (rad/sec)

แรงบิดที่มอเตอร์สามารถสร้างขึ้นจะเป็นสัดส่วนกับกระแสที่ไหลผ่านขดลวดอาร์เมเจอร์ ซึ่งสามารถเขียนได้ดังนี้

$$T = K_i i_a \qquad \text{ก.2}$$

K_i = ค่าคงตัวแรงบิด (torque constant) $\left(\frac{\text{N-m}}{\text{amp}}\right)$

จากกฎข้อที่สองของนิวตัน สามารถเขียนสมการการเคลื่อนที่ของมวลที่ติดอยู่ที่ปลายของแกนมอเตอร์ได้ดังนี้คือ

$$J\frac{d\omega}{dt} = T - c\omega \qquad \text{ก.3}$$

โดยที่ J คือค่าความเฉื่อยหรือ inertia ของภาระ (load) ในที่นี้ ไม่ได้คำนึงถึงระบบส่งกำลัง (gear box) ระหว่างมอเตอร์กับภาระ รูปที่ ก.4 เป็นรูปบล็อกไดอะแกรมของระบบ รวมจากสมการที่ ก.1, ก.2 และสมการที่ ก.3

จากสมการที่ ก.3 ถ้านำมาปรับปรุงโดยคำนึงถึงพลศาสตร์ของตัวมอเตอร์ จะได้ว่า

$$K_i I - J_1\dot{\omega} = T \qquad \text{ก.4}$$

โดยที่ J_1 คือค่าความเฉื่อยหรือ inertia ของมอเตอร์

ถ้าพิจารณาสมการที่ ก.1, ก.3 และ ก.4 จะเห็นว่ามีอยู่สองแนวทางในการควบคุมการทำงานของมอเตอร์กระแสตรงกล่าวคือ

ก) การควบคุมแรงบิดของมอเตอร์โดยการควบคุมกระแสที่ไหลเข้าไปในขดลวดของมอเตอร์ ตามสมการที่ ก.4

ข) การควบคุมความเร็วในการหมุนของมอเตอร์โดยการควบคุมขนาดแรงดัน (Voltage) ที่ป้อนเข้าที่ขดลวดของมอเตอร์

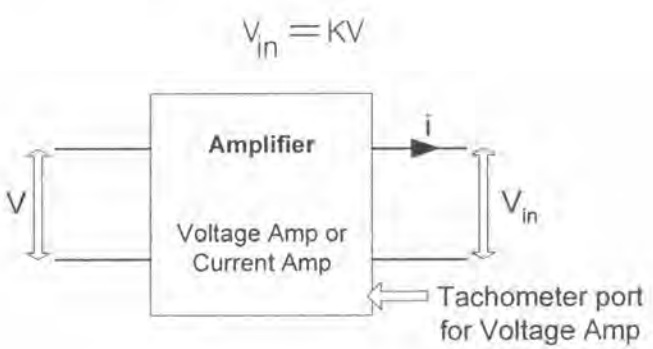
การควบคุมแบบแรก (ก) หรือการควบคุมกระแสที่ไหลเข้าไปในขดลวดของมอเตอร์ จะคำนึงว่าแรงบิดหรือ Torque (T) ที่เกิดจากตัวมอเตอร์จะเป็นค่าสัญญาณอินพุตที่ส่งเข้าไปในระบบหรือคือสัญญาณอินพุตที่ควบคุมการเคลื่อนที่ของภาระ ส่วนการควบคุมแบบที่สอง (ข) หรือการควบคุมแรงดันแรงไฟฟ้า (voltage) ที่ป้อนเข้าที่ขดลวดของมอเตอร์ จะเป็นวิธีที่มองว่า

ตัวภาระเป็นสัญญาณรบกวนหรือ disturbance ที่กระทำที่แกนของมอเตอร์ การควบคุมทั้งสองแบบนี้ให้ผลลัพธ์ของการควบคุมแตกต่างกัน

ก.2 อุปกรณ์ขยายกำลังงาน (Power Amplifier) ที่ใช้กับมอเตอร์

อุปกรณ์ขยายกำลังงาน (Power Amplifier) ที่ใช้กับมอเตอร์จะทำหน้าที่ให้พลังงานกับตัวมอเตอร์ บางครั้งจะเรียกว่า motor driver, motor controller เป็นต้น จะเห็นว่าในแบบจำลองทางพลศาสตร์ (Dynamic model) ของมอเตอร์มีส่วนของแรงดันย้อนกลับ (back emf, electromotive force) ค่าแรงดันนี้เกิดขึ้นเนื่องจากขวดหมุนตัดผ่านสนามแม่เหล็กทำให้เกิดความต่างศักดิ์ขึ้นในขวดลวดอาร์เมเจอร์ซึ่งมีทิศทางตรงข้ามกับแรงดังที่จ่ายให้กับมอเตอร์ ค่าแรงย้อนกลับของตัวมอเตอร์ที่เข้ามาเกี่ยวข้องนี้ทำให้ระบบดูซับซ้อนขึ้น และจะใช้ในกรณีที่ตัวขยายกำลังงานหรือ Power Amplifier เป็นชนิดตัวขยายกำลังงานแบบใช้งานทั่วไป (general purposed amplifier) เช่นเมื่อใช้ตัวขยายกำลังงานของเครื่องขยายเสียงที่มีขายตามท้องตลาดแล้วนำมาดัดแปลงภาคสัญญาณขาเข้าและขาออกให้ใช้ได้กับสัญญาณไฟฟ้ากระแสตรงเพื่อสามารถนำต่อเข้ากับมอเตอร์ได้ แต่ในกรณีที่ใช้ชุดขับหรือขยายกำลังที่ออกแบบมาสำหรับใช้กับมอเตอร์กระแสตรงแบบเซอร์โว (DC Servo Motor) นั้น ส่วนใหญ่แล้วจะออกแบบมาเพื่อให้สามารถเลือกโหมด (Mode) ของการทำงานได้ 2 แบบ คือแบบ Voltage Amplifier และ Current Amplifier ทั้งสองแบบนี้มีลักษณะการใช้งานแตกต่างกันกล่าวคือ ในกรณีของ Voltage Amplifier จะใช้กับมอเตอร์ที่มีตัววัดรอบหรือมี Tachometer ติดตั้งมาด้วย และชุดขับมอเตอร์หรือตัวขยายกำลังไฟฟ้าจะมีช่องสำหรับต่อสัญญาณจากตัว Tachometer ของมอเตอร์เพื่อใช้สำหรับทำการชดเชย (compensate) กับสัญญาณแรงดันย้อนกลับ (Back emf. Voltage) สมการตรงขดลวดของมอเตอร์ที่ใช้กับโหมดการทำงานแบบ Voltage Amplifier สามารถเขียนได้ดังนี้

$$V_{in} - K_e\omega + K_e\omega = L\frac{di}{dt} + iR \Rightarrow V_{in} = iR \quad (L << R) \qquad \text{ก.5}$$



ในกรณีที่ทำงานในโหมดของ Current Amplifier ตัวมอเตอร์ไม่จำเป็นต้องติดตั้งอุปกรณ์วัดความเร็วหรือ Tachometer ชุดขับหรือตัวขยายสัญญาณจะทำหน้าที่เป็นตัวขยายสัญญาณกระแสซึ่งหมายถึงแรงบิดที่มอเตอร์สามารถสร้างได้ โดยค่ากระแสนี้จะแปรตามขนาดของสัญญาณแรงดัน (Voltage) ที่ส่งเข้าไปยังตัวขยายสัญญาณ ดังนั้นจะได้ว่า

$$i = KV \quad ; \quad K = Amp/volt \qquad \text{ก.6}$$

ชุดขับที่ทำงานในโหมด Current Amplifier จะนิยมใช้กับการควบคุมตำแหน่งการหมุนและควบคุมแรงบิดของมอเตอร์กับภาระงาน ส่วนชุดขับที่ทำงานในโหมด Voltage Amplifier นั้นจะนิยมใช้ในกรณีที่ต้องการควบคุมความเร็วของมอเตอร์เป็นหลัก

ก.3 หลักการพื้นฐานของระบบควบคุมอัตโนมัติ

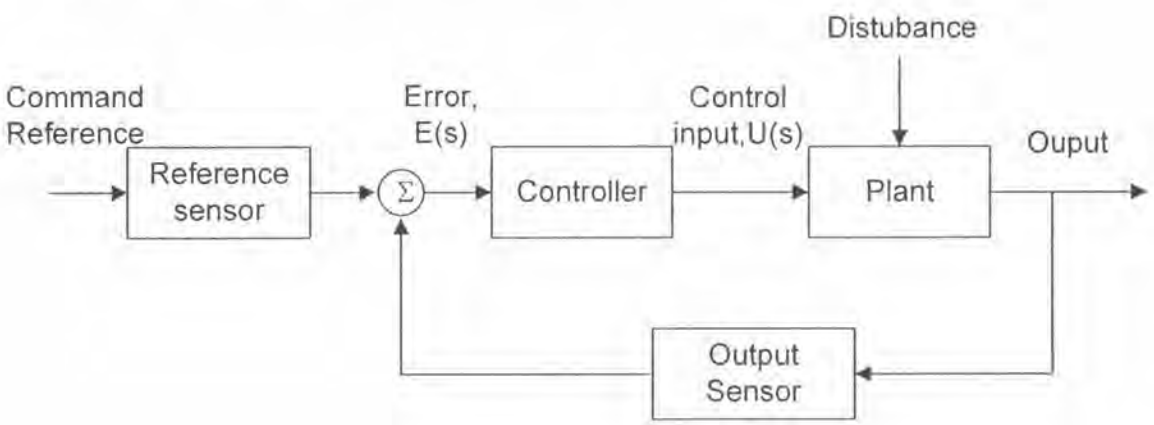
เครื่องเจียระไนพลอยนี้สามารถทำงานได้โดยอัตโนมัติ ดังนั้นจึงต้องมีระบบควบคุมทั้งตำแหน่งและความเร็วโดยใช้ระบบควบคุมที่เรียกว่า Feedback Control หรือระบบควบคุมแบบอัตโนมัติ (Automatic Controller)

ระบบควบคุมแบบอัตโนมัติ จะทำหน้าที่เป็นตัวควบคุมการทำการของระบบพลศาสตร์ที่จะทำการควบคุม โดยระบบควบคุมแบบอัตโนมัตินี้จะมีส่วนที่ทำการเปรียบเทียบสัญญาณที่วัดค่าตัวแปรสถานะหรือ State variables จริงของระบบด้วยตัวตรวจรู้ (sensor) แล้วนำมาเปรียบเทียบกับค่าตัวแปรที่ต้องการพิจารณาการเปลี่ยนแปลงที่เกิดขึ้น ค่าความแตกต่างนี้จะถูกใช้สำหรับสร้างสัญญาณควบคุมซึ่งจะสามารถลดการเปลี่ยนแปลงหรือความผิดพลาดลงไปได้ เพื่อให้มีค่าเป็นศูนย์หรือเป็นค่าตัวเลขที่มีขนาดเล็ก การกระทำของระบบควบคุมอัตโนมัติที่สร้างสัญญาณควบคุม (Control signal) เรียกว่าปฏิกิริยาควบคุม (Control Action)

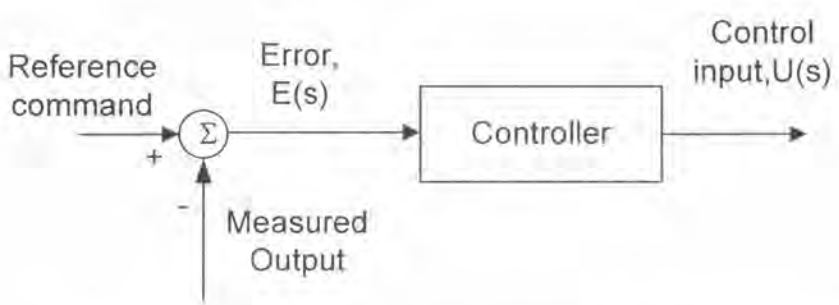
การใช้งานของปฏิกิริยาควบคุม (Control Action) ที่เหมาะสมสำหรับควบคุมการทำงานของเครื่องเจียระไน โดยจะใช้ตัวควบคุมที่ใช้ในระบบควบคุมในโรงงานอุตสาหกรรมซึ่งสามารถนำมาประยุกต์ใช้ในการพัฒนาเครื่องเจียระไนพลอยแบบอัตโนมัติ ตัวควบคุมที่จะใช้ในเครื่องเจียระไนพลอยนี้จะเป็นตัวควบคุมแบบพีดี (PD, Proportional plus Derivative) หรืออาจจะมีส่วนของอินทิกรัลเข้ามาช่วยด้วย โดยที่ตัวควบคุมแบบสัดส่วน (Proportional action, P) นั้นจะช่วยให้การตอบสนองเร็วขึ้น ตัวควบคุมแบบอนุพันธ์ (Derivative action, D) จะช่วยเพิ่มค่าอัตราส่วนการหน่วงให้กับระบบทำให้มีค่าโอเวอร์ชูตน้อยลงระบบจะทำงานราบเรียนขึ้น และตัวกิริยาแบบอินทิกรัล (Integral action, I) จะช่วยให้ลดค่าความผิดพลาดที่สถานะอยู่ตัวได้ (steady state error) การเข้าใจบทบาทของตัวกิริยาหรือ action เพื่อให้เข้าใจบทบาทของแต่ละตัวเพื่อ

นำมาลดผลกระทบที่เกิดขึ้นเนื่องจากค่าพารามิเตอร์ต่าง ๆ ของระบบมีการเปลี่ยนแปลงในขณะที่ทำงานในสภาวะแตกต่างกัน

รูปที่ ก.4 เป็นรูปแผนภาพบล็อกแสดงระบบควบคุมแบบปิด สัญญาณคำสั่งอ้างอิง (Command Reference) จะถูกเปลี่ยนด้วยระบบตรวจรู้อ้างอิง (Reference sensor) เพื่อเปลี่ยนแปลงให้สัญญาณคำสั่งอ้างอิงดังกล่าวมีลักษณะที่มีหน่วยเดียวกับสัญญาณที่ออกจากระบบตรวจรู้ขาออก (Output sensor) เพื่อต้องการเปรียบเทียบกัน เมื่อเปรียบเทียบกันแล้ว จะได้สัญญาณผิดพลาด (Error Signal) สัญญาผิดพลาดนี้จะถูกส่งเข้าไปยังระบบควบคุม (Controller) เพื่อสร้างสัญญาณควบคุม (Control Input) ดังแสดงในรูปที่ ก.5 และสัญญาณที่ออกจากตัวควบคุมนี้จะใช้เป็นสัญญาณควบคุมที่จะส่งเข้าไปยังระบบที่ต้องการควบคุม (Plant) ตัวควบคุมหรือ Controller จะมีบทบาทมากกับลักษณะตอบสนองของสัญญาณเอาต์พุตที่ต้องการ ตัวควบคุมมาตรฐานมีด้วยกันหลายประเภทซึ่งจะทำการศึกษาอย่างละเอียดต่อไป



รูปที่ ก.4 แผนภาพบล็อกแสดงระบบควบคุมแบบปิด

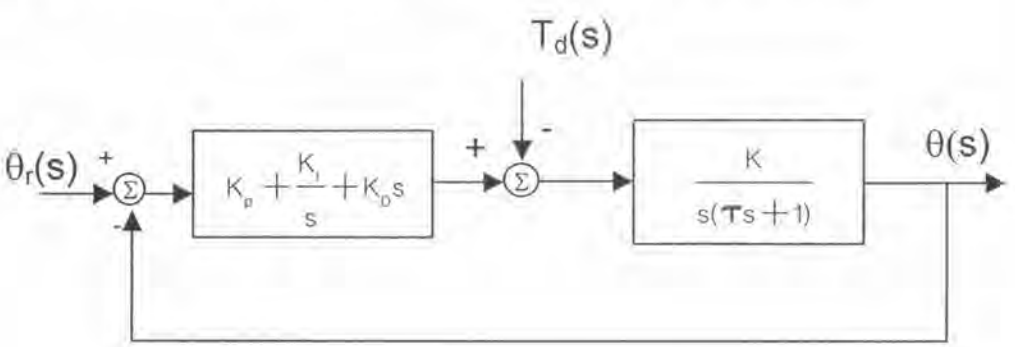


รูปที่ ก.5 ความสัมพันธ์ระหว่างสัญญาณควบคุมและสัญญาณผิดพลาด

ในทางปฏิบัติแล้วการกำหนดลักษณะเฉพาะสำหรับออกแบบระบบควบคุมมีรายละเอียดมากกว่านี้ เช่นอาจจะต้องกำหนดความกว้างแถบ (bandwidth) ที่ทำให้ระบบมีความปลอดภัย ความปลอดภัยในที่นี้หมายถึงระบบปลอดภัยจากการที่ระบบจะไม่มีเสถียรภาพหรือไม่มีความสมดุล ไม่สามารถรู้ได้แน่ชัดเลยเกี่ยวกับค่าพารามิเตอร์ของระบบนั้น ว่ามีค่าถูกต้องแม่นยำหรือมีความแน่นอนเท่าไรตามค่าที่ใช้ในสมการจำลองการทำงานของระบบ ระบบควบคุมหรือตัวควบคุม (Controller) ชนิดต่าง ๆ หรือที่มีรูปแบบต่างกัน (เช่นตัวควบคุมแบบ P, PD, PID เป็นต้น) ก็สามารถออกแบบให้ระบบรวมมีความไว (sensitivity) มากน้อยต่อการเปลี่ยนแปลงของค่าพารามิเตอร์แตกต่างกันออกไป ดังนั้นการออกแบบบางครั้งอาจจะต้องครอบคลุมถึงความไว (sensitivity) ของระบบควบคุมต่อการเปลี่ยนแปลงของพารามิเตอร์ของระบบด้วย เพื่อทำให้ระบบควบคุมโดยรวมมีความปลอดภัยเมื่อค่าพารามิเตอร์ของระบบเปลี่ยนแปลงในขณะที่ระบบกำลังทำงาน

ก.4 การควบคุมแบบสัดส่วนบวกกับแบบอินทิกรัลบวกกับแบบอนุพันธ์ (PID-control) กับการควบคุมตำแหน่งมุมของแกนหมุนของมอเตอร์

การควบคุมแบบสัดส่วนบวกกับแบบอินทิกรัลและตัวควบคุมแบบอนุพันธ์เป็นตัวควบคุมพีไอดี (PID-control) ตัวควบคุมพีดีที่ใช้ควบคุมมอเตอร์ซึ่งมีอันดับสองข้างต้นจะเห็นว่าสามารถปรับปรุงการตอบสนองช่วง Transient ได้โดยเฉพาะเมื่อสัญญาณอินพุตเป็นฟังก์ชันขั้นหนึ่งหน่วย แต่ถ้าสัญญาณอินพุตเป็นสัญญาณแรมป์แล้วค่าความผิดพลาดที่สถานะอยู่ตัวก็ยังคงมีอยู่ ทำให้ยังไม่สามารถบรรลุการออกแบบที่สมบูรณ์ ในหัวข้อนี้จะศึกษาตัวควบคุมแบบพีไอดีในการควบคุมตำแหน่งการหมุนของมอเตอร์ ซึ่งแสดงไว้ในแผนภาพบล็อกข้างล่างนี้



จากแผนภาพบล็อกข้างต้น สามารถเขียนสมการฟังก์ชันถ่ายโอนได้ดังนี้คือ

$$\frac{\theta(s)}{\theta_r(s)} = \frac{K(K_D s^2 + K_p s + K_I)}{\tau s^3 + (1+KK_D)s^2 + KK_p s + KK_I}$$

และ

$$\frac{\theta(s)}{T_d(s)} = -\frac{Ks}{\tau s^3 + (1+KK_D)s^2 + KK_p s + KK_I}$$

ซึ่งค่าผิดพลาดในสถานะอยู่ตัว (Steady state error) อันเนื่องมาจากสัญญาณอินพุตที่เป็นฟังก์ชันขั้นหนึ่งหน่วยจะเป็นศูนย์เนื่องมาจากมีตัวควบคุมแบบอินทิกรัล ระบบนี้จะมีเสถียรภาพเมื่อค่าอัตราขยาย K_p, K_I และ K_D มีค่าเป็นบวกและ

$$(1+KK_D)KK_p - \tau KK_I > 0$$

ซึ่งค่าอัตราขยาย K_D ที่อยู่ในระบบควบคุมนี้จะช่วยทำให้สมการข้างต้นเป็นไปได้โดยง่าย

ค่าอัตราขยาย K_p, K_I และ K_D ที่ใช้ในระบบควบคุมเครื่องเจียระไนพลอยอัตโนมัติ แสดงไว้ที่ตารางที่ ก.1

ตารางที่ ก.1 ค่าอัตราขยาย K_p, K_I และ K_D

	K_p	K_I	K_D
แกนการเคลื่อนที่ขึ้นและลง	20	670	200
แกนการหมุนทำมุมการเจีย	150	300	50
แกนการหมุนเหลี่ยม	7500	250	50
แกนการหมุนเปลี่ยนจาน	100	350	250

ภาคผนวก ข

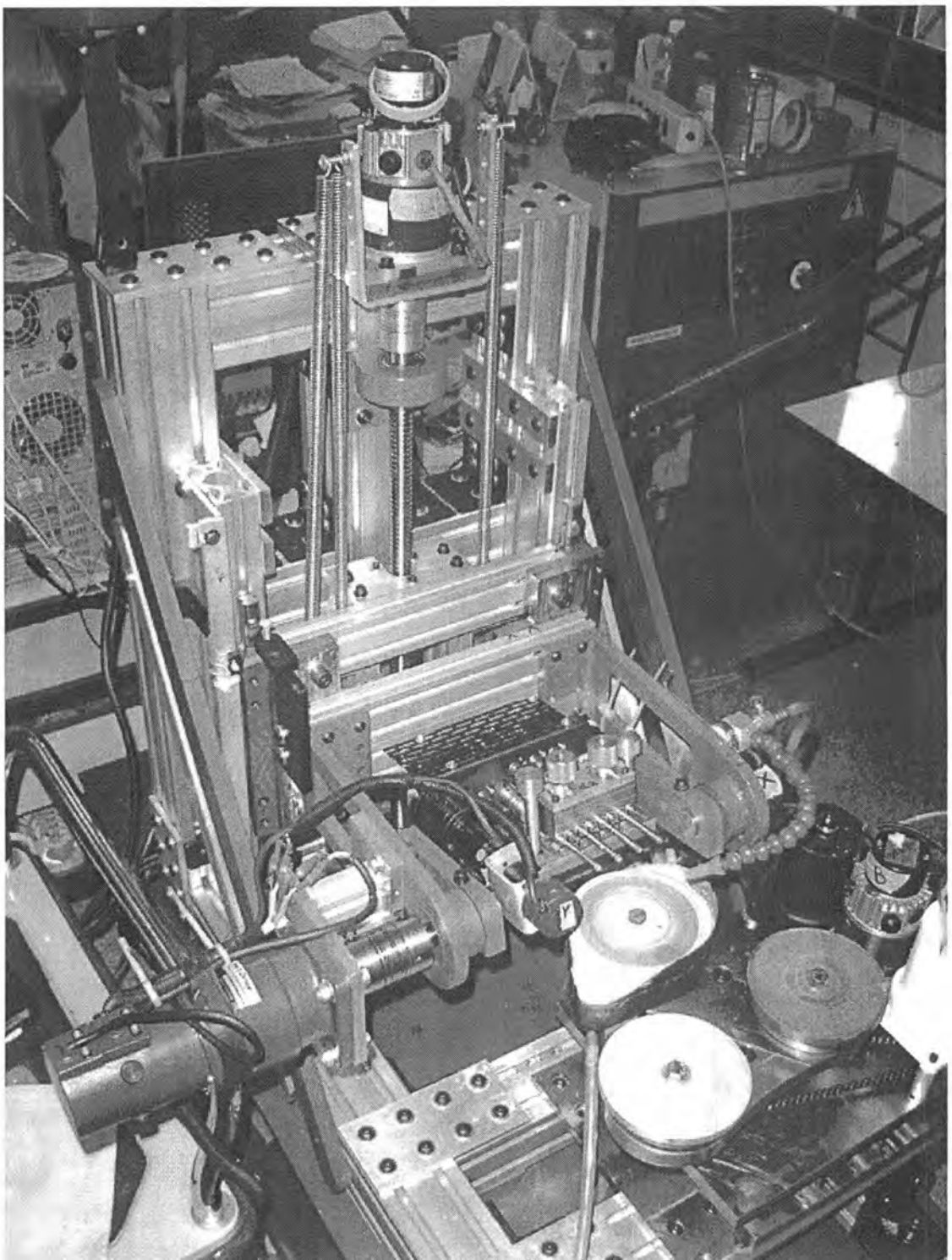
คู่มือการใช้โปรแกรมสำหรับเครื่องเจียระไนอัตโนมัติ

ข.1 บทนำ

เครื่องเจียระไนพลอยที่พัฒนาขึ้นมานั้น ได้มีการปรับปรุงเครื่องในส่วนต่าง ๆ หลายส่วนเพื่อลดข้อผิดพลาดที่เกิดขึ้นจากเครื่องทดสอบหลักการเจียระไน ยกตัวอย่างเช่น เสริมความแข็งแกร่งในส่วนโครงสร้าง เพื่อให้เครื่องมีความมั่นคงมากยิ่งขึ้นในขณะที่ทำการเจียระไน เสริมระบบเปลี่ยนจานเจียระไนพลอยได้อัตโนมัติเพื่อลดเวลาในการใช้แรงงานคนในการเปลี่ยนจานเจียระไน ดังนั้น เครื่องเจียระไนพลอยในรุ่นที่ 2 จึงตอบสนองในทุกฟังก์ชั่นในการใช้งาน และลดการมีส่วนร่วมในการใช้งานของมนุษย์ลงให้น้อยที่สุด

ส่วนประกอบของเครื่องเจียระไนพลอยอัตโนมัติที่พัฒนาขึ้น มีดังนี้

- แกนที่ 1 เป็นแกนที่ใช้ในการยกแท่นที่ติดตั้งก้านจับพลอยขึ้นลง (Robotic Arm) เพื่อปรับระดับความลึกในการเจียระไน
- แกนที่ 2 คือแกนหมุนของแท่นที่ติดตั้งก้านจับพลอย (Drop stick grinding angle) เพื่อปรับมุมองศาในการเจียระไน
- แกนที่ 3 คือแกนหมุนของก้านจับพลอย (Stone Attachment angle) เพื่อปรับเหลี่ยมในการเจียระไน
- แกนที่ 4 คือแกนการเคลื่อนที่หมุนเพื่อเปลี่ยนชนิดของจานเจียระไน
- แกนที่ 5 แกนการเคลื่อนที่ของจานเจียระไนในแนวเชิงเส้น เพื่อไม่ให้เกิดการเจียระไนที่ตำแหน่งเดียวกันตลอด
- แกนที่ 6 แกนการหมุนของจานเจียระไน (Spindle)
- ปั๊มน้ำหล่อเย็น ใช้ในการระบายความร้อนของพลอย เพื่อลดการแตกร้าวของพลอย
- ไฟแสดงสัญญาณสถานการณ์เจียระไน
- โปรแกรมและระบบควบคุมเครื่องเจียระไนอัตโนมัติ



รูปที่ ข.1 เครื่องเจียระไนอัตโนมัติที่พัฒนาขึ้น (Chula Faceting Machine Model 2)

ข.2 การตั้งค่าโปรแกรมควบคุมเครื่องเจียระไนพลอยอัตโนมัติ

ข.2.1 ลักษณะทั่วไปของโปรแกรมควบคุมการทำงาน

ในการเจียระไนพลอยด้วยเครื่องเจียระไนพลอยอัตโนมัติ จำเป็นที่จะต้องมีโปรแกรมที่ใช้สำหรับการควบคุมการเคลื่อนที่ในการเจียระไน รวมถึงการควบคุมฟังก์ชั่นเสริมบางอย่าง เช่น การเปิดและปิดน้ำหล่อเย็น โดยการควบคุมการเคลื่อนที่จะใช้ข้อมูลพารามิเตอร์ที่ได้จากโปรแกรมการออกแบบการเจียระไนพลอย เช่น ระยะความลึกในการเจียระไน สำหรับโปรแกรมที่ใช้ควบคุมการเคลื่อนที่สำหรับเครื่องเจียระไนอัตโนมัตินี้ จึงต้องมีความสามารถดังที่กล่าวมาข้างต้น

ข.2.2 ขั้นตอนการเปิดโปรแกรม

การเรียกใช้โปรแกรมสามารถทำได้โดยเลือกไอคอนของโปรแกรม SPiiPlus MMI ซึ่งคอมพิวเตอร์จะเริ่มทำการเปิดโปรแกรมควบคุม

รูปที่ ข.2 ไอคอนของโปรแกรม SPiiPlus MMI

โปรแกรมจะถามวิธีเชื่อมต่อของอุปกรณ์ควบคุม ซึ่งในที่นี้ใช้การเชื่อมต่อแบบ PCI Bus ให้ตอบตกลง

สำหรับวิธีการเชื่อมต่อไปยังอุปกรณ์ควบคุมสามารถทำได้หลายวิธี เช่น ผ่านช่องสัญญาณอนุกรม (Serial Port) ผ่านระบบเครือข่าย (Ethernet) และผ่าน PCI Bus

รูปที่ ข.3 การยืนยันการเชื่อมต่อ

หลังจากที่ได้ทำการเชื่อมต่อกับอุปกรณ์ควบคุมแล้ว โปรแกรมจะแสดงหน้าต่างหลักของโปรแกรม ซึ่งมีไฟสีเขียวที่แสดงสถานะของการเชื่อมต่อ ว่ามีการเชื่อมต่อที่สมบูรณ์แล้ว

รูป ข.4 หน้าต่างหลักของโปรแกรมการควบคุมเครื่องเจียระไนพลอยอัตโนมัติ

ข.2.3 การตั้งค่า (Configuration and Setup)

การใช้งานของเครื่องเจียระไนพลอยอัตโนมัติจำเป็นที่จะต้องมีการปรับตั้งค่าของระบบ เพื่อเป็นการปรับตั้งการใช้งานให้เหมาะสมกับรูปแบบ เช่น ปรับตั้งค่าขอบเขตความปลอดภัย การกำหนดความเร็วสูงสุด เป็นต้น

ข.2.3.1 การตรวจสอบระบบรักษาความปลอดภัย (Safety Monitor)

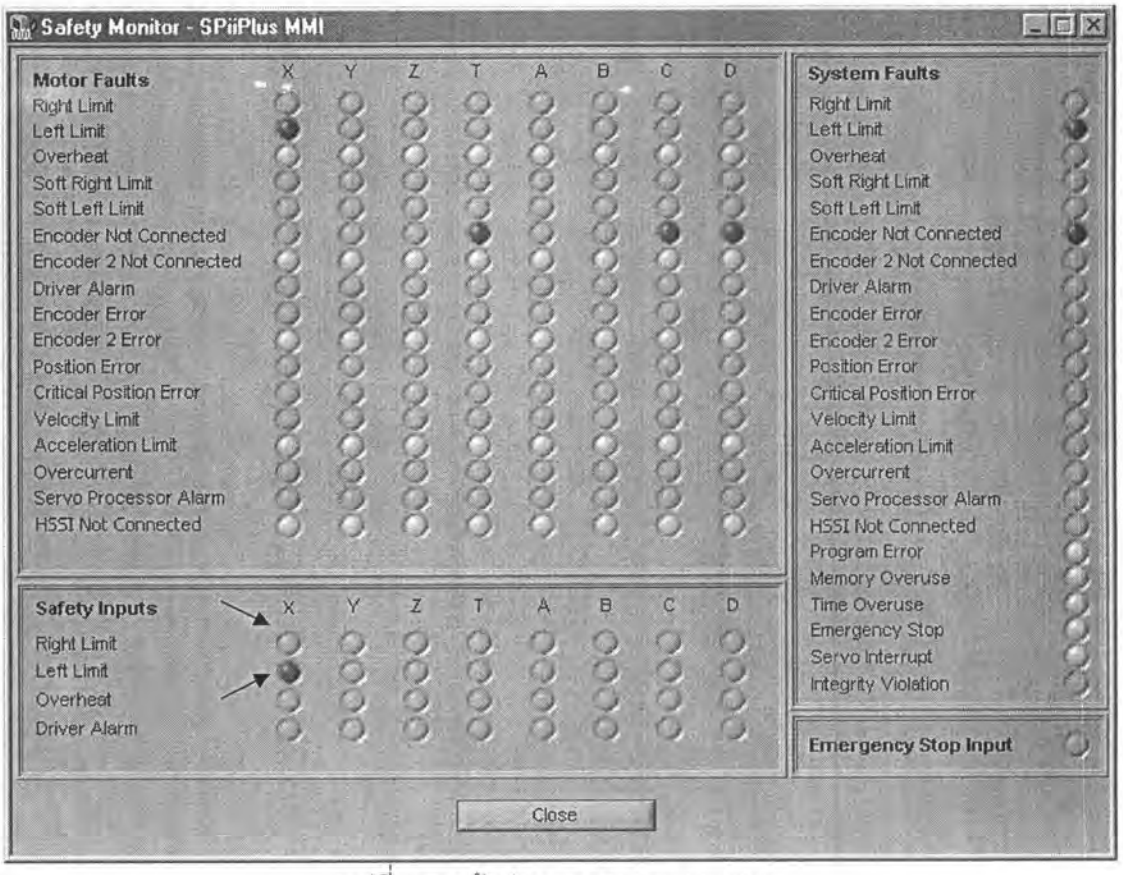
เพื่อป้องกันการเคลื่อนที่ออกนอกขอบเขตที่กำหนด ที่จะทำให้เกิดความเสียหายกับโครงสร้างของเครื่องเจียระไนพลอยอัตโนมัติ จึงต้องมีการกำหนดขอบเขตการเคลื่อนที่

สำหรับเครื่องเจียระไนพลอยอัตโนมัติ จะใช้การตรวจสอบการเคลื่อนที่ 2 แกน คือ แกนการเคลื่อนที่ในแนวดิ่ง และแกนหมุนของแท่นที่ติดตั้งก้านจับพลอย (Drop stick grinding angle) เพื่อป้องกันไม่ให้เคลื่อนที่เลยออกนอกขอบเขต เป็นเหตุให้เกิดความเสียหาย ซึ่งการกำหนดขอบเขตนั้น สามารถนำมาประยุกต์ใช้ในการหาระยะอ้างอิงของเครื่อง

ระบบหยุดฉุกเฉิน (Emergency Stop) เพื่อใช้ในการหยุดการเคลื่อนที่ทั้งหมดทุกแกน เมื่อเกิดเหตุการณ์ฉุกเฉิน โดยระบบจะตัดการเคลื่อนที่ทั้งหมดทุกแกน

ขั้นตอนในการตรวจสอบระบบรักษาความปลอดภัย มีดังนี้

1. เลือก Safety Monitor บนหน้าต่างหลักของโปรแกรมการควบคุม
2. โปรแกรมแสดงหน้าต่างของระบบรักษาความปลอดภัย



รูปที่ ข.5 หน้าต่างของ Safety Monitor

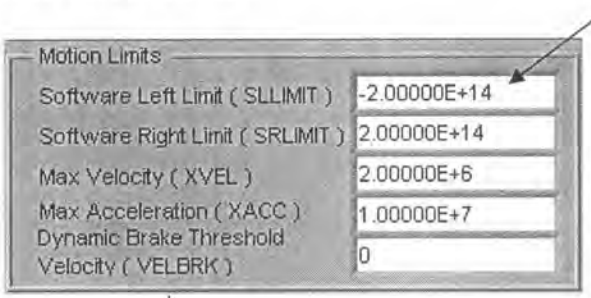
3. โปรแกรมจะแสดงสัญลักษณ์เป็นสีเพื่อแสดงสถานะของระบบ โดยสีเขียวแสดงว่าระบบอยู่ในสภาวะปลอดภัย และสีแดงหมายถึงระบบอยู่ในสภาวะที่มีการเคลื่อนที่ออกนอกจากขอบเขต ส่งผลให้โปรแกรมหยุดสั่งการเคลื่อน

ข.2.3.2 การตั้งค่าขอบเขตโดยใช้ซอฟแวร์ (Software Limit)

โดยปกติแล้วการใช้การตั้งค่าขอบเขตโดยใช้ซอฟแวร์จะใช้สำหรับการตั้งค่าเพื่อเป็นการเตือนการเคลื่อนที่ก่อนที่จะเคลื่อนที่ไปถึงขอบเขตที่แท้จริง เพื่อเป็นการป้องกันล่วงหน้าไม่ให้เกิดความเสียหาย มีขั้นตอนในการตั้งค่าดังนี้

1. เปิดโปรแกรม SPiiPlus MMI เลือก Motion Manager และทำการเคลื่อนที่ไปยังตำแหน่งศูนย์
2. ที่หน้าต่าง Motion Manager เลือก Zero Feedback Position เพื่อเป็นการตั้งค่าศูนย์ให้กับโปรแกรม
3. เคลื่อนการเคลื่อนที่ไปยังด้านซ้าย (นับลง) จนกระทั่งถึงตำแหน่งที่ต้องการตั้งเป็นขอบเขตโดยใช้ซอฟแวร์ด้านซ้าย

4. ที่หน้าต่าง Motion Manager สังเกตค่าของ Feedback Position (FPOS) แล้วบันทึกค่าเก็บเอาไว้
5. ที่หน้าต่าง SPiiPlus MMI เลือก Setup แล้วเลือก Adjuster
6. เลือกแกนที่ต้องการแล้วเลือก Safety Parameters
7. ที่หน้าต่างของ Safety Parameters ในส่วนของ Software Left Limit ให้ใส่ค่าที่ได้จากขั้นตอนที่ 4



รูปที่ ข.6 Software Left Limit

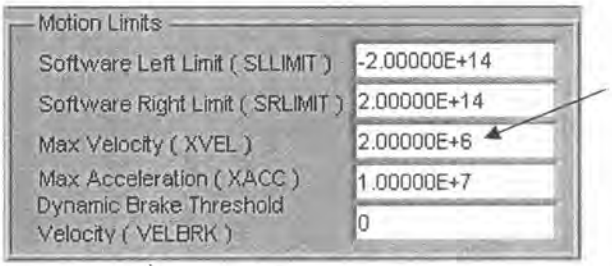
8. ทำการเคลื่อนที่ไปยังด้านขวา (นับขึ้น) จะกระทั่งถึงตำแหน่งที่ต้องการตั้งเป็นขอบเขตโดยใช้ซอฟแวร์ด้านขวา
9. ที่หน้าต่าง Motion Manager สังเกตค่าของ Feedback Position (FPOS) แล้วบันทึกค่าเก็บเอาไว้อีกครั้ง
10. ที่หน้าต่างของ Safety Parameters ในส่วนของ Software Right Limit ให้ใส่ค่าที่ได้จากขั้นตอนที่ 9

รูปที่ ข.7 Software Right Limit

ข.2.3.3 การกำหนดค่าความเร็วและความเร่งสูงสุด (Maximum Velocity and Acceleration)

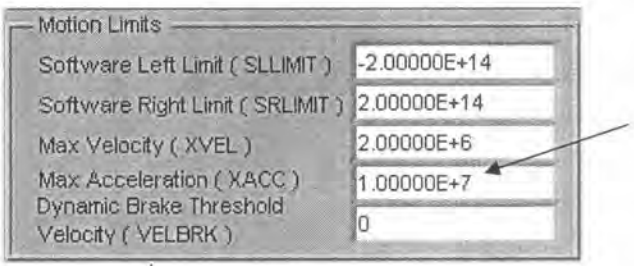
ความเร็วและความเร่งสูงสุดสามารถที่จะกำหนดได้โดยผู้ใช้ หากว่าเมื่อใดที่มีค่าความเร็วและความเร่งเลยจากค่าที่กำหนดไว้ ระบบก็จะทำการเตือน โดยมีขั้นตอนในการตั้งค่าดังนี้

1. เปิดโปรแกรม SPiiPlus MMI เลือก Setup แล้วเลือก Adjuster
2. เลือกแกนที่ต้องการแล้วเลือก Safety Parameters
3. ใส่ค่าความเร็วสูงสุด (Maximum Velocity: XVEL) สำหรับแกนที่เลือก



รูปที่ ข.8 การใส่ค่าความเร็วสูงสุด

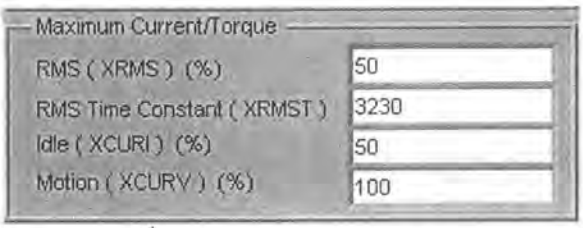
4. ใส่ค่าความเร่งสูงสุด (Maximum Acceleration: XACC) สำหรับแกนที่เลือก



รูปที่ ข.9 การใส่ค่าความเร่งสูงสุด

ข.2.3.4 การกำหนดค่ากระแสสูงสุด (Axis Maximum Current: Torque)

1. เปิดโปรแกรม SPiiPlus MMI เลือก Setup แล้วเลือก Adjuster
2. เลือกแกนที่ต้องการแล้วเลือก Safety Parameters
3. ใส่ค่า Idle (XCURV) ซึ่งเป็นค่าที่แสดงถึงระดับกระแสน้อยที่ใช้สำหรับรักษาตำแหน่งของมอเตอร์ให้อยู่ที่ตำแหน่งเดิม
4. ใส่ค่า Motion (XCURV) ซึ่งเป็นค่ากระแสสูงสุดที่ใช้เมื่อมอเตอร์มีการเคลื่อนที่
5. ใส่ค่า RMS (XRMS) ซึ่งเป็นค่ากระแสปกติที่ใช้เมื่อมอเตอร์มีการเคลื่อนที่
6. ใส่ค่า RMS Time Constant ในหน่วย msec (XRMST) ซึ่งเป็นค่าของช่วงเวลาที่ใช้สำหรับตรวจสอบสภาวะที่กระแสเกิน (Over current Fault)



รูปที่ ข.10 การใส่ค่ากระแสสูงสุด

ตัวอย่างการใส่ค่ากระแสสูงสุด

- ลักษณะของอุปกรณ์ขับ : กระแสสูงสุด 20 แอมแปร์ กระแสปกติ 10 แอมแปร์
- ลักษณะของมอเตอร์ : กระแสสูงสุด 10 แอมแปร์ กระแสปกติ 3 แอมแปร์

จะได้

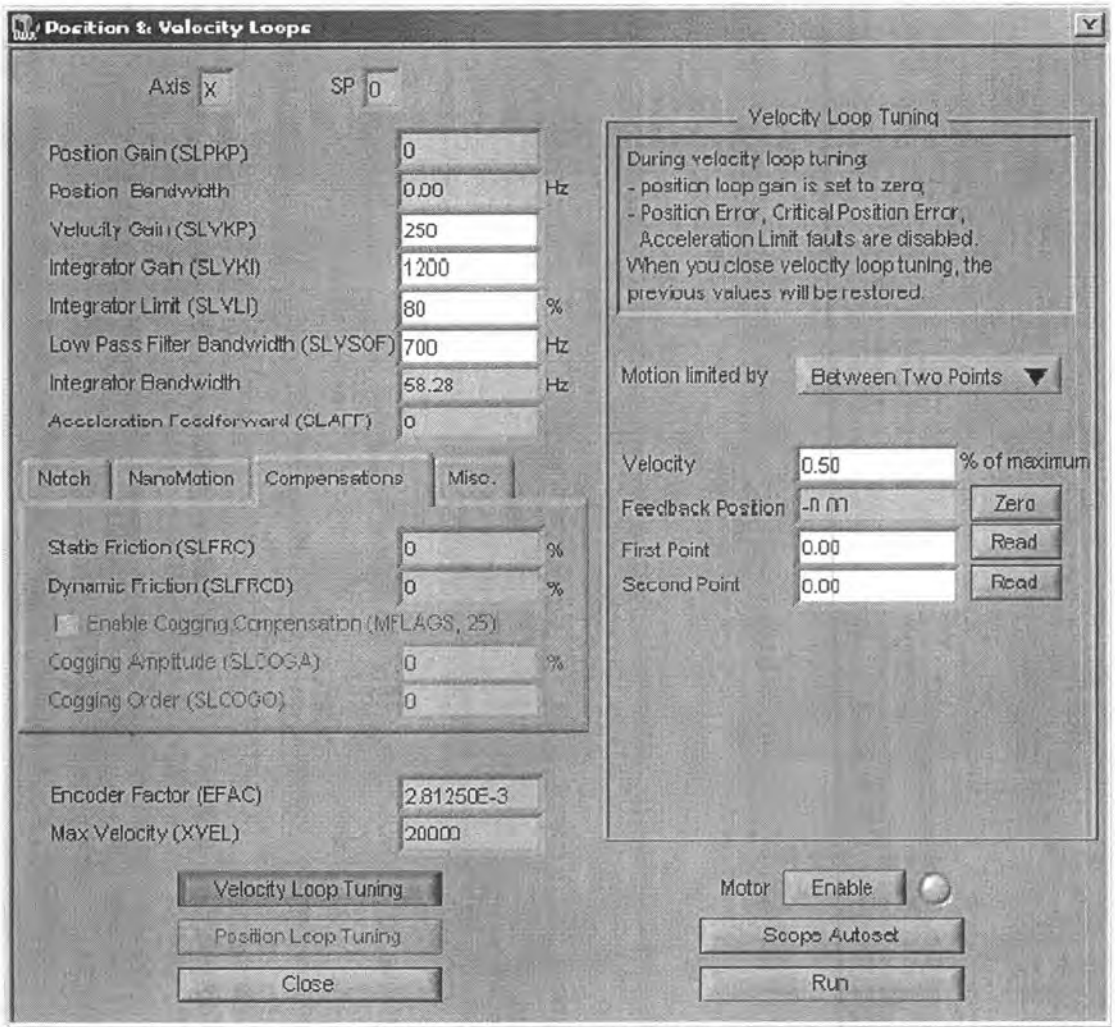
- XCURV = (10/20)*100 = 50%
- XRMS = (3/20)*100 = 15%

ข.2.4 การปรับค่าระบบควบคุม (Position and Velocity Loops)

ในการควบคุมการเคลื่อนที่ของเครื่องเจียระไนพลอย ซึ่งมีอุปกรณ์ขับที่สำคัญคือมอเตอร์ ดังนั้นจึงต้องมีการปรับตั้งค่าตัวควบคุมให้เหมาะสม เพื่อให้สามารถเคลื่อนที่ได้อย่างสมบูรณ์ ทั้งตอบสนองด้วยระยะเวลาที่สั้น และมีค่าความผิดพลาดที่น้อยที่สุด

2.4.1 การปรับตั้งค่าวังวนความเร็ว (Adjusting the Velocity Loop)

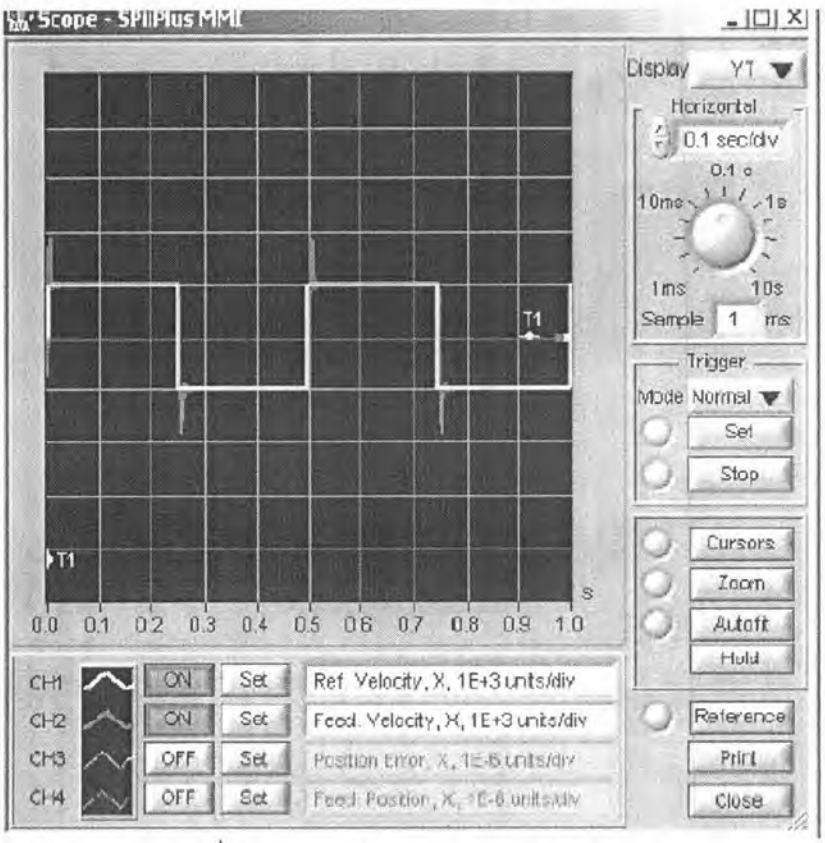
1. เปิดโปรแกรม SPiiPlus MMI เลือก Setup เลือก Adjuster แล้วเลือก Position & Velocity Loops โปรแกรมจะแสดง Position & Velocity Loops และหน้าต่างของ Scope
2. เลือก Velocity Loops Tuning หน้าต่างของ Position & Velocity Loops จะขยายออก



รูปที่ ข.11 หน้าต่าง Velocity Loop

3. ที่ช่อง Motion Limited By ให้ใส่ค่าลักษณะการเคลื่อนที่ ที่ใช้ในการปรับตั้งค่าของ Velocity Loops Tuning
4. ที่ช่อง Low Pass Filter Bandwidth ใส่ค่า 650 Hz
5. ที่ช่อง Integrator Gain ใส่ค่า 0
6. ที่ช่อง Velocity Gain ใส่ค่า 100
7. ที่ช่อง Integrator Limit ใส่ค่า 50%
8. ที่ช่อง Velocity ใส่ค่า 10
9. เลือก Enable เพื่อเปิดมอเตอร์
10. เลือก Run เพื่อให้มอเตอร์เริ่มเคลื่อนไหว
11. เลือก Scope Autoset เพื่อเลือกความเร็วอ้างอิง และความเร็วที่ป้อนกลับให้แสดงบนหน้าต่าง Scope

12. ที่หน้าต่าง Position & Velocity Loop ใส่ค่า Velocity Gain เพิ่มขึ้นเป็นสองเท่าจนกระทั่งได้กราฟสี่เหลี่ยมและมีโอเวอร์ชูตน้อย (Overshoot)
13. เพิ่มค่า Integrator Gain ทีละ 100 จนกระทั่งได้ความกว้างของโอเวอร์ชูตเป็น 10-20% ของขนาดคลื่น
14. ใส่ค่า Integrator Bandwidth ไม่เกิน 50Hz โดยมีค่าตามสมการของ (Integrator Gain)/20 = Integrator Bandwidth

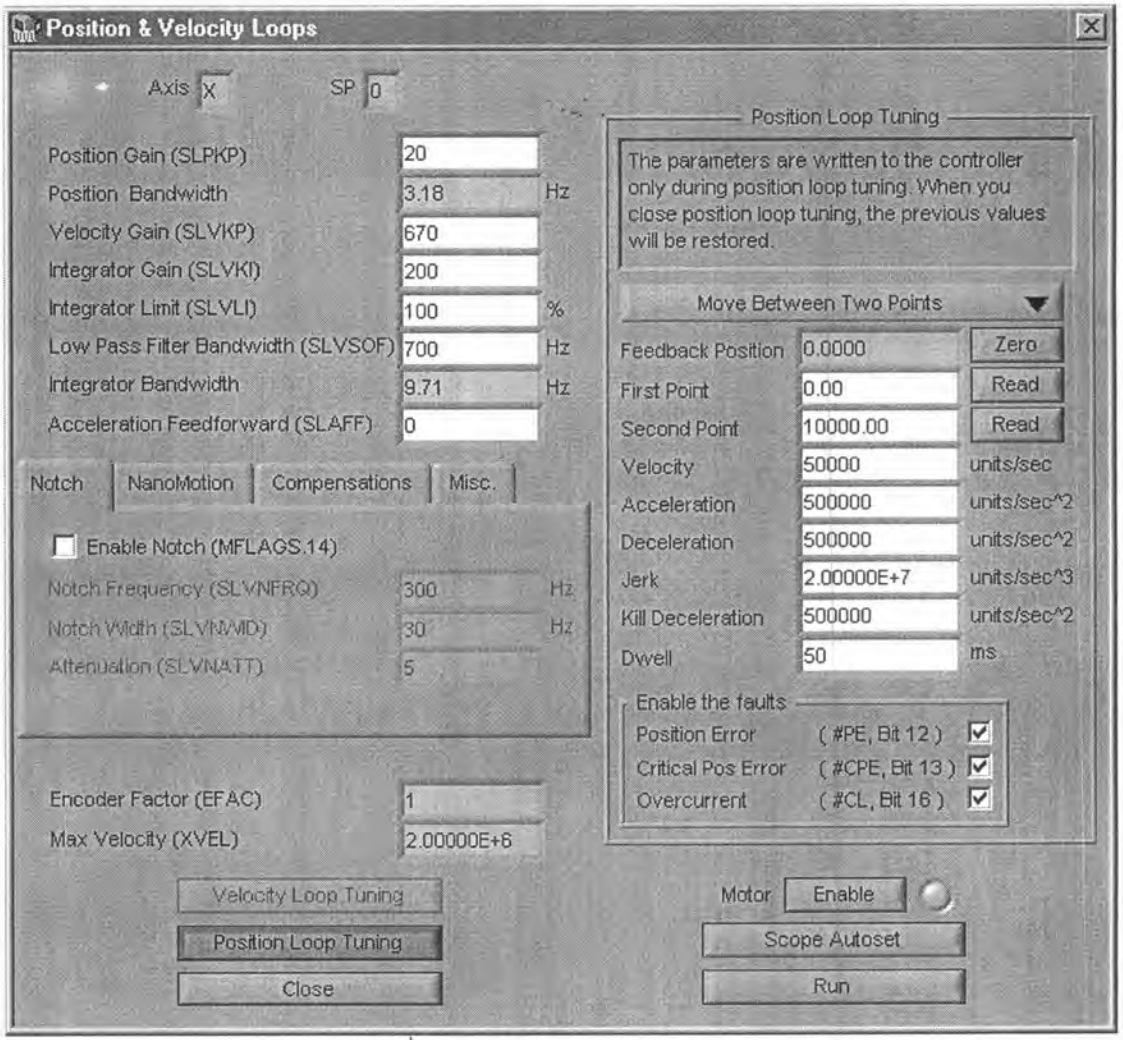


รูปที่ ข.12 ผลของ Velocity Loop Tuning

15. ปรับค่า Velocity Gain และ Integrator Gain จนกระทั่งได้กราฟเป็นที่น่าพึงพอใจ

2.4.2 การปรับตั้งค่าวังวนตำแหน่ง (Adjusting the Position Loop)

1. เปิดโปรแกรม SPiiPlus MMI เลือก Setup เลือก Adjuster แล้วเลือก Position & Velocity Loops โปรแกรมจะแสดง Position & Velocity Loops และหน้าต่างของ Scope
2. เลือก Velocity Loops Tuning หน้าต่างของ Position & Velocity Loops จะขยายออก



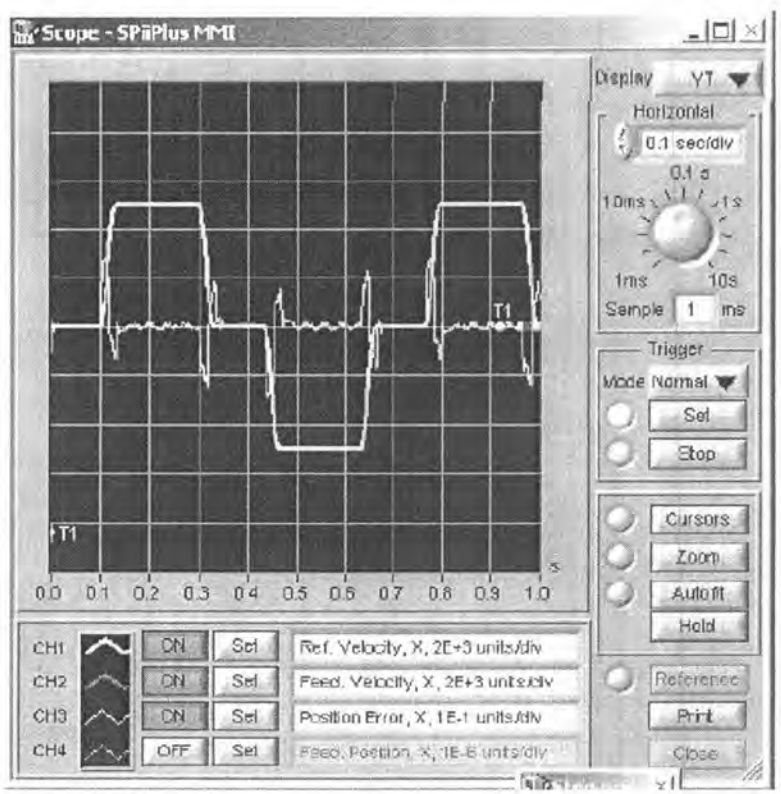
รูปที่ ข.13 หน้าต่าง Position Loop

ก่อนที่จะเริ่มปรับตั้งค่า ควรที่จะทราบความสำคัญของข้อมูลเหล่านี้ก่อน

- *First Point* และ *Second Point* เพื่อกำหนดตำแหน่งเริ่มต้นและสุดท้ายของการเคลื่อนที่
- *Velocity, Acceleration, Deceleration* และ *Jerk* เป็นค่าที่ใช้เฉพาะในการปรับตั้งค่าเท่านั้น ไม่ได้ใช้ในการเคลื่อนที่ปกติ
- *Dwell* คือเวลาที่ใช้สำหรับหยุดรอในแต่ละทิศทางการเคลื่อนที่

3. เลือก Overcurrent เพื่อให้ระบบทำการตรวจสอบความผิดพลาดที่จะเกิดขึ้น
4. ที่ตำแหน่งศูนย์ให้เลือก Zero และให้เลื่อนไปที่ตำแหน่งอื่นแล้วเลือก Read

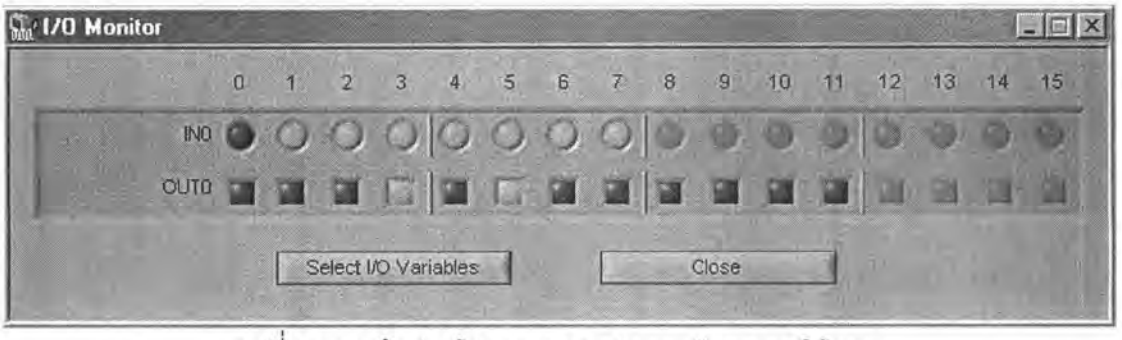
5. การกำหนดค่าของจุดแรกและจุดที่สองจะต้องมีระยะห่างพอสมควรที่มอเตอร์สามารถเคลื่อนที่โดยมีความเร็วคงที่ได้ ภายในขอบเขตที่กำหนด
6. ใส่ค่าตัวแปรต่าง ๆ สำหรับทดสอบ ทั้ง Velocity, Acceleration และ Deceleration ให้มีค่า 50% ของค่าสูงสุดที่กำหนดไว้ในการเคลื่อนที่ปกติ
7. เลือก Run เพื่อให้มอเตอร์เริ่มเคลื่อนที่
8. เลือก Scope Autoset เพื่อเลือกช่องสัญญาณของ RVEL, FVEL และ PE
9. เพิ่มค่า Position Gain เพื่อลดค่าตำแหน่งผิดพลาดขณะทำความเร่งและทำความหน่วง
10. เพิ่มค่า Acceleration Feedforward เพื่อลดค่าตำแหน่งผิดพลาดขณะทำความเร่งและทำความหน่วง โดยค่าทางทฤษฎีของ Acceleration Feedforward มีค่าดังสมการ ACC_FF=[2^16*10^7*EFAC/ACC]*0.1
11. เพื่อเป็นการปรับปรุงผลที่ได้จากการปรับตั้งค่า สามารถปรับเปลี่ยนค่าของ Velocity Gain, Integrator Gain หรือ Low Pass Filter Bandwidth



รูปที่ ข.14 ผลของ Position Loop Tuning

2.5 โปรแกรมส่งออกสัญญาณดิจิตอล

โปรแกรมสามารถควบคุมฟังก์ชั่นเสริมต่าง ๆ ได้ เช่น การเปิดและปิดน้ำหล่อเย็น การเปิดและปิดสัญญาณไฟเตือน การเคลื่อนที่เข้าและออกของจานเจียระไน โดยผ่านฟังก์ชั่นการส่งออกสัญญาณทางดิจิตอล ซึ่งสามารถส่งสัญญาณได้ถึง 8 สัญญาณ



รูปที่ ข.15 หน้าต่างโปรแกรมการส่งออกสัญญาณดิจิตอล

การสั่งคำสั่งส่งออกสัญญาณดิจิตอล สามารถสั่งผ่านโปรแกรมการเจียระไนพลอยโดยอัตโนมัติได้ ตัวอย่างเช่น

- เมื่อเริ่มทำการเจียระไน สั่งให้จานเจียระไนเคลื่อนที่เข้าและออก จนกระทั่งทำการเจียระไนเสร็จครบขั้นตอนแล้วจึงหยุด
- เมื่อทำการเจียระไนหยาบให้เปิดน้ำหล่อเย็น และเมื่อเสร็จขั้นตอนการเจียระไนหยาบแล้วให้ปิดน้ำหล่อเย็น
- เมื่อทำการเจียระไนเสร็จทุกขั้นตอนแล้ว ให้ส่งสัญญาณไฟกระพริบ เพื่อเป็นการส่งสัญญาณให้ผู้ใช้งานเข้ามานำพลอยที่เจียระไนเสร็จออกไปดำเนินการในขั้นตอนไป รวมถึงการใส่พลอยชุดใหม่เข้ามาดำเนินการต่อ

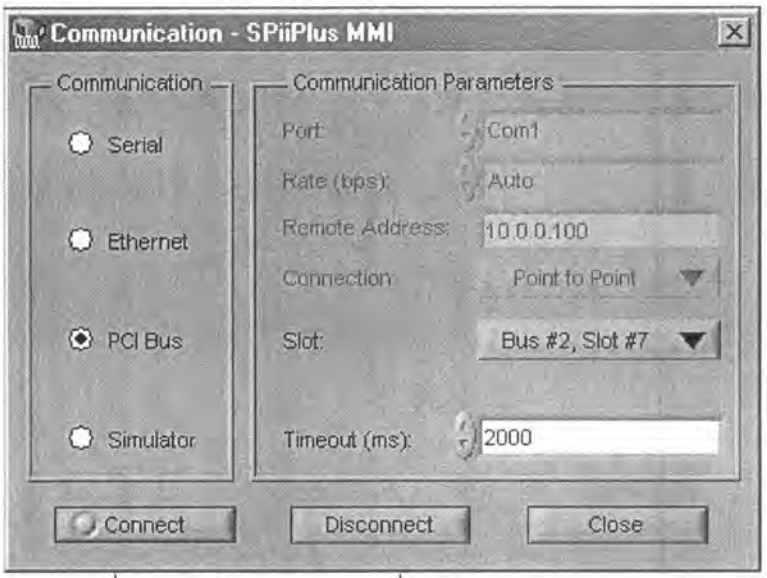
การแสดงค่าสัญญาณดิจิตอล จะแสดงโดยสัญลักษณ์ไฟสีเขียว ซึ่งหมายถึงอุปกรณ์นั้นกำลังดำเนินงานอยู่ โดยขั้นตอนในการเข้าสู่โปรแกรมตรวจสอบอุปกรณ์ดิจิตอลสามารถทำได้ ดังนี้

1. เปิดโปรแกรม SPiiPlus MMI แล้วเลือก I/O Monitor
2. สังเกตค่าสัญลักษณ์ของอุปกรณ์ต่าง ๆ เพื่อดูว่ามีอุปกรณ์ใดบ้างที่กำลังทำงานอยู่

2.6 ขั้นตอนการปิดการใช้โปรแกรม

เมื่อต้องการปิดการใช้โปรแกรม สามารถยกเลิกได้โดยการตัดการเชื่อมต่อของโปรแกรมโดยมีวิธีการดังต่อไปนี้

1. ปิดหน้าต่างทุกโปรแกรม ให้เหลือเพียงแต่หน้าต่างของ SPiiPlus MMI
2. เลือกคำสั่ง Communication



รูปที่ ข.16 หน้าต่างของการเชื่อมต่อ (Communication)

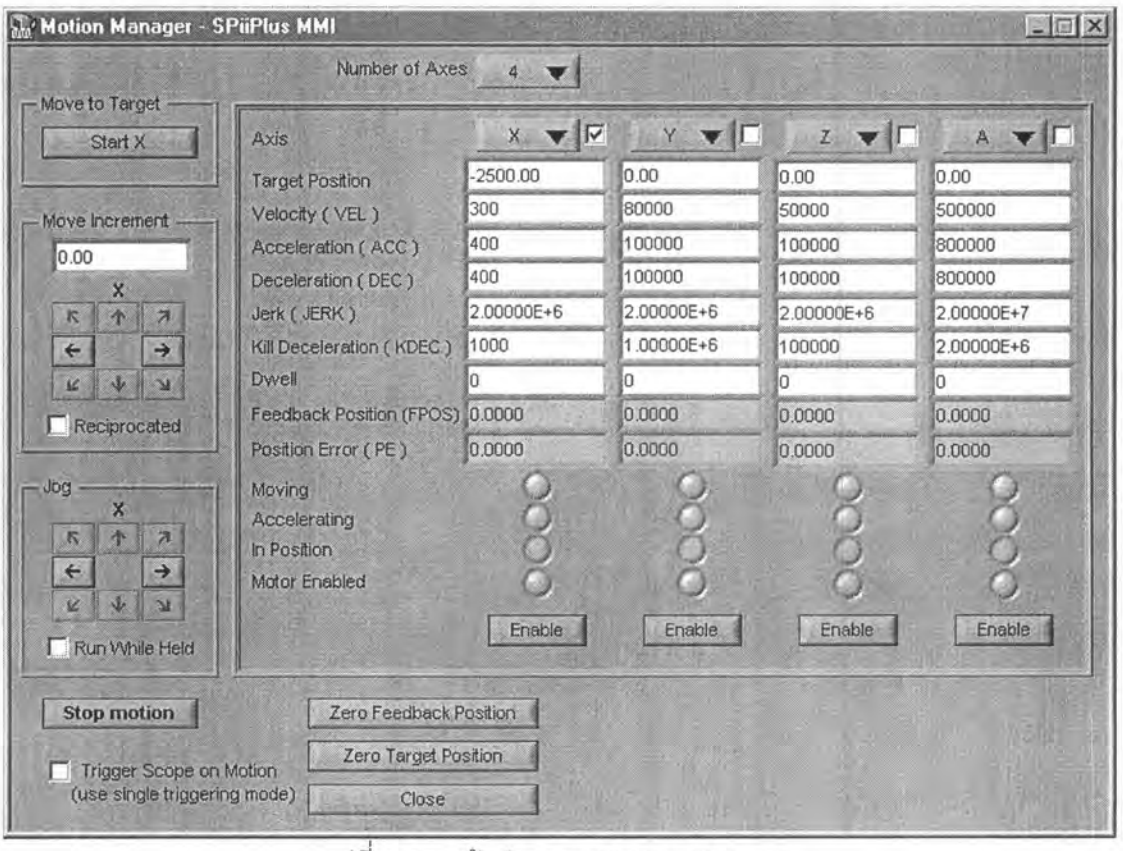
3. เลือกคำสั่ง Disconnect แล้วเลือกClose

ข.3 การใช้โปรแกรมควบคุมเครื่องเจียระไนพลอยอัตโนมัติ

ข.3.1 การควบคุมการเคลื่อนที่

ในส่วนของการเคลื่อนที่แบบไม่อัตโนมัติ สามารถควบคุมให้เคลื่อนที่ได้แบบอิสระในแต่ละแกน โดยจะเป็นการเคลื่อนที่ให้ถึงเป้าหมาย (Move to Target) การเคลื่อนที่เป็นระยะที่กำหนด (Move Increment) การเคลื่อนที่กลับไปกลับมา (Reciprocated) และการเคลื่อนที่ไปตลอดจนสั่งหยุด (Jog) ซึ่งขณะเคลื่อนที่ก็จะแสดงตำแหน่งปัจจุบัน (Feedback Position) พร้อมแสดงค่าผิดพลาด (Position Error) ตลอดการเคลื่อนที่ การเคลื่อนที่ในแต่ละแกนนั้นสามารถปรับได้ทั้งความเร็ว ความเร่ง ความหน่วง ได้อย่างอิสระในแต่ละแกน รวมถึงการตั้งค่าตำแหน่งศูนย์ (Zero Feedback Position) ซึ่งนำไปใช้ในการตั้งจุดอ้างอิงของการเคลื่อนที่ได้

1. เปิดโปรแกรม SPiiPlus MMI แล้วเลือก Motion Manager



รูปที่ ข.17 หน้าต่างของ Motion Manager

2. ที่ช่องของ Number of Axes ให้ใส่ค่าของจำนวนของแกนที่ให้แสดงค่าบนหน้าต่าง

รูปที่ ข.18 จำนวนแกนที่จะให้แสดงผล

3. ที่ช่องของ Axis ให้เลือกแกนที่จะแสดงผลบนหน้าต่างของโปรแกรม

รูปที่ ข.19 แกนที่เลือกใช้งาน

4. ใส่ค่าพารามิเตอร์ต่างที่ต้องการควบคุมลงในช่องของแต่ละแกนต่าง ๆ เช่น Velocity (VEL), Acceleration (ACC), Deceleration (DEC), Jerk (JERK), Kill deceleration (KDEC) และ Dwell

5. เลือก Enable สำหรับแกนที่จะให้มอเตอร์ทำงาน โดยเมื่อเลือก Enable แล้วจะมีสัญลักษณ์ไปสีเขียวแสดงอยู่
6. เลือกแกนที่จะต้องการทำงาน โดยทำเครื่องหมายถูกที่ช่องว่างหลังแกน
7. สำหรับการเคลื่อนที่ไปยังเป้าหมายที่กำหนด ให้ใส่ค่าตำแหน่งเป้าหมายลงในช่อง Target Position แล้วเลือกเครื่องหมาย Start

รูปที่ ข.20 หน้าต่าง Move to Target

8. สำหรับการเคลื่อนที่แบบเป็นช่วง ให้ใส่ค่าของช่วงการเคลื่อนที่ในช่องของ Move Increment เสร็จแล้วให้เลือกเครื่องหมายซ้ายเพื่อเคลื่อนที่ถอยหลัง หรือเลือกเครื่องหมายขวาเพื่อเคลื่อนที่ไปข้างหน้า หากต้องการให้เคลื่อนที่แบบกลับไปกลับมา ให้เลือก Reciprocated

รูปที่ ข.21 หน้าต่าง Move Increment

9. สำหรับการเคลื่อนที่แบบเคลื่อนที่ตลอดจนกระทั่งเจอคำสั่งหยุด ให้เลือกคำสั่งที่อยู่ในหมวดของ Jog โดยหากต้องการให้เคลื่อนที่ถอยหลังให้เลือกเครื่องหมายซ้าย หรือหากต้องการให้เคลื่อนที่ไปข้างหน้าให้เลือกเครื่องหมายขวา ถ้าคำสั่ง Run While Held ถูกเลือกอยู่หมายความว่า มอเตอร์จะหยุดหมุนเมื่อผู้ใช้ได้ทำการปล่อยเมาส์

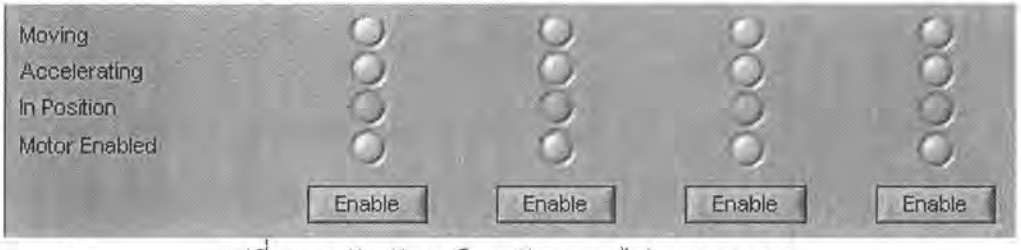
รูปที่ ข.22 หน้าต่าง Jog

10. หากต้องการหยุดการเคลื่อนที่ให้เลือกคำสั่ง Stop Motion
11. เมื่อต้องการให้มีการตั้งค่าอ้างอิงที่ตำแหน่งศูนย์ให้เลือก (Zero Feedback Position)

รูปที่ ข.23 หน้าต่าง Zero Feedback Position

สัญลักษณ์สัญญาณไฟสีเขียวที่แสดงบนหน้าต่างของ Motion Manager มีความหมายดังนี้

- Moving หมายความว่า มอเตอร์กำลังทำการเคลื่อนที่อยู่
- Accelerating หมายความว่า มอเตอร์กำลังอยู่ในช่วงของการเร่งความเร็ว
- In Position หมายความว่า มอเตอร์ได้เคลื่อนที่เข้าสู่เป้าหมายอย่างสมบูรณ์แล้ว
- Enabled หมายความว่า มอเตอร์อยู่ในสถานะพร้อมใช้งาน



รูปที่ ข.24 สัญลักษณ์ของสัญญาณไฟแสดงสถานะ

ข.3.2 การควบคุมการเคลื่อนที่แบบป้อนคำสั่งโดยตรง (Manual Data Input)

ในบางโอกาสเราจำเป็นที่จะต้องป้อนคำสั่งโดยตรง ให้เกิดการทำงานอย่างง่าย ๆ เพื่อเป็นการทดสอบคำสั่งบางอย่างที่ที่เราต้องการ โดยคำสั่งนั้นจะเป็นคำสั่งสั้น ๆ เช่น การสั่งให้มอเตอร์ทำงาน การเปิดจานเจียระไนให้หมุน เป็นต้น ในส่วนของโปรแกรมนี้จะช่วยให้เราทำงานได้สะดวกขึ้น สำหรับขั้นตอนการใช้การควบคุมการเคลื่อนที่แบบป้อนคำสั่งโดยตรง มีวิธีการดังนี้

1. เปิดโปรแกรม SPiiPlus MMI เลือก Terminal
2. ใส่คำสั่งที่ต้องการลงในช่องของ Command
3. ส่งคำสั่งโดยเลือกปุ่ม Send
4. ถ้าต้องการลบคำสั่งบนหน้าต่างให้เลือกปุ่ม Clear

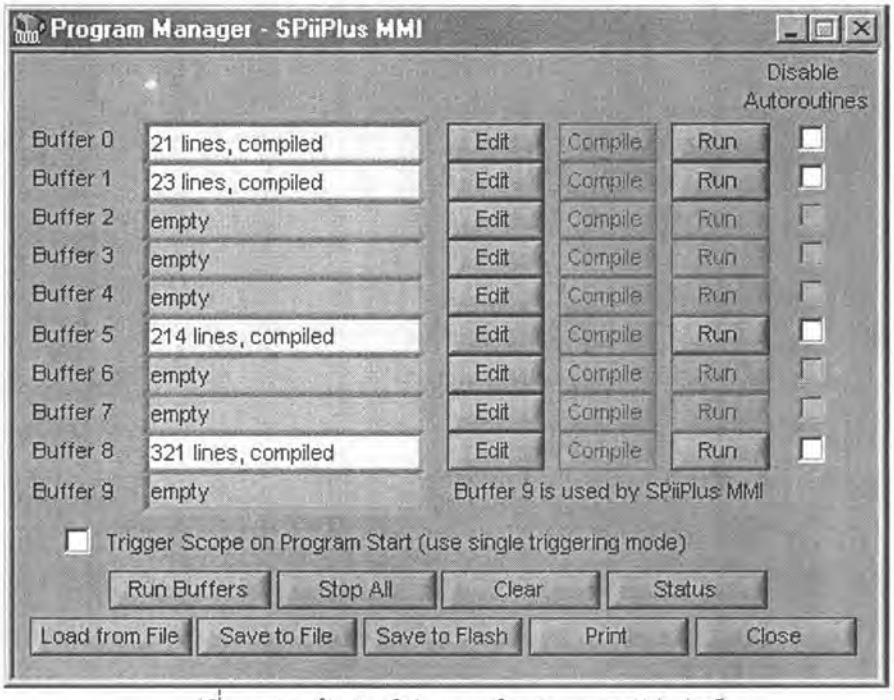
รูปที่ ข.25 หน้าต่างโปรแกรมป้อนคำสั่งโดยตรง

ข.3.3 การควบคุมการเคลื่อนที่โดยใช้โปรแกรมอัตโนมัติ

การเจียระไนพลอยโดยใช้เครื่องเจียระไนอัตโนมัติ จำเป็นที่จะต้องมีชุดโปรแกรมคำสั่งเพื่อที่จะให้เครื่องเจียระไนดำเนินการเจียระไนตามลำดับที่เราต้องการ เช่น โปรแกรมการเจียระไนด้านก้นพลอย (Pavilion) โปรแกรมการเจียระไนด้านหน้าพลอย (Crown) ดังนั้นในส่วนนี้จะเป็นส่วนที่มีความสำคัญมากในการกำหนดการเจียระไน

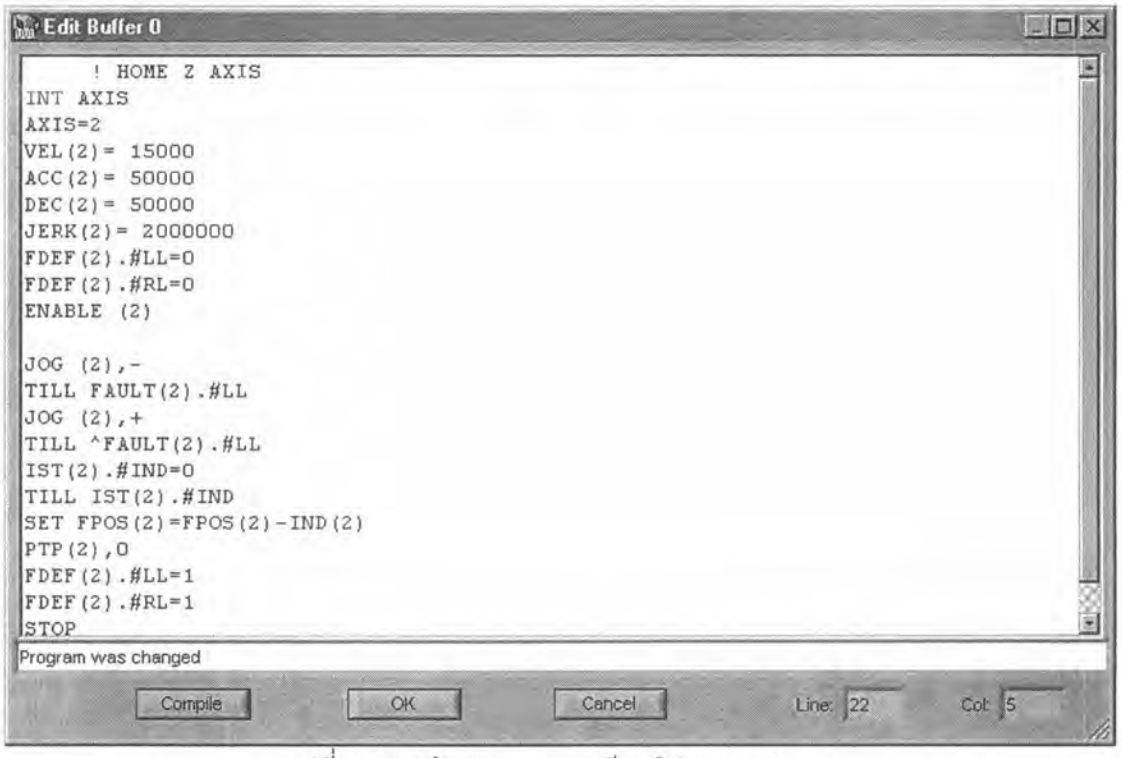
สำหรับการเขียนโปรแกรมการเจียระไนพลอยนั้น สามารถทำได้ไม่ยาก โดยใช้การเขียนแบบเป็นลำดับการทำงาน สามารถใช้การเขียนแบบวนลูป เพื่อลดความยาวในการเขียนโปรแกรม

การเขียนโปรแกรมนั้นจะเขียนอยู่ในส่วนของบัฟเฟอร์ เพื่อที่จะพร้อมในการเรียกใช้งาน โดยสามารถเก็บโปรแกรมได้ถึง 9 บัฟเฟอร์ ซึ่งหลังจากที่ได้ทำการเขียนโปรแกรมไว้แล้ว สามารถแก้ไขโปรแกรมได้ตามที่ต้องการ และยังสามารถที่จะทำการจัดเก็บโปรแกรมไว้ในรูปแบบเอกสารที่พร้อมใช้งานได้ในอนาคต โดยมีวิธีใช้ดังนี้



รูปที่ ข.26 หน้าต่างโปรแกรมในส่วนของบัฟเฟอร์

1. เปิดโปรแกรม SPiiPlus MMI แล้วเลือก Program Manager
2. การใส่และแก้คำสั่งของโปรแกรมทำได้โดยการเลือก Edit หลังจากนั้นให้เลือก Compile
3. เมื่อต้องการใช้งานโปรแกรมที่ได้เขียนไว้ ให้เลือก Run
4. สามารถบันทึกโปรแกรมที่เขียนไว้ โดยใช้คำสั่ง Save to File
5. เมื่อต้องการนำโปรแกรมที่บันทึกเอาไว้ออกมาใช้ ให้เลือกคำสั่ง Load from File



รูปที่ ข.27 หน้าต่างของการเขียนโปรแกรม (Edit)

ข.4 ภาษาเบื้องต้นสำหรับโปรแกรมควบคุมเครื่องเจียระไนพลอยอัตโนมัติ

ข.4.1การประกาศตัวแปร

ในการเขียนโปรแกรมทั่วไปจำเป็นที่จะต้องใช้ตัวแปรเข้ามาเกี่ยวข้อง แต่ตัวแปรที่จะกำหนดขึ้นมาอาจจะสามารถใช้ได้และใช้ไม่ได้ เพราะว่าได้มีการจองตัวแปรนั้น ๆ เป็นที่เรียบร้อยแล้ว

abs	ceil	edge	getdsp	jog
acos	connect	else	getdspini	kill
all	cos	elseif	global	killall
arc1	dc	enable	go	ldexp
arc2	deadzone	enableon	goto	lag
asin	disable	end	group	let
atan	disableall	ends	halt	line
atan2	disableon	extin	hypot	local
avg	disp	extout	if	log
binp	do	exp	imm	log10
break	dsign	floor	int	loop
call	dynamic	getsp	intgr	map
mapby1	mref	read	setdsp	stopall
mapby2	mseg	real	setdspini	stopdc
map2	on	resume	sign	stopper
map2free	path	ret	sin	tan
master	pause	roll	slave	till
max	peg	sat	split	vsp
min	point	save	splitall	wait
monsp	pow	send	sqrt	while
mpoint	projection	set	start	write
mptp	ptp	setsp	stop	

รูปที่ ข.28 ตัวแปรที่ถูกจองชื่อไว้แล้ว

ข.4.1.1 Case Sensitivity

นอกจากตัวแปรที่ถูกจองชื่อไว้แล้ว ภาษาที่ใช้สำหรับโปรแกรมการเจียระไนพลอยยังเป็นประเภท Case sensitive คือ ตัวพิมพ์ใหญ่กับตัวพิมพ์เล็ก จะมีความหมายแตกต่างกันเช่น step_z และ step_Z ต่างก็มีความหมายต่างกัน

ข.4.1.2 การประกาศการใช้แกนการเคลื่อนที่

ภาษาที่ใช้ในการเจียระไนพลอยนั้นจะมีการประกาศการใช้แกนการเคลื่อนที่ ซึ่งสามารถประกาศการใช้ได้ 2 แบบ คือ การประกาศด้วยตัวอักษร เช่น X, XYZ, ZAD เป็นต้น หรือจะประกาศการใช้ด้วยตัวเลข เช่น (0, 2, 4) เทียบได้กับ XZA ซึ่งตัวเลขจะทำหน้าที่แทนตัวแปรของแต่ละแกน

ตารางที่ ข.1 ความสัมพันธ์ของชื่อแกนกับตัวเลข

ตัวแปร	X	Y	Z	T	A	B	C	D
ตัวเลข	0	1	2	3	4	5	6	7

ข.4.1.3 การแทรกข้อความ (Comment)

การแทรกข้อความลงในโปรแกรมสามารถช่วยให้ผู้ใช้สามารถอ่านโปรแกรมได้สะดวกมากยิ่งขึ้น แต่ไม่ต้องการให้โปรแกรมเข้ามาทำงานในส่วนที่ผู้ใช้ได้ใส่ข้อความเอาไว้ สามารถทำได้ใส่เครื่องหมาย ! ไว้ข้างหน้าของข้อความที่แสดงความคิดเห็น ซึ่งหมายความว่าเป็นข้อความที่แสดงความคิดเห็นทั้งบรรทัด เช่น

```
ptp x, 100        ! Move X axis 100 pulses
```

ข.4.1.4 ชนิดของข้อมูล int และ real

ในการเขียนโปรแกรม ผู้ใช้จำเป็นต้องรู้ถึงชนิดของข้อมูลต่าง ๆ ก่อนเสมอ เพราะว่า เมื่อโปรแกรมประมวลผลทำงาน จะต้องใช้หน่วยความจำในการเก็บข้อมูล ซึ่งจะต้องใช้ตัวแปร และตัวแปรก็จะถูกประกาศเป็นชนิดข้อมูลต่าง ๆ ถ้าการประกาศตัวแปรไม่เหมาะสมต่อการใช้งาน ก็จะทำให้ โปรแกรมทำงานผิดพลาด และอาจจะทำให้โปรแกรมใช้ทรัพยากรของเครื่องคอมพิวเตอร์มากเกินไป ดังนั้นการเลือกใช้ชนิดข้อมูลจึงเป็นสิ่งที่จำเป็น

ชนิดข้อมูล int ใช้เก็บเลขจำนวนเต็ม ซึ่งจะเก็บในหน่วยของขนาดความจำ 4 ไบท์ หรือ 32 บิท

ชนิดข้อมูล real ใช้เก็บเลขจำนวนจริง ซึ่งจะเก็บในหน่วยของขนาดความจำ 8 ไบท์ หรือ ใช้เก็บตัวเลย 52 บิท และเก็บเลขยกกำลัง 12 บิท

ตัวอย่างการประกาศการใช้ชนิดข้อมูล

```
int Var1! Declare Var1 integer variable
real Var2         ! Declare Var2 real variable
Var1 = Var2       ! Controller automatically converts real to integer
```

ข.4.2 การใช้ฟังก์ชัน (Function)

ในการเขียนโปรแกรมควบคุมจำเป็นที่จะต้องมีการใช้ฟังก์ชัน เพื่อใช้ในการคำนวณค่าต่าง ๆ ฟังก์ชันที่ใช้อาจจะเป็นไปได้ทั้งฟังก์ชั่นการคำนวณตัวเลข ฟังก์ชันทางสถิติ เป็นต้น

ตารางที่ ข.2 Arithmetic Function

ARITHMETICAL FUNCTIONS	
NAME	ACTION
abs	Calculates the absolute value
acos	Calculates the arccosine
asin	Calculates the arcsine
atan	Calculates the arctangent
atan2	Calculates the arctangent of Y/X
eil	Calculates the ceiling of a value
cos	Calculates the cosine
exp	Calculates the exponential
floor	Calculates the floor of a value
hypot	Calculates the hypotenuse
idexp	Calculates a value of x*2exp
log	Calculates the natural logarithm
log10	Calculates the base-10 logarithm
pow	Calculates x raised to the power of y
sin	Calculates the sine
sqrt	Calculates the square root
tan	Calculates the tangent
sign	Returns -1, 0 or 1 depending on the sign of x

ตารางที่ ข.3 Statistical Function

STATISTICAL FUNCTIONS	
NAME	ACTION
min	Finds the minimal value in an array
max	Finds the maximal value in an array
avg	Finds the average of all values in an array

ตารางที่ ข.4 Signal Processing Function

SIGNAL PROCESSING FUNCTIONS	
NAME	ACTION
deadzone	Implements dead-zone routine
dsign	Implements a dynamic version of the standard **sign** function.
edge	Returns 1 on positive edge of x
intgr	Implements an integrator with deadzone and saturation.
lag	Provides delayed switching on argument change (anti-bouncing effect)
map	Implements a table-defined function with constant step
mapby1	Implements a table-defined function with variable step.
mapby2	Implements a table-defined function with variable step.
map2	Implements a table-defined function with two arguments and constant step along each argument.
map2free	Implements a table-defined function with two arguments and variable step along each argument.
roll	Calculates a result rolled-over to within one pitch.
sat	Implements a saturation characteristic

ตารางที่ ข.5 Servo Processor Function

SERVO PROCESSOR FUNCTIONS	
NAME	ACTION
getsp	Reads a value from the specified SP address
setsp	Writes a value to the specified SP address
monosp	Initiates monitoring of the specified SP address through analog output

ตารางที่ ข.6 Miscellaneous Function

MISCELLANEOUS FUNCTIONS	
NAME	ACTION
sysinfo	Returns a value containing system information.
input	Receives numerical values from a communication channel and assigns them to variables.
getconf	Reads hardware and firmware parameters.
setconf	Writes hardware and firmware parameters.

ข.4.3 การใช้คำสั่งเคลื่อนที่ (Motion)

การใช้คำสั่งเคลื่อนที่ถือว่าเป็นคำสั่งที่มีความสำคัญเป็นอย่างมาก เพราะการเจียระไนพลอยจะเกิดขึ้นได้ จะต้องเกิดจากการเคลื่อนที่เพื่อทำการเจียระไน โดยการเคลื่อนที่ในเบื้องต้นของเครื่องเจียระไนพลอยจะควบคุมทั้งหมด 4 แกน คือ แกนการเคลื่อนที่ขึ้นและลง แกนการเคลื่อนที่ทำมุมในการเจียระไน แกนการเคลื่อนที่เพื่อเปลี่ยนหน้าเหลี่ยมของพลอย และแกนการเคลื่อนที่เพื่อเปลี่ยนจานเจียระไรทั้ง 3 จาน

การเคลื่อนที่สามารถทำได้หลายวิธี เช่น การเคลื่อนที่ไปยังตำแหน่งที่กำหนด (Point to Point) หรือการเคลื่อนที่ไปเรื่อย ๆ (Jog) ซึ่งผู้ใช้สามารถที่จะเสริมรายละเอียดต่าง ๆ ให้กับการเคลื่อนที่ได้ เช่น การกำหนดความเร็วเฉพาะสำหรับการเคลื่อนที่นั้น ๆ

ข.4.3.1 การเคลื่อนที่ไปยังตำแหน่งที่กำหนด (Point to Point Motion)

ตารางที่ ข.7 รายละเอียดคำสั่ง ptp

คำสั่ง	ptp
คำต่อท้าย	e - รอจนกระทั่งการเคลื่อนที่เป็นไปอย่างสมบูรณ์จึงเริ่มคำสั่งใหม่ f - กำหนดค่าความเร็วสุดท้ายไม่ให้เป็นศูนย์ m – ใช้การข้อมูลการเคลื่อนที่ที่มีค่าสูงสุดในกลุ่มของแกน โดยมีความสำคัญมากกว่าลำดับของแกน r – เป็นการเคลื่อนที่แบบสัมพัทธ์ v – กำหนดความเร็วเฉพาะแทนที่จะใช้ความเร็วเดิม w – สร้างการเคลื่อนที่แต่ยังไม่กระทำจนกระทั่งได้รับคำสั่ง go

ตัวอย่างเช่น

ptp X, 1000 หมายถึงการเคลื่อนที่ในแกน X ไปยังตำแหน่งสัมบูรณ์ 1000 ทันที แต่ถ้ามีการเติม e ต่อท้ายคำสั่ง ระบบควบคุมจะหยุดรอจนกระทั่งการเคลื่อนที่ก่อนหน้าจะเคลื่อนที่เสร็จสมบูรณ์แล้วจึงเริ่มเคลื่อนที่ ยกตัวอย่างเช่น

ptp/e Y,1000

มีความหมายเทียบเท่ากับ

Ptp Y, 1000

Till ^Y_AST.#MOVE

การเติม w ต่อท้ายทำสั่งจะช่วยป้องกันไม่ให้การเคลื่อนที่เริ่มขึ้น แม้ว่าระบบจะเตรียมพร้อมแล้ว ยกตัวอย่างเช่น

ptp/w x, 1000

ระบบจะเตรียมการเคลื่อนที่ไปยังตำแหน่ง 1000 แต่จะยังไม่เริ่มเคลื่อนที่จนกระทั่งจะได้รับคำสั่ง go X

ในสองตัวอย่างที่กล่าวมาค่า 1000 เป็นค่าตำแหน่งสัมบูรณ์ ถ้าผู้ใช้เติม r ต่อท้ายหนังคำสั่งจะเป็นการใช้ตำแหน่งสัมพัทธ์ ยกตัวอย่างเช่น

ptp X, 1000

ptp/r X, 1000

ได้ผลลัพธ์เหมือนกับ

ptp X, 1000

ptp X, 2000

ตัวอย่างทั้งหมดที่กล่าวมาเบื้องต้น การเคลื่อนที่จะใช้ความเร็วตามค่าที่กำหนดไว้ใน X_VEL ของแต่ละแกนนั้น ๆ ในกรณีที่ผู้ใช้ต้องการกำหนดความเร็วเฉพาะ ให้ผู้ใช้เติม v ไว้หลังคำสั่งแล้วตามด้วยความเร็วที่กำหนด ยกตัวอย่างเช่น

ptp/v X, 1000, 15000

การเคลื่อนที่ที่เกิดขึ้นจะไม่สนใจค่าความเร็ว X_VEL แต่ใช้ความเร็ว 150000 เพื่อเคลื่อนที่ไปยังตำแหน่ง 1000 และค่าของ X_VEL ก็ยังไม่เปลี่ยนแปลง

การใช้คำสั่งต่อท้ายสามารถที่จะใช้รวมกันได้พร้อมกันหลายคำสั่ง ยกตัวอย่างเช่น

ptp/rv X, 1000, 15000

หมายถึงการเคลื่อนที่แบบสัมพัทธ์ไปตำแหน่งถัดไป 1000 และใช้ความเร็ว 15000

ข.4.3.2 การเคลื่อนที่ไปยังตำแหน่งที่กำหนดแบบหลายแกน (Multi-Axis Point to Point Motion)

หัวข้อที่กล่าวมาเบื้องต้นนั้นเป็นการเคลื่อนที่แบบทีละแกน อย่างไรก็ตามการใช้คำสั่งการเคลื่อนที่ ptp สามารถที่จะใช้การกับแกนเดียวหรือใช้พร้อมกันหลายแกนได้ ยกตัวอย่างเช่น

ptp XYT, 1000, 2000, 3000

การเคลื่อนที่พร้อมกันหลายแกนนั้น การใช้ค่าพารามิเตอร์ในการเคลื่อนที่นั้นจะมีความแตกต่างกันสำหรับแต่ละแกน โดยปกติแล้วระบบควบคุมจะใช้ค่าพารามิเตอร์ของแกนที่อยู่ที่ตัวอักษรตัวแรกสุด อย่างไรก็ตามผู้ใช้สามารถค่าพารามิเตอร์ที่มีค่าสูงสุดโดยการเติม m ต่อท้าย ยกตัวอย่างเช่น

Ptp/m XY, 1000, 2000

ข.4.3.3 การเคลื่อนที่แบบ Jog (Jogging Motion)

การเคลื่อนที่แบบ jog นั้นจะเป็นการเคลื่อนที่ด้วยความเร็วคงที่ แต่ไม่มีการระบุตำแหน่งปลายเอาไว้ การเคลื่อนที่จะดำเนินต่อไปเรื่อย ๆ จนกระทั่งได้รับคำสั่งให้หยุด หรืออาจจะหยุดเมื่อการเคลื่อนที่ไม่สมบูรณ์ เช่น เลยออกนอกขอบเขต

ตารางที่ ข.8 รายละเอียดคำสั่ง jog

คำสั่ง	jog
คำต่อท้าย	v – กำหนดความเร็วเฉพาะแทนที่จะใช้ความเร็วเดิม w – สร้างการเคลื่อนที่แต่ยังไม่กระทำจนกระทั่งได้รับคำสั่ง go

ยกตัวอย่างเช่น

jog X

เมื่อได้รับคำสั่งแล้ว ระบบจะทำการเคลื่อนที่แกน X ไปในทิศทางบวก และใช้ความเร็ว X_VEL

jog X, -

เมื่อได้รับคำสั่งแล้ว ระบบจะทำการเคลื่อนที่แกน X ไปในทิศทางลบ และใช้ความเร็ว X_VEL

โดยคำสั่ง

jog X

มีความหมายเทียบเท่ากับ

jog X, +

การเติม v ต่อท้ายจะเป็นการกำหนดความเร็วที่ใช้เฉพาะการเคลื่อนที่นั้น ๆ เช่น

jog/v X, 30000

การเคลื่อนที่ด้วยคำสั่ง jog ถูกยกเลิกเมื่อได้รับคำสั่ง halt, kill หรือ break อย่างไรก็ตาม คำสั่ง jog สามารถถูกยกเลิกด้วยคำสั่งการเคลื่อนที่ถัดไป ยกตัวอย่างเช่น

jog X, +

wait 500

jog X, -

ระบบจะทำการเคลื่อนที่แบบ jog ที่แกน X ไปในทิศทางบวก เป็นระยะเวลา 500 msec แล้วจะทำการกลับทิศการเคลื่อนที่ให้ไปในทิศทางลบ

ข.4.3.4 การเคลื่อนที่หลายแกนแบบ Jog (Multi-Axis Jogging Motion)

การใช้คำสั่ง jog สามารถทำพร้อมกันได้หลายแกน ยกตัวอย่างเช่น

jog XYZ, -++

ระบบจะทำการเคลื่อนที่ 3 แกน คือ เคลื่อนที่แกน X ไปในทิศทางลบ และเคลื่อนที่แกน Y ไปในทิศทางบวก โดยการเคลื่อนที่ที่เกิดขึ้นนั้นจะใช้ความเร็วของ X_VEL ในการเคลื่อนที่ของทั้ง 3 แกน

ข.4.4 การใช้คำสั่งโปรแกรมส่งออกสัญญาณดิจิตอล (Digital Output)

การใช้โปรแกรมส่งออกสัญญาณสามารถใช้ควบคุมฟังก์ชั่นเสริมต่าง ๆ ได้ เช่น การเปิดและปิดน้ำหล่อเย็น การเปิดและปิดสัญญาณไฟเตือน การเคลื่อนที่เข้าและออกของจาน

เจียระไน โดยผ่านฟังก์ชันการส่งออกสัญญาณทางดิจิตอล ซึ่งสามารถส่งสัญญาณได้ถึง 8 สัญญาณ

โดยการใช้โปรแกรมส่งออกสัญญาณดิจิตอลสามารถใช้ได้โดยการส่งสัญญาณของสถานะไปให้กับอุปกรณ์นั้น ๆ ซึ่งการส่งสถานะ 1 หมายถึงสถานะเปิด และการส่งสถานะ 0 หมายถึงสถานะปิด

ยกตัวอย่างเช่น ให้ตำแหน่งของปั๊มน้ำหล่อเย็นอยู่ที่ 3

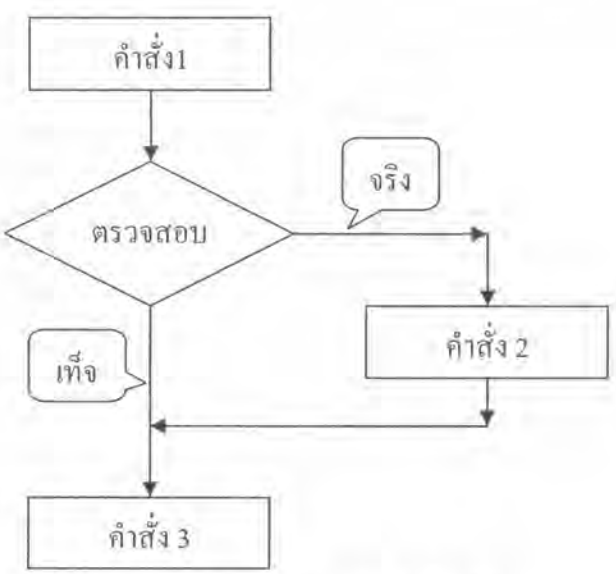
OUT0.3 = 1 หมายถึงทำการเปิดปั๊มน้ำ

OUT0.3 = 0 หมายถึงทำการปิดปั๊มน้ำ

ข.4.5 การใช้คำสั่งเงื่อนไขและคำสั่งวนรอบ (Condition and Loop Statement)

ในหัวข้อนี้จะอธิบายถึงคำสั่งเงื่อนไขและวนรอบ ซึ่งเป็นคำสั่งที่ใช้ตัดสินใจว่าจะทำหรือไม่ทำ วนรอบอีก หรือไม่วนรอบ และเป็นส่วนที่สำคัญในการเขียนโปรแกรมการเจียระไนพลอย เช่นการเจียระไนโดยให้ทำการวนรอบของเหลี่ยมให้ครบ 16 เหลี่ยม เป็นต้น นอกจากนี้ยังต้องใช้คำสั่งในการเพิ่มค่า (Increment) มาใช้ร่วมกับคำสั่งเงื่อนไขและวนรอบ

ข.4.5.1คำสั่งเงื่อนไข (Condition Statement)



รูปที่ ข.29 ผังงาน (Flow Chart) ของคำสั่งเงื่อนไข

คำสั่งนี้ใช้เพื่อตัดสินใจว่า จะเลือกทางใดทางหนึ่ง ในการตัดสินใจว่าจะเลือกทางใดทางหนึ่งนั้น จะขึ้นอยู่กับผลลัพธ์ที่เป็นบูลลีน (Boolean) ซึ่งจะมีเฉพาะค่าจริงหรือเท็จ ถ้าจริงก็จะเลือกทำทางเลือกหนึ่ง ถ้าไม่จริงก็จะตัดสินใจทำอีกทางเลือกหนึ่ง คำสั่งที่อยู่ในคำสั่งเงื่อนไขได้แก่ if, elseif, else, end โดยมีโครงสร้างดังนี้

" If เงื่อนไข คำสั่ง end "

" If เงื่อนไข คำสั่ง else คำสั่ง end "

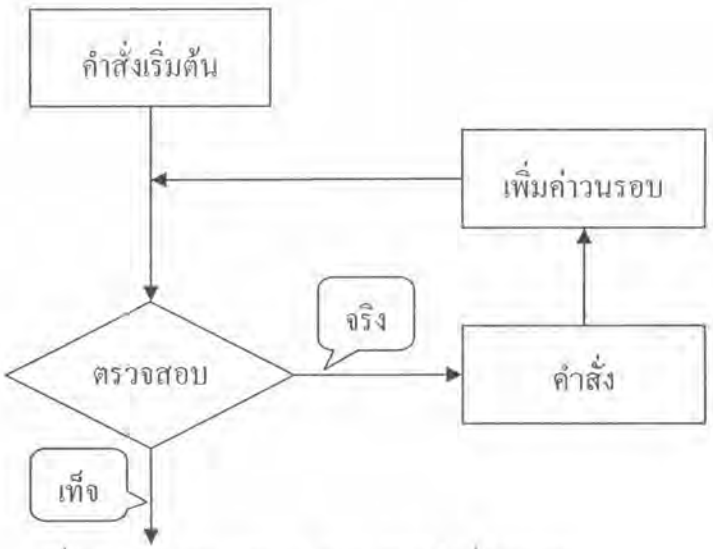
" If เงื่อนไข คำสั่ง elseif เงื่อนไขต่อไป คำสั่ง end "

การทำงานของคำสั่งจะมีขั้นตอนดังต่อไปนี้

1. ทำการตรวจสอบเงื่อนไขก่อนเป็นอันดับแรก โดยดูค่าผลลัพธ์ซึ่งสามารถเป็นไปได้ 2 กรณี คือ
   - ถ้าผลลัพธ์ไม่เป็น 0 หมายถึงเงื่อนไขนั้นเป็นจริง
   - ถ้าผลลัพธ์เป็น 0 หมายถึงเงื่อนไขนั้นเป็นเท็จ
2. ถ้าผลลัพธ์ไม่เป็น 0 (จริง) ให้ทำคำสั่งต่อไป
3. ถ้าผลลัพธ์เป็น 0 (เท็จ) ให้ข้ามคำสั่งไป
4. ถ้าโปรแกรมนั้นมีหลายเงื่อนไขให้ตรวจสอบเงื่อนไขต่อไป แล้วทำตามข้อ 2 หรือข้อ 3 จนกระทั่งเจอคำสั่ง end

ข.4.5.2 คำสั่งวนรอบ (Loop Condition)

การใช้คำสั่งวนรอบ จะวนรอบหรือไม่วนรอบหรือไม จะขึ้นอยู่กับผลลัพธ์ที่เป็นบูลลีน ถ้าเป็นจริง ก็จะวนรอบ แต่ถ้าเป็นเท็จ ก็จะออกจากการวนรอบทันที ในโปรแกรมการเจียระไนพลอย สามารถนำมาใช้โดยให้ทำการวนรอบของเหลี่ยมให้ครบ 16 เหลี่ยม แล้วจึงออกจากการใช้คำสั่งวนรอบ คำสั่งวนรอบที่ใช้ ได้แก่ คำสั่ง loop



รูปที่ ข.30 ผังงาน (Flow Chart) ของคำสั่งวนรอบ

สำหรับคำสั่ง loop นั้นมีโครงสร้างดังต่อไปนี้

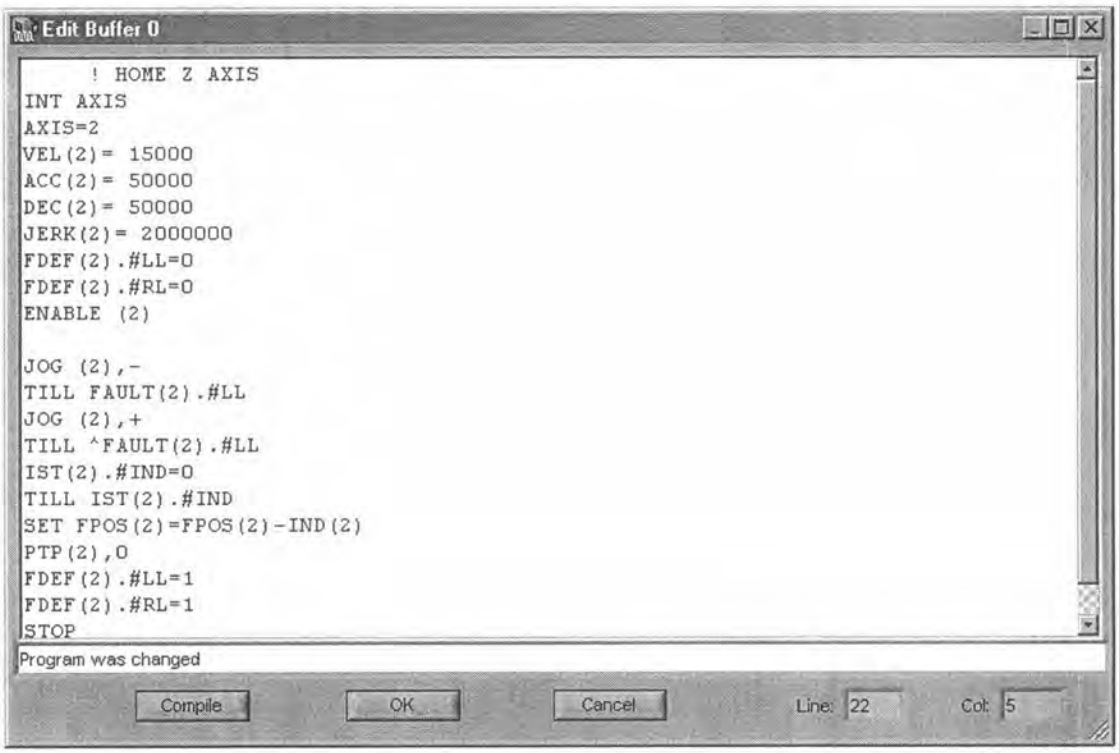
" Loop เงื่อนไข คำสั่ง end "

โดยมีขั้นตอนในการใช้งาน ดังนี้

1. คำนวณเงื่อนไข
2. ถ้าผลของเงื่อนไขที่ได้มีค่าเป็น 0 หรือมีค่าติดลบ คำสั่งต่อมาจะไม่ถูกกระทำ และให้ข้ามไปที่คำสั่ง end เพื่อออกจากการวนรอบ
3. ถ้าผลลัพธ์ที่ได้ไม่เป็นจำนวนเต็ม ให้ทำการปัดค่าไปยังค่าจำนวนเต็มที่ใกล้สุด
4. ถ้าผลของเงื่อนไขที่ได้มีค่าเป็นบวก ให้ทำตามคำสั่งเป็นจำนวนครั้งตามที่ได้จากเงื่อนไข
5. ทำตามคำสั่งจนกระทั่งครบจำนวนตามเงื่อนไข และไปที่คำสั่ง end เพื่อออกจากการวนรอบ

ข.4.6 ตัวอย่างโปรแกรม

หลังจากได้ทราบถึงวิธีการเขียนโปรแกรมเบื้องต้นแล้ว ในหัวข้อนี้จะแสดงถึงตัวอย่างโปรแกรมที่ใช้ในเครื่องเจียระไนพลอย ซึ่งเป็นโปรแกรมของการกลับจุดอ้างอิงของแกนแนวดิ่ง



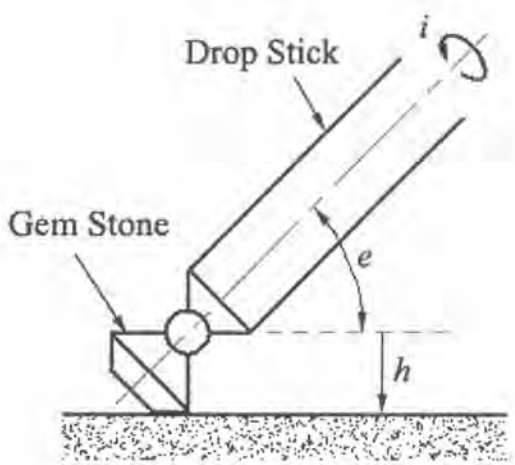
รูปที่ ข.31 ตัวอย่างโปรแกรมการกลับจุดอ้างอิง (Home Position)

ภาคผนวก ค

การใช้โปรแกรม CU-Solid ช่วยในการเจียระไน

การเจียระไนพลอยด้วยเครื่องเจียระไนอัตโนมัติ สามารถที่จะเจียระไนพลอยได้หลายรูปแบบ ซึ่งการที่จะเจียระไนได้สะดวกนั้น จำเป็นที่จะต้องใช้โปรแกรมอื่นเข้ามาช่วย เพื่อที่จะเป็นการช่วยกำหนดระยะความลึกในการเจียระไน รวมถึงมุมที่ใช้ในการเจียระไนต่าง ๆ โปรแกรม CU-Solid ที่ได้พัฒนาขึ้นมาควบคู่ด้วยนั้น มีส่วนเสริมที่สามารถช่วยให้ผู้ใช้สามารถออกแบบรูปแบบของพลอย รวมถึงการนำมาประยุกต์ในการหาระยะความลึกในการเจียระไนในแต่ละหน้าของพลอย

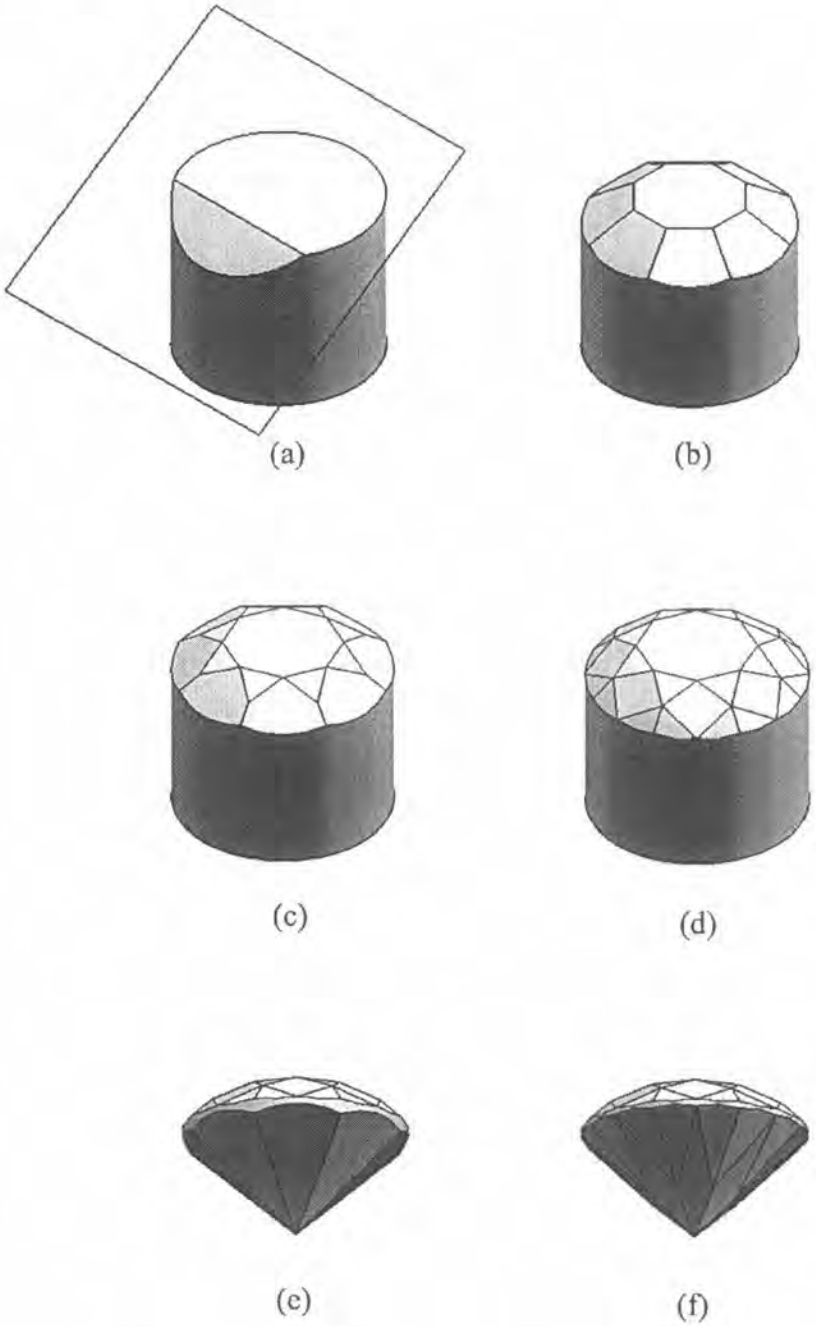
การเจียระไนพลอยด้วยเครื่องเจียระไนแบบอัตโนมัติจะเริ่มจากการนำพลอยที่ผ่านการแต่งรูปทรงเบื้องต้น (Pre-shape) มาเจียระไนเกิดเป็นระนาบขึ้นบนพลอย โดยเครื่องเจียระไนแบบอัตโนมัติที่สร้างขึ้น จะอาศัยพารามิเตอร์ 3 ตัวในการเจียระไนคือ มุมยก (Elevation angle: e), มุมดรรชนี (Index angle: i) และความลึกในการกัด (Depth of cut: h) นั่นคือพารามิเตอร์ทั้ง 3 เป็นตัวกำหนดระนาบที่เกิดขึ้นบนพลอย ดังแสดงในรูปที่ ค.1



รูปที่ ค.1 พารามิเตอร์ในการเจียระไนพลอย

สำหรับซอฟแวร์ CU-Solid นี้ ได้แนวคิดในการสร้างโมเดลพลอยมาจากกระบวนการข้างต้น คือ ในขั้นแรกจะกำหนดรูปทรงที่มีลักษณะใกล้เคียงกับพลอยที่ผ่านการแต่งรูปทรงเบื้องต้น (Pre-shape) จากนั้นกำหนดค่าพารามิเตอร์ มุมยก มุมดรรชนี และความลึกในการ

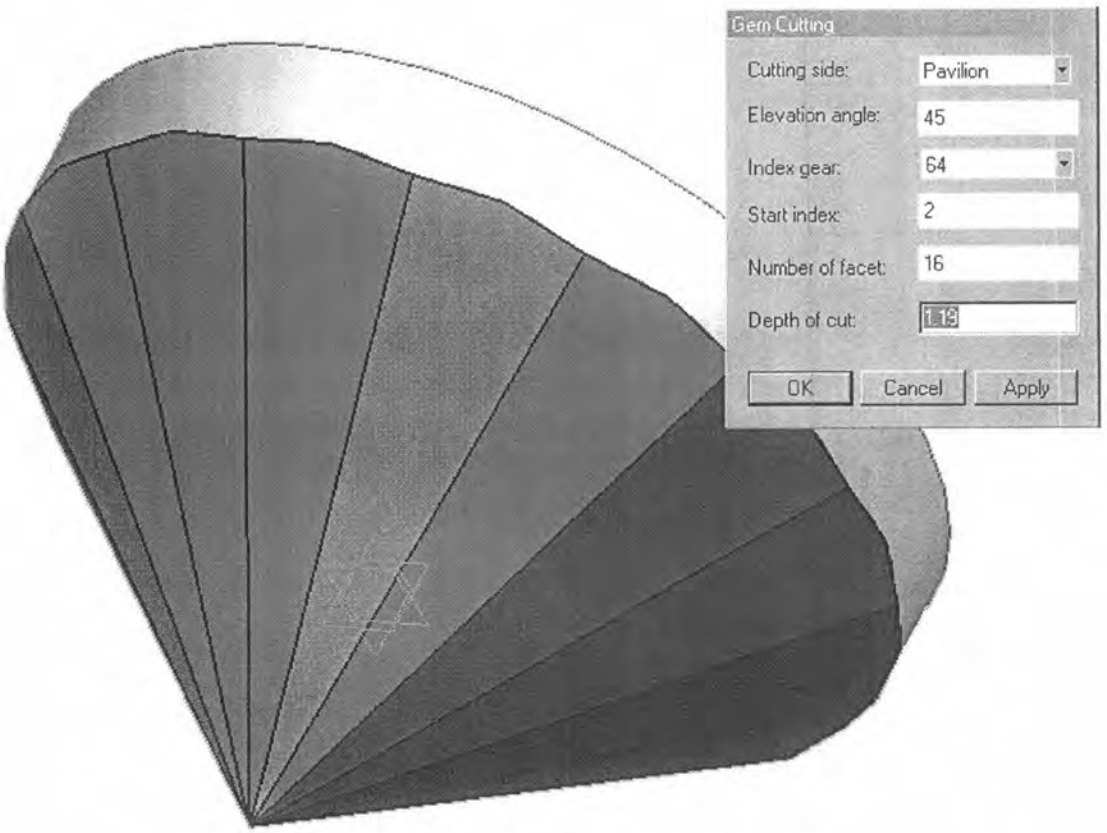
กัด เพื่อคำนวณหาระนาบที่จะนำมาตัดกับรูปทรงเบื้องต้นที่สร้างไว้ในขั้นแรกเกิดเป็นระนาบบนโมเดลของพลอย



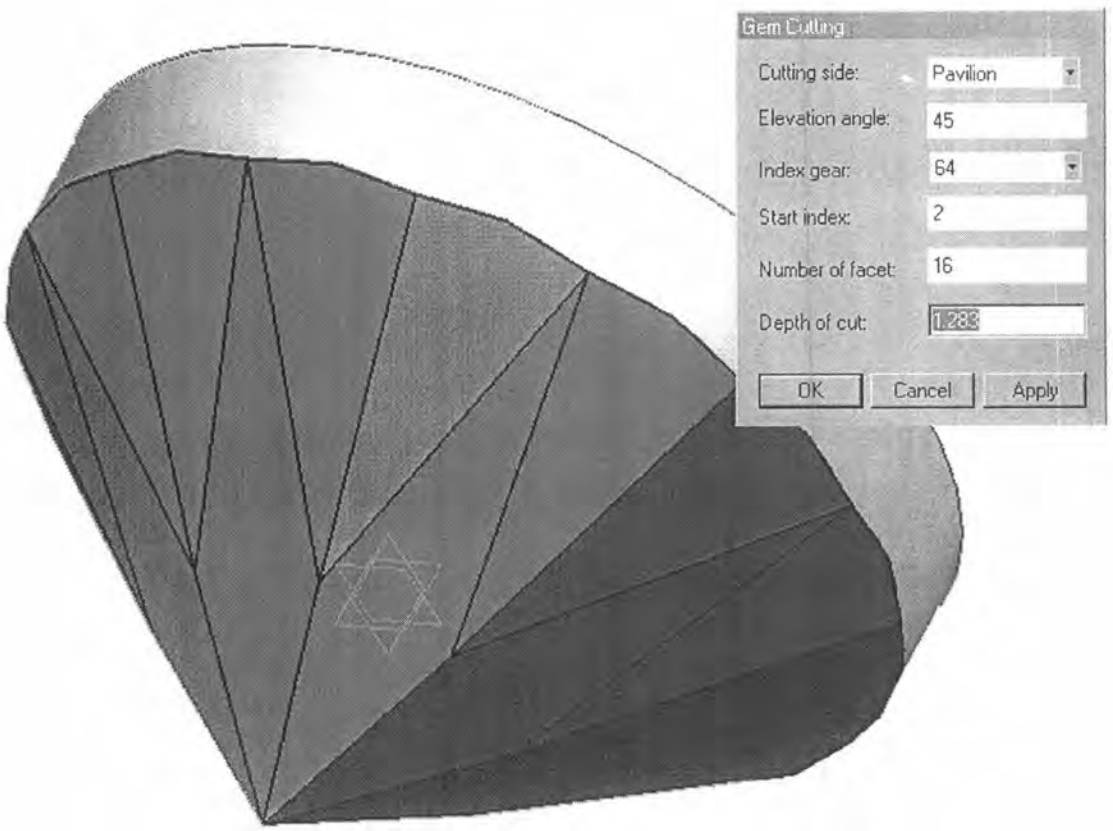
รูปที่ ค.2 การตัดรูปทรงด้วยระนาบจนได้โมเดลพลอยที่ต้องการ

การประยุกต์ใช้โปรแกรม CU-Solid กับเครื่องเจียระไนพลอยอัตโนมัติ เพื่อที่จะเป็นข้อมูลในการเจียระไนพลอย มีประโยชน์ทั้งในด้านของเหลี่ยมมุมการเจียระไน รวมถึงระยะความลึกในการเจียระไน ทำให้เราสามารถทราบระยะความลึกในการเจียระไนในแต่ละมุม โดย

การนำระยะตัดจนถึงยอดแหลม มาลบกับระยะของความลึกที่ต้องการ จะได้เป็นระยะความลึกที่ต้องลดลงจากยอดแหลม ซึ่งสามารถนำไปใช้เป็นข้อมูลความลึกในการเจียระไนของพลอยได้



รูปที่ ค.3 ข้อมูลความลึกเมื่อตัดเหลี่ยม Half ของก้นพลอยจนแหลม



รูปที่ ค.4 ข้อมูลความลึกเมื่อตัดเหลี่ยม Half ของก้นพลอยพอดี

จากรูปที่ ค.3 จะได้ระยะความลึกเมื่อตัดเหลี่ยม Half ของก้นพลอยจน (Pavilion) แหลมเท่ากับ 1.19 มิลลิเมตร และจากรูปที่ ค.4 จะได้ระยะความลึกเมื่อตัดเหลี่ยม half ของก้นพลอย (Pavilion) พอดี เท่ากับ 1.238 มิลลิเมตร ดังนั้นระยะที่จะนำไปใช้ในการเจียระไนให้ลึกพอดีเพื่อใช้เลื่อน (Offset) จากระยะที่ยอดแหลมเท่ากับ

$$\text{Offset} = Ds - De \qquad \text{(ค.1)}$$

$$= 1.19 - 1.238$$

$$= -0.048 \text{ มิลลิเมตร}$$

เมื่อ Offset คือ ระยะที่จะนำไปใช้ในการเจียระไนให้ลึกพอดีเพื่อใช้เลื่อน (Offset) จากระยะที่ยอดแหลม

Ds คือ ระยะความลึกเมื่อตัดเหลี่ยม Half ของก้นพลอยจน (Pavilion) แหลม

De คือ ระยะความลึกเมื่อตัดเหลี่ยม Half ของก้นพลอย (Pavilion) พอดี

จากสมการที่ (ค.1) จะได้ระยะที่จะนำไปใช้ในการเจียระไนให้ลึกพอดีเพื่อใช้เลื่อน (Offset) จากระยะที่ยอดแหลม เพื่อที่จะนำไปใส่ในโปรแกรมการเจียระไนพลอยในหน้าก้นพลอย (Pavilion) เท่ากับ -0.048 มิลลิเมตร

ประวัติผู้เขียนวิทยานิพนธ์

นายสมจินต์ ธนานันทศิริ เกิดเมื่อวันที่ 21 กุมภาพันธ์ พ.ศ. 2524 เป็นชาวกรุงเทพมหานคร เข้าศึกษาชั้นประถมศึกษา ที่โรงเรียนสีตบุตรบำรุง เมื่อสำเร็จการศึกษาชั้นประถมปีที่หก ได้ทำการสอบเข้าศึกษาต่อชั้นมัธยมศึกษา ที่โรงเรียนเทพศิรินทร์ หลังจากนั้นได้เข้าศึกษาต่อในคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ และได้สำเร็จปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมเครื่องกล ในปีการศึกษา 2545 และเข้าศึกษาต่อในระดับปริญญามหาบัณฑิต สาขาวิศวกรรมเครื่องกล จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2546